Indian Institute of Technology, Madras
Department of Electrical Engineering
Applied Programming Lab

Lab Report
Assignment 4

Fourier Approximations

Nithin Uppalapati
EE18B035

12-02-2020

Contents

1. Abstract

Aim :
The aim of this assignment is to accurately fit the two functions, e^t and $cos(cos(t))$ by using the first 25 coefficients of Fourier series, and comparing it with the function which is determined by the lstsq method.

Implementation :
So, in order to fit the functions, we take 401 samples of data, of the both functions and try to find the coefficients by lstsq function, imported from scipy.linalg() library.

2. Introduction

Assignment 4 is based on

- Calculating the Fourier coefficients, by using quad function
- Comparing the Fourier coefficients with the predicted coefficients, and
- Plotting graphs using the pylab library for both of the function, in loglog scale and semilogy scale.

3. Theory

We compute the Fourier coefficients, as shown below.
Where the coefficients are given as: $a_n, b_n,\ n\epsilon(0, 25)$
where,

$$\begin{aligned} a_0 &= \tfrac{1}{2\pi}\int_0^{2\pi} f(x)dx \\ a_n &= \tfrac{1}{\pi}\int_0^{2\pi} f(x)cos(nx)dx \\ b_n &= \tfrac{1}{\pi}\int_0^{2\pi} f(x)sin(nx)dx \end{aligned}$$

and compare them to the coefficients, which are computed by the 400 smaples (in the range of $(0, 2\pi)$) which are defined as follows

```
x=linspace(0,2*pi,401)
```

3.1 Defining the functions:

As, we need to define the functions, in order to compute the numeric integrals, and to get the samples which are to be sent to the $lstsq()$ function, in order to predict accurately the coefficients.

```
def f_1(t): return exp(t)
def f_2(t): return cos(cos(t))
def u_1(x,k): return cos(k*x)*f_1(x)
def u_2(x,k): return cos(k*x)*f_2(x)
def v_1(x,k): return sin(k*x)*f_1(x)
def v_2(x,k): return sin(k*x)*f_2(x)
```

where,

$$\begin{aligned} f_1(t) = e^t \quad &and \quad f_2(t) = cos(cos(t)) \\ u_1(x,k) = cos(kx)e^t \quad &and \quad u_2(x,k) = cos(kx)cos(cos(t)) \\ v_1(x,k) = sin(kx)e^t \quad &and \quad v_2(x,k) = sin(kx)cos(cos(t)) \end{aligned}$$

3.2 Computing the Fourier coefficients

The coefficients are computed theoretically as follows:

$$f(x) = a_n cos(x) + b_n sin(x)$$

The coefficients are given as: $a_n, b_n,\ n\epsilon(0, 25)$
where,

$$a_0 = \frac{1}{2\pi}\int_0^{2\pi} f(x)dx$$
$$a_n = \frac{1}{\pi}\int_0^{2\pi} f(x)cos(nx)dx$$
$$b_n = \frac{1}{\pi}\int_0^{2\pi} f(x)sin(nx)dx$$

Whereas, by using the built in integrator in Python, we can implement is as follows:

```
        coeff_1=zeros(51)
        coeff_2=zeros(51)

for i in range(0,26):
  if i:
    coeff_1[2*i-1]=(quad(u_1,0,2*pi,args=(i))[0])*(1/pi)
    coeff_2[2*i-1]=(quad(u_2,0,2*pi,args=(i))[0])*(1/pi)
    coeff_1[2*i]=(quad(v_1,0,2*pi,args=(i))[0])*(1/pi)
    coeff_2[2*i]=(quad(v_2,0,2*pi,args=(i))[0])*(1/pi)
  else:
    coeff_1[0]=(quad(u_1,0,2*pi,args=(0))[0])*(0.5/pi)
    coeff_2[0]=(quad(u_2,0,2*pi,args=(0))[0])*(0.5/pi)
```

Where, the coefficients are stored in the fashion,

$$coeff = \begin{bmatrix} a_0 \\ a_1 \\ b_1 \\ a_2 \\ \vdots \\ b_{24} \\ a_{25} \\ b_{25} \end{bmatrix} \tag{3.1}$$

3.3 Least Squares Approach to find the Fourier Coefficients:

We choose 401 samples from the actual function and try to fit it by determining the coefficients by lstsq method, which is done as follows

$$a_0 + \sum_{n=1}^{25} a_n cos(nx_i) + \sum_{n=1}^{25} b_n sin(nx_i) \approx f(x_i)$$

and x_i are given as:

```
x=linspace(0,2*pi,401)
```

and the coefficients are predicted as follows:
The matrix can be constructed using zeros((400,51)) and then filling in the columns in a for loop.

```
A=zeros((400,51))
A[:,0]=1
x=x[:-1]
b_1=f_1(x)
b_2=f_2(x)
for k in range(1,26):
      A[:,2*k-1]=cos(k*x)
      A[:,2*k]=sin(k*x)
p_coeff_1=lstsq(A,b_1,rcond=-1)[0]
p_coeff_2=lstsq(A,b_2,rcond=-1)[0]
# best fit vector. lstsq returns a list
```

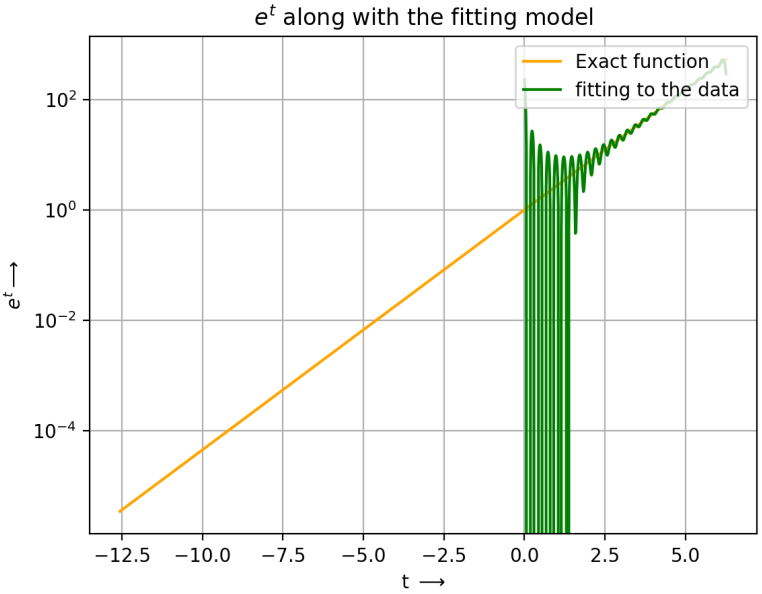
So, in this way we got the predicted Fourier coefficients, and stored them in the p_coeff_1 and p_coeff_2 arrays, respectively.

3.3.1 Least Squares Method

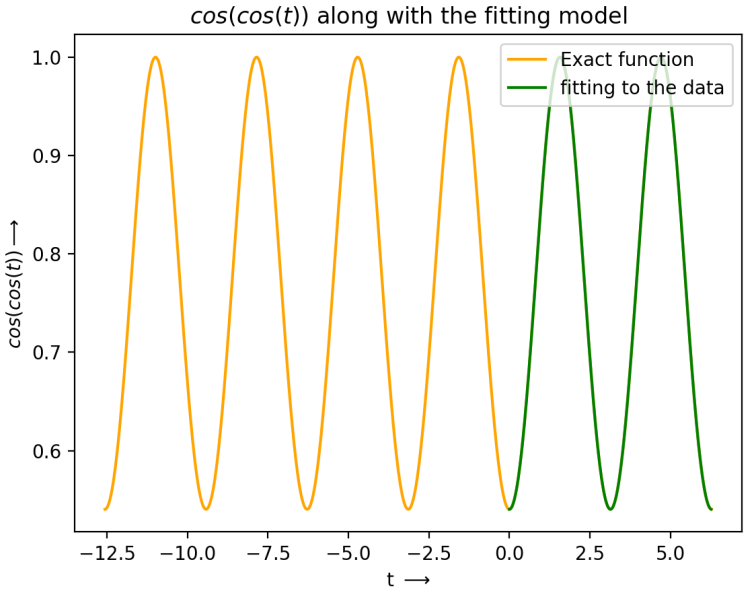
The method of least squares is a standard approach in regression analysis to approximate the solution of overdetermined systems (sets of equations in which there are more equations than unknowns) by minimizing the sum of the squares of the residuals made in the results of every single equation. The most important application is in data fitting. The best fit in the least-squares sense minimizes the sum of squared residuals (a residual being: the difference between an observed value, and the fitted value provided by a model).
So, as the coefficients are varied to minimize the error, we differentiate the error wrt. each coefficient and equate it to zero, in order to obtain the vector of optimal coefficients ($p_i's$).
And in python, in scipy.linalg library, there is a dedicated function, named $lstsq()$ [leastsqares], which accepts the arguments as : $lstsq(F, x)$

3.4 Plots of Functions and Coefficients:

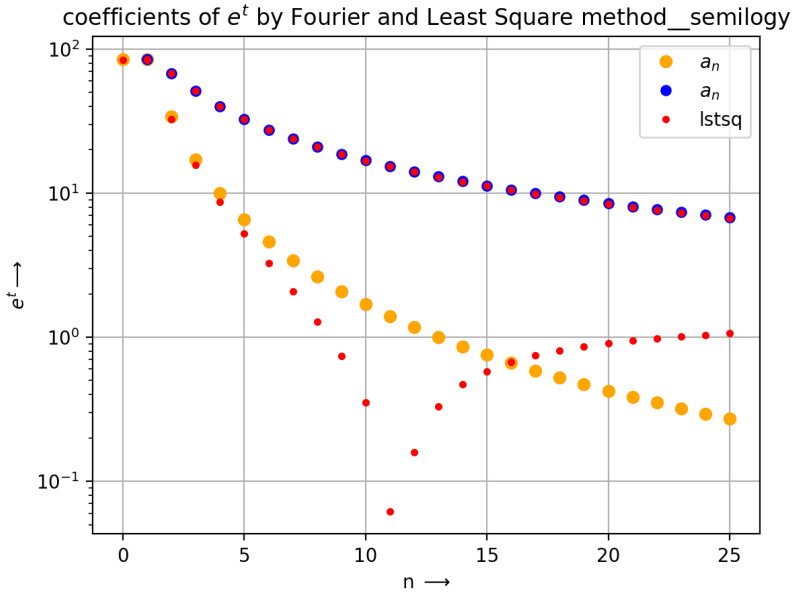
3.4.1 Plot of e^t along with the Estimated Function in Semilog Scale:



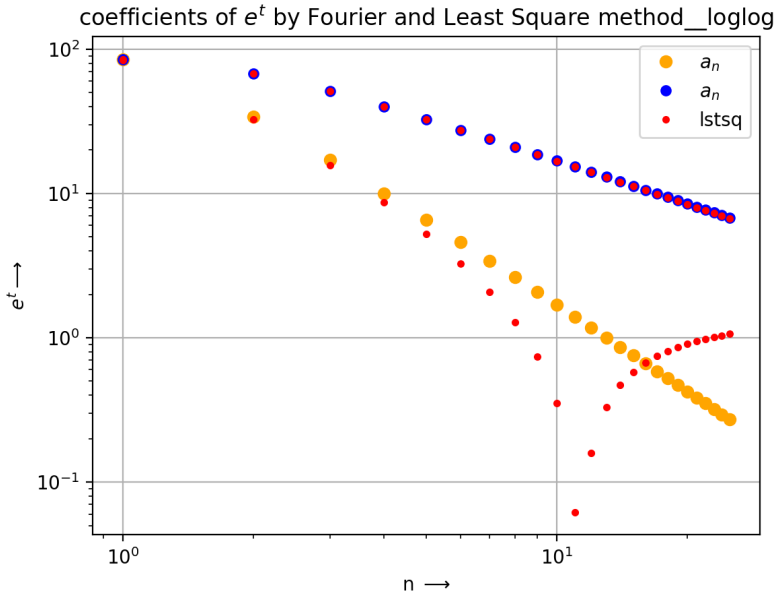
3.4.2 Plot of $cos(cos(t))$ along with the Estimated Function:



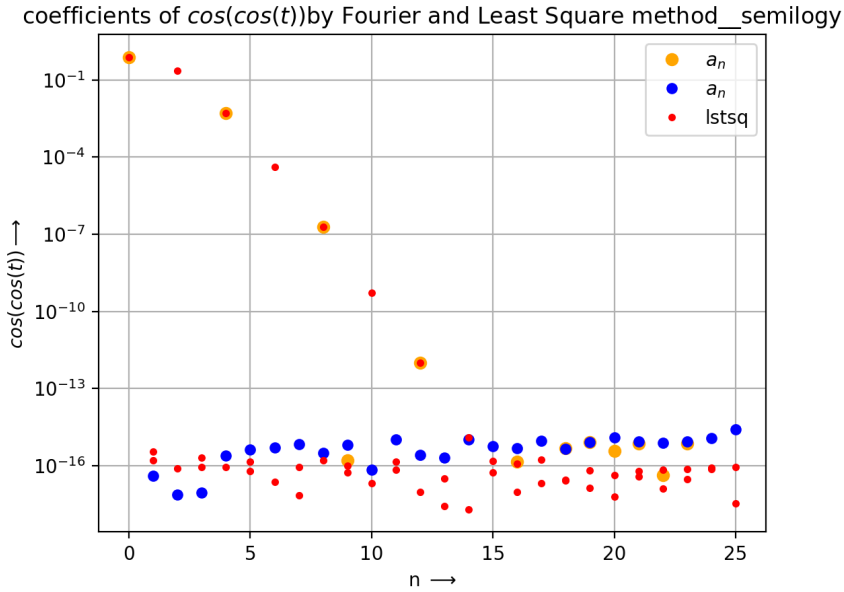
3.4.3 Plot of Coefficients of e^t along with the Estimated Coefficients in Semilog Scale:



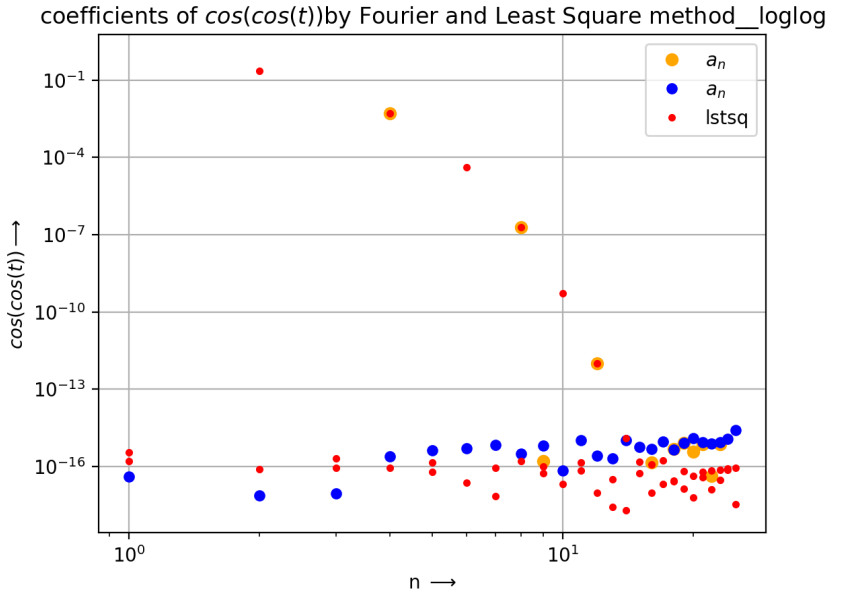
3.4.4 Plot of Coefficients of e^t along with the Estimated Coefficients in loglog Scale:



3.4.5 Plot of Coefficients of $cos(cos(t))$ along with the Estimated Coefficients in Semilog Scale:

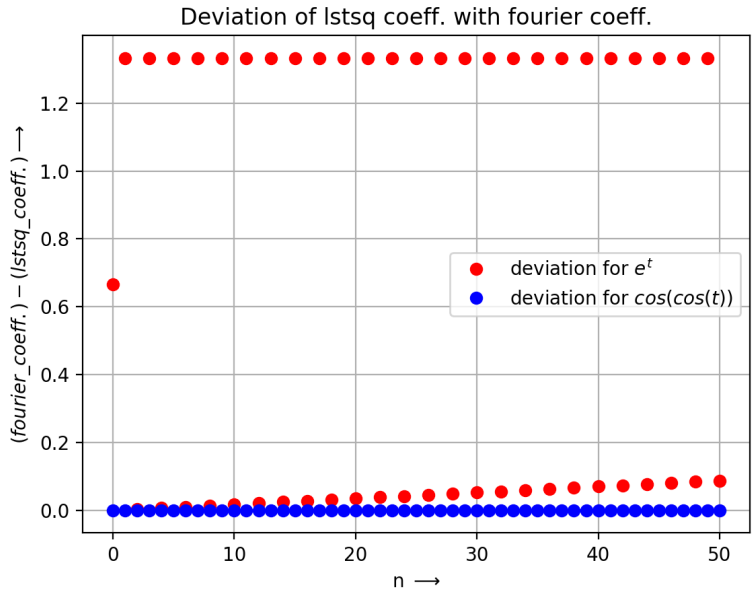


3.4.6 Plot of Coefficients of $cos(cos(t))$ along with the Estimated Coefficients in Loglog Scale:



3.4.7 Deviation of Fourier coefficients from lstsq coefficients:

The below graph shows the absolute difference between the Fourier and lstsq coefficients.



3.4.7.1 Finding the Maximun Deviation of the Coefficients:

The max of the deviation of the lstsq coefficients with respect ot the Fourier coefficients can be computed as follows:

```
amax(abs(coeff_1-p_coeff_1))  ### by using amax function
amax(abs(coeff_2-p_coeff_2))
```

The *amax* function returns the maximun value in the input array. And the output is shown below:

```
nithinuppalapati@Nithins-MacBook-Pro Exp_4 % python fourier.py
The largest deviation in the coeff. for e^{t} is  1.3327308703353395
The largest deviation in the coeff. for cos(cos({t}))$ is  2.5705914064275956e-15
```

4. Conclusions

4.1 On the Deviation of Coefficients:

So, as $cos(cost)$ is periodic with period π, and which is equal to the twice of the fundamental period 2π,

```
amax(abs(coeff_1-p_coeff_1))  ### by using amax function
amax(abs(coeff_2-p_coeff_2))
```

4.1.1 Comment on Lstsq, and Fourier Coefficients:

- So, as we are computing the first few Fourier coefficients, whereas in lstsq method, we are trying to fit the entire function to the 51 coefficients, the two sets of coefficients are not going to be the same.

- So, in order to account the entire function, the first few coefficients will deviate and gradually they are in good agreement.

- And in the case of Fourier coefficients of $cos(cost)$, the $b'_n s$ are nearly zero, (*actually they should be exactly zero, but it is not; due to the floating error while computing...*) as the function, $cos(cost)$ is an even function.

- And in the case of Fourier coefficients of e^t, they are not decaying as in the case of $cos(cost)$, because, the function e^t is not a periodic function, and thus it contains all harmonics of the fundamental frequency.