Student: Garrett Nadauld

Course: ECE 3210

Subject: Lab 4, Fourier Series Reconstruction

Date: October 13, 2023

1 Introduction

In this lab, we explore Fourier series reconstruction, a vital tool for breaking down complex time-domain signals into a sum of sinusoidal components. We begin by deriving the trigonometric Fourier series representation for a triangle wave. This theoretical foundation serves as the basis for our practical Python implementation, which not only reconstructs signals but also quantifies the accuracy of the approximation. We can then translate our Python implementation into a C program.

As we calculate error signals and measure their power, we discover how the quality of the reconstructed signal improves with the inclusion of more harmonics ('m'). By carefully selecting a discrete sampling step size, we ensure the accuracy of our signal representation. Additionally, we delve into C programming to assess the computational efficiency of both Python and C implementations. This lab bridges theory and practice, equipping us with essential skills for signal analysis in electrical engineering.

2 Theory

The Fourier series is a powerful mathematical tool used to represent periodic time-domain signals. It provides a means to decompose a complex signal $f(t)$ into simpler components, primarily sine and cosine functions. The fundamental representation of a periodic signal with period T_0 is given by:

$$f(t) = a_0 + \sum_{n=1}^{\infty} a_n \cos(n\omega_0 t) + b_n \sin(n\omega_0 t) \tag{1}$$

$$a_0 = \frac{1}{T_0}\int_{T_0} f(t)dt \tag{2}$$

$$a_n = \frac{2}{T_0}\int_{T_0} f(t)\cos(n\omega_0 t)dt \tag{3}$$

$$b_n = \frac{2}{T_0}\int_{T_0} f(t)\sin(n\omega_0 t)dt \tag{4}$$

Given Fig. 1 we solved for the coefficients.

$$a_0 = 0 \tag{5}$$

$$a_n = 0 \tag{6}$$

$$b_n = \frac{\cos\left(\frac{3}{2}\pi n\right) - \cos\left(\frac{1}{2}\pi n\right)}{\pi n} + \frac{6\sin\left(\frac{1}{2}\pi n\right) - 2\sin\left(\frac{3}{2}\pi n\right)}{n^2\pi^2} \tag{7}$$

3 Results

Fig. 2 displays the Fourier series reconstruction using the trigonometric Fourier representation. As the number of harmonics ('m') increases, the reconstructed signal converges closer to the actual function, illustrating the trend of enhanced accuracy. Fig. 3 depicts the diminishing error waveform with increasing 'm,' showcasing improved precision in the reconstruction.

Additionally, Fig. 4 provides a performance comparison between Python and C implementations, with the C implementation showing notably faster execution times. Finally, Table **??** quantifies the power of the error waveform for various 'm' values, offering insights into its spectral properties. These results collectively provide insights into the quality of Fourier series reconstructions and the efficiency of the implemented algorithms.

4 Discussion and Conclusions

The data and analysis presented in this lab reveal several noteworthy insights into the Fourier series reconstruction of time-domain signals, demonstrating the convergence of the representation as the number of harmonics (m) increases.

Fig. 2 showcases the reconstruction of f(t) using the Fourier series with varying numbers of harmonics. It is evident that as the number of harmonics increases, the representation of f(t) becomes increasingly accurate. This aligns with the theoretical understanding that a higher number of harmonics in the Fourier series allows for a more detailed description of the original signal. This idea is reinforced in Fig. 3 where the amplitude of e(t) decreases as the number of harmonics increases.

Fig. 4 presents the timing comparison between the C and Python implementations on a logarithmic scale. The results reveal that the C program consistently outperforms the Python program for all values of m. This performance disparity aligns with the expectations regarding the computational efficiency of C over Python and highlights the practical implications of choosing an efficient programming language for signal processing tasks.

Table 1 quantifies the power of the error function for each value of m. As expected, it demonstrates that as m increases, the power of the error function decreases rapidly. This reduction in error power reflects the improved accuracy in the Fourier series reconstruction as more harmonics are incorporated.

In conclusion, the data and analysis confirm the theoretical concepts underpinning Fourier series reconstruction. The trend towards more accurate representation with increasing harmonics, the decreasing error signal amplitude, and the superior computational performance of C over Python all align with the principles learned in class.

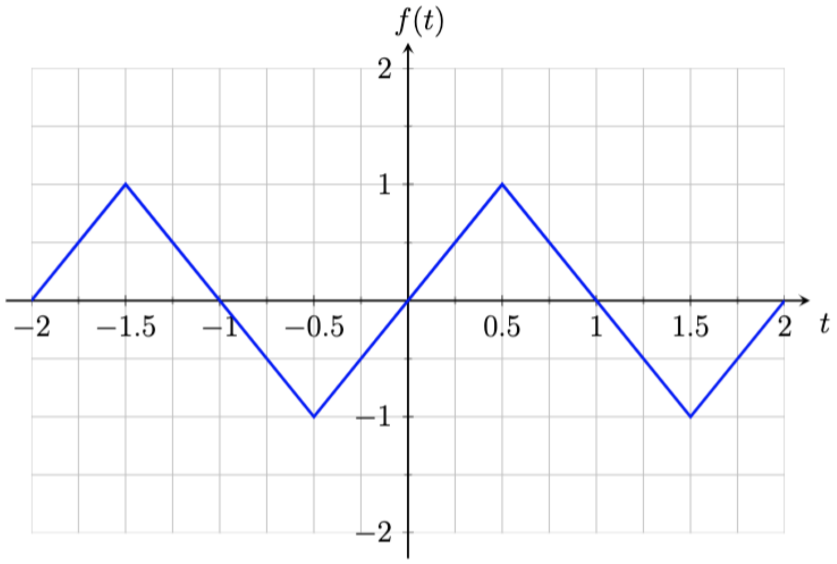

Figure 1: This figure depicts f(t) which is the signal that we were tasked with reconstructing using the trigonometric Fourier series

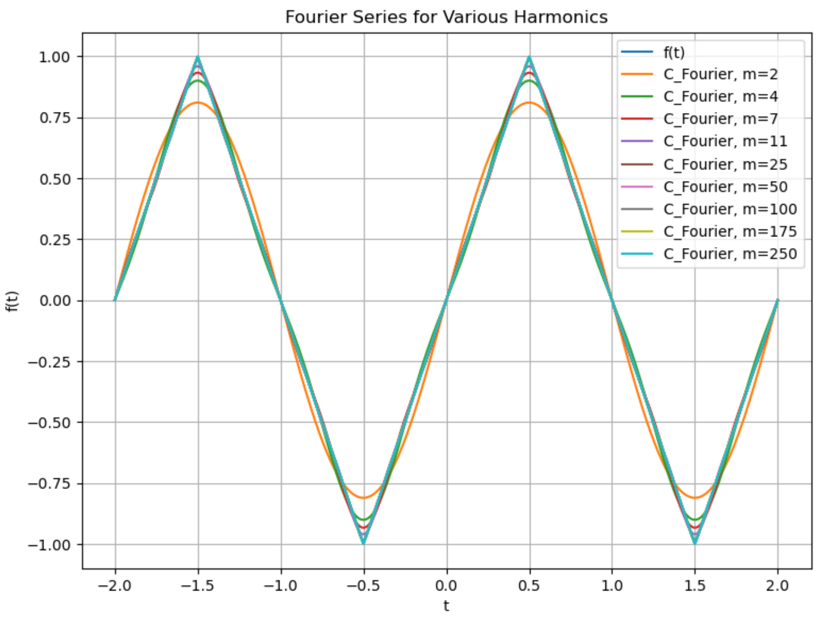

Figure 2: This figure depicts the reconstructed signal using trigonometric Fourier series for each number of harmonics m.

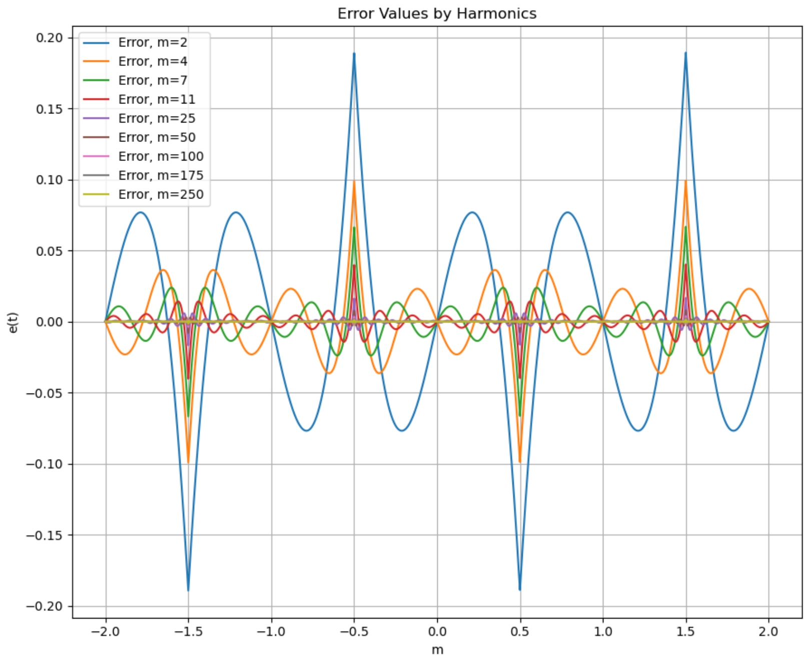

Figure 3: This plot shows the error signal e(t) which is obtained through $f_m(t) - f(t)$ for each value m ranging from 2 to 250. As m increases the amplitude of e(t) decreases. e(t) is obtained by subtracting f(t) from the reconstructed signal.

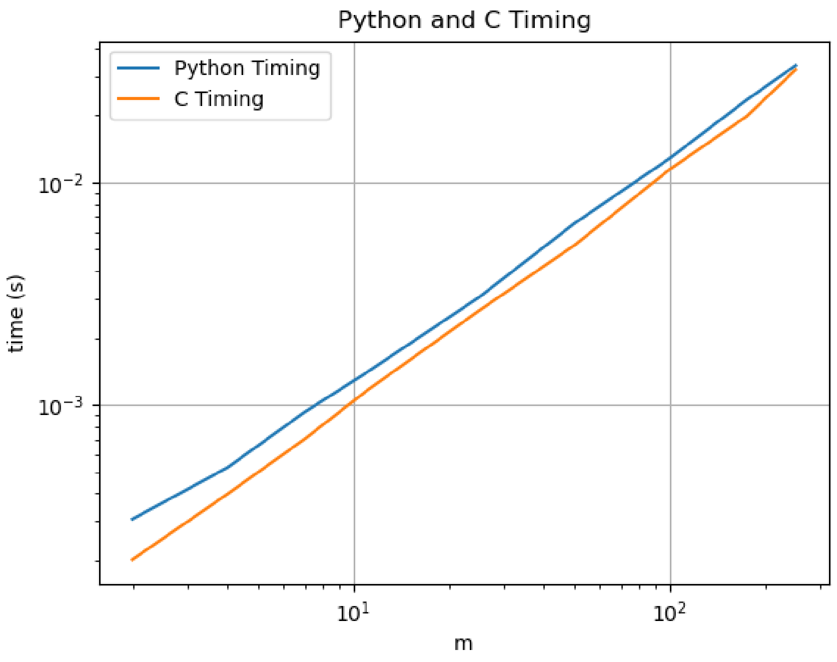

Figure 4: Shown is the timing of the Python and C implementations. This is plotted on a logarithmic scale.

m	P_e
2	9.64379731e-03
4	1.53240633e-03
7	4.81170612e-04
11	1.07385153e-04
25	7.89349336e-06
50	8.75114449e-07
100	1.09373839e-07
175	2.07230609e-08
250	6.96486564e-09

Table 1: This table shows the power of the error signal corresponding to each value of m.