Assignment No 4

Shivanshu Shekhar

February 26, 2022

1 Abstract

In this weeks assignment we are going to fit two function e^x and $\cos(\cos x)$ over the interval $[0, 2pi)$ using Fourier series coefficients.

2 Introduction

The Fourier Series of a function f(x) with period 2π is computed as follows:

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

Where,

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

For the sake of this assignment, Since $\exp x$ doesn't have a period of 2π,We choose to change its definition in a piece-wise manner to satisfy periodicity

3 Problems and Solutions

3.1 Question 1

```
#Exponential function that takes both scalar and vector as input
def exp(x):
    return np.exp(x)

#Nested cos function that takes both scalar and vector as input
def cc(x):
    return np.cos(np.cos(x))


#helper
a_s = np.array([True]+ [True if i%2 != 0 else False for i in range(1,51)])
b_s = np.array([False] + [True if i%2 == 0 else False for i in range(1,51)])

#Using 100 points in between -2pi and 4pi for plot
x = np.linspace(-2*math.pi, 4*math.pi, 99)
temp = exp(np.linspace(0,2*math.pi,33)).reshape(-1,1)
ffit = np.concatenate((temp,temp,temp), axis = 0)
fig = plt.figure(figsize=(70,70))
plt.subplot(10,10,1)
plt.title("Plot of $e^{x}$ in semilog y scale")
plt.plot(x, exp(x), label = "$e^{x}$")
plt.plot(x, ffit, linestyle = "dashed" ,label = "Fourier fit")
plt.grid()
plt.legend()
plt.ylabel("$e^{x}$", fontsize =15);plt.xlabel("x", fontsize = 15)
plt.yscale("log")
plt.subplot(10,10,2)
plt.title("Plot of $cos(cos(x))$ in linear scale")
plt.plot(x, cc(x), label = "cos(cos(x))")
plt.plot(x, cc(x),linestyle = "dashed" ,label = "Fourier fit")
plt.xlabel("x", fontsize =15);plt.ylabel("$cos(cos(x))$", fontsize =15)
plt.legend()
plt.grid()
plt.show()
```

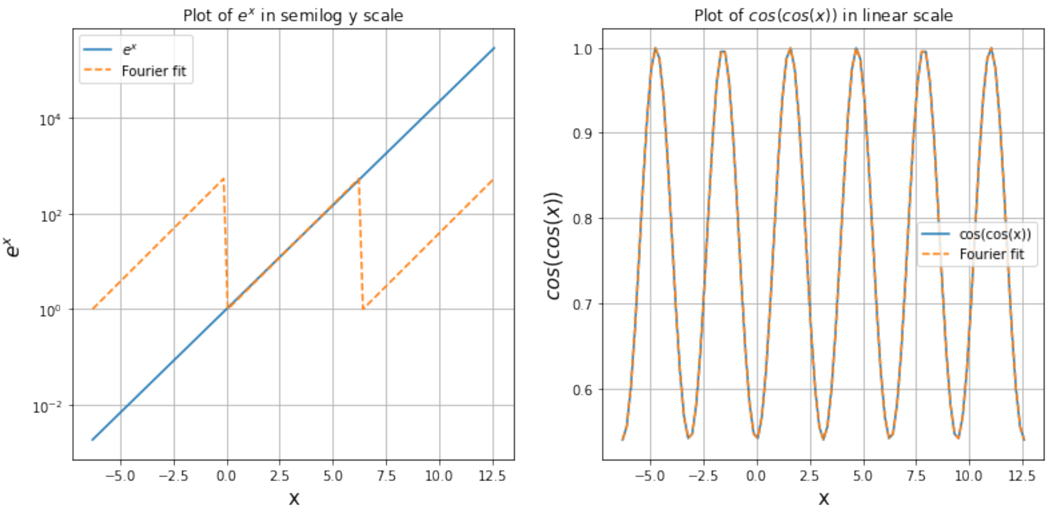

Figure 1: "Question 1"

3.2 Question 2

We use the **scipy quad** function for integration and instead of defining a new 2nd redundant function we used lambda function to get the job done and stored the result in the following format as a numpy column matrix.

$$\begin{pmatrix} a_0 \\ a_1 \\ b_1 \\ ... \\ a_{25} \\ b_{25} \end{pmatrix}$$

```
a = np.zeros((26,2))
b_ = np.zeros((26,2))
a[0,:]  = 1/(2*math.pi)*np.array([integrate.quad(exp, 0, 2*math.pi)[0],
integrate.quad(cc, 0, 2*math.pi)[0]])
plotting = np.zeros((1,2))
plotting[0,:] = a[0,:]
for i in range(1, 26):
    a[i,:] = 1/math.pi*np.array([integrate.quad(lambda x, k:
     exp(x)*math.cos(k*x), 0, 2*math.pi, args=(i,))[0]
              , integrate.quad(lambda x, k: cc(x)*math.cos(k*x), 0, 2*math.pi,
              args=(i,))[0]])
    b_[i,:] = 1/math.pi*np.array([integrate.quad(lambda x, k:
     exp(x)*math.sin(k*x), 0, 2*math.pi, args=(i,))[0]
              , integrate.quad(lambda x, k: cc(x)*math.sin(k*x), 0, 2*math.pi,
              args=(i,))[0]])
    plotting = np.concatenate((plotting,a[i,:].reshape(-1,2), b_[i,:].reshape(-1,2)))
```

3.3 Question 3

We plot the magnitude of the coefficients we got in question 2 in the same order as the matrix given there, in semilog scale(y-axis) and loglog scale.

```
plt.figure(figsize=(70,70))
plt.subplot(10,10,1)
plt.title("Coefficients of fourier series of $e^{x}$ in semilog y scale")
plt.semilogy(np.abs(a[:,0]), "ro", label = "$a_n$")
plt.semilogy(range(1,26), np.abs(b_[1:,0]), "bo", label = "$b_n$")
plt.grid()
plt.legend()
plt.subplot(10,10,2)
plt.loglog(np.abs(a[:,0]), "ro", label = "$a_n$")
plt.loglog(range(1,26), np.abs(b_[1:,0]), "bo", label = "$b_n$")
plt.title("Coefficients of fourier series of $e^{x}$ in loglog scale")
plt.grid()
plt.legend()
plt.show()
plt.figure(figsize=(70,70))
plt.subplot(10,10,1)
plt.semilogy(np.abs(a[:,1]), "ro", label = "$a_n$")
plt.semilogy(range(1,26), np.abs(b_[1:,1]), "bo", label = "$b_n$")
plt.grid()
plt.title("Coefficients of fourier series of $cos(cos(x))$ in semilog y scale")
plt.legend()
plt.subplot(10,10,2)
plt.loglog(np.abs(a[:,1]), "ro", label = "$a_n$")
plt.loglog(range(1,26), np.abs(b_[1:,1]), "bo", label = "$b_n$")
plt.grid()
plt.title("Coefficients of fourier series of $cos(cos(x})$ in loglog scale")
plt.legend()
plt.show()
```

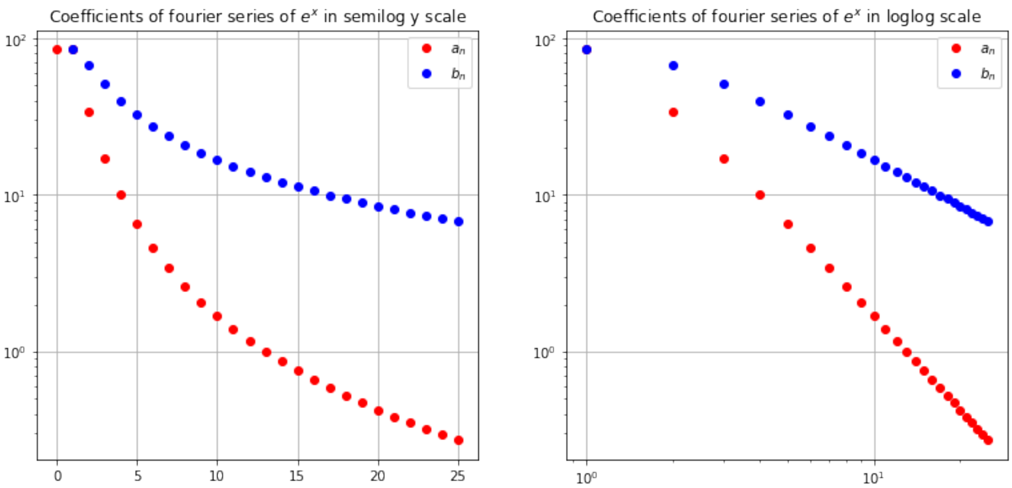

Figure 2: Question 3

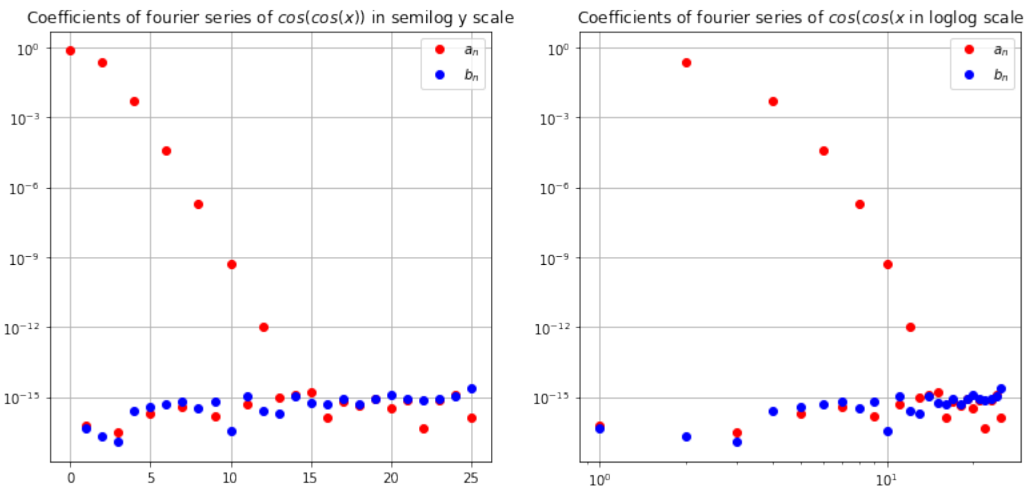

Figure 3: Question 3

3.3.1 If you did Q1 correctly, the b_n coefficients in the second case should be nearly zero. Why does this happen?

This happens because $\cos(cosx)$ is a periodic function with period of π and is also an even function so all the odd harmonics will be zero so the error is very low.

3.3.2 In the first case, the coefficients do not decay as quickly as the coefficients for the second case. Why not?

In the first case, as an exponential has a number of frequencies in it, it has a wide range of frequencies in its Fourier approximation. On the other hand, the second case has only a low frequency of $\frac{1}{\pi}$, and hence has a low contribution from higher sinusoids.

3.3.3 Why does loglog plot in Figure 4 look linear, whereas the semilog plot in Figure 5 looks linear?

The magnitude of the coefficients of e^x vary as:

$$|a_n|, |b_n| \propto \frac{1}{1+n^2}$$

Thus, with larger values of n, it becomes proportional to $\frac{1}{n^2}$ and hence the log-log plot has a slope of -2$\log n$ and appears to become linear

Similarly for $\cos(\cos x)$, the coefficients vary exponentially with n, and hence, $\log y$ vs x is linear.

3.4 Question 4

We find the Fourier coefficient using **scipy lstsq** function for the following matrix equation:

$$\begin{pmatrix} 1 & \cos x_1 & \sin x_1 & ... & \cos 25x_1 & \sin 25x_1 \\ 1 & \cos x_2 & \sin x_2 & ... & \cos 25x_2 & \sin 25x_2 \\ ... & ... & ... & ... & ... & ... \\ 1 & \cos x_{400} & \sin x_{400} & ... & \cos 25x_{400} & \sin 25x_{400} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ b_1 \\ ... \\ a_{25} \\ b_{25} \end{pmatrix} = \begin{pmatrix} f(x_1) \\ f(x_2) \\ ... \\ f(x_{400}) \end{pmatrix}$$

```
x = np.linspace(0,2*math.pi,400, endpoint = False).reshape(-1,1)
b = np.c_[exp(x), cc(x)]

def get_A(dim1, dim2, x):
    A = np.ones((dim1,1))
    tmp = 1
    for i in range(1,dim2):
        if(i%2 != 0):
            A = np.c_[A, np.multiply(np.cos(tmp*x).reshape(-1,1),np.ones((dim1,1)))]
        else:
            A = np.c_[A, np.multiply(np.sin(tmp*x).reshape(-1,1),np.ones((dim1,1)))]
            tmp+=1
    return A
A = get_A(400,51,x)
```

3.5 Question 5

We plot the best fit points and true Fourier coefficients for each function together in both loglog scale and semilog scale.

```
c_0 = sp.linalg.lstsq(A,b[:,0])[0].reshape(-1,1)
c_1 = sp.linalg.lstsq(A,b[:,1])[0].reshape(-1,1)

c_0_a = c_0[a_s,:]; c_0_b = c_0[b_s,:]

c_1_a = c_1[a_s,:]; c_1_b = c_1[b_s,:]
plt.show()
plt.figure(figsize=(70,70))
plt.subplot(10,10,1)
plt.title("Plot of least squre fit vs true fourier coefficients")
plt.semilogy(np.abs(a[:,0]), "ro", label = "True $a_n$")
plt.semilogy(range(1,26), np.abs(b_[1:,0]), "ro", label = "True $b_n$")
plt.semilogy(np.abs(c_0_a[:,0]), "go", label = "Estimated $a_n$")
plt.semilogy(range(1,26), np.abs(c_0_b[:,0]), "go", label = "Estimated $b_n$")
plt.grid()
plt.legend()
plt.subplot(10,10,2)
plt.title("Plot of least squre fit vs true fourier coefficients")
plt.loglog(np.abs(a[:,0]), "ro", label = "True $a_n$")
plt.loglog(range(1,26), np.abs(b_[1:,0]), "ro", label = "True $b_n$")
plt.loglog(np.abs(c_0_a[:,0]), "go", label = "Estimated $a_n$")
plt.loglog(range(1,26), np.abs(c_0_b[:,0]), "go", label = "Estimated $b_n$")
plt.grid()
plt.legend()
plt.show()
plt.figure(figsize=(70,70))
plt.subplot(10,10,1)
plt.title("Plot of least squre fit vs true fourier coefficients")
plt.semilogy(np.abs(a[:,1]), "ro", label = "True $a_n$")
plt.semilogy(range(1,26), np.abs(b_[1:,1]), "ro", label = "True $b_n$")
plt.semilogy(np.abs(c_1_a[:,0]), "go", label = "Estimated $a_n$")
plt.semilogy(range(1,26), np.abs(c_1_b[:,0]), "go", label = "Estimated $b_n$")
plt.grid()
plt.legend()
plt.subplot(10,10,2)
plt.title("Plot of least squre fit vs true fourier coefficients")
plt.loglog(np.abs(a[:,1]), "ro", label = "True $a_n$")
plt.loglog(range(1,26), np.abs(b_[1:,1]), "ro", label = "True $b_n$")
plt.loglog(np.abs(c_1_a[:,0]), "go", label = "Estimated $a_n$")
plt.loglog(range(1,26), np.abs(c_1_b[:,0]), "go", label = "Estimated $b_n$")
plt.grid()
plt.legend()
plt.show()
```

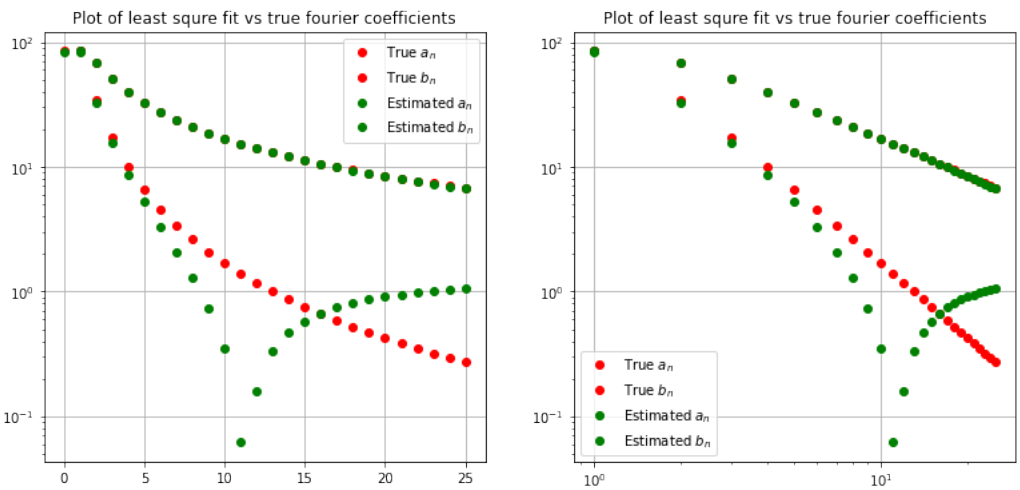

Figure 4: Question 5

Figure 5: Question 5

3.6 Question 6

We find the absolute error between the best fit values and actual Fourier coefficients and plot them in linear scale.

```
err = abs(plotting - np.c_[c_0,c_1])
plt.plot(err[:,0],"bo", label = "Error")
plt.title("Error in estimated vs true value")
plt.legend()
plt.show()
print(f"The max error is {err.max(axis = 0)[0]}")
plt.plot(err[:,1], "bo", label = "Error")
plt.title("Error in estimated vs true value")
plt.legend()
plt.show()
print(f"The max error is {err.max(axis = 0)[1]}")
```

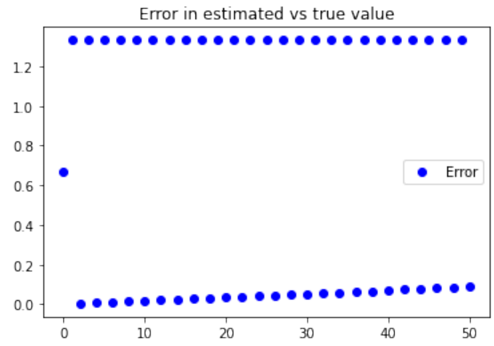

Figure 6: Question 6

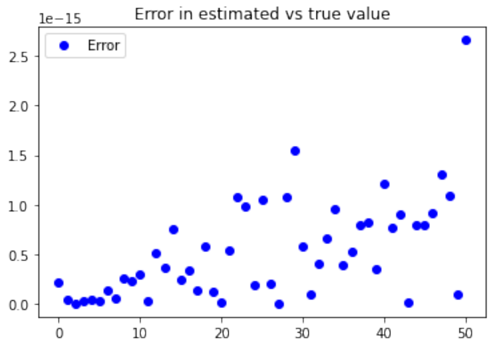

Figure 7: Question 6

3.7 Question 7

e^x is a non periodic function, so we have considered the variation of e^x with period 2π that has the actual value of e^x only in the range $[0,2\pi)$. Hence it is acceptable that there is a large discrepancy in the predicted value of e^x at the boundaries.

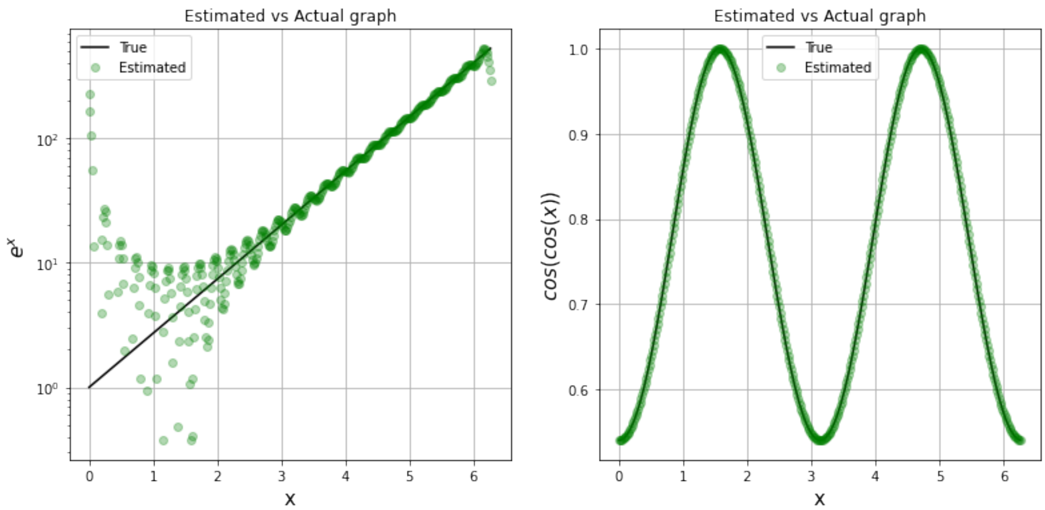

Figure 8: Question 7

```
f_1 = np.dot(A,c_0)
f_2 = np.dot(A,c_1)
fig = plt.figure(figsize=(70,70))
plt.subplot(10,10,1)
plt.title("Estimated vs Actual graph")
plt.plot(x, exp(x), "k", label = "True")
plt.plot(x, f_1, "go", alpha = 0.3, label = "Estimated")
plt.legend()
plt.grid()
plt.ylabel("$e^{x}$", fontsize =15);plt.xlabel("x", fontsize = 15)
plt.yscale("log")
plt.subplot(10,10,2)
plt.title("Estimated vs Actual graph")
```

```
plt.plot(x, cc(x),"k", label = "True")
plt.plot(x,f_2, "go", alpha = 0.3, label = "Estimated")
plt.xlabel("x", fontsize =15);plt.ylabel("$cos(cos(x))$", fontsize =15)
plt.legend()
plt.grid()
plt.show()
```

4 Conclusion

We found that Fourier series converged for periodic function whereas for a non periodic function it failed to converge outside the region of $[0, 2\pi)$.