Assignment #4: Fourier analysis using Python (**total 10 points**), due by 3:00 pm Monday, 27 March 2023

This assignment introduces the following:

- Fourier analysis of both periodic and non-periodic signals (Fourier series, Fourier transform, Discrete Fourier transform);
- The use of Simpson's rule for numerical integration.

Fourier analysis is an extremely important tool in the investigation of signals; it essentially decomposes a signal into constituent harmonic vibrations. Sometimes the signals are periodic, and the period can be measured directly. But Fourier analysis is also useful for investigating signals which are not periodic. In the first part of this assignment, you will deal with periodic signals, while in the second part you will see that the numerical Fourier method applies with some modification also to the case of non-periodic signals (known as the Discrete Fourier Transform, or DFT.)

1 Fourier series

Any periodic function $f(t)$, with period $T = 2\pi/\omega$, can be represented as a Fourier series

$$f(t) = a_0 + \sum_{n=1}^{\infty} [a_n \cos(n\omega t) + b_n \sin(n\omega t)] \tag{1}$$

The sine and cosine functions are harmonic functions, and the series (1) contains a possibly infinite set of harmonic functions with discrete frequencies $\omega_n = n\omega$, $n = 1, 2, \ldots$. The frequency $\omega_1 = \omega$ is known as the fundamental frequency, and ω_n, $n > 1$, are the harmonics. The coefficients a_n and b_n measure the 'amount' of $\cos(n\omega t)$ and $\sin(n\omega t)$ present in the function $f(t)$. The result of Fourier-analysing a signal is a set of values for these coefficients for all n. In practice, the coefficients will be obtained for all n up to some finite maximum N. The Fourier coefficients are evaluated using the orthogonality properties of sines and cosines:

$$\frac{2}{T}\int_0^T \sin(n\omega t)\sin(k\omega t)dt = \delta_{nk} \tag{2}$$

$$\frac{2}{T}\int_0^T \cos(n\omega t)\sin(k\omega t)dt = 0 \tag{3}$$

$$\frac{2}{T}\int_0^T \cos(n\omega t)\cos(k\omega t)dt = \delta_{nk} \tag{4}$$

where δ_{nk} is the Kronecker delta. Applying these orthogonality rules onto equation (1), we obtain the coefficients

$$
\begin{aligned}
a_0 &= \frac{1}{T}\int_0^T f(t)dt && (5)\\
a_k &= \frac{2}{T}\int_0^T f(t)\cos(k\omega t)dt \qquad k=1,2,\ldots && (6)\\
b_k &= \frac{2}{T}\int_0^T f(t)\sin(k\omega t)dt \qquad k=1,2,\ldots && (7)
\end{aligned}
$$

1.1 Simpson's Rule

A Fourier analysis program must perform the integrations as shown in the equations above for any function $f(t)$ of interest. Simpson's rule will be used in this assignment to perform our numerical integrations. Simpson's rule approximates the integral $\int_a^b dx\, f(x)$ by splitting the interval from $x = a$ to $x = b$ into n steps of equal length $h = (b - a)/n$ where n is an even number. The integration is approximated as

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left[f(x_0) + 2\sum_{j=1}^{n/2-1} f(x_{2j}) + 4\sum_{j=1}^{n/2} f(x_{2j-1}) + f(x_n)\right] \tag{8}$$

where $x_j = a + jh$ for $j = 0, 1, \ldots, n-1, n$ with $h = (b-a)/n$, in particular, $x_0 = a$ and $x_n = b$.

1.2 Problems

(a) Write a Python code for the computation of the integral $I = \int_0^1 \exp(x)\,dx$ using Simpson's rule. Compare your numerical result with the analytical value for I. Test your integration for $n = 50$ and $n = 100$ steps for the Simpson's rule. This exercise item does not render points but it is an important one to start the exercise and to test the integration routine that will be used in the next exercises.

(b) Write a Python code to compute and plot the Fourier coefficients a_k and b_k for the functions below. You should plot $a_k \times k$ and $b_k \times k$ in the same plot. Calculate a maximum of 10 coefficients, i.e., $k_{\max} = 10$. Note that these functions are chosen so that you can check the performance of your program, as the functions are already in the form of equation (1). Set $\omega = 1$ in the numerical computations. **(1 point)**

- $f(t) = \sin(\omega t)$
- $f(t) = \cos(\omega t) + 3\cos(2\omega t) - 4\cos(3\omega t)$
- $f(t) = \sin(\omega t) + 3\sin(3\omega t) + 5\sin(5\omega t)$
- $f(t) = \sin(\omega t) + 2\cos(3\omega t) + 3\sin(5\omega t)$

(c) Analyse the square wave with period $T = 2\pi/\omega$ and $\omega = 1$. We define $\theta = \omega t$ where the square wave function can be written as: **(0.5 points)**

$$f(\theta) = \begin{cases} 1 & 0 \leq \theta \leq \pi \\ -1 & \pi < \theta \leq 2\pi \end{cases}$$

- Plot the function $f(\theta) \times \theta$, compute its Fourier coefficients and compare your output with the analytic result:

$$\begin{aligned} a_k &= 0 \quad k = 1, 3, 5, \ldots & (9) \\ b_k &= \begin{cases} \dfrac{4}{\pi k} & k = 1, 3, 5, \ldots \\ 0 & k = 2, 4, 6, \ldots \end{cases} & (10) \end{aligned}$$

  To generate the square wave signal, use the function `square` from `scipy.signal` library. Its documentation can be found in `https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.square.html`.

- Plot the reconstructed function, i.e. use equation (1) and the coefficients you just calculated to reconstruct the original signal. Check how your signal is reconstructed using 10, 20, and 30 Fourier coefficients.

(d) Analyse the pulse train with period $T = 2\pi/\omega$ as: **(0.5 points)**

$$f(\omega t) = \begin{cases} 1 & 0 \leq \omega t \leq \omega\tau \\ -1 & \omega\tau < \omega t \leq 2\pi \end{cases}$$

in the same way as in item (c) (including the reconstruction of the signal). The analytic results, setting $\omega\tau = 2\pi/\alpha$ where α is a predefined parameter of your choice that will introduce an asymmetry to your signal (over a period), are:

$$\begin{aligned} a_0 &= \frac{2}{\alpha} - 1 & (11) \\ a_k &= \frac{2}{k\pi} \sin\left(\frac{2k\pi}{\alpha}\right) \qquad k = 1, 2, 3, \ldots & (12) \\ b_k &= \frac{2}{k\pi}\left[1 - \cos\left(\frac{2k\pi}{\alpha}\right)\right] \qquad k = 1, 2, 3, \ldots & (13) \end{aligned}$$

(e) **IMPORTANT:** In order to be considered for the full marks assigned in items (b), (c), and (d) above, discuss the results obtained in each of these items in your report.

(f) **IMPORTANT:** Attach ALL `.ipynb` files you generated for items (b), (c), and (d) above, in addition to your report. **(0.5 points)**

2 Fourier transform for non-periodic functions

A non-periodic function $f(t)$ may be expanded in terms of cosine and sine functions but, in this case, the expansion is a Fourier integral over a continuous range of frequencies, instead of a sum over a discrete set of frequencies. The Fourier integral may be viewed as the limit of a Fourier series (1) in the limit $T \to \infty$. The summation over n in equation (1) is replaced by an integration over ω,

$$f(t) = \int_0^\infty d\omega \, [a(\omega)\cos(\omega t) + b(\omega)\sin(\omega t)] \tag{14}$$

and the equations (5), (6), and (7) are replaced by

$$a(\omega) = \frac{1}{\pi}\int_{-\infty}^{\infty} dt\, f(t)\cos(\omega t) \tag{15}$$

$$b(\omega) = \frac{1}{\pi}\int_{-\infty}^{\infty} dt\, f(t)\sin(\omega t) \tag{16}$$

It is frequently more convenient to use complex notation to express functions that carry cos(...) and sin(...) elements because $e^{i\omega t} = \cos(\omega t) + i\sin(\omega t)$. We can then define

$$f(t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} d\omega\, F(\omega)\, e^{i\omega t} \tag{17}$$

$$F(\omega) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} dt\, f(t)\, e^{-i\omega t} \tag{18}$$

where the function $F(\omega)$ is called the Fourier transform of $f(t)$. If the signal function has the dimensions of energy, for instance, then its Fourier transform has the dimensions of power, and its magnitude $|F(\omega)|$ is a measure of the total power in the signal at frequency ω. $|F(\omega)|$ is given by

$$|F(\omega)| = \sqrt{\mathbb{R}\mathrm{e}\{F(\omega)\}^2 + \mathbb{I}\mathrm{m}\{F(\omega)\}^2} = \sqrt{\frac{\pi}{2}}\sqrt{a^2(\omega) + b^2(\omega)} \tag{19}$$

2.1 The Discrete Fourier Transform - DFT

In practice, the numerical solution of (18) will involve replacing the integration with a discrete summation. The exact integrals are approximated by a Discrete Fourier transform (DFT) as defined below. DFTs are useful for analysing physical amplitude-time or intensity-time data. In such cases, it is not known whether or not the signal is periodic and, even if it is, its period is unknown. Suppose we have a time-dependent physical signal represented by a function $f(t)$ and we sample it N times at intervals h from $t = 0$ to $t = (N-1)h$. We can define a discrete timeline as $t_m = mh$ with $m = 0, 1, 2, \ldots, N-1$. The function is approximated by the discrete set of values at these instants, and time $\tau = Nh$ becomes the period of the approximated function. We need τ to be the longest time over which we are interested in the behaviour of $f(t)$, and we assume

$$f(t) = f(t+\tau) \text{ i.e. } f(t_m) = f(t_{m+N}) \text{ or in shorthand form, } f_m = f_{m+N} \tag{20}$$

In this case, the lowest frequency in the DFT will be $\nu_1 = 1/\tau = 1/(Nh)$. This is the fundamental frequency in the case where the function $f(t)$ is periodic and $\tau = T$. The frequency spectrum is given by

$$\nu_n = \frac{n}{\tau} = n\frac{1}{Nh} = n\nu_1 \text{ for } n = 1, 2, \ldots, N. \tag{21}$$

The DFT evaluates equations (17) and (18) as

$$f_m = \frac{1}{N}\sum_{n=0}^{N-1} F_n\, e^{i\,2\pi\nu_n t_m} = \frac{1}{N}\sum_{n=0}^{N-1} F_n\, e^{i\,2\pi mn/N} \tag{22}$$

$$F_n = \sum_{m=0}^{N-1} f_m\, e^{-i\,2\pi\nu_n t_m} = \sum_{m=0}^{N-1} f_m\, e^{-i\,2\pi mn/N} \tag{23}$$

Note that there exists an orthogonality relation for sums that leads to an identity when inserting f_m from (22) into (23) and reverse. F_n are generally complex numbers. Not all the Fourier components F_n are independent of each other and one can show that $F_{N/2-n} = F^*_{N/2+n}$ where F^*_n is the complex conjugate of F_n. The highest frequency component is thus $F_{N/2-1}$, corresponding to [see equation (21)] a frequency of $\nu_{\max} = (N/2-1)/(Nh) = 1/(2h) - 1/(Nh) \approx 1/(2h)$ for large N. This is called the *Nyquist frequency* expressed as ν_{Nyquist}. If the signal has a component with frequency $\nu > \nu_{\text{Nyquist}}$, there are less than two sample points per period. In this case, there will be one or more frequencies less than ν_{Nyquist} for which the amplitude equals the true amplitude at the sample points, and these lower (incorrect) frequencies will appear in the calculated spectrum. This is known as *aliasing*.

The power spectrum of the DFT [see equation (23)] is given by a plot of all the values $P_n^2 = \mathbb{R}\mathrm{e}\{F_n\}^2 + \mathbb{I}\mathrm{m}\{F_n\}^2$ as a function of n, where $\mathbb{R}\mathrm{e}\{F_n\}$ is the real part of F_n and $\mathbb{I}\mathrm{m}\{F_n\}$ its imaginary part.

2.2 Exercises - DFT

Modify your Python script for the Fourier series exercise to calculate the DFT in the example below. Use the following relations for $\mathbb{R}\mathrm{e}\{F_n\}$ and $\mathbb{I}\mathrm{m}\{F_n\}$:

$$\mathbb{R}\mathrm{e}\{F_n\} = \sum_{m=0}^{N-1} f_m \cos\left(\frac{2\pi mn}{N}\right) \tag{24}$$

$$\mathbb{I}\mathrm{m}\{F_n\} = \sum_{m=0}^{N-1} f_m \sin\left(\frac{2\pi mn}{N}\right) \tag{25}$$

The original signal is then reconstructed via equation (22), resulting in

$$f_m = \frac{1}{N}\sum_{n=0}^{N-1}\left\{\mathbb{R}\mathrm{e}\{F_n\}\cos\left(\frac{2\pi mn}{N}\right) + \mathbb{I}\mathrm{m}\{F_n\}\sin\left(\frac{2\pi mn}{N}\right)\right\} \tag{26}$$

Remember that the effective fundamental frequency, ν_1, to be used in each case is determined by the total sampling time $\tau = T$. It is recommended to make your program evaluate and print out the sampling rate $\nu_s = 1/h$ as well as ν_1 so that you can see how many Fourier components you would expect to be able to reconstruct the signal accurately.

(a) Consider the function $f(t) = \sin(0.45\pi t)$: **(1 point)**

- Set $N = 128$ and $h = 0.1$ and plot the function for a total time $\tau = Nh$, together with the points where it is sampled.
- Write a code that returns the Fourier components $\mathbb{R}\mathrm{e}\{F_n\}$ and $\mathbb{I}\mathrm{m}\{F_n\}$ using equations (24) and (25) and plot them as a function of n. Plot in a separate figure the power spectrum of the signal.
- You should find one dominant Fourier component $\nu_n = n/(Nh)$. How does its value compare with what you expect for the frequency from the given function $f(t)$ above?
- From the Fourier components $\mathbb{R}\mathrm{e}\{F_n\}$ and $\mathbb{I}\mathrm{m}\{F_n\}$, reconstruct the initial function $f(t)$ and plot it as a function of time $t_m = mh$. Include $f(t)$ calculated directly from the $\sin(0.45\pi t)$ function in this figure to see how well the reconstruction works.
- Repeat the four steps above for the function $f(t) = \cos(6\pi t)$. Start with $N = 32$ and $h = 0.6$. In the following, start decreasing the time spacing as $h = 0.5$, $h = 0.4$, and $h = 0.1$ and keeping $N = 32$ fixed. What is happening with the reconstruction of your signal? Is it improving the reconstruction?

(b) *Radio and TV Transmission*: Radio, television, and some other forms of communication transmit information via electromagnetic waves. The various sources in these applications can be transmitting simultaneously and in the same geographic region. But how is it that we can "tune in" to a specific radio station, or television program for example, rather than hearing the jumble of all the various transmissions put together? Luckily different transmissions operate at different frequencies. Thus, even though all the signals are "jumbled" together in the time domain, they are distinct in the frequency domain. With some basic frequency domain processing based on Fourier analysis, one can separate the signals and "tune in" to the frequency we wish.

You will now process a real sound data stored in the file `pitch.txt`. Modify your code to read the content of the file which is in one-column format representing vibrations in the air over time in seconds. Obtain the Fourier components and plot them as a function of n or as a function of frequency (in Hz). Plot in a separate figure the power spectrum of the signal. Using the Fourier components, reconstruct the original signal. Which frequencies can you identify in your power spectrum? Can you 'denoise' the signal? To do that, reconstruct the signal using equation (26) but impose a threshold of $\epsilon = 50$ in which $F_n = 0$ only if $|F_n| < \epsilon$. **(1 point)**

(c) **IMPORTANT:** In order to be considered for the full marks assigned in items (a), and (b) above, discuss the results obtained in each of these items in your report.

(d) **IMPORTANT:** Attach ALL `.ipynb` files you generated for items (a), and (b) above, in addition to your report. **(0.5 points)**

3 Report writing guide (5 points)

After generating your codes and results for the exercises above, you will summarize them in the form of a written report. Please, follow the instructions provided in the report template located in the sub-module of the assignment to write your report. All margins, font style/size, line spacing, and page limits set on the template need to be respected. **We will deduct a significant amount of points from reports that do not align with the template provided or we will not be able to assess the report, meaning we will have to deduct -5/10 points from the total marks.**

There is a `docx` file (generated in MS Word editor) available in the sub-module of the assignment that can, in principle, be used and exported to other text editors. However, some of its pre-defined macros may not be read correctly by other editors. Moreover, inserting text on the editable `docx` file may shift or duplicate some of the original headers. Make sure your written content obeys the format set in the read-only `PHYS 381 Assignment 4 Template.pdf` file. In case you have problems with the `docx` file provided, a suggestion is to start a blank document in the text editor of your choice and set the layout as below:

- Page size: 8.5" $\times$ 11" (letter);
- Margins: 2.54 cm (top, bottom, left, right);
- Font type and size: Times New Roman, 12 pt, non-italic, non-bold;
- Line spacing: between 1.0 and 1.15 pt;
- Figures must be numbered and contain captions. Axes must carry names and their quantities must be expressed with the proper units whenever required. Do not insert too large figures just to cover space. Choose figure sizes typically seen in textbooks and scientific manuscripts.
- Follow the exact page configuration and order as described in the report template, e.g., title page on page 1, abstract on page 2, introduction on page 3, etc. The title page needs to contain the exact same information as in the report template.
- **NOTE FOR THIS ASSIGNMENT ONLY:** Points will not be deducted if the section 'Results and Analysis' go over the page limit set in the report template. But this is valid ONLY for the 'Results and Analysis' section; all other sections need to follow the template requirements.

3.1 Submission

The due date for the submission of your report is depicted on the first page of the assignment. Once you conclude the writing of your report in accordance with the template provided, generate a `pdf` file and name it as `report_assignment_4.pdf` (since this is report #4). You will upload the `pdf` file of your report plus your Notebook `.ipynb` files requested in the assignment to the Gradescope platform. **It is important that you log in to Gradescope.ca and not Gradescope.com.** You can upload multiple files to the Gradescope platform and you can resubmit your work until the due date. We will test-run all your submitted codes to test for errors. We will also check if the results/figures presented in your report match the output of your codes.

For Groups: If you worked in a group, remember to include your group members in Gradescope after you submit all the files on behalf of the group. You will see an option "Add Group Member" after you upload all files to Gradescope. Students can work in groups of up to 4 members.

Autograder option: Ignore the autograder results that may appear after you upload files to Gradescope. There is no autograder configured for this assignment since everything will be evaluated manually by the PHYS 381 Team.

Please, remember to add comments in all your submitted Notebook codes! Pure code lines without explanatory comments or written Markdown cells will have reduced marks.

Standard `.py` Python codes are not accepted in this submission. Work only with Notebook environments. If any member of the group is experiencing problems with their coding environment, please, contact the instructor immediately.

ONLY `pdf` (for the written report) AND `.ipynb` (for codes) FILES ARE ACCEPTED.

*****