EE2703: Applied Programming Lab

Assignment # 4

Sourav Sahoo

February 23, 2019

Contents

1 Introduction

A Fourier series is an expansion of a periodic function $f(x)$ in terms of an infinite sum of sines and cosines. Fourier series make use of the orthogonality relationships of the sine and cosine functions. The computation and study of Fourier series is known as harmonic analysis and is extremely useful as a way to break up an arbitrary periodic function into a set of simple terms that can be plugged in, solved individually, and then recombined to obtain the solution to the original problem or an approximation to it to whatever accuracy is desired or practical.[1]
In this assignment we find the Fourier Expansions of $f(x) = e^x$ and $f(x) = cos(cos(x))$.

2 Actual Functions

2.1 $f(x) = e^x$

The functions is generated by executing he code :

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

The function is plotted by executing the code, both linear and semilog plot:

```
plt.grid()
plt.semilogy(x,f(x))
plt.title('Plot of f(x) in semilog scale')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.show()

plt.grid()
plt.plot(x,f(x))
plt.title('Plot of f(x)')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.show()
```

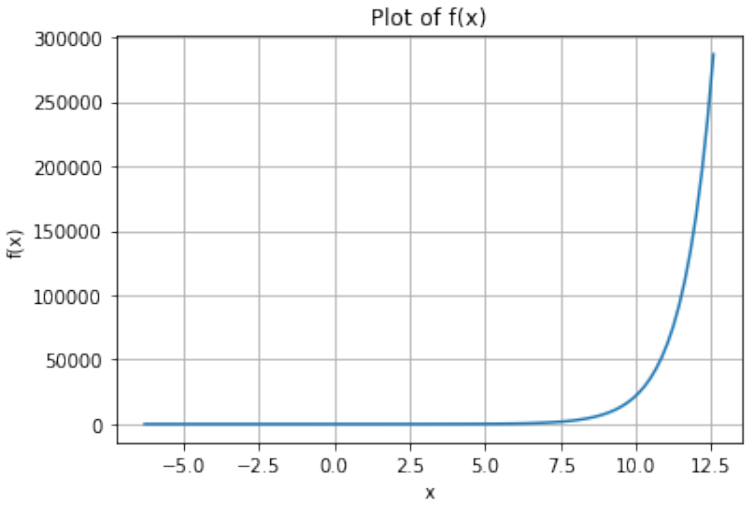

Figure 1: e^x in linear scale

[1] http://mathworld.wolfram.com/FourierSeries.html

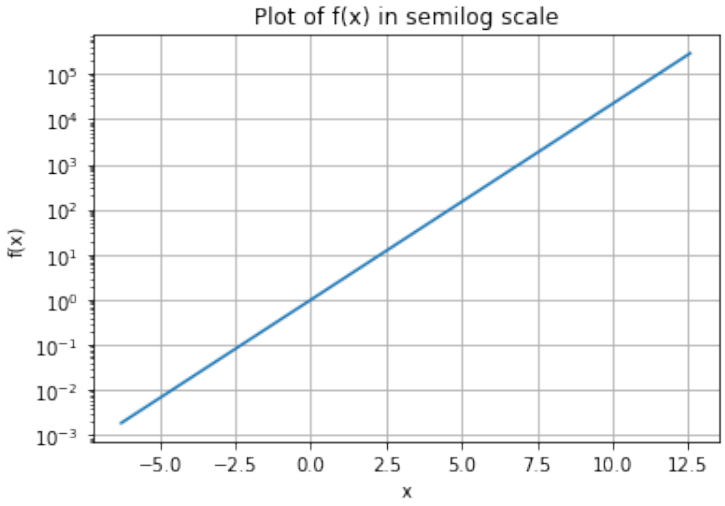

Figure 2: e^x in semi-log scale

2.2 $g(x) = cos(cos(x))$

The function is generated by executing he code :

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

The functions is plotted by executing the code :

```
x = np.linspace(-2*pi,4*pi,100)
plt.grid()
plt.plot(x,g(x))
plt.title('Plot of g(x)')
plt.xlabel('x')
plt.ylabel('g(x)')
plt.show()
```

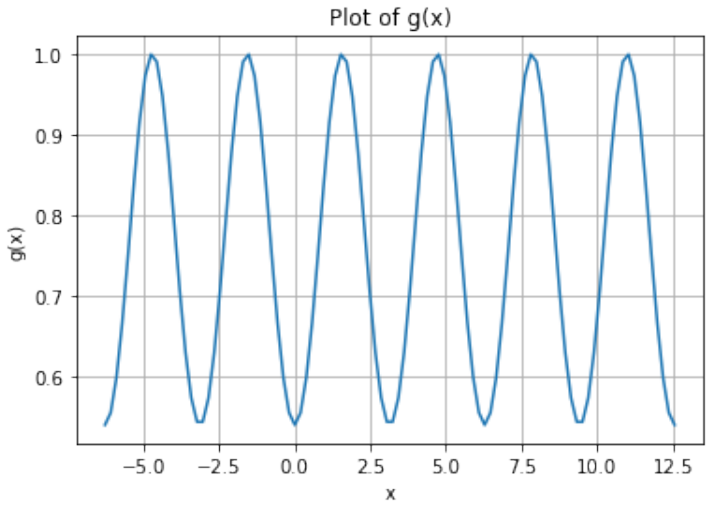

Figure 3: $cos(cos(x))$

The function $f(x) = e^x$ is not periodic whereas $g(x) = cos(cos(x))$ is periodic with period π.

3 Fourier Coefficients

The Fourier coefficients are calculated using the given formulae:

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

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

Now we separately generate the first 51 Fourier coefficients of the $f(x)$ and $g(x)$.

3.1 $f(x) = e^x$

The integration function is calculated for $f(x) = e^x$:

```
def u(x,k):
    return f(x)*np.cos(k*x)
def v(x,k):
    return f(x)*np.sin(k*x)
for i in range(26):
    fa[i],_ = sp.integrate.quad(u,0,2*pi,(i,))
for i in range(25):
    fb[i],_ = sp.integrate.quad(v,0,2*pi,(i,))
fa /= pi
fb /= pi
fa[0] /= 2
```

Then we merge the coefficients in a single numpy array.

```
F = [None]*(len(fa)+len(fb))
F[0] = fa[0]
F[1::2] = fa[1:]
F[2::2] = fb
F = np.asarray(F)
```

Then plot the coefficients in semilog and loglog scale.

```
plt.grid()
plt.semilogy(abs(F),'o',color = 'r',markersize = 4)
plt.title('Fourier Coefficients for f(x) by direct integration')
plt.xlabel('n')
plt.ylabel('Fourier Coefficients')
plt.show()

plt.grid()
plt.loglog(abs(F),'o',color = 'r',markersize = 4)
plt.title('Fourier Coefficients for f(x) by direct integration')
plt.xlabel('n')
plt.ylabel('Fourier Coefficients')
plt.show()
```

3.2 $g(x) = cos(cos(x))$

The integration function is calculated for $g(x) = cos(cos(x))$:

```
def w(x,k):
    return g(x)*np.cos(k*x)
def z(x,k):
    return g(x)*np.sin(k*x)
```

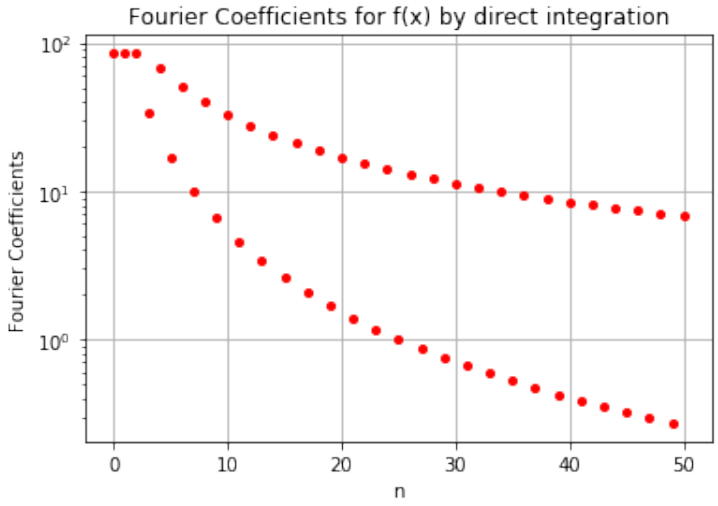

Figure 4: Fourier Coefficients for $f(x)$ by direct integration in semilog scale

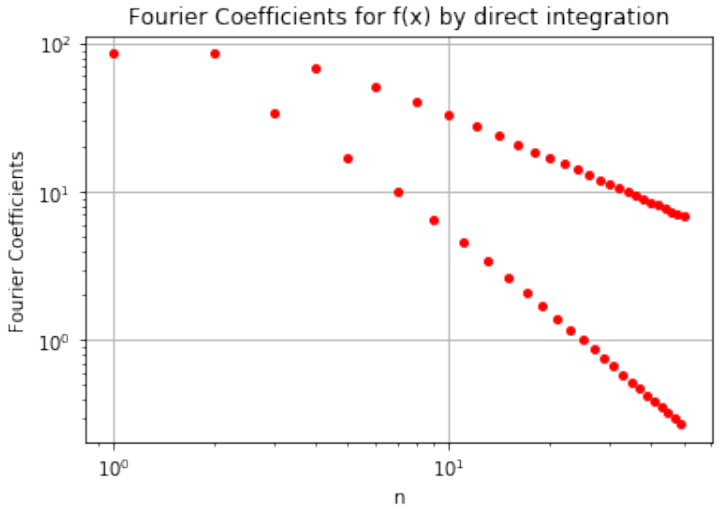

Figure 5: Fourier Coefficients for $f(x)$ by direct integration in loglog scale

```
for i in range(26):
    ga[i],_ = sp.integrate.quad(w,0,2*pi,(i,))
for i in range(25):
    gb[i],_ = sp.integrate.quad(z,0,2*pi,(i,))
ga /= pi
gb /= pi
ga[0] /= 2
```

Then we merge the coefficients in a single numpy array.

```
G = [None]*(len(ga)+len(gb))
G[0] = ga[0]
G[1::2] = ga[1:]
G[2::2] = gb
G = np.asarray(G)

plt.grid()
plt.semilogy(abs(G),'o',color = 'r',markersize = 4)
plt.title('Fourier Coefficients for g(x) by direct integration')
```

```
plt.xlabel('n')
plt.ylabel('Fourier Coefficients')
plt.show()

plt.grid()
plt.loglog(abs(G),'o',color = 'r',markersize = 4)
plt.title('Fourier Coefficients for g(x) by direct integration')
plt.xlabel('n')
plt.ylabel('Fourier Coefficients')
plt.show()
```

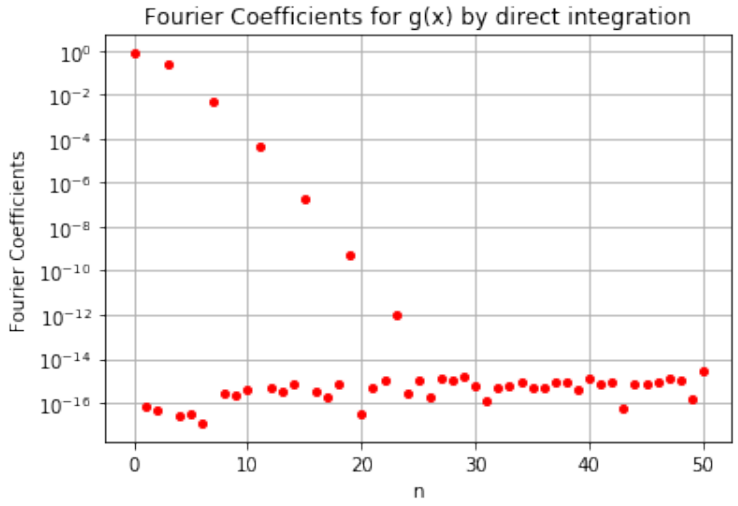

Figure 6: Fourier Coefficients for $g(x)$ by direct integration in semilog scale

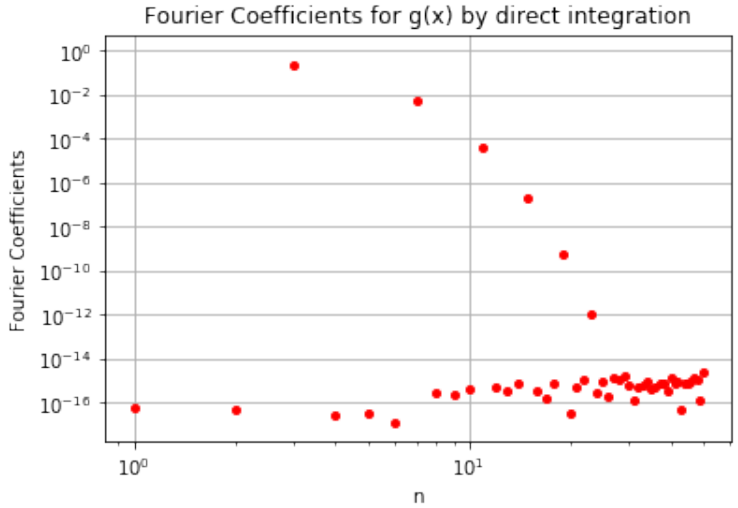

Figure 7: Fourier Coefficients for$g(x)$ by direct integration in loglog scale

3.3 Inline Questions

3.3.1 Question 1

The function $g(x) = cos(cos(x))$ is an even function. So,ideally it has zero odd harmonics i.e. sine components in Fourier series.So, b_n are approximately 0 in the second function.

3.3.2 Question 2

The more the function sinusoidal, lesser the number of non-zero coefficients it has in its Fourier Expansion. For example ,$f(x) = sin\,x$ has only one non-zero Fourier coefficient. Similarly $g(x) = cos(cos(x))$ is much more sinusoidal than $f(x) = e^x$. So, the Fourier coefficients converge to zero very quickly in case of $g(x)$.

3.3.3 Question 3

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

$$= \frac{1}{\pi}\int_0^{2\pi} e^x\,cos(nx)\,dx \;= \frac{e^{2\pi}-1}{n^2+1}$$

Taking log on both sides;

$$log(a_n) = C - log(n^2+1) \approx C - log(n^2) = C - 2log(n)$$

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

$$= \frac{1}{\pi}\int_0^{2\pi} e^x\,sin(nx)\,dx \;= \frac{n-e^{2\pi}}{n^2+1}$$

Taking log on both sides;

$$log(b_n) = log(n-e^{2\pi}) - log(n^2+1) \approx log(n) - log(n^2) = -log(n)$$

Hence, the log-log plot for $f(x) = e^x$ is almost linear for higher n.

$$a_n = \frac{1}{\pi}\int_0^{2\pi} g(x)\,cos(nx)\,dx$$

$$= \frac{1}{\pi}\int_0^{2\pi} cos(cos(x))\,cos(nx)\,dx$$

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

Taking log on both sides;

$$log(a_n) \approx -n for\ higher\ values\ of\ n.$$

Hence, the semilog plot for $g(x) = cos(cos(x))$ is almost linear for higher n.

4 Reconstructing Actual Functions

4.1 Generating Fourier Coefficients

The original functions are tried to be reconstructed using the Fourier expansion upto first 51 terms. The Fourier coefficients are calculated using `np.linalg.lstsq` function.

```
X = np.linspace(0,2*pi,401)
X = X[:-1]
b1 = f(X)
b2 = g(X)
A = np.zeros((400,51))
A[:,0] = 1
for k in range(1,26):
    A[:,2*k-1] = np.cos(k*X)
    A[:,2*k] = np.sin(k*X)

c1 = np.linalg.lstsq(A,b1,rcond = -1)[0]
c2 = np.linalg.lstsq(A,b2,rcond = -1)[0]
```

Then the calculated Fourier coefficients are plotted.

```
plt.grid()
plt.plot(c1,'o',color = 'g')
plt.title('Fourier Coefficients calculated using lstsq method')
plt.xlabel('n')
plt.ylabel('Coefficients')
plt.show()

plt.grid()
plt.plot(c2,'o',color = 'g')
plt.title('Fourier Coefficients calculated using lstsq method')
plt.xlabel('n')
plt.ylabel('Coefficients')
plt.show()
```

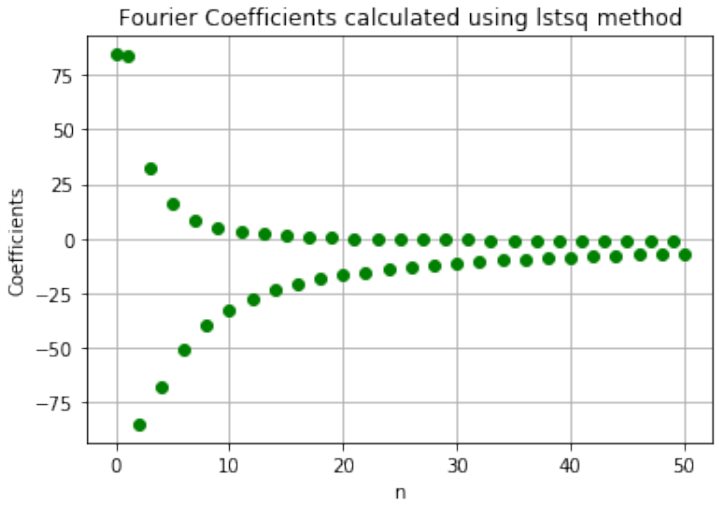

Figure 8: Fourier Coefficients for $f(x)$ by lstsq method

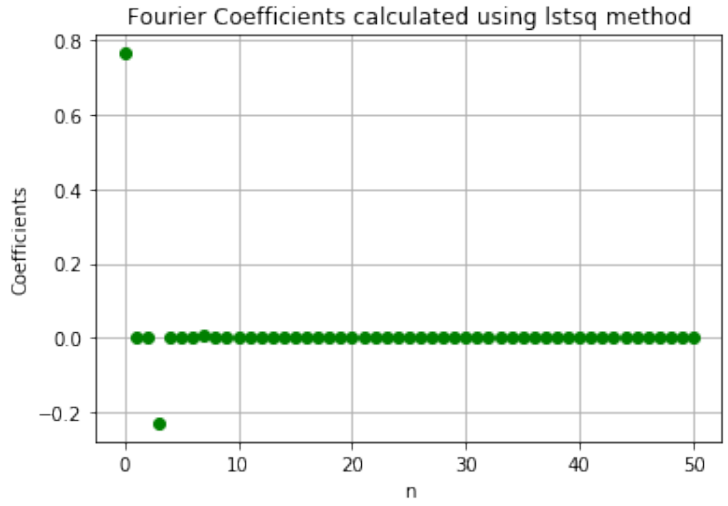

Figure 9: Fourier Coefficients for $g(x)$ by lstsq method

4.2 Actual and Reconstructed Functions

The function reconstructed after obtaining the Fourier coefficients using `np.linalg.lstsq` method is plotted on the same graph to find the deviation.

```
final_f = A.dot(c1)
plt.grid()
plt.plot(X,final_f,'-',color = 'g')
plt.plot(X,f(X),'-.',color = 'b')
plt.title(' Original Function and Reconstructed Function from Fourier Coefficients')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.legend(['original','reconstructed'])
plt.show()
final_g = A.dot(c2)
plt.grid()
plt.plot(X,final_g,'-',color = 'g')
plt.plot(X,g(X),'-.',color = 'b')
plt.title(' Original Function and Reconstructed Function from Fourier Coefficients')
plt.xlabel('x')
plt.ylabel('g(x)')
plt.legend(['original','reconstructed'])
plt.show()
```

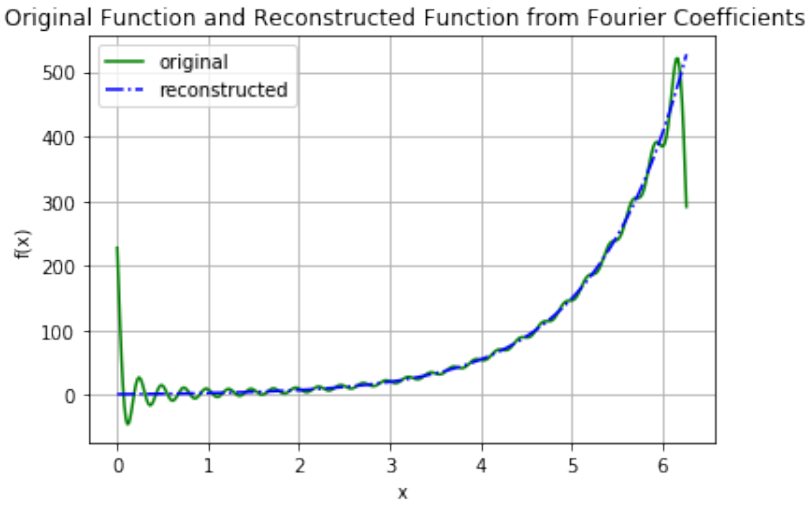

Figure 10: Original Function and Reconstructed Function from Fourier Coefficients

4.3 Error Calculation

The mean error is then calculated executing the code:

```
print("Mean Error in f(x) = exp(x) is {}".format(np.mean(abs(c1 - F))))
print("Mean Error in g(x) = cos(cos(x)) is {}".format(np.mean(abs(c2 - G))))
```

The mean error in $f(x)$ is 0.68868. The mean error in $g(x)$ is 5.359481e-16.
The maximum deviation is the $f(x) = e^x$ occurs in the a_1 whereas the maximum deviation in $g(x) = cos(cos(x))$ occurs in a_{25}.

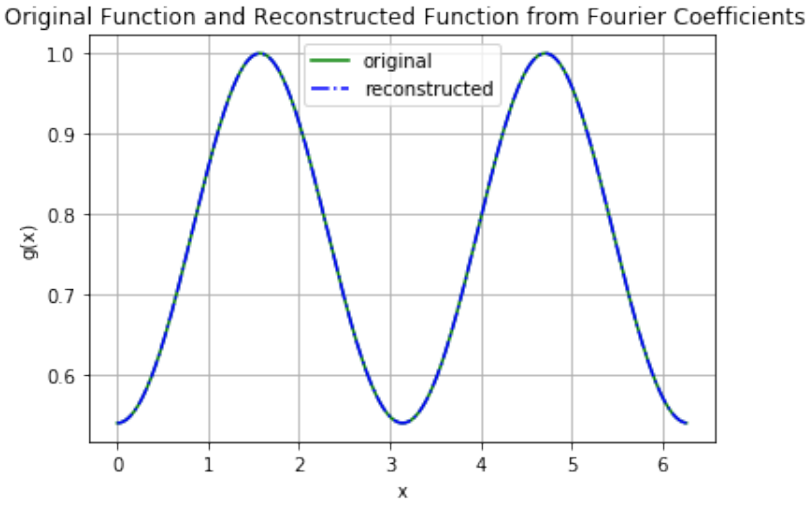

Figure 11: Original Function and Reconstructed Function from Fourier Coefficients

4.4 Inline Question

4.4.1 Question 4

The function $f(x) = e^x$ is neither periodic nor sinusoidal , hence the Fourier expansion upto 51 terms shows large deviation as compared to $g(x) = cos(cos(x))$ which periodic as well as sinusoidal. This is precisely the reason we find 10^{15} times error in the first case compared to the second. If we calculate the Fourier series upto more number of terms, then the deviation will decrease.

5 Conclusion

In this assignment, we calculated the Fourier Expansion of two functions $f(x) = e^x$ and $g(x) = cos(cos(x))$ and calculated the deviation the reconstructed function from the actual function.