Lab 4: Fourier Analysis using Python

This laboratory introduces the following

- Fourier analysis of both periodic and non-periodic signals
- Fourier series, Fourier transforms, discrete Fourier transforms
- The use of Simpson's rule for numerical integration.

Fourier analysis is an important tool in the analysis of signals - it decomposes a signal into constituent harmonic vibrations – i.e. sines and cosines (such as the saw-tooth profile in Fig.1). Sometimes signals are periodic, but more often they are not. A middle C on a piano is a periodic signal containing a fundamental (261.6 Hz) and multiple higher harmonics. On the other hand, sound waves generated in speech are aperiodic signals. Fourier analysis is used for analysing signals which are either periodic or aperiodic. In the former case a Fourier series is used and in the latter a Fourier transform is used. A Fourier transform is essentially a Fourier series for a function with infinite period. Both require integrations of products of sines and cosines times the function to be transformed over the period or full duration of the signal. Numerical integration is done by sampling a function at regular intervals. Sampling a signal in this way has implications for the fidelity of the Fourier transform, which you will investigate later. A numerical Fourier transform which uses regular (discrete) sampling of a function is called a discrete Fourier transform.

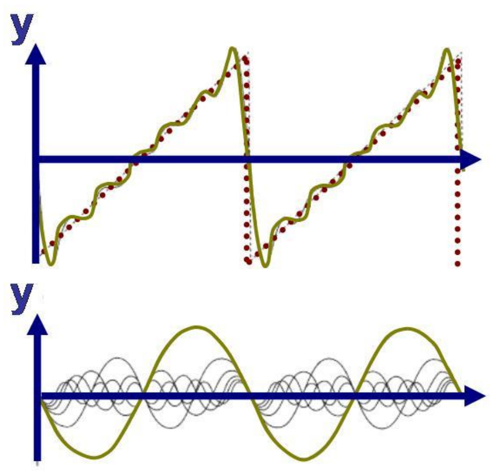


Fig.1 Periodic saw-tooth profile approximated as a Fourier series (https://commons.wikimedia.org/wiki/File:Sawtooth_Fourier_Analysis.JPG)

In the first part of this practical you will deal with periodic signals, while in the second part you will see that the numerical method applies with very little modification also to the case of non-periodic signals when it is known as the discrete Fourier transform, or DFT.

Fourier Series of Periodic Functions

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 series contains a possibly infinite set of so-called harmonic functions with discrete frequencies $\omega_n = n\omega$. n is a positive integer. The frequency $\omega_1 = \omega$ is known as the fundamental frequency, and ω_n for n >1, are harmonics. Coefficients a_n and b_n measure the 'amount' of cos(n ωt) and sin(n ωt) present in the function f(t). The result of Fourier analysing a signal is a set of values for these coefficients for all n. Obviously in practice the coefficients will be obtained for all n up to some finite maximum, N. A faithful reconstruction of the original function requires all sines and cosines with non-zero a or b coefficients.

Fourier coefficients are evaluated using the following orthogonality properties of sines and cosines with period $T = 2\pi/\omega$.

$$\frac{2}{T}\int_0^T dt \sin(n\omega\, t)\sin(k\omega\, t) = \delta_{nk} = \begin{cases} 0 & n \neq k \\ 1 & n = k \end{cases} \tag{2a}$$

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

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

If we integrate each of the functions f(t), cos(n ωt) * f(t), and sin(n ωt) * f(t) over one period, and use equations (1) and (2a)-(2c), we find

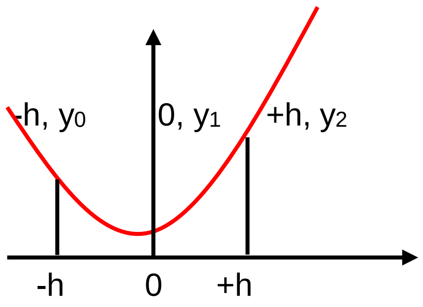
$$a_0 = \frac{1}{T}\int_0^T dt\ \ f(t) \tag{3a}$$

$$a_k = \frac{2}{T}\int_0^T dt\ \ f(t)\cos(k\omega\, t) \qquad \text{k=1,2, ….} \tag{3b}$$

$$b_k = \frac{2}{T}\int_0^T dt\ \ f(t)\sin(k\omega\, t) \qquad \text{k=1,2, ….} \tag{3c}$$

Simpson's Rule

A Fourier analysis program must perform the integrations shown in equations (3), for any function f(t) of interest. Simpson's rule will be used for this. Simpson's rule uses parabolas to approximate the area under a curve which is more accurate than the Trapezoidal rule that we previously looked at.



The integral of a general parabolic function from –h to h is

$$\int_{-h}^{+h} (a\,x^2 + b\,x + c)\,dx = \frac{h}{3}(2a\,h^2 + 6c)$$

The values of the function at x = -h, 0 and +h are

$$y_0 = a\,h^2 + b\,h + c \qquad y_1 = c \qquad y_2 = a\,h^2 - b\,h + c$$

From these equations we find that $2a\,h^2 = y_0 - 2\,y_1 + y_2$ and $c = y_1$. Hence the area, A_0, under the curve from –h to +h is

$$A_0 = \frac{h}{3}(2a\,h^2 + 6c) = \frac{h}{3}(y_0 + 4\,y_1 + y_2)$$

The area, A_1, under an adjacent interval from +h to +3h is

$$A_1 = \frac{h}{3}(y_2 + 4\,y_3 + y_4)$$

$$A_0 + A_1 = \frac{h}{3}(y_0 + 4\,y_1 + 2\,y_2 + 4\,y_3 + y_4)$$

Extending this to a larger number of intervals of length, 2h, the rule to obtain the area under a series of parabolic fits to each interval along the curve is 'h/3 times the sum of the two end points $(y_0 + y_4)$ + 4 times odd values $4\,(y_1 + y_3)$ + 2 times even values, $2\,y_2$. [Adapted from www.intmath.com].

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

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

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

Exercise 1: Simpson's rule Python script for evaluating Fourier series

(1) Write a Python script to integrate a function between two limits, a and b, with an even number of steps, n, using Simpson's rule. Define the function to be integrated in a separate Python function for convenience later.

(2) Use your script evaluate the integral below.

$$\int_0^1 e^x \, dx$$

n = 8 steps should be sufficient for this problem, but some of the tasks below may require a greater number of steps.

(3) Compare your numerical result with the analytical value of the integral. If the agreement is not excellent then your script is not functioning properly.

(4) Modify your Python script to compute the Fourier series for any input function. Your Simpson's rule integral evaluator from (1) can be extended so that it evaluates the integrals in Eqns. (3a) to (3c) for a range of values of k to obtain coefficients a_k and b_k. Define the function to be Fourier analysed in a separate Python function.

(5) Add lines to your script which plot the function to be analysed as well as the Fourier series coefficients as discrete points.

(6) Use your script to obtain the Fourier series for the following functions with fundamental frequency $\omega = 1$. These functions are chosen so that you can check the validity of your programme.

$f(t) = \sin \omega t$

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

$f(t) = \sin \omega t + 3 \sin 3\omega t + 5 \sin 5\omega t$

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

(7) Explain why you expect to obtain $b_1 = 1$, for the first function, $a_1 = 1$, $a_2 = 3$, $a_3 = -4$ for the second function, etc.

Exercise 2: Python script for Fourier series of square and rectangular waves

(1) Insert the function below into your Python script from Exercise 1. This is a square wave signal with period $T = 2\pi / \omega$, where $\theta = \omega t$. You will need to use an "if" statement to generate the function as there is no step function defined in Python.

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

(2) Compute the Fourier coefficients and compare your output with the analytic result

$$a_k = 0 \qquad k = 1,3,5\ldots$$

$$b_k = \begin{cases} 4/\pi k & k = 1,3,5\ldots \\ 0 & k = 2,4,6\ldots \end{cases}$$

(3) Add a section to your script which reconstructs the original function using equation (1).

(4) Plot the reconstructed square wave function using limited sets of a and b coefficients in equation (1). Use the first 1, 2, 3, 5, 10, 20 and 30 terms in the sum to see how the square wave emerges as the number of terms in the reconstruction increases.

(5) Make plots which compare the partly reconstructed square wave to the exact square wave and include them in your report.

(6) Modify your Python script for the square wave so that it analyses a rectangular wave. The function in this case is

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

The exact Fourier coefficients obtained by evaluating the integrals in equations (3a) and (3b) analytically (with $\omega\tau = 2\pi / \alpha$) are

$$a_0 = (2/\alpha) - 1$$

$$a_k = (2/k\pi) \sin(2k\pi/\alpha) \qquad k = 1,2,3\ldots$$

$$b_k = (2/k\pi)(1 - \cos(2k\pi/\alpha)) \qquad k = 1,2,3\ldots$$

(7) Repeat (4) and (5) for the rectangular wave.

Fourier Transform for Non-Periodic Functions

A non-periodic function, f(t), may also be expanded in terms of cosine and sine functions. In this case, however, the expansion is a Fourier integral over a **continuous** range of frequencies ω, instead of a sum over a **discrete** set of frequencies k ω, i.e. an integer multiple of a fundamental frequency, ω. A Fourier integral may be viewed as the limit of a Fourier series as the period $T\rightarrow\infty$. As the period, $T = 2\pi / \omega$, increases, the fundamental frequency, $\omega = 2\pi / T$, decreases. In the limit that $T \rightarrow \infty$ the fundamental frequency becomes smaller and smaller so that the separation between discrete frequencies eventually vanishes. In this way, a sum over discrete frequencies becomes an integral over a continuous frequency variable.

The sum over n in equation (1) is replaced by an integral over ω,

$$f(t) = \int_0^\infty d\omega \{a(\omega)\cos\omega t + b(\omega)\sin\omega t\} \qquad (5)$$

Equations (3a) - (3c) are replaced by

$$a(\omega) = \frac{1}{\pi}\int_{-\infty}^{+\infty} dt\, f(t)\cos\omega t \qquad (6a)$$

$$b(\omega) = \frac{1}{\pi}\int_{-\infty}^{+\infty} dt\, f(t)\sin\omega t \qquad (6b)$$

It is frequently more convenient to use complex notation, remembering that

$$e^{i\omega t} = \cos(\omega t) + i\sin(\omega t)$$

and defining

$$F(\omega) = \int_{-\infty}^{+\infty} dt\, f(t)e^{-i\omega t} \qquad (7)$$

$$f(t) = \int_{-\infty}^{+\infty} \frac{d\omega}{2\pi} F(\omega)e^{+i\omega t} \qquad (8)$$

F(ω) in equation (7) is the Fourier transform of f(t) and equation (8) is referred to as the back-transform. The factor outside the integral in the definition of the Fourier transform is a matter of convention. Changing that factor simply means that the factor in front of the integral for the back transform must be changed accordingly to yield the original function again. The factor $1/2\pi$ in equation (8) is one particular convention. In some texts $F(\omega)$ is redefined so that both equations have a factor of $1/\sqrt{(2\pi)}$. In equations (6a) and (6b) the factor $1/\pi$ means that the Fourier coefficients used in reconstructing the function look as much like the Fourier series in equation (1) as possible.

In engineering applications the energy E of a time-dependent signal x(t) is often written as $E = \int_{-\infty}^{+\infty} |x(t)|^2 dt$. Here $|x(t)|^2$ is called the energy spectral density, i.e. the energy per time interval, with dimensions of a power. The energy contained per frequency interval, $|F(\omega)|^2$, is also called the power spectrum.

Note that

$$|F(\omega)| = \sqrt{\Re e(F(\omega)^2) + \Im m(F(\omega)^2)} = \sqrt{\frac{\pi}{2}}\sqrt{a(\omega)^2 + b(\omega)^2} \qquad (9)$$

The Discrete Fourier Transform

In practice, numerical solutions of equations (6a), (6b) and (8), replace integration with respect to a continuous variable by discrete sampling (numerical quadrature), as does numerical solution of equations (5) and (7). Exact integrals are replaced 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 a signal is periodic, and even if it is, its period is unknown.

Suppose we have a time-dependent signal represented by a function, f(t), and we sample it N times at intervals, h, from t = 0 to t = (N - 1) * h. (It is usual to use t = 0 as the lower limit when making measurements). Define

$$t_m = m * h \qquad m = 0, 1, 2, \ldots, N - 1 \qquad (10)$$

The function is approximated by the discrete set of values at these instants, and time $\tau = N * h$ becomes the period of the function. We need τ to be the longest time over which we are interested in the behaviour of f(t), and we assume

$f(t + \tau) = f(t)$, i.e. $f(t_m) = f(t_{m+N})$ or in short form, $f_m = f_{m+N}$. (11)

In this case the lowest frequency in the DFT is $\omega_1 = 2\pi / \tau = 2\pi / (N * h)$. This is the fundamental frequency in the case where the function f(t) is periodic and $\tau \equiv T$. The frequency spectrum is the discrete spectrum

$$\omega_n = 2\pi n / (N * h) = n\omega_1, \text{ for } n = 1, 2, \ldots, N. \qquad (12)$$

The DFT evaluates equations (8) and (7) as

$$F_n = \sum_{m=0}^{N-1} f_m e^{-i\omega_n t_m} = \sum_{m=0}^{N-1} f_m e^{\frac{-2\pi imn}{N}} \qquad (13a)$$

and

$$f_m = \frac{1}{N}\sum_{n=0}^{N-1} F_n\, e^{+i\omega_n t_m} = \frac{1}{N}\sum_{n=0}^{N-1} F_n\, e^{\frac{+2\pi imn}{N}} \tag{13b}$$

Note that there is an orthogonality relation for sums of complex exponentials that leads to an identity when inserting f_m from equation (13b) into (13a) and vice versa. Coefficients F_n are generally complex numbers even when f is a real function.

Not all the Fourier components F_n are independent of each other and one can show that $F_{N/2-n} = F^*_{N/2+n}$ (or $F_{N-n} = F^*_n$) where F^*_n is the complex conjugate of F_n. The highest frequency component is thus $F_{N/2-1}$, corresponding to (equation (12)) a frequency

$\nu_n = (N / 2 - 1) / (N * h) = 1 / (2 * h) – 1 / (N * h) \approx 1 / 2h$ for large N.

This is called the Nyquist frequency. If the signal has a component with frequency greater than the Nyquist frequency, there are fewer than two sample points per period. In this case there will be one or more frequencies below the Nyquist frequency 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 (equation (9)) of the DFT is given by a plot of all the values $P_n = F_{n,real}^2 + F_{n,imaginary}^2$ as a function of n, where $F_{n,real}$ is the real part of F_n and $F_{n,imaginary}$ its imaginary part.

$$F_{n,\,real} = \sum_{m=0}^{N-1} f_m \cos\left(\frac{2\pi mn}{N}\right) \tag{14a}$$

$$F_{n,\,imag} = \sum_{m=0}^{N-1} -f_m \sin\left(\frac{2\pi mn}{N}\right) \tag{14b}$$

$$f_m = \frac{1}{N}\sum_{n=0}^{N-1} F_n \left(\cos\left(\frac{2\pi mn}{N}\right) + i\sin\left(\frac{2\pi mn}{N}\right)\right) \tag{15}$$

The effective fundamental frequency to be used in each case is determined by the total sampling time τ.

Exercise 3: Discrete Fourier Transform Python Script

(1) Modify your Python script for the Fourier series to calculate the DFT using the relations above for $F_{n,real}$ and $F_{n,imaginary}$. Note that Python uses the engineering convention for the square root of minus one, j, i.e. F_n = 3.0 + 4.0j defines F_n as a

complex number (3.0, 4.0j). You will find that using lists of data useful for storing Fourier components, time values, etc. Declare required lists at the top of your script where other variables, such as h, are initialised using

```
my_list=[]
```

Add values to the list using

```
my_list.append(value)
```

Plot using

```
matplotlib.pyplot.plot(my_list)
```

(2) Make your program evaluate and print out the sampling rate ν_s (samples per unit time) as well as the fundamental frequency, ω_1, so that you can see how many Fourier components you would expect to be able to calculate accurately.

(3) For the function $f(t) = \sin(0.45\pi t)$, set N = 128 and h = 0.1 and plot the function for a total time $\tau = N * h$, together with the points where it is sampled as symbols.

(4) Add code that plots Fourier components $F_{n,real}$ and $F_{n,imaginary}$ as a function of n.

(5) In your plot of the real and imaginary parts of the Fourier components you should find one dominant component at some integer value of n. How does its value compare with what you expect for the frequency from the given function f(t) above? (Equate $0.45\pi t$ to $2\pi nm/N$ and find the value of n. NB $t_m = m * h$).

(6) An ideal sampling time for a periodic signal is a multiple of the period. Use this criterion to pick an ideal value for h, keeping N = 128 fixed. Perform the Fourier calculation again with this ideal sampling time. How do the Fourier components differ in this case? What is the ideal sampling interval, h, for $\omega_1 = 0.45\pi$ when N = 128?

(7) Compute the components of the Fourier back-transform, $f_{m,real}$ and $f_{m,imaginary}$. Plot these as a function of time $t_m = m * h$ to reconstruct the initial function f(t). Compare f(t) with the reconstructed function in your report.

(8) Note that $F_{N-n} = F^*_n$ (Nyquist 'symmetry'). Use the definition of the Fourier transform in equations (14a) and (14b) to prove this and hence that you should use the following equation to reconstruct the real part of the original function.

$$f_{m,real} = \frac{1}{N}\sum_{n=0}^{N-1}\left(\mathrm{Re}[F_n]\cos(\frac{2\pi mn}{N}) - \mathrm{Im}[F_n]\sin(\frac{2\pi mn}{N})\right)$$

(9) Change the function in your DFT script to $f(t) = \cos(6\pi t)$. Treat it as a function of unknown period, and sample it every second for time t = N * h (N = 32, h = 0.6).

(10) Note the sampling frequency, and plot the Fourier coefficients (real and imaginary parts of F_n), power spectrum and reconstructed function.

(11) Repeat (10) with h = 0.2, h = 0.1, and h = 0.04, keeping N = 32 fixed.

(12) Compare the sampling rate to the Nyquist frequency in each case and explain your results.

Supplementary Exercise: DFT of damped sine wave and finite square pulse

(1) Change the function in your DFT script to f(t) = cos(3 t) and sample it every second for 8 seconds (h = 1, N * h = 8).

(2) Plot the coefficients and the power spectrum as a function of frequency.

(3) Repeat the calculation, sampling the function every second for a total sampling time τ of 16, 32 and 64 seconds.

(4) Compare the spectra obtained. What went wrong in the first calculation? (This is known as leakage - the amplitude leaks into nearby frequency bins.)

(5) Evaluate the DFT for the damped sine wave $f(t) = \sin t\, e^{-kt}$. For a suitable value of k (e.g. 0.2) choose τ long enough for the signal to have decreased to at least 1 / e of its initial value.

(6) Make a plot of the function f(t) which also shows the sampled values.

(7) Plot real and imaginary parts of F_n, and the power spectrum as functions of n.

(8) Reconstruct the function from the DFT and note how good the approximation is.

(9) Evaluate the DFT for a single pulse.

$$f(t) = 1 \quad 0 < t < \tau$$
$$f(t) = 0 \quad t < 0,\ t > \tau$$

(10) Compare the power spectrum with that obtained in (5) of Exercise 2. Note that increasing N for a fixed total time interval *τ* (i.e. decreasing h and increasing N so that N * h is unchanged) increases the number of lines in the frequency spectrum, so that in the limit $N \to \infty$, $h \to 0$, the spectrum would become continuous.