Eric Näser
ID: 202300343

Report Assignment 2

Task 1 - Problem and Numerical Method

Which task was given and which numerical method did I use to solve it?

GitHub Repository:

https://github.com/eric-official/advanced-numerical-methods

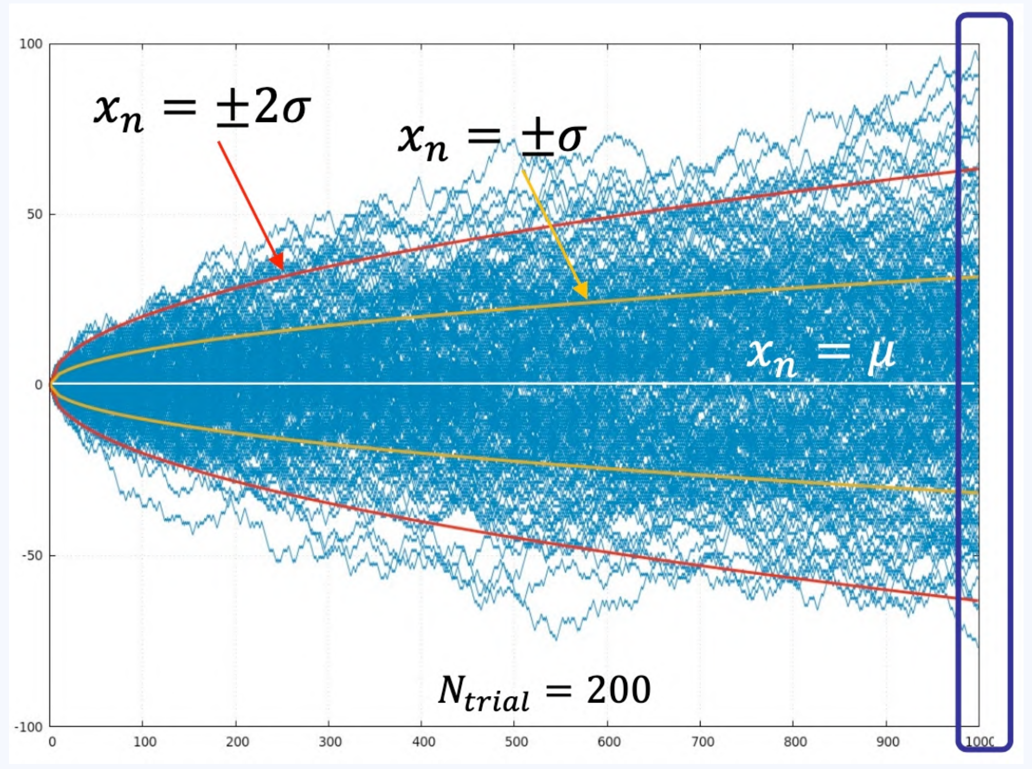
Random Walk

In the first task, random walk has to be modeled. At each step during the walk, the position is changed by a certain value with a defined probability. The following details were provided:

- The random walk starts at 0
- At each step, the position changes by either -1 or +1
- The probability for each value of position change is 0,5 respectively
- The random walks follows a binomial, or for a large number of trials, a normal distribution

Illustration of Problem

The given problem can illustrated in the following diagram:



Numerical Method

To solve the described problem, the random walks have been modelled with Monte Carlo simulation:

- Simulations use a random number in the algorithm
- Here: position change is randomized with a value either -1 or +1
- Random walks are non-deterministic
- Still, they are related via statistical error and result in a certain distribution for a large number of trials
- Here: 10,000 trials

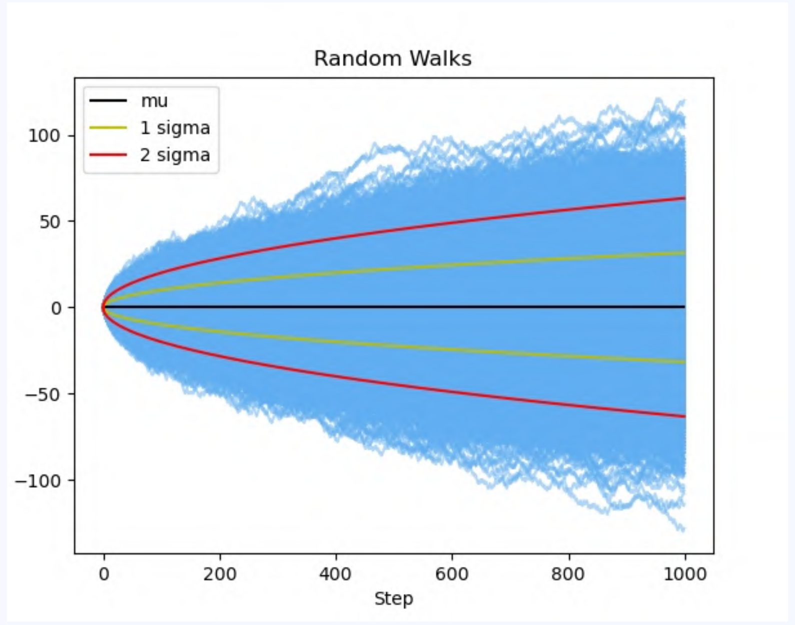
Task 1 - Results and Discussion

Which results have I achieved with my solution?

GitHub Repository:

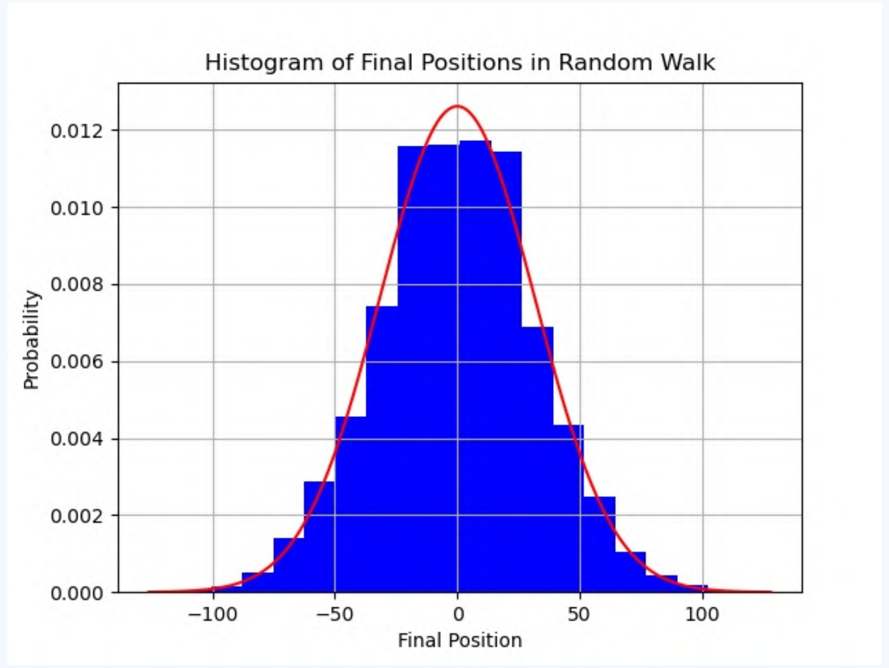
https://github.com/eric-official/advanced-numerical-methods

Random Walk Plot



Random walks have been conducted for 1,000 steps in 10,000 trials. During each step, the position was changed by -1 or +1 with a probability of 0.5 respectively. Furthermore, the mean (mu), single standard deviation (1 sigma) and double standard deviation (2 sigma) has been plotted. In the plot, it can be seen that the distribution of random walks is centred around the mean (mu). Also, the bell shape of the random walk distribution can be easily seen. This distribution of random walks is expected due to an equal probability of 0.5 for a -1 or a +1 walk. Extreme values for final positions, which random walks reach after 1,000 steps, are approximately -100 and +100. The number of random walks gets less the closer their end position is at the extreme values. Again, this observation is as expected due to the increasing distance from the mean as well as the decreasing share of samples outside of one-sigma and two-sigma by theory.

Random Walk Histogram



The Monte Carlo Simulations for random walks have also been visualized in a histogram. The histogram grouped the random walks in 20 bins and shows their respective probability. Furthermore, the plot shows the probability density function of the normal distribution as a red line. Similar to the plot of random walks, a bell-shaped distribution is visible. The highest probabilities are concentrated around the mean of zero. With a greater distance of the final position from zero, the probability is decreasing. The visibility of the bell-shaped curve gets even more clearer with the probability density function of the normal distribution.

Task 2 - Problem and Numerical Methods

Which task was given and which numerical methods did I use to solve it?

GitHub Repository:

https://github.com/eric-official/advanced-numerical-methods

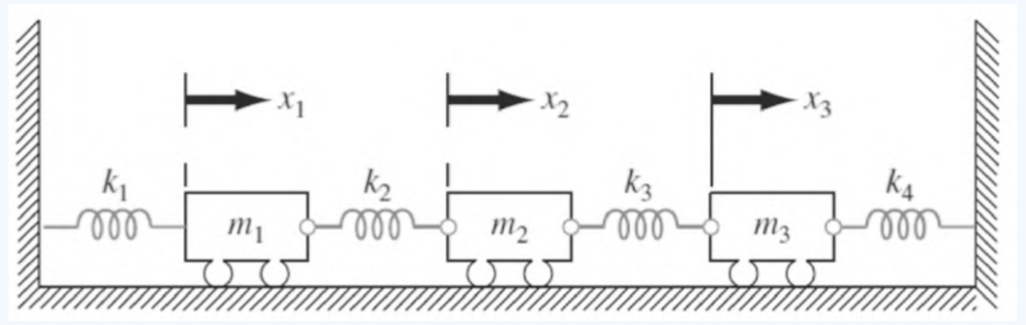
Three-mass vibrating system

In the second task, the three-mass vibrating system has to be solved.

- The differential equations of motions had to be derived
- The eigenvalues and natural frequencies had to be calculated for k1=k4=15 N/m, k2=k3=35 N/m, and m1=m2=m3=1.5 kg
- For calculating the eigenvalues two numerical methods had be used
- The modal shapes had be drawn.

Illustration of Problem

The given problem can illustrated in the following figure:



Numerical Methods

To calculate the eigenvalues of the three-mass vibrating system, two numerical methods were used:

1. Shifted Power Method
   - power method determines the eigenvalue farthest from zero
   - matrix gets iteratively multiplied with vector and the result will be normalized
   - matrix can be shifted iteratively and power method applied respectively for finding all eigenvalues
2. QR Algorithm
   - finds all eigenvalues iteratively
   - each iteration
     - matrix gets QR decomposed
     - R and Q matrices get multiplied
     - new eigenvalues are diagonal of matrix

Task 2 - Matrix Construction

How did I came up with the matrix?

GitHub Repository:

https://github.com/eric-official/advanced-numerical-methods

1. Initial Equations of Motions

$$\begin{aligned} m_1\ddot{x}_1 &= -k_1x_1 + k_2(x_2 - x_1) \\ m_2\ddot{x}_2 &= -k_2(x_2 - x_1) + k_3(x_3 - x_2) \\ m_3\ddot{x}_3 &= -k_3(x_3 - x_2) - k_4x_3 \end{aligned}$$

2. Rearrange Equations of Motions

$$\begin{aligned} \ddot{x}_1 &= -\frac{k_1 + k_2}{m_1}x_1 + \frac{k_2}{m_1}x_2 \\ \ddot{x}_2 &= \frac{k_2}{m_2}x_1 - \frac{k_2 + k_3}{m_2}x_2 + \frac{k_3}{m_2}x_3 \\ \ddot{x}_3 &= \frac{k_3}{m_3}x_2 - \frac{k_3 + k_4}{m_3}x_3 \end{aligned}$$

3. Bring Equations of Motions in Matrix Notation

$$\begin{bmatrix} -\frac{k_1+k_2}{m_1} & \frac{k_2}{m_1} & 0 \\ \frac{k_2}{m_2} & -\frac{k_2+k_3}{m_2} & \frac{k_3}{m_2} \\ 0 & \frac{k_3}{m_3} & -\frac{k_3+k_4}{m_3} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \ddot{x}_1 \\ \ddot{x}_2 \\ \ddot{x}_3 \end{bmatrix}$$

4. Calculate Matrix Values for k1=k4=15, k2=k3=35, and m1=m2=m3=1.5

$$\begin{bmatrix} -\frac{50}{1.5} & \frac{35}{1.5} & 0 \\ \frac{35}{1.5} & -\frac{70}{1.5} & \frac{35}{1.5} \\ 0 & \frac{35}{1.5} & -\frac{50}{1.5} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \ddot{x}_1 \\ \ddot{x}_2 \\ \ddot{x}_3 \end{bmatrix}$$

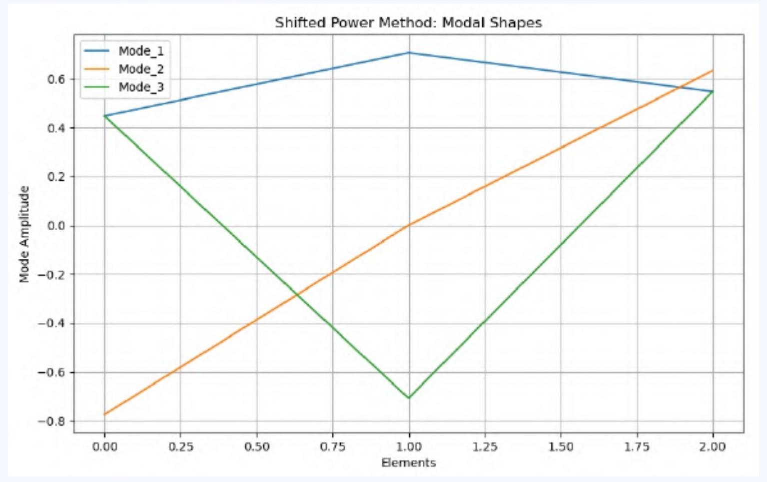
Task 2 - Results and Discussion

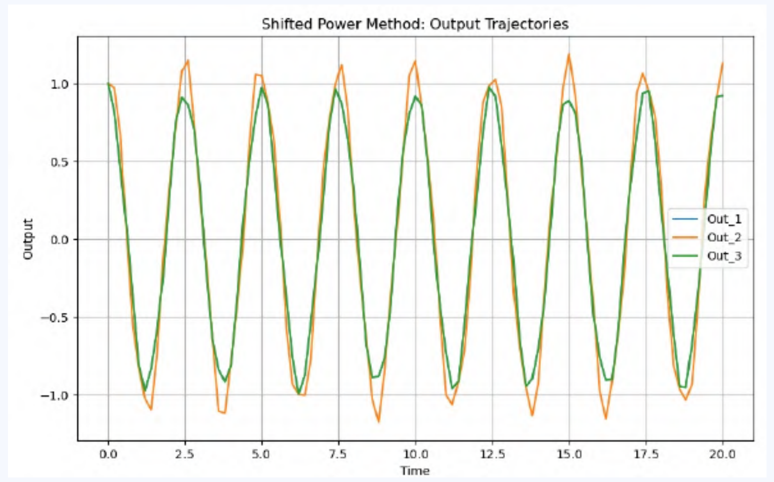
Which results have I achieved with my solution?

GitHub Repository:

https://github.com/eric-official/advanced-numerical-methods

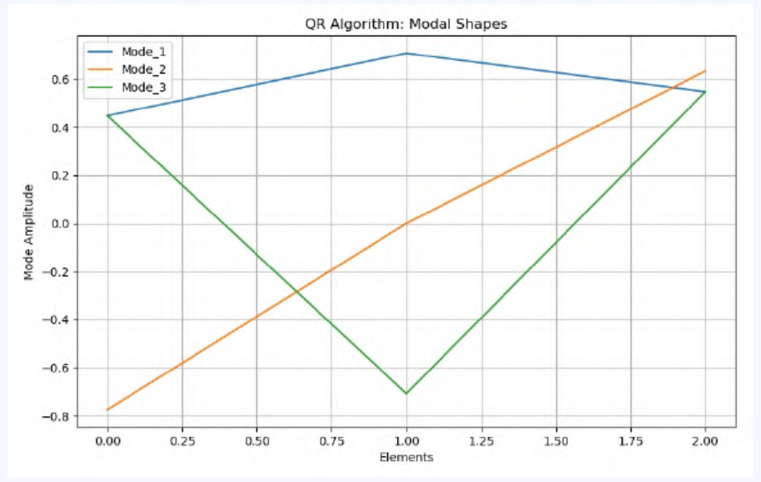
Shifted Power Method

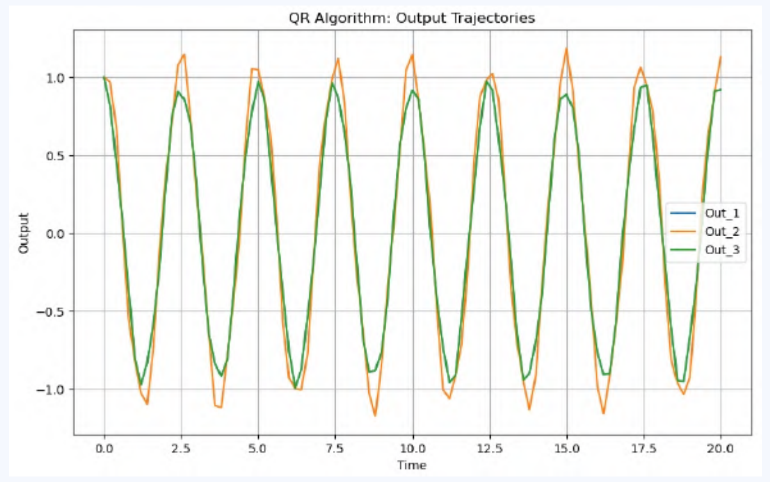




For analyzing the results of the Shifted Power Method, the eigenvalues and natural frequencies are presented as well as modal shapes are plotted. The Shifted Power Method yields the eigenvalues -73.665, -40.0 and -6.335. The first and the third eigenvalues are identical with the solution of the Python library Numpy. Only the second eigenvalue is slightly different. Furthermore, the Shifted Power Algorithm yields the natural frequencies 8.58, 6.32 and 2.52. Those values suggest that the first frequency has the fastest vibration oscillations.

QR Algorithm





The analysis of the QR algorithm follows the steps of the analysis of the Shifted Power Method. Again, the eigenvalues and natural frequencies are presented as well as modal shapes are plotted. The QR algorithm yields the eigenvalues -73.665, -33.333 and -6.335. Those results are all identical with the solution of the Python library Numpy. Furthermore, the QR algorithm yields the natural frequencies 8.58, 5.77 and 2.52. Those values again suggest that the first frequency has the fastest vibration oscillations.

Try Pitch

Task 3 - Problem and Numerical Methods

Which task was given and which numerical method did I use to solve it?

GitHub Repository:

https://github.com/eric-official/advanced-numerical-methods

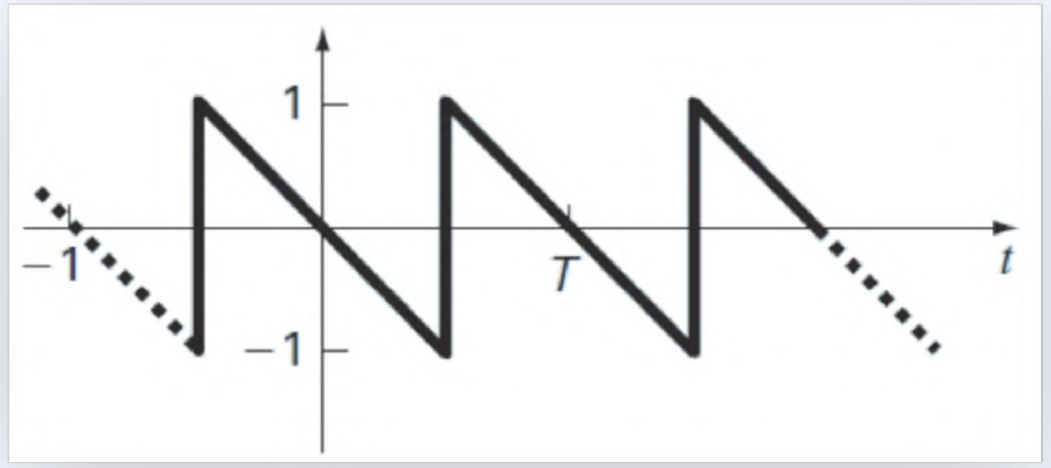
Sawtooth Wave

The third task is concerned with the sawtooth wave . Said sawtooth wave is a function that decreases linearly during a certain period and then is set back to a starting value, which forms the sawtooth shape. For the sawtooth wave the Fourier series expansion had to be found and plotted for four terms as well as Fast Fourier Transform carried out. The following details were provided:

- Sawtooth wave oscillates between -1.0 and 1.0
- T and N could be chosen freely

Illustration of Problem

The given problem can illustrated in the following diagram:



Numerical Methods

To solve the described problem, the **Fourier series expansion** had to be found an the **Fast Fourier Transformation** had to be carried out:

The **Fourier series expansion** represents a periodic function as the sum of sine and cosine functions. For a given periodic function f(t), the expansion involves coefficients a0, an and bn, calculated from integrals. Therefore, this method simplifies the analysis of complex periodic functions. The **Fast Fourier Transform** (FFT) is an algorithm for computing the Discrete Fourier Transform (DFT) of a sequence or signal. It speeds up the computation of the Fourier transform compared to the standard DFT calculation.

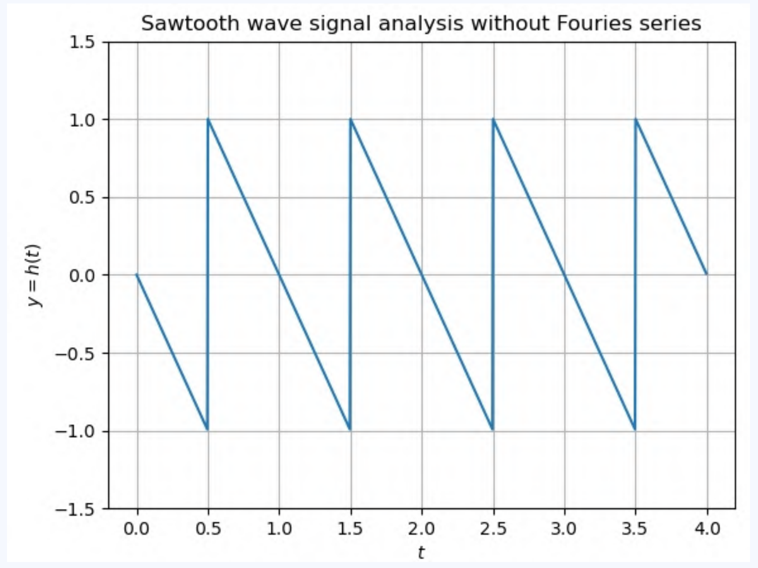
Task 3 - Results and Discussion

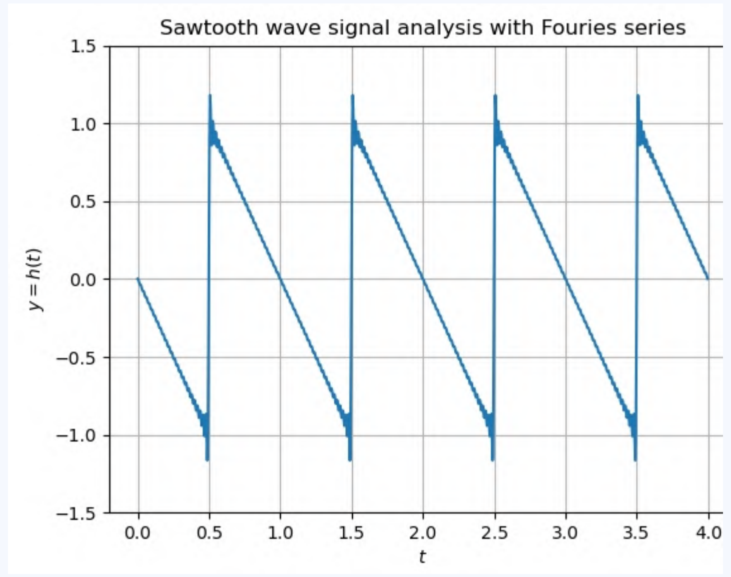
Which results have I achieved with my solution?

GitHub Repository:

https://github.com/eric-official/advanced-numerical-methods

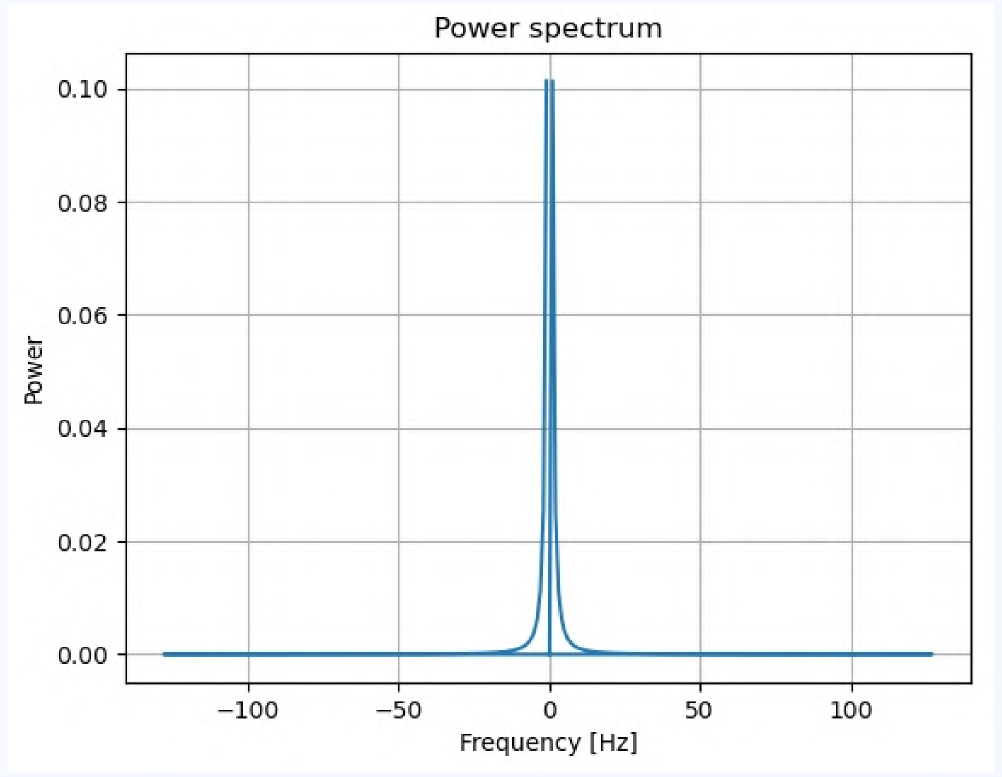
Sawtooth Wave without and with Fourier Series Expansion





The plots above show the original sawtooth wave on the left and the Fourier expansion of the sawtooth wave on the right. Each plot shows the first four terms. The sawtooth wave has been initialized with with T=1 and a frequency of 1 Hz. Furthermore, the sawtooth wave oscillates between y-values of -1.0 and 1.0. The resulting wave of Fourier series expansion (N=50) shows the same pattern. A noticeable difference to the original wave is the function behavior. The Fourier series expansions shows little oscillations around the turning points with values -1.0 and 1.0.

Power Spectrum



This plot shows the power spectrum of the Fast Fourier Transform. The power at the y-axis is normalized with the number of samples (256). It can be seen that power spectrum is symmetric to the y-axis. For large negative frequencies the power stays at 0 Hz. At approximately a frequency of -10 Hz the power dramatically increases. In the interval of 0 Hz and 10 Hz the power then dramatically decreases again and afterwards remains at 0 Hz.

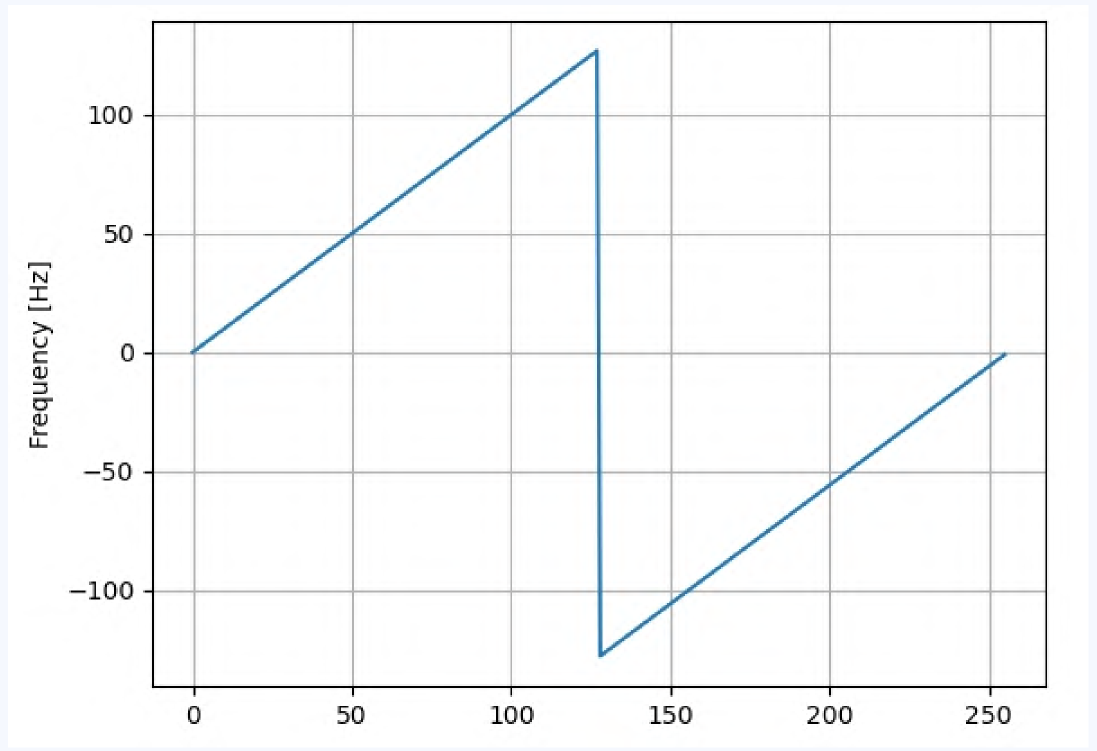
Task 3 - Results and Discussion

Which results have I achieved with my solution?

GitHub Repository:

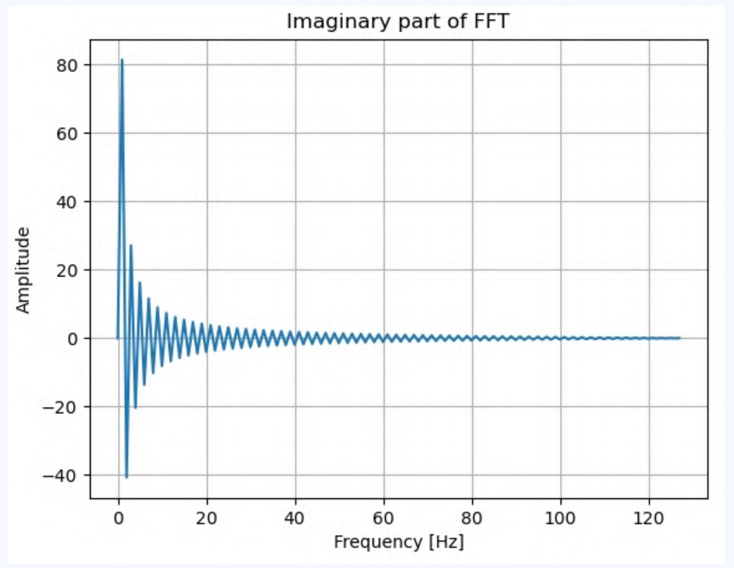
https://github.com/eric-official/advanced-numerical-methods

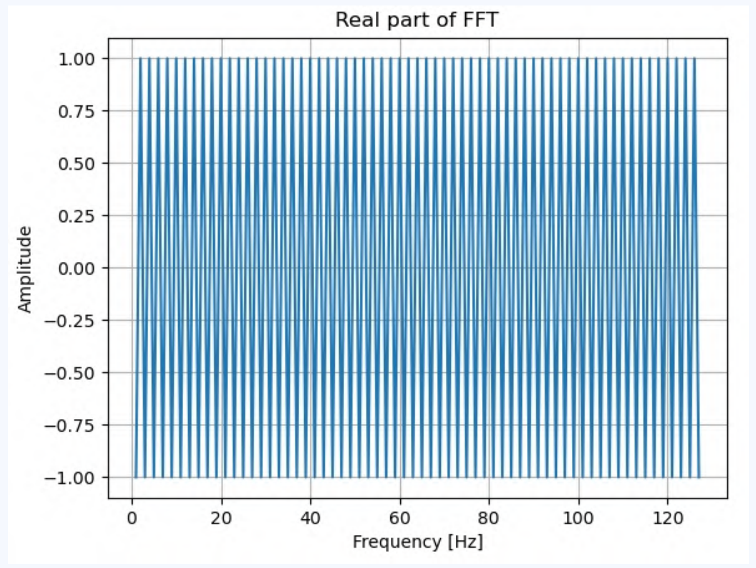
FFT Frequency



The plot above shows the frequency axis. As it can be seen, the frequency starts at 0 Hz and increases until 128 Hz. Then it drastically drops until -128 Hz at half of the sampling rate and increases again until 0 Hz. The maximum frequency that can hold information from the signal, 128 Hz, is half of the sampling rate and this is true according to the Sampling Theorem.

Imaginary and Real Part of FFT





The left plot shows the imaginary part of the Fast Fourier Transform. For low values of frequency, the amplitude is the highest. The maximum amplitude has value 80 and the minimum amplitude has value -40. With an increase of frequency, the amplitude decreases. As it can be seen, the amplitude converges towards zero. The right plot shows the real part of the Fast Fourier Transform. Contrary to the imaginary part, the real part show constant fluctuations of the amplitude between -1.0 and 1.0.

References

Which books, documents and websites have I used to solve the homework?

[1]‘Python Programming And Numerical Methods: A Guide For Engineers And Scientists — Python Numerical Methods’. Accessed: Oct. 31, 2023. [Online]. Available: https://pythonnumericalmethods.berkeley.edu/notebooks/Index.html
[2]J. Kiusalaas, ‘Numerical Methods in Engineering with Python 3’., p. 336-351
[3]D. E. Stewart, *Numerical Analysis: A Graduate Course*, vol. 4. in CMS/CAIMS Books in Mathematics, vol. 4. Cham: Springer International Publishing, 2022. doi: 10.1007/978-3-031-08121-7., p. 316-321
[4]T. A. Beu, J. Adler, K. Roos, and J. Driscoll, ‘INTRODUCTION TO NUMERICAL PROGRAMMING: A Practical Guide for Scientists and Engineers Using Python and C/C++’.
[5]T. A. Beu, J. Adler, K. Roos, and J. Driscoll, ‘INTRODUCTION TO NUMERICAL PROGRAMMING: A Practical Guide for Scientists and Engineers Using Python and C/C++’.p. 233-234
[6]‘Fourier Transform, the Practical Python Implementation | by Omar Alkousa | Towards Data Science’. Accessed: Dec. 07, 2023. [Online]. Available: https://towardsdatascience.com/fourier-transform-the-practical-python-implementation-acdd32f1b96a
[7]‘Fourier Series Examples’. Accessed: Dec. 07, 2023. [Online]. Available: https://lpsa.swarthmore.edu/Fourier/Series/ExFS.html
[8]‘Eigenvalues for Vibration Problems’. Accessed: Dec. 07, 2023. [Online]. Available: https://lpsa.swarthmore.edu/MtrxVibe/EigApp/EigVib.html
[9]‘Discrete Fourier Transform (DFT) — Python Numerical Methods’. Accessed: Dec. 07, 2023. [Online]. Available: https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.02-Discrete-Fourier-Transform.html
[10]*Day 151: Generate Sine , Square , Triangle and Sawtooth in Python*, (May 12, 2023). Accessed: Dec. 07, 2023. [Online Video]. Available: https://www.youtube.com/watch?v=6lETCqsDmOw
[11]D. S. Bhar, ‘Analysis of Fourier series using Python Code’.
[12]‘3. Fourier Series of Even and Odd Functions’. Accessed: Dec. 07, 2023. [Online]. Available: https://www.intmath.com/fourier-series/3-fourier-even-odd-functions.php