Upload Date:	`[Moodle Assignments]`	April 25th, 2023
Hand-In Deadline:	`[Moodle Assignments]`	May 2nd, 2023, 08:45
Correction Session:	`[VU class]`	May 2nd, 2023

11 **Basic Fourier Transform.** In this exercise, you are going to implement and plot numerical Fourier transform, as well as numerical inverse Fourier transform. You may use modules like `numpy` to handle operations on complex numbers and arrays, but the Fourier transform and its inverse must both be coded from scratch.

1. Define a function $f(x)$. Create arrays with $n = \{10, 50, 100\}$ points $\in [-\pi; \pi]$ and visualize $f(x)$ for all three cases.

$$f(x) = \sin(x) + 2\cos(2x) + 2\sin(3x). \tag{1}$$

2. Define a function `FourierTrafo()` which takes in an f-array $\{f_0, f_1, ..., f_{N-1}\}$, and returns a Fourier-transformed array of values $f^* = \{f_0^*, f_1^*, ..., f_{N-1}^*\}$. The following equation must be satisfied:

$$f_k^* = \sum_{l=0}^{N-1} f_l \cdot e^{-\frac{2\pi i}{N} \cdot k \cdot l}$$

3. Define a function `InverseFourierTrafo()` which takes in an f^*-array $\{f_0^*, f_1^*, ..., f_{N-1}^*\}$, and returns an inverse-Fourier-transformed array of values $f = \{f_0, f_1, ..., f_{N-1}\}$. The following equation must be satisfied:

$$f_k = \frac{1}{N} \sum_{l=0}^{N-1} f_l^* \cdot e^{\frac{2\pi i}{N} \cdot k \cdot l}$$

4. Compute the Fourier transform f^* for your function values. Then, for the resulting array f^*, compute the inverse Fourier transform f'. Compute the Euclidean-2 norm $\|f - f'\|$.
5. Plot f' and f^* into two subplots for all 3 values of n (think carefully which values to use on your x-axis when plotting f^*), and label your graphs properly.
6. Explain the results and give an interpretation. **Hint:** You can look at the documentation for `numpy.fft.fftfreq` by calling `help(numpy.fft.fftfreq)`, if you are unsure about which x-axis values to use for the Fourier-transformed function values. To be clear: Do not use the `numpy.fft` for this task, but you can look at its individual function documentations to find out more about what you want to do.

12 **Filtering noise.** In this exercise, you are going to "clean" some rather noisy data and extract the underlying function.

1. Read the data from `ex12_data.csv`, which can be found on Moodle, store it in an array and visualize the data.
2. Check if your x-data is equidistant, and then use `numpy.fft` to calculate the Fourier transform f^*. Plot f^* with the aid of `numpy.fft.fftfreq`.
3. Have a look at the plot of f^* and define a threshold ϵ such that you set any values where $\text{abs}(x^*) > \epsilon$ to 0. Compute the inverse Fourier transform of your modified data (use `numpy.fft.ifft`, for example). Generate another plot showing the noisy data and the filtered data and their corresponding Fourier transforms for different values of ϵ.
4. What is a good choice for ϵ?

Note: When comparing the run time of your own Fourier Transform to the one in this exercise, you might notice a significant speed increase. Here, we have been using Fast Fourier Transform (FFT), which indeed significantly outperforms the hard-coded Fourier Transform we have employed in the prior exercise.