Graduate Computational Physics Problem Set #8
Prithivi Rana. (11/26)

Problem 1. *Ex 7.3: Fourier transforms of musical instruments*

In the online resources, you will find files called piano. txt and trumpet. txt, which contains data representing the waveform of a single note, played on, respectively, a piano and a trumpet.

a) Write a program that loads a waveform from one of these files, plots it, then calculates its discrete Fourier transform and plots the magnitudes of the first 10,000 coefficients in a manner similar to Fig. 7.4. Note that you will have to use a fast Fourier transform for the calculation because there are too many samples in the files to do the transforms the slow way in any reasonable amount of time. Apply your program to the piano and trumpet waveforms and discuss briefly what one can conclude about the sound of the piano and trumpet from the plots of Fourier coefficients.

Solution.

a) The first part of the problem tasks us with loading the waveform from the files and plotting them. This is pretty straightforward and was achieved using Numpy's `np.loadtxt` and matplotlib. The waveforms from the files are plotted in figure 1.

The piano's waveform looks to be clear and suggests it is playing one fundamental frequency and harmonics. The trumpet's waveform however looks more noisy and complex and suggests there are multiple frequencies with high magnitudes present in the waveform.

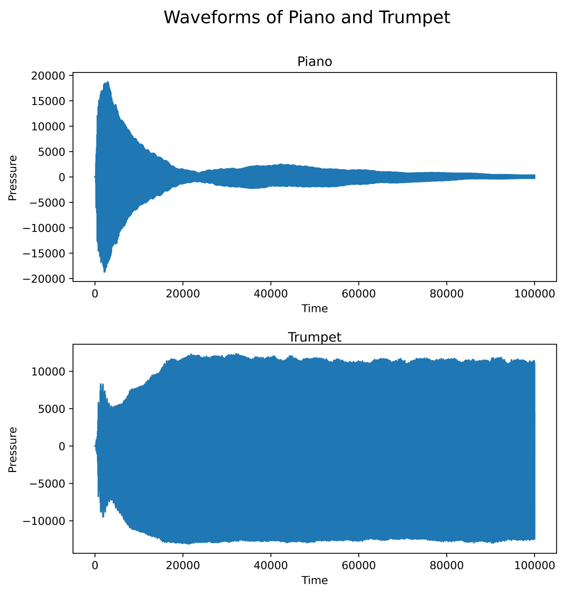


Figure 1: Waveforms from the data files

Performing a Fourier transform on the waveforms using Numpy's `ftt` routine and plotting the magnitude (using `np.abs()`) of the first 10,000 coefficients gives us the plots in figures 2 and 3.

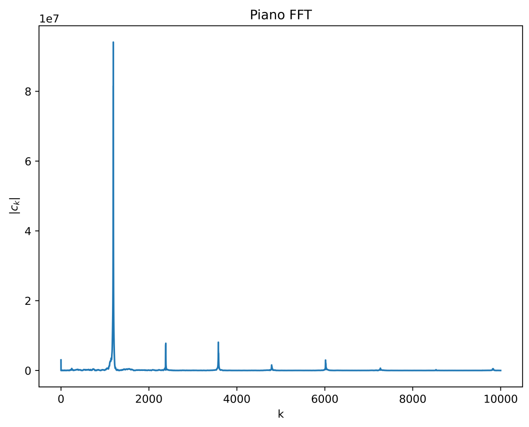


Figure 2: FFT performed on the piano waveform

The Fourier transform of the piano waveform shows that it has one distinct fundamental frequency and several evenly spaced harmonics that are small in magnitude. The smaller peaks don't contribute much to the overall sound of the piano which supports our claim that the piano produces a clear note.

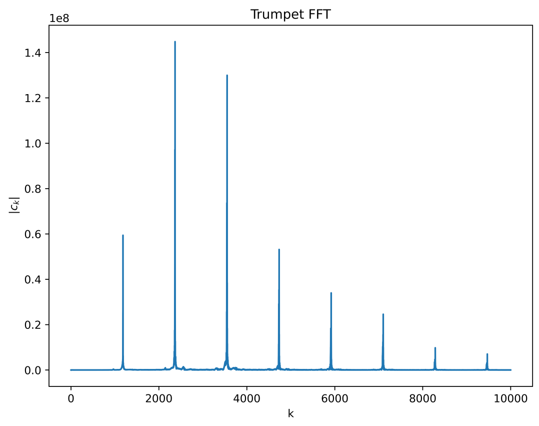


Figure 3: FFT performed on the trumpet waveform

The Fourier transform of the trumpet's sound shows that its waveform has numerous evenly spaced peaks, each with significant magnitudes. This explains why the trumpet's waveform looks more complex since it has significant contributions from several frequencies.

b) Both waveforms were recorded at the industry-standard rate of 44100 samples per second and both instruments were playing the same musical note when the recordings were made.

From your Fourier transform results calculate what note they were playing. (Hint: The musical note middle C has a frequency of 261 Hz.)

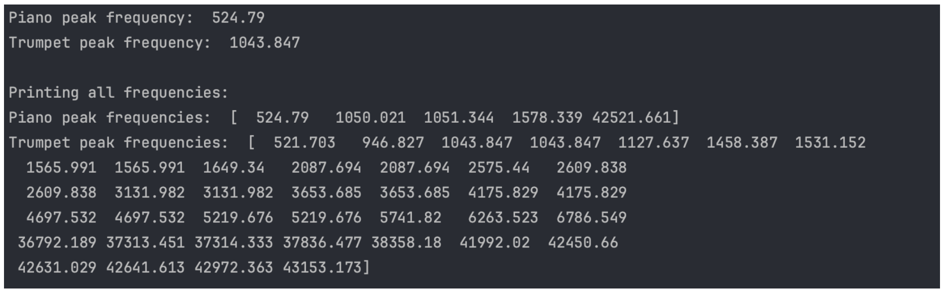
In order to find the frequencies defined by the peaks of our FFT, we first need to find the frequency resolution (or interval). The frequency resolution is given by:

$$res_f = \frac{f_s}{N}$$

where, f_s is the sampling rate (44,100 in our case) and `N` is the total number of samples (`len(piano or trumpet)`). The frequency related to a sample point `k` is then $k * res_f$.

Finding the frequencies related to the greatest peaks using `np.argmax()` and multiplying them with res_f gives us 524.79 Hz for the piano and 1043.85 Hz for the trumpet (Figure 4). These frequencies are multiples of the musical note middle C which suggests that these are higher octaves of middle C. Comparing these values with a database of frequencies of musical notes, the most dominant frequency of the piano corresponds to C_5 (523.25 Hz) and the trumpet corresponds to C_6 (1046.50 Hz) [1]

Although the dominant frequencies are indicative of the final waveform, the note being played might be better explained by the first significant peak in the graph. In the case of the piano, the first significant peak is the peak at 524.79 Hz. However, 1043.85 Hz is the second peak in the FFT plot of the trumpet. In order to find unique frequencies, I grouped frequencies that appeared in a close range and took the frequency with the highest magnitude as the true dominant frequency of that range. These values can be observed in figure 4. Comparing these values, we can see that the trumpet's first significant peak is at 521.70 Hz, which also corresponds to C_5. This confirms that both instruments were playing the same note and that the trumpet has dominant contributions from higher octaves.



```
Piano peak frequency:  524.79
Trumpet peak frequency:  1043.847

Printing all frequencies:
Piano peak frequencies:  [  524.79   1050.021  1051.344  1578.339 42521.661]
Trumpet peak frequencies:  [  521.703   946.827  1043.847  1043.847  1127.637  1458.387  1531.152
  1565.991  1565.991  1649.34   2087.694  2087.694  2575.44   2609.838
  2609.838  3131.982  3131.982  3653.685  3653.685  4175.829  4175.829
  4697.532  4697.532  5219.676  5219.676  5741.82   6263.523  6786.549
 36792.189 37313.451 37314.333 37836.477 38358.18  41992.02  42450.66
 42631.029 42641.613 42972.363 43153.173]
```

Figure 4: Peak frequencies

Problem 2. *Exercise 8.3: The Lorenz equations*

One of the most celebrated sets of differential equations in physics is the Lorenz equations:

$$\frac{dx}{dt} = \sigma(y-x), \quad \frac{dy}{dt} = rx - y - xz, \quad \frac{dz}{dt} = xy - bz$$

where σ, r, and b are constants. (The names σ, r, and b are odd, but traditional -they are always used in these equations for historical reasons.)

These equations were first studied by Edward Lorenz in 1963, who derived them from a simplified model of weather patterns. The reason for their fame is that they were one of the first incontrovertible examples of deterministic chaos, the occurrence of apparently random motion even though there is no randomness built into the equations. We encountered a different example of chaos in the logistic map of Exercise 3.6.

a) Write a program to solve the Lorenz equations for the case σ = 10, r = 28, and $b = \frac{8}{3}$ in the range from $t = 0$ to $t = 50$ with initial conditions (x,y,z) = (0,1,0). Have your program make a plot of y as a function of time. Note the unpredictable nature of the motion.

Solution. This problem tasks us with numerically solving a set of three coupled differential equations that exhibit deterministic chaos. In order to solve the Lorenz equations, I first wrote a function that accepts the current x,y, and z values as arguments (along with the constants) and outputs an array containing the $\frac{dx}{dt}, \frac{dy}{dt}, and \frac{dz}{dt}$ values calculated with the current x,y, and z state.

Setting the values of the constants and initial conditions as defined by the problem lets us use SciPy's `integrate.solve_ivp` routine to solve the Lorenz equations defined in our function. For this problem, I used the 4^{th} and 5^{th} order Runge-Kutta methods implemented in the SciPy package as `RK45` using a maximum step of 0.01 since that resulted in the smoothest plots after trying out various maximum step sizes.

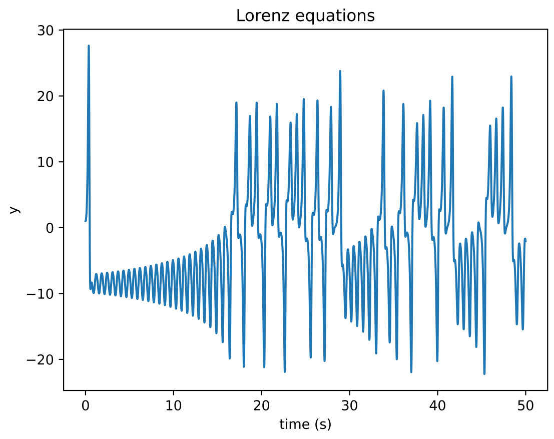


Figure 5: y vs time

Plotting the y values against time gives us the plot in figure 5. This plot exhibits an irregular and unpredictable pattern. There are no repeating patterns that can be distinguished in the plot and it seems to fluctuate randomly indicating the chaotic nature of the system.

b) Modify your program to produce a plot of z against x. You should see a picture of the famous "strange attractor" of the Lorenz equations, a lopsided butterfly-shaped plot that never repeats itself.

The plot of z against x exhibits a butterfly-shaped plot with two elliptical wings (figure 6. The line follows a complex non-repeating path which again indicates the chaotic nature of the Lorenz system.

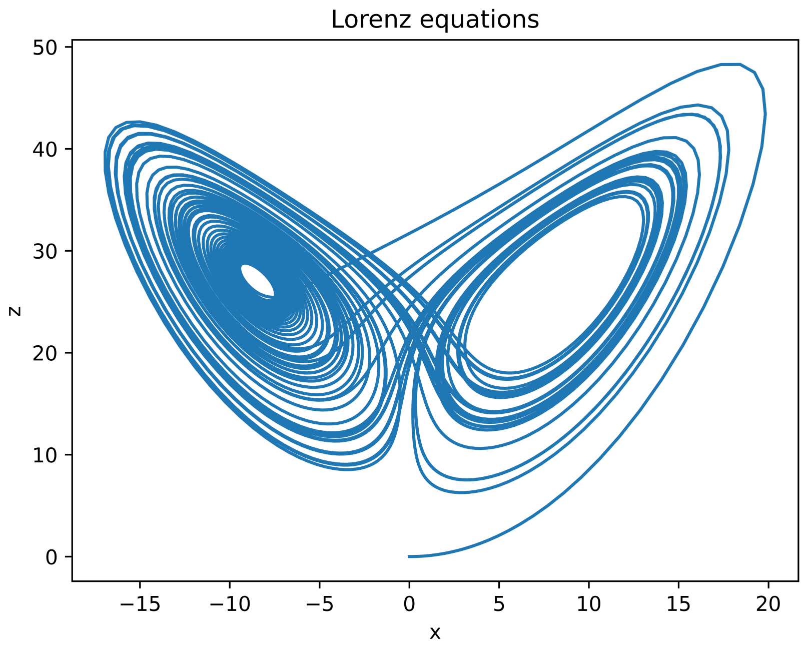


Figure 6: Lorenz "strange attractor"

References

[1] Michigan Tech University. Physics of music - notes. `https://pages.mtu.edu/~suits/notefreqs.html`.