1 Introduction

The goal of this lab is to use Fourier Series to approximate periodic time-domain signals.

2 Equations

For this lab, we are given the following functions:

$$x(t) = \frac{1}{2}a_0 + \sum_{k=1}^{\infty} a_k \cos(kw_0 t) + b_k \sin(kw_0 t) \tag{1}$$

$$a_k = \frac{2}{T}\int_0^T x(t)\cos(kw_0 t)dt \tag{2}$$

$$b_k = \frac{2}{T}\int_0^T x(t)\sin(kw_0 t)dt \tag{3}$$

3 Methodology and Results

This lab consists of two tasks:

- Task 1:
- Input the expressions for ak and bk into Spyder. Use Python to solve for a0, a1, b1, b2, and b3 and display a numerical value for each.:

```

#%% part 1 task 1

T = 8
w = 2*np.pi/T
steps = 1e-3
t = np.arange(0, 20, steps)


ak = []
bk = []

a0 = (1/T)*(T-T)
ak.append(a0)
for k in range(1, 2):
a = 2/(k*w*T)*(2*np.sin(k*w*T/2) - np.sin(k*w*T))
ak.append(a)

for k in range(1, 4):
b = 2/(k*w*T)*(-2*np.cos(k*w*T/2) + np.cos(k*w*T) + 1)
bk.append(b)

print("ak: ", ak)
```

```
    print("bk: ", bk)


```

- The output of this code is the following:

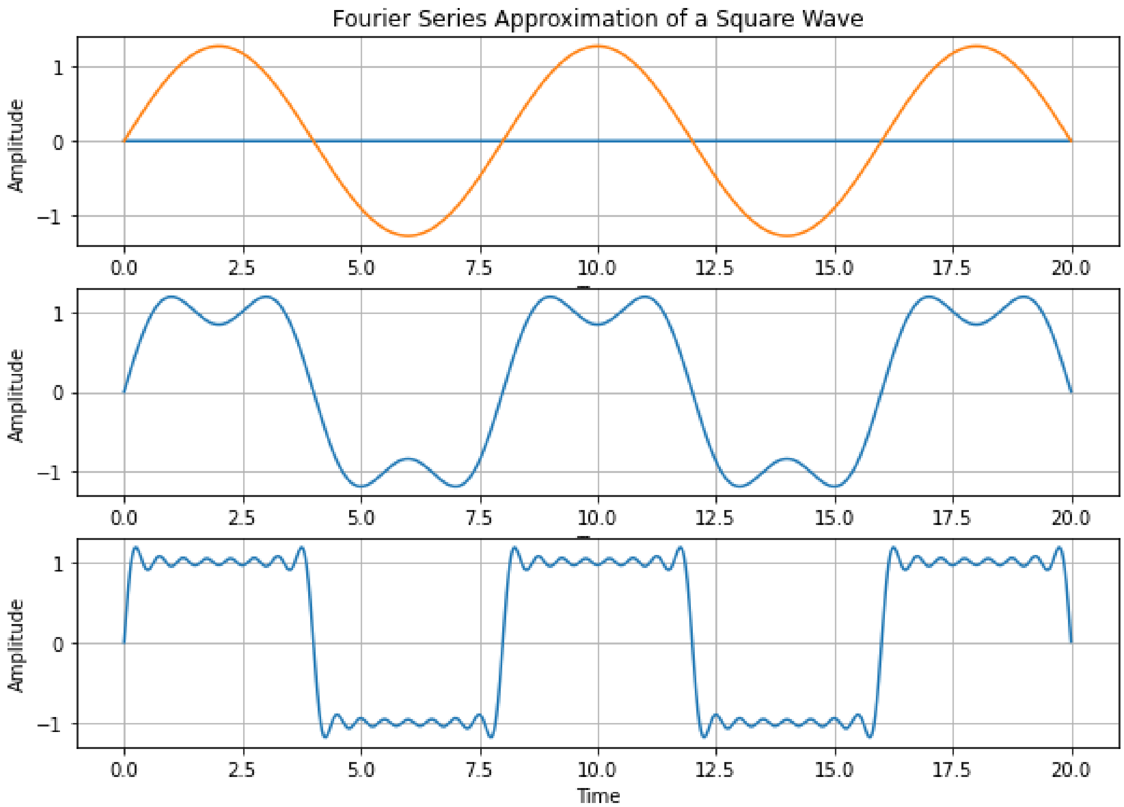
```

    ak:  [0.0, 1.5592687330077502e-16]
    bk:  [1.2732395447351628, 0.0, 0.4244131815783876]


```

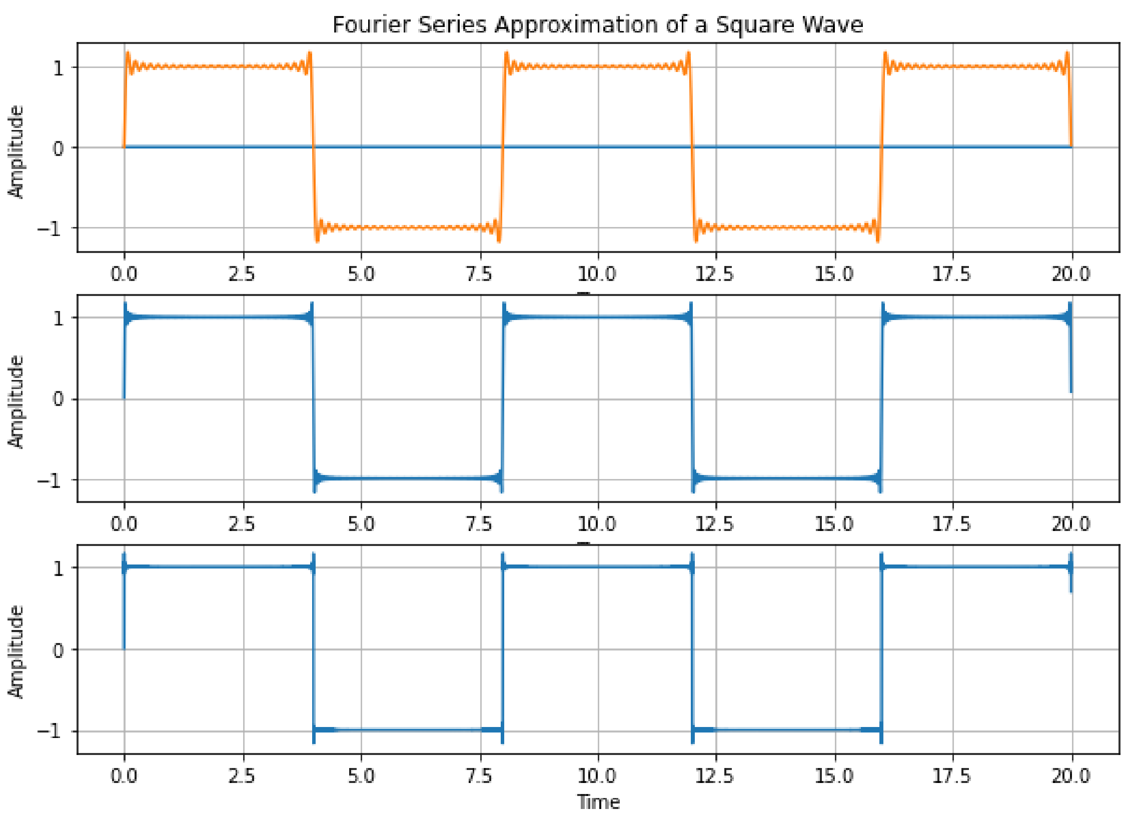
We can see that a0 and a1 are both zero (a1 is extemely small we can consider it as zero).

- Task 2: plot the Fourier series approximation for N = 1; 3; 15; 50; 150; 1500:

```

    #%% part 1 task 2

    N = [1, 3, 15, 50, 150, 1500]

    func = []
    f = np.zeros(len(t))

    for n in N:
    f = np.zeros(len(t))
    print(n)
    if n == 1 or n == 50:
    plt.figure(figsize=(10,7))
    if n == 1 or n == 50:
    plt.subplot(3, 1, 1)
    plt.title ('Fourier Series Approximation of a Square Wave')
    plt.plot(t,f)
    if n == 3 or n == 150:
    plt.subplot(3, 1, 2)

    if n == 15 or n == 1500:
    plt.subplot(3, 1, 3)

    for k in range(1, n+1):
    b = 2/(k*w*T)*(-2*np.cos(k*w*T/2) + np.cos(k*w*T) + 1)
    a = 2/(k*w*T)*(2*np.sin(k*w*T/2) - np.sin(k*w*T))
    f = f+b*np.sin(w*k*t)+a*np.cos(w*k*t)
    plt.ylabel("Amplitude")
    plt.xlabel("Time")
    plt.plot(t, f)
    plt.grid()
    plt.show()


```

- The outputs of fourier wave approximation for different values of N are given below:


Fourier Series Approximation of a Square Wave
Amplitude
1
0
−1
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Time


Fourier Series Approximation of a Square Wave
Amplitude
1
0
−1
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Time

4 Questions

1. Is x(t) an even or an odd function? Explain why
   The function is an odd function because it consists of sine waves.

2. Based on your results from Task 1, what do you expect the values of a2, a3, . . . , an to be? Why?

   The rest of the values of ak will also be zeros this is because all the values of ak depend on sine of pi with an integer multiple. Since all the integer values of sine pi are zero therefore all the values of ak are zero.

3. How does the approximation of the square wave change as the value of N increases? In what way does the Fourier series struggle to approximate the square wave?
   As the values of N increases the approximation gets better and better this is because more and more sinusoids are summed together to form a square wave and the sinusoids being summed are all of varying frequencies, phases and amplitudes therefore we get a more refined square wave as we increase the value of N.

4. What is occurring mathematically in the Fourier series summation as the value of N increases?

   As the value of N increases, different sinusoids are added together and the greater the value of N the greater the number of sinusoids. And as more and more sinusoids are added the final wave will look more and more similar to a square wave. The different sinusoids will have varying phase, amplitude and frequencies and adding them together will cause the waves to interfere constructively and destructively and hence the final output will be more square wave like.

5. Leave any feedback on the clarity of lab tasks, expectations, and deliverables.

5 Conclusion

In this lab, we learned how to use Fourier series to approximate periodic time domain signals in python.