Assignment 4: Fourier Approximations

T. Pavan Kumar [EE20B140]

March 9, 2022

Abstract

The goal of this assignment is the following.

- To fit two functions e^x and $cos(cos(x))$ using the Fourier series.
- To use least squares fitting to simplify the process of calculating Fourier series.
- To plot graphs to understand the above

1 The functions e^x and $cos(cos(x))$

The following python snippet is used to declare the functions e^x and $cos(cos(x))$. The x values are also declared from -2π to 4π.

```
def exp(x):
    return np.exp(x)


def cos(x):
    return np.cos(np.cos(x))


t = linspace(-2 * pi, 4 * pi, 401)
Exp = exp(t)
Cos = cos(t)
Exp2 = exp(t % (2 * pi))
```

The following code is used to plot the graphs of e^x and $cos(cos(x))$.

```
plt.semilogy(t, Exp, label="original_exp(t)")
plt.semilogy(t, Exp2, label="periodic_exp(t) with period 2*pi")
plt.grid(True, color="grey", linewidth="1.4", linestyle="--")
```

```
xlabel(r"$t$", size=20)
ylabel(r"exp($t$)", size=20)
legend(loc="upper right")
title(r"exponential of $t$ in semilog axis")
```

The plots of e^x and $cos(cos(x))$ are as shown below:

As it is evident from the plots, e^x is aperiodic whereas $cos(cos(x))$ is periodic. The functions generated by the fourier series should be a periodic extension of the actual functions. The plots showing the periodic extensions of e^x and $cos(cos(x))$ are as shown below:

2 The Fourier coefficients

The fourier series used to approximate a function is as follows:

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

The equations used here to find the Fourier coefficients are as follows:

$$a_0 = \frac{1}{2\pi} \int_0^{2\pi} f(x)dx \tag{2}$$

$$a_n = \frac{1}{\pi} \int_0^{2\pi} f(x) \cos(nx)dx \tag{3}$$

$$b_n = \frac{1}{\pi} \int_0^{2\pi} f(x) \sin(nx)dx \tag{4}$$

Hence, in python we will use the *quad()* function to perform an intergration function. First we'll have to create functions which contains the variable k also. The python code snippet for declaring the functions with an additional variable k is as follows:

```
def u(x, k):  # defning the functions that are needed to be integrable
    return np.cos(k * x) * f(
        x
    )  # integrating the u(x,k) gives the a_n coefficients of fourier series


def v(x, k):
    return np.sin(k * x) * f(
        x
    )  # integrating the u(x,k) gives the b_n coefficients of fourier series
```

The python code snippet for finding the fourier coefficients is as follows:

```
for i in range(
    51
):  # integrating the function then scaling and finding the fourier coefficients for
    if i == 0:
        coef1[i] = quad(u, 0, 2 * pi, args=(i))[0] / (2 * pi)
    elif i % 2 != 0:
        coef1[i] = quad(u, 0, 2 * pi, args=((i + 1) / 2))[0] / pi
    elif i % 2 == 0:
        coef1[i] = quad(v, 0, 2 * pi, args=(i / 2))[0] / pi


def f(
    x,
):  # integrating the function then scaling and finding the fourier coefficients for
    return cos(x)


for i in range(51):
    if i == 0:
        coef2[i] = quad(u, 0, 2 * pi, args=(i))[0] / (2 * pi)
    elif i % 2 != 0:
        coef2[i] = quad(u, 0, 2 * pi, args=((i + 1) / 2))[0] / pi
    elif i % 2 == 0:
        coef2[i] = quad(v, 0, 2 * pi, args=(i / 2))[0] / pi
```

3 The plots of Fourier coefficients

The semilog and log plots of the Fourier coefficients of e^x and $cos(cos(x))$ is as shown:

a. As it is evident from the plots, b_n is nearly zero for $cos(cos(x))$. This is because $cos(cos(x))$ is an even function, hence in the fourier series expansion, all the b_n terms should be zero for the series to be an even function.

b. The magnitude of the coefficients would represent how much of certain frequencies happen to be in the output. $cos(cos(t))$ does not have very many frequencies of harmonics, so it dies out quickly. However, since the periodic extension of e^t is discontinuous. To represent this discontinuity as a sum of continuous sinusoids, we would need high frequency components, hence coefficients do not decay as quickly.

c. The loglog plot is linear for e^t since Fourier coefficients of e^t decay with

$1/n$ or $1/n^2$. The semilog plot seems linear in the $cos(cos(t))$ case as its fourier coefficients decay exponentially with n.

4 The Least Squares Approach

For the least squares approach, we'll have to create matrices and then use *lstsq()* function inorder to get the most approximate values of the fourier coefficients.
The python code snippet to create the matrices and to get the least squared value of the coefficients is as follows:

```
x = linspace(
    0, 2 * pi, 401
)  # defining the time space array for the least square method
x = x[:-1]
b1 = np.exp(x)  # the source array for the least square matrix
b2 = np.cos(x)
A = np.zeros(
    (400, 51)
)  # the coefficient matrix of the variables for the least square method
A[:, 0] = 1

for k in range(1, 26):  # applying the least squares method
    A[:, 2 * k - 1] = np.cos(k * x)
    A[:, 2 * k] = np.sin(k * x)
c1 = lstsq(A, b1, rcond=1)[0]
c2 = lstsq(A, b2, rcond=1)[0]
```

The plots in order to show the differences between the actual and predicted values of the fourier coefficients are shown below

5 Estimated Functions

Using the predicted values of the fourier coefficients, we can calculate the functional values for both e^x and $cos(cos(x))$.
The plots showing both the actual and predicted functional values are as shown below:

The $cos(cos(t))$ vs t graph, agrees almost perfectly, beyond the scope of the precision of the least squares fitter. The Fourier approximation of e^t does not agree very well to the ideal case near the discontinuity. The cause for this is the Gibbs Phenomenon, which can be described as below. The partial sums of the Fourier series will have large oscillations near the discontinuity of the function. These oscillations do not die out as n increases, but approaches a finite limit. This is one of the causes of ringing artifacts in signal processing, and is very undesirable. Plotting the output on a linear graph would make this ringing much more apparent.

Conclusions

- We saw two different ways to calculate the Fourier series of a periodic signal.
- We saw how least squares fitting can be used to simplify the process of calculating the Fourier Series.
- We observed Gibbs phenomenon at the discontinuity in the Fourier approximation of e^t.

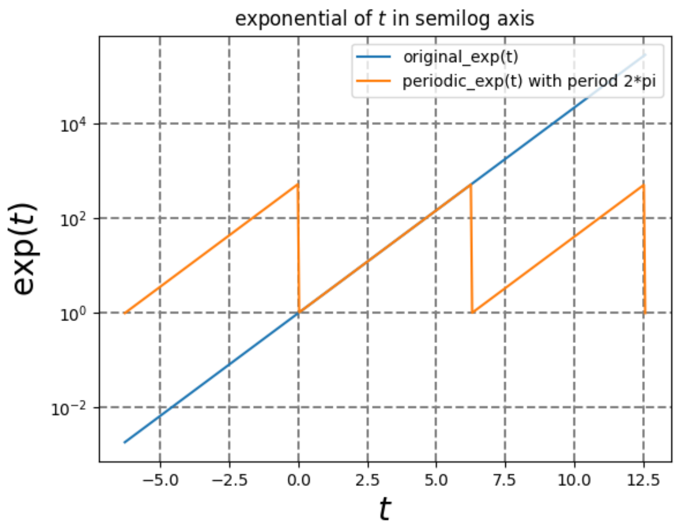

Figure 1: Data plot

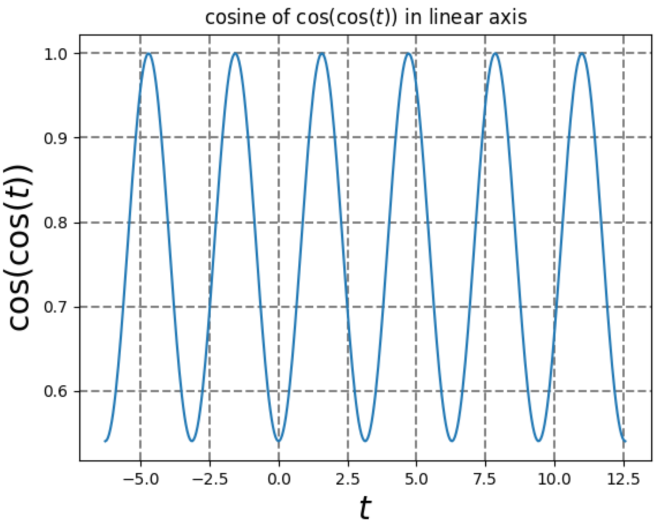

Figure 2: Data plot

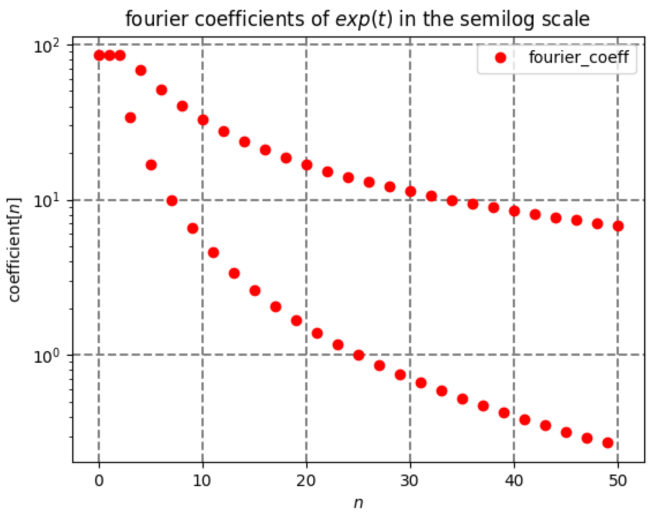

Figure 3: Data plot

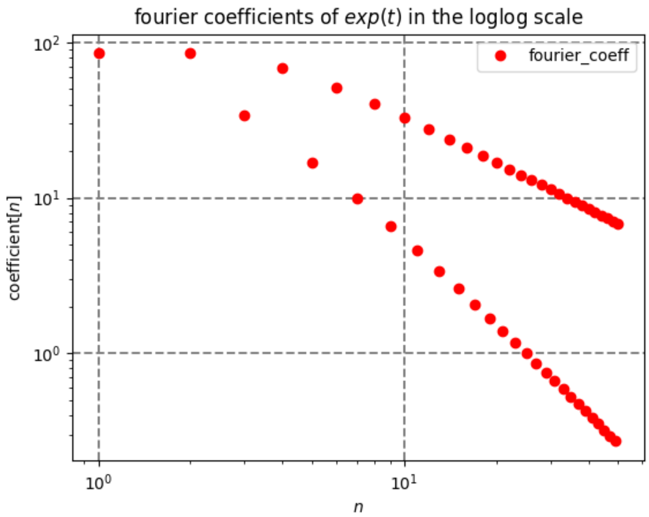

Figure 4: Data plot

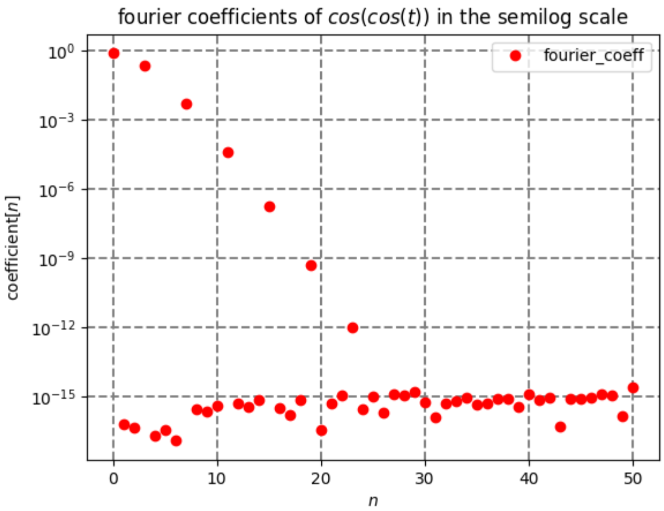

Figure 5: Data plot

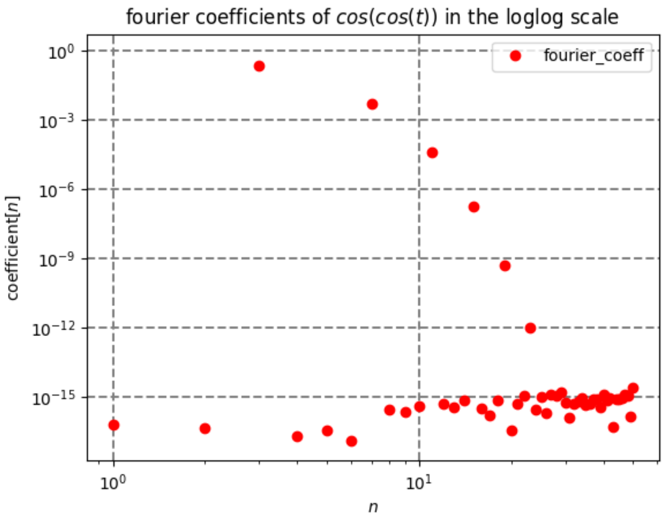

Figure 6: Data plot

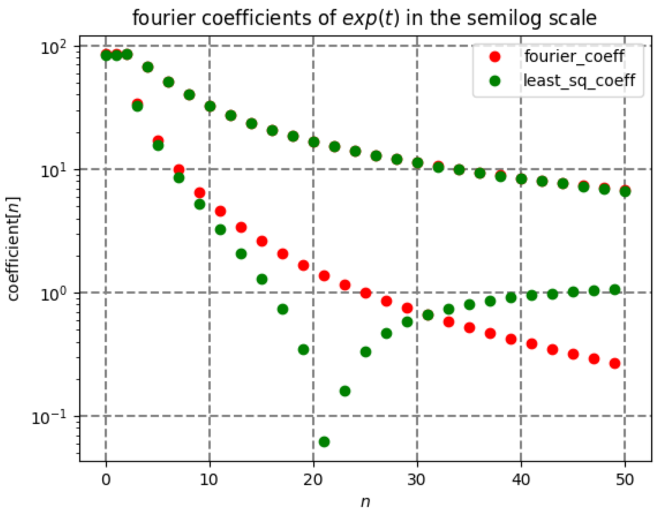

Figure 7: Data plot

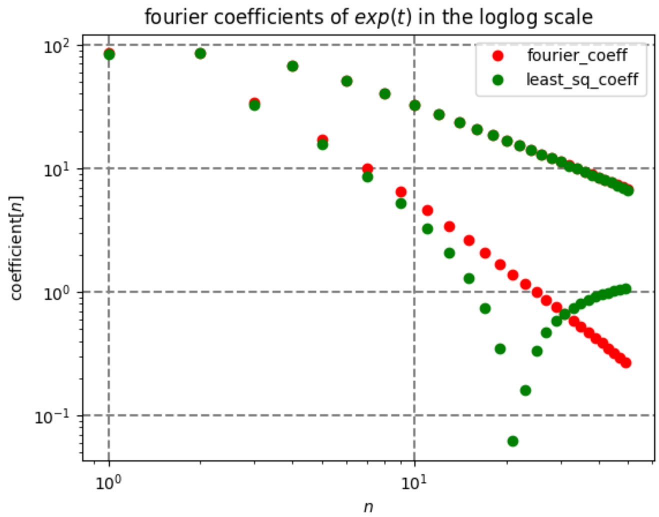

Figure 8: Data plot

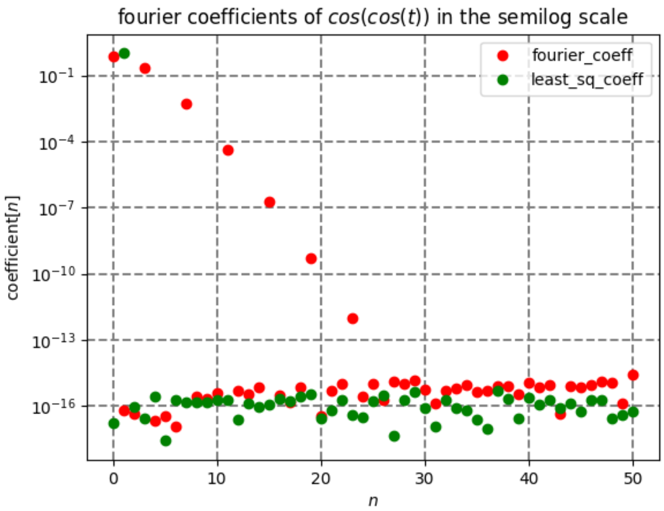

Figure 9: Data plot

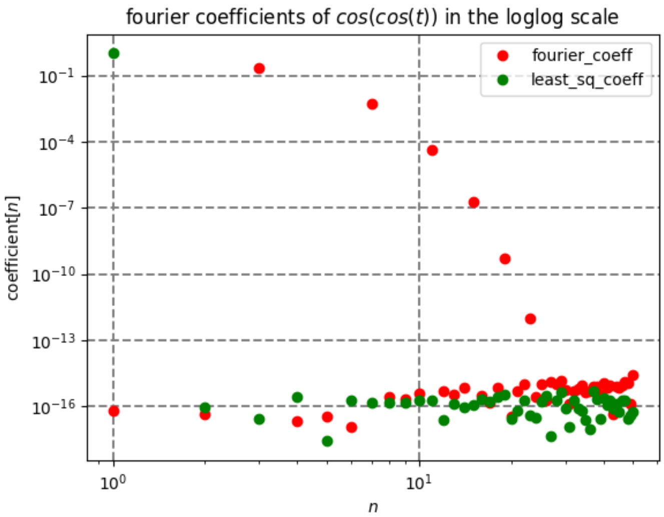

Figure 10: Data plot

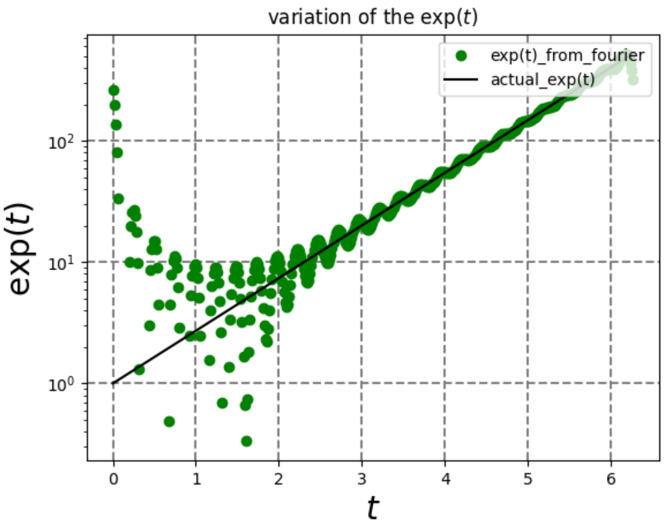

Figure 11: Data plot

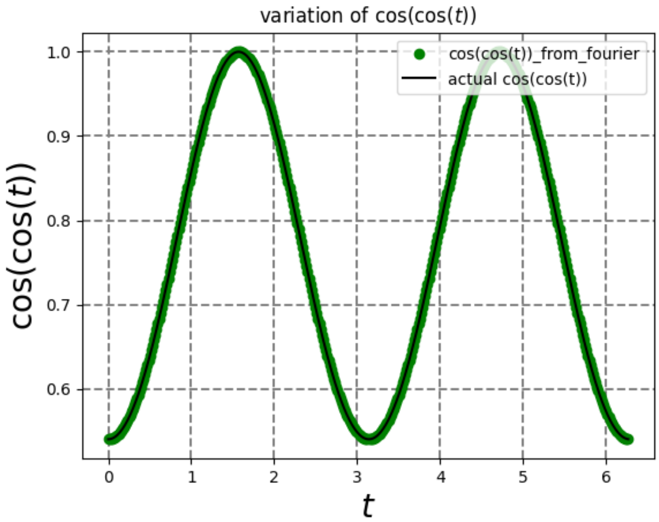

Figure 12: Data plot