TRINITY COLLEGE DUBLIN, SCHOOL OF MATHEMATICS

Assignment #2, Module: MA5633

Gustavo Ramirez

November 23, 2016

1 Analytical calculation of Fourier series expansion

In the first part of the assignment, the function give to calculate it's expansion (in multiple ways) is: $f(x) = 1 - x^4$, over the interval $[-1, 1]$.
The analytical calculation for the infinite series is done in the following lines:

$$s_N = \frac{a_0}{2} + \sum_{n=1}^{N} (a_n \cos(2\pi x n/P) + b_n \sin(2\pi x n/P))$$

where:

$$a_n = \frac{2}{P} \int_{x_0}^{x_0+P} f(x) \cos(2\pi x n/P) dx$$

$$b_n = \frac{2}{P} \int_{x_0}^{x_0+P} f(x) \sin(2\pi x n/P) dx$$

and in the case of the given function:

$$b_n = \frac{2}{2} \int_{-1}^{1} (1 - x^4) \sin(2\pi x n/2) dx = 0$$

$$a_n = \frac{2}{2} \int_{-1}^{1} (1 - x^4) \cos(2\pi x n/2) dx = \int_{-1}^{1} (1 - x^4) \cos(\pi x n) dx \Rightarrow$$

$$\frac{a_n}{2} = \int_0^1 \cos(\pi x n) dx - \int_0^1 x^4 \cos(\pi x n) dx = \left(\frac{\sin(\pi x n)}{\pi n}\right)_0^1 - I_1$$

$$I_1 = \left(\frac{x^4 \sin(\pi x n)}{\pi n}\right)_0^1 - \frac{4}{\pi n} I_2$$

$$I_2 = -\left(\frac{x^3 \cos(\pi x n)}{\pi n}\right)_0^1 + \frac{3}{\pi n} I_3$$

$$I_3 = \left(\frac{x^2 \sin(\pi x n)}{\pi n}\right)_0^1 - \frac{2}{\pi n} I_4$$

$$I_4 = -\left(\frac{x \cos(\pi x n)}{\pi n}\right)_0^1 + \frac{1}{\pi n} \left(\int_0^1 \cos(\pi x n)\right) = -\left(\frac{x \cos(\pi x n)}{\pi n}\right)_0^1 + \frac{1}{\pi n} \left(\frac{\sin(\pi x n)}{\pi n}\right)_0^1 = -\frac{1}{\pi n} \cos(\pi n)$$

$$\Rightarrow I_3 = \frac{1}{\pi n} \sin(\pi n) + \frac{2}{(\pi n)^2} \cos(\pi n)$$

$$I_2 = -\frac{1}{\pi n} \cos(\pi n) + \frac{3}{(\pi n)^2} \sin(\pi n) + \frac{6}{(\pi n)^3} \cos(\pi n)$$

$$I_1 = \frac{1}{\pi n} \sin(\pi n) + \frac{4}{(\pi n)^2} \cos(\pi n) - \frac{12}{(\pi n)^3} \sin(\pi n) - \frac{24}{(\pi n)^4} \cos(\pi n)$$

$$\frac{a_n}{2} = \frac{1}{\pi n} \sin(\pi n) - \frac{1}{\pi n} \sin(\pi n) - \frac{4}{(\pi n)^2} \cos(\pi n) + \frac{12}{(\pi n)^3} \sin(\pi n) + \frac{24}{(\pi n)^4} \cos(\pi n)$$

$$= -\frac{4}{(\pi n)^2} \cos(\pi n) + \frac{24}{(\pi n)^4} \cos(\pi n)$$

From the way the Fourier series coefficients are expressed, n takes both odd and even values. For odd n:

$$\frac{a_n}{2} = -\left(-\frac{4}{(\pi n)^2} + \frac{24}{(\pi n)^4}\right)$$

and for even n:

$$\frac{a_n}{2} = \left(-\frac{4}{(\pi n)^2} + \frac{24}{(\pi n)^4}\right)$$

and then:

$$a_n = \frac{8}{(\pi n)^2}\left(-1 + \frac{6}{(\pi n)^2}\right)(-1)^n$$

Also, in particular:

$$a_0 = \int_{-1}^{1}(1-x^4)dx = 2 - \frac{2}{5} = \frac{3}{5}$$

So, in total, the series expansion takes the form:

$$s_N(x) = \frac{3}{10} + \sum_{n=1}^{N}\left(\frac{8}{(\pi n)^2}\left(-1 + \frac{6}{(\pi n)^2}\right)(-1)^n \cos(\pi x n)\right)$$

2 General specification of even finite Fourier series

First, let's take this form of Fourier transform (for even functions), and let's discretize it explicitly:

$$f(\theta) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos(n\theta)$$

$$\Rightarrow f(\theta) \approx \frac{a_0}{2} + \sum_{n=1}^{N-1} a_n \cos(n\theta) + \frac{a_N}{2} cos(N\theta)$$

and now, discretizing the Fourier coefficients (the interval from 0 to π is split into intervals of length $\frac{\pi}{m}$, which leads to the substitution $\theta \to \frac{\pi}{m} j$):

$$a_k = \frac{2}{\pi}\int_0^{\pi} f(\theta)\cos(k\theta)d\theta$$

$$\Rightarrow a_k \approx \frac{2}{\pi}\sum_{j=0}^{m-1} f\left(\frac{\pi}{m} j\right)\cos\left(\frac{\pi}{m} jk\right)\frac{\pi}{m}$$

$$\Rightarrow a_k \approx \frac{2}{m}\sum_{j=0}^{m-1} f\left(\frac{\pi}{m} j\right)\cos\left(\frac{\pi}{m} jk\right)$$

Then, the analogue to s_N (but in the case of a simple numerical integral expansion) would take, in this case, the form:

$$s_N(\theta) = \frac{a_0}{2} + \sum_{n=1}^{N-1} a_n \cos(n\theta) + \frac{a_N}{2}\cos(N\theta)$$

where:

$$a_k = \frac{2}{m}\sum_{j=0}^{m-1}\left(f\left(\frac{\pi}{m}j\right)\cos\left(\frac{\pi}{m}jk\right)\right)$$

and where m defines the order of the expansion, due to the relation:

$$N = [m/2]$$

3 On program output (Fourier calculations)

After implementation and execution of a Python script to numerically calculate the three types of forms for $f(x) = 1 - x^4$ (direct numerical implementation, finite Fourier series and infinite Fourier series), an output such as this is obtained:

First part: approximating a function by a Fourier Series expansion:

–**EVAL. POINT: 1.0

–Value from original function: 0.0

–Finite Fourier approx: 0.509162775247

–Infinite Fourier approx: -0.846004651234

–**EVAL. POINT: 0.5

–Value from original function: 0.9375

–Finite Fourier approx: 1.32787356839

–Infinite Fourier approx: 0.88151174379

–**EVAL. POINT: 0.333333333333

–Value from original function: 0.987654320988

–Finite Fourier approx: 0.983528210584

–Infinite Fourier approx: 0.974860571657

–**EVAL. POINT: 0.25

–Value from original function: 0.99609375

–Finite Fourier approx: 0.939494665181

–Infinite Fourier approx: 0.995468974791

–**EVAL. POINT: 0.2

–Value from original function: 0.9984

–Finite Fourier approx: 1.0330407653

–Infinite Fourier approx: 0.989617829925

From which, is very clear that the truncated infinite Fourier series is better than the finite Fourier series, because by construction the latter is a discretization of the former.

4 On finding roots of functions

Functions for this part and the previous, were defined in external files, and then imported from the main program program.py.
The slowest algorithm for finding the root of a function, in terms of number of global steps, was bisection, followed by secant and finally Newton's method was the fastest.

5 On R_n

As can be seen from the output of the execution of the Python script program.py:

R_0(p=2.05000)= 0.65048595

R_0(p=2.02500)= 0.64455289

R_0(p=2.01667)= 0.64258725

R_0(p=2.01250)= 0.64160668

R_0(p=2.01000)= 0.64101906

R_0(p=2)= 0.63867394

R_1(p=2.05000)= 0.48008942

R_1(p=2.02500)= 0.46611748

R_1(p=2.01667)= 0.46155112

R_1(p=2.01250)= 0.45928474

R_1(p=2.01000)= 0.45793026

R_1(p=2)= 0.45255216

R_2(p=2.05000)= 0.57722715

R_2(p=2.02500)= 0.53343925

R_2(p=2.01667)= 0.51959430

R_2(p=2.01250)= 0.51280716

R_2(p=2.01000)= 0.50877751

R_2(p=2)= 0.49297307

R_3(p=2.05000)= 0.70998044

R_3(p=2.02500)= 0.59572168

R_3(p=2.01667)= 0.56187850

R_3(p=2.01250)= 0.54568488

R_3(p=2.01000)= 0.53619360

R_3(p=2)= 0.49985085

R_4(p=2.05000)= 1.04378884

R_4(p=2.02500)= 0.72223363

R_4(p=2.01667)= 0.63880169

R_4(p=2.01250)= 0.60077276

R_4(p=2.01000)= 0.57905090

R_4(p=2)= 0.49973845

---

the closer p gets to 2, the more convergent becomes $R_n(p)$, which means that, indeed, the sequence converges with order 2 to the root value.