# Digital Communication Lab

Signal Sampling and Fourier Transform Analysis

**Name: Sachin Kumar**
**Roll No: 220104026**
**Section: D**
**Lab No. : 1**

October 19, 2024

## 1 Objective

To sample a sinusoidal signal using an impulse train and analyze the characteristics of the sampled signal.

## 2 Introduction

This document demonstrates the simulation of signal sampling using an impulse train and a sinusoidal message signal, followed by frequency analysis using the Fourier Transform. Below is the Python code used to perform the simulation and the resulting plots showing both the time domain and frequency domain representations of the signals.

## 3 Theory

In this experiment, we investigate the behavior of a continuous-time sinusoidal signal represented in the time domain. The continuous-time message signal is defined as:

$$m(t) = \cos(2\pi f_m t)$$

where $f_m$ is the frequency of the message signal. For this experiment, we set $f_m = 5\,Hz$. This cosine function represents a periodic waveform that oscillates between -1 and 1, completing one full cycle every:

$$\frac{1}{f_m} = 0.2\,s.$$

To convert this continuous signal into a discrete signal, we utilize an impulse train. The impulse train can be expressed mathematically as:

$$g(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT)$$

where $T$ is the sampling period, given by:

$$T = \frac{1}{F_s}$$

and $F_s$ is the sampling frequency. In our experiment, we select a sampling frequency of $F_s = 100\,Hz$, resulting in a sampling period of:

$$T_s = \frac{1}{100} = 0.01\,s.$$

The time vector is constructed as:

$$t = 0 : T_s : 1\,s$$

which provides discrete time instances at which the message signal is sampled. The sampled signal $M_d(t)$ is obtained by multiplying the message signal with the impulse train:

$$M_d(t) = m(t) \cdot g(t).$$

This operation results in a discrete representation of the continuous signal at specified intervals, allowing for analysis and processing in digital systems.

The time-domain representation provides insights into the waveform characteristics, including amplitude, frequency, and phase, and is essential for understanding how the signal behaves over time.

# 4 Python Code

```
import numpy as np
import matplotlib.pyplot as plt

# Parameters
f = 100
t_max = 0.4
interval = 0.01

# Time vector
t = np.arange(0, t_max + interval, interval)

# Impulse train
train = np.ones_like(t)

# Plot impulse train
plt.figure(figsize=(10, 8))

plt.subplot(3, 2, 1)
plt.stem(t, train)
plt.title('Impulse train with frequency 100 Hz')
plt.xlabel('Time (seconds)')
plt.ylabel('Amplitude')
N = len(t)

# Fourier Transform of the impulse train
train_fft = np.fft.fft(train/N)
train_freq = np.fft.fftfreq(train.size, d=interval)

plt.subplot(3, 2, 2)
plt.stem(train_freq, np.abs(train_fft))
plt.title('FFT of Impulse Train')
plt.xlabel('Frequency (Hz)')
plt.ylabel('Amplitude')

# Sinusoidal message signal
fm = 5
m = np.sin(2 * np.pi * fm * t)

# Plot sinusoidal message signal
plt.subplot(3, 2, 3)
plt.plot(t, m)
plt.title('Sinusoidal message signal')
```

```
plt.xlabel('Time (seconds)')
plt.ylabel('Amplitude')

# Fourier Transform of the sinusoidal message signal
m_fft = f * (np.fft.fft(m/N))
m_freq = np.fft.fftfreq(m.size, d=interval)

plt.subplot(3, 2, 4)
plt.stem(m_freq, np.abs(m_fft))
plt.title('FFT of Sinusoidal Message Signal')
plt.xlabel('Frequency (Hz)')
plt.ylabel('Amplitude')

# Sampled signal
s = m * train

# Plot sampled signal
plt.subplot(3, 2, 5)
plt.stem(t, s)
plt.title('Sampled signal m(delta)(t)')
plt.xlabel('Time (seconds)')
plt.ylabel('Amplitude')

# Fourier Transform of the sampled signal
s_fft = f * (np.fft.fft(s/N))
s_freq = np.fft.fftfreq(s.size, d=interval)

plt.subplot(3, 2, 6)
plt.stem(s_freq, np.abs(s_fft))
plt.title('FFT of Sampled Signal')
plt.xlabel('Frequency (Hz)')
plt.ylabel('Amplitude')

# Show plots
plt.tight_layout()
plt.show()
```

# 5 Results

The figure below shows the generated plots: the impulse train, the sinusoidal message signal, and their respective Fourier Transforms. The sampled signal and its Fourier Transform are also presented.

# 6 Conclusion

In conclusion, the experiment successfully demonstrated the process of sampling a sinusoidal signal using an impulse train. We verified that the sampling frequency was adequate for accurately reconstructing the original signal, adhering to the Nyquist theorem. Future experiments could explore different signal types and sampling frequencies to further understand the effects on signal fidelity.

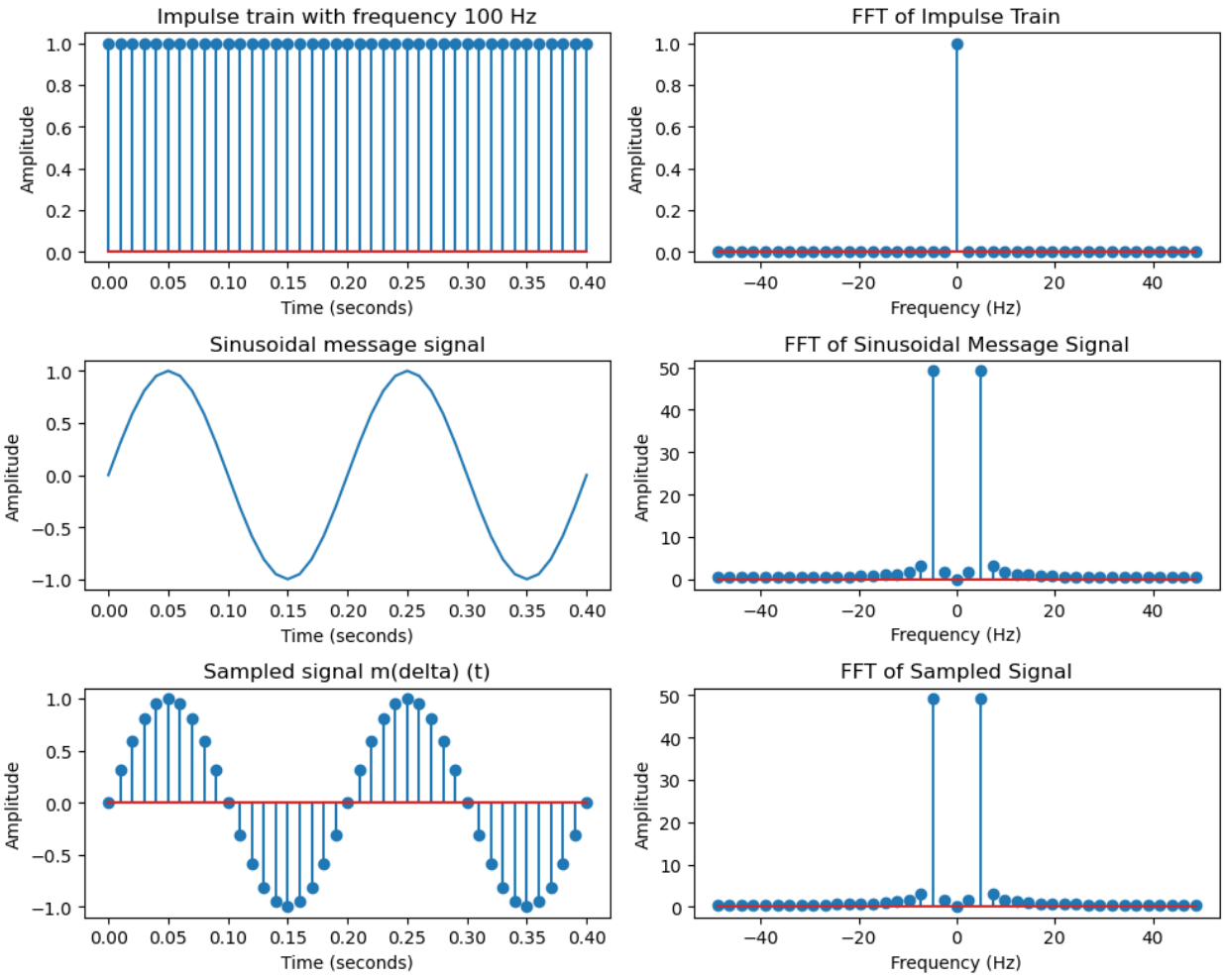


Figure 1: Impulse Train, Message Signal, and Sampled Signal in Time and Frequency Domains