Toronto Metropolitan University

Department of Electrical, Computer, & Biomedical Engineering
Faculty of Engineering & Architectural Science

Course Number	ELE 532
Course Title	Signals and Systems I
Semester/Year	Fall 2025
Instructor	Beheshti, Soosan
TA Name	Punya Cheema

Lab/Tutorial Report No.	03
Report Title	Fourier Series Analysis using Matlab

Submission Date	2025.11.02
Due Date	Nov 9, 2025

Student Name	Student ID	Signature
Vinci Fajardo	501239903	VF
Keunhyeok Choi	500958125	KH

1. Objective

The objective of this experiment was to analyze and synthesize periodic signals using the Fourier series representation. Key goals included learning to compute Fourier series coefficients (Dn) in MATLAB, visualizing the magnitude and phase spectra of different signals, and investigating the effects of series truncation on the quality of signal reconstruction. A central part of this investigation involved observing and understanding the Gibbs phenomenon in the reconstruction of signals with discontinuities.

2. Methodology

This experiment was performed using a single, comprehensive MATLAB script. The core of the methodology revolved around the Fourier series synthesis and analysis equations:

- Synthesis: $x(t) = \sum_{n=-\infty}^{+\infty} D_n e^{jn\omega_0 t}$
- Analysis: $D_n = \frac{1}{T_0}\int_{T_0} x(t)e^{-jn\omega_0 t}dt$

The process was implemented through two main custom functions within the MATLAB script:

calculate_Dn(n, signal_type): This function served as the implementation of the analysis equation. Based on the signal_type ('x1', 'x2', or 'x3'), it calculated the analytical Fourier series coefficients for a given range of harmonic indices n. For x1(t), the coefficients were discrete and assigned directly. For the rectangular pulse trains x2(t) and x3(t), the coefficients were calculated using the sinc function formula derived from the analysis integral.

reconstruct_signal(Dn, n, signal_type, t): This function implemented the synthesis equation. It took a truncated set of Fourier coefficients (Dn) and reconstructed the time-domain signal x(t) by summing the complex exponential terms over the specified range of n.

The experiment was divided into two main parts: spectral analysis and signal reconstruction. For analysis (Problem A.4), the stem command was used to plot the magnitude spectra. For reconstruction (Problem A.6), the plot command was used to visualize the synthesized waveforms. All results were organized into a tabbed figure window for clear comparison.

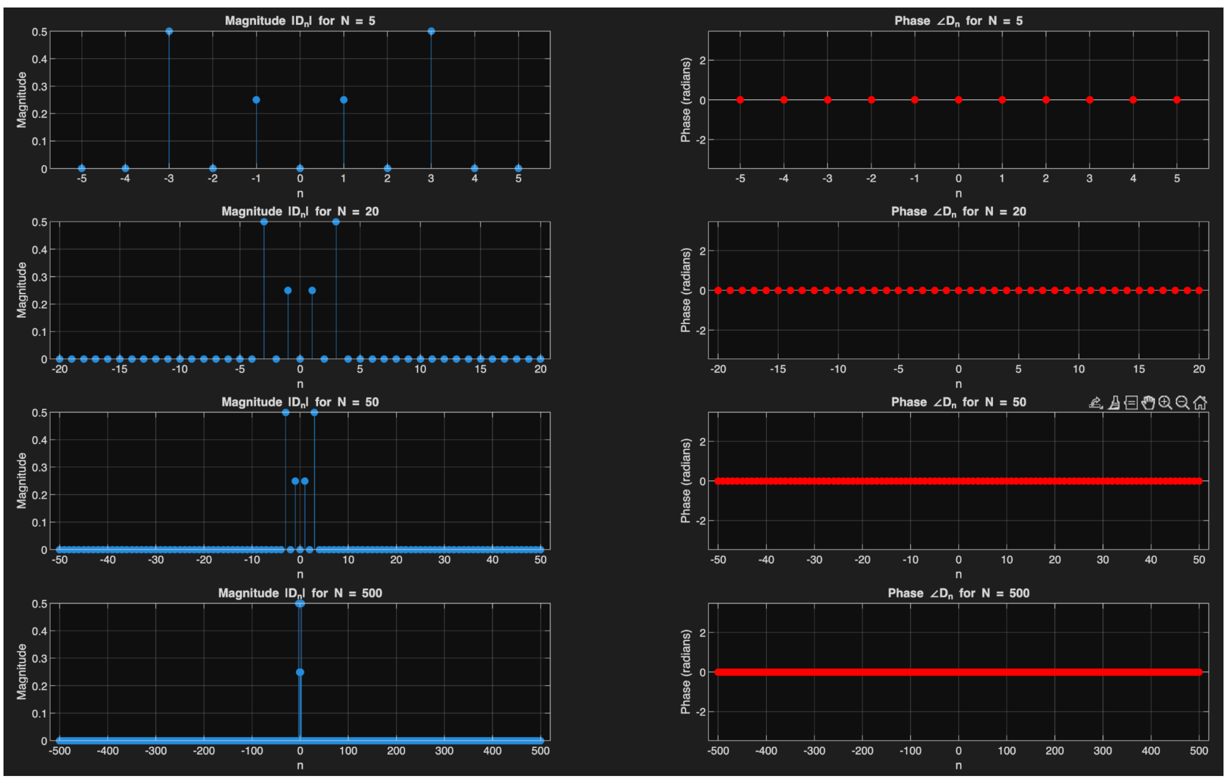
3. Results and Analysis

Part A.4: Spectral Analysis

The magnitude spectra for signals x1(t), x2(t), and x3(t) were generated using an increasing number of coefficients (N = 5, 20, 50, 500).

Signal x1(t) - Sum of Cosines:

The signal x1(t) is composed of only two frequency components. Its spectrum is therefore discrete and contains a finite number of non-zero coefficients.

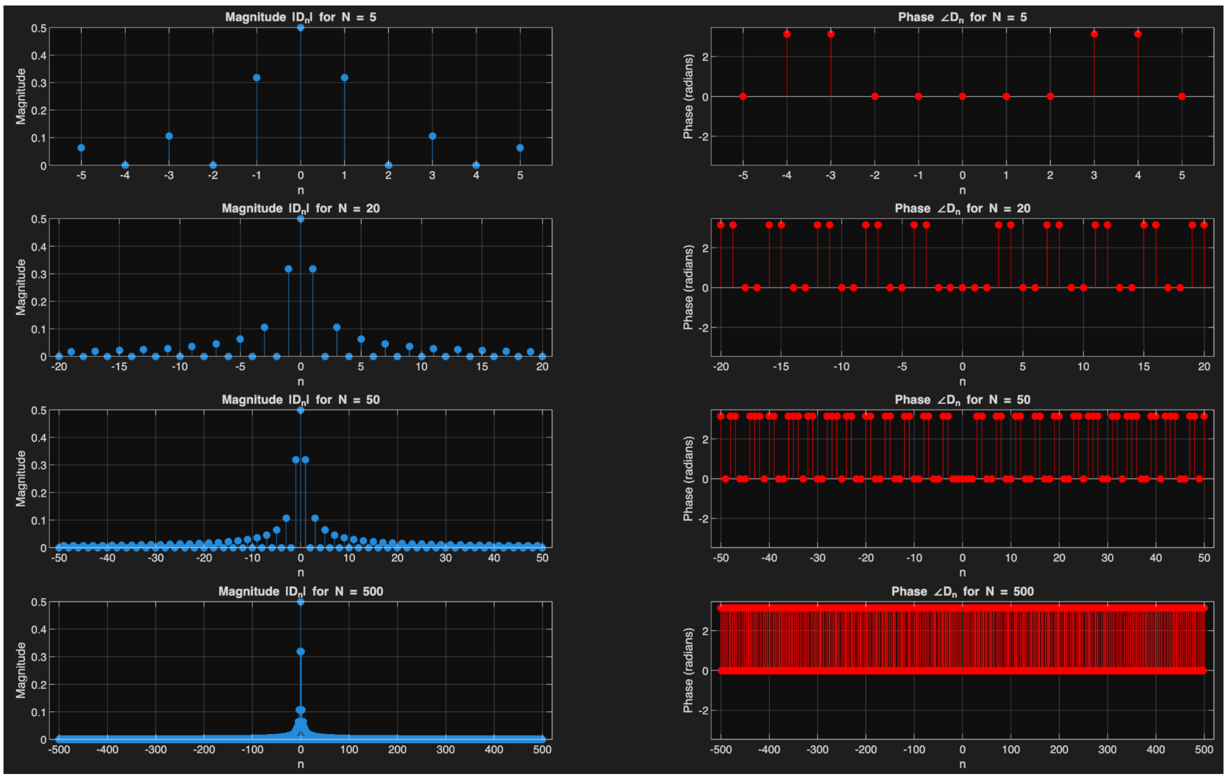


[Figure 1: Magnitude and Phase spectra for x1(t).]

Analysis: As shown in Figure 1, the magnitude spectrum for x1(t) has only four non-zero components at n = ±1 and n = ±3. This is expected, as Euler's identity expands the two cosines into four complex exponentials. The spectrum is sparse. The phase spectrum is zero for all components. This is because the Fourier coefficients (Dn) for this signal are purely real and positive.

Signal x2(t) - Rectangular Pulse Train (T_0=20):

x2(t) is a periodic square wave, which contains an infinite number of harmonics.

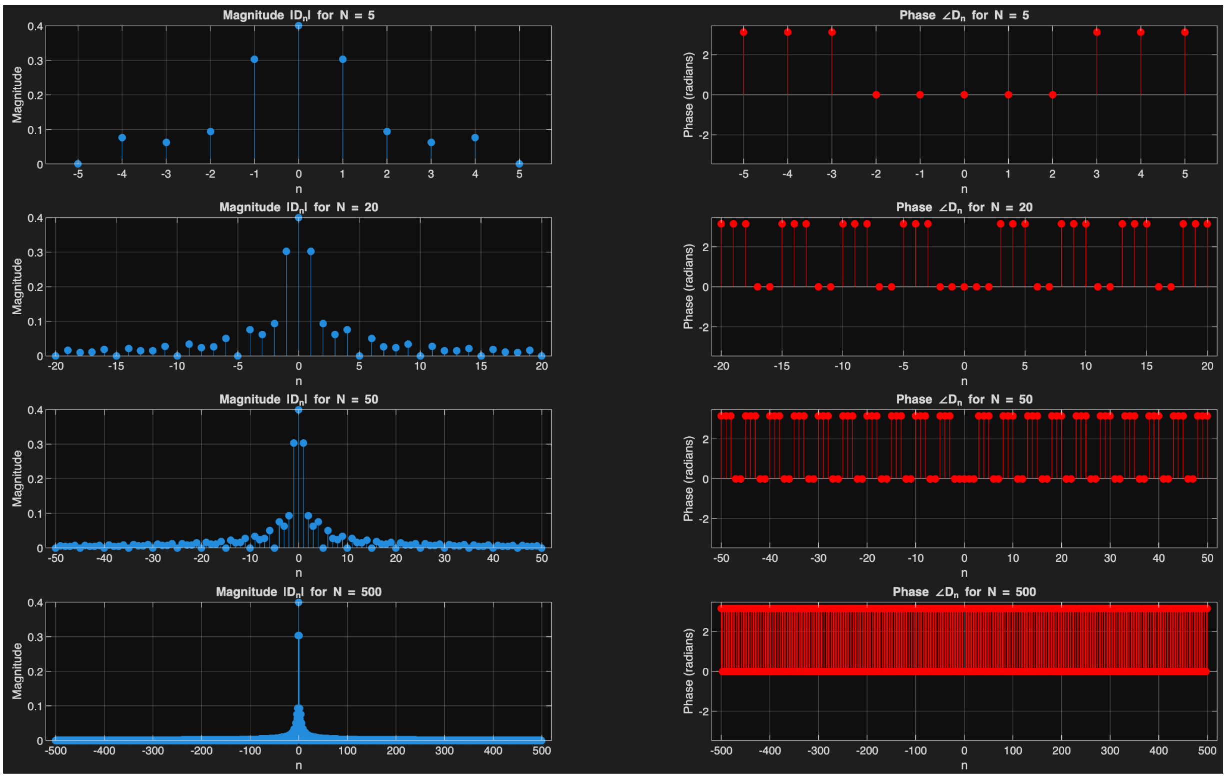


[Figure 2: Magnitude and Phase spectra for x2(t).]

Analysis: Figure 2 shows that the magnitude spectrum of x2(t) has the characteristic shape of a sinc function, with nulls at all even harmonics. The phase spectrum shows a pattern of alternating between 0 and $\pm\pi$ radians ($\pm 180°$). The phase is 0 when the sinc function is positive and flips to $\pm\pi$ when the sinc function becomes negative. This phase flip accounts for the sign change in the Fourier coefficients.

Signal x3(t) - Rectangular Pulse Train (T_0=25):

x3(t) has the same pulse shape as x2(t) but a longer period.



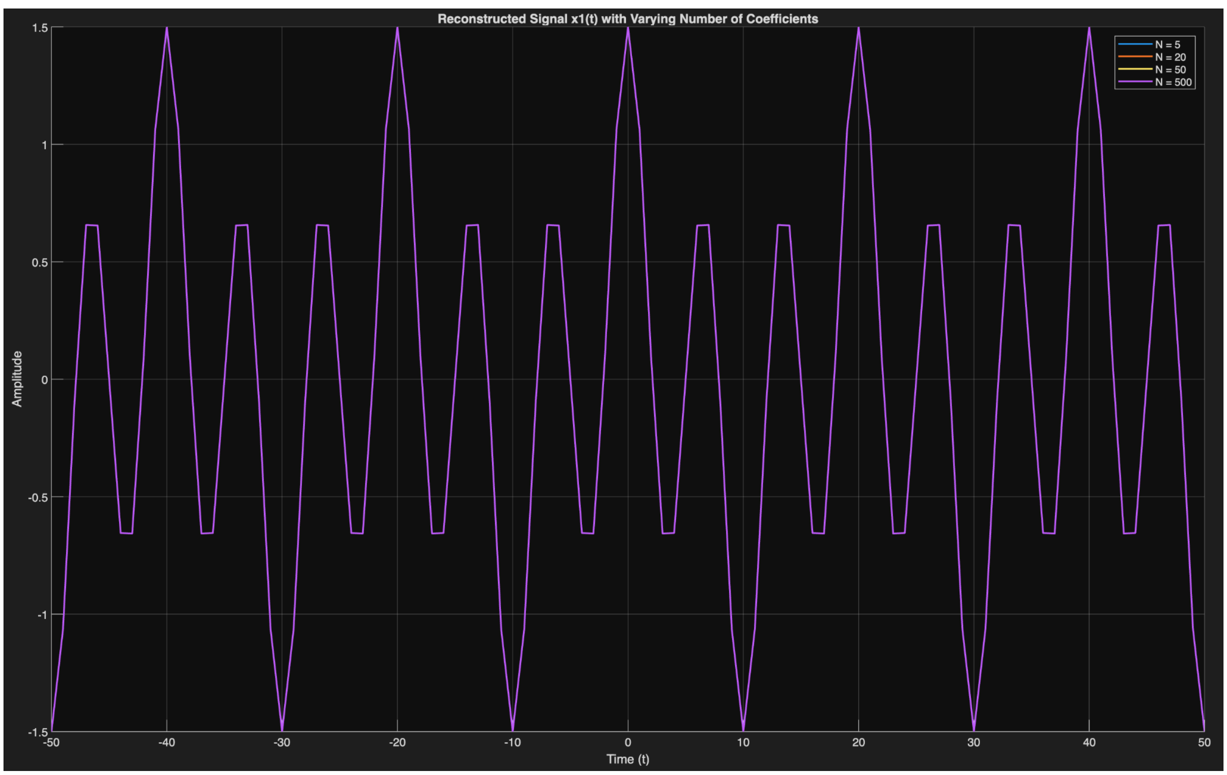
[Figure 3: Magnitude and Phase spectra for x3(t).]

Analysis: The spectrum of x3(t) also follows a sinc envelope, but the components are more densely packed. Similarly to x2(t), its phase spectrum alternates between 0 and ±π corresponding to the positive and negative lobes of its sinc function.

Part A.6: Signal Reconstruction

The signals were reconstructed using the coefficients generated in the previous part.

Reconstruction of x1(t):



[Figure 4: Reconstructed signal x1(t) with varying N. This is the screenshot of the 'x1 Reconstruction' tab.]

Analysis: As seen in Figure 4, x1(t) is perfectly reconstructed with N=3 or higher (as N=5 is the first case shown). Since the signal is composed of a finite number of harmonics, adding more coefficients beyond N=3 does not change the reconstructed signal.

Reconstruction of x2(t):

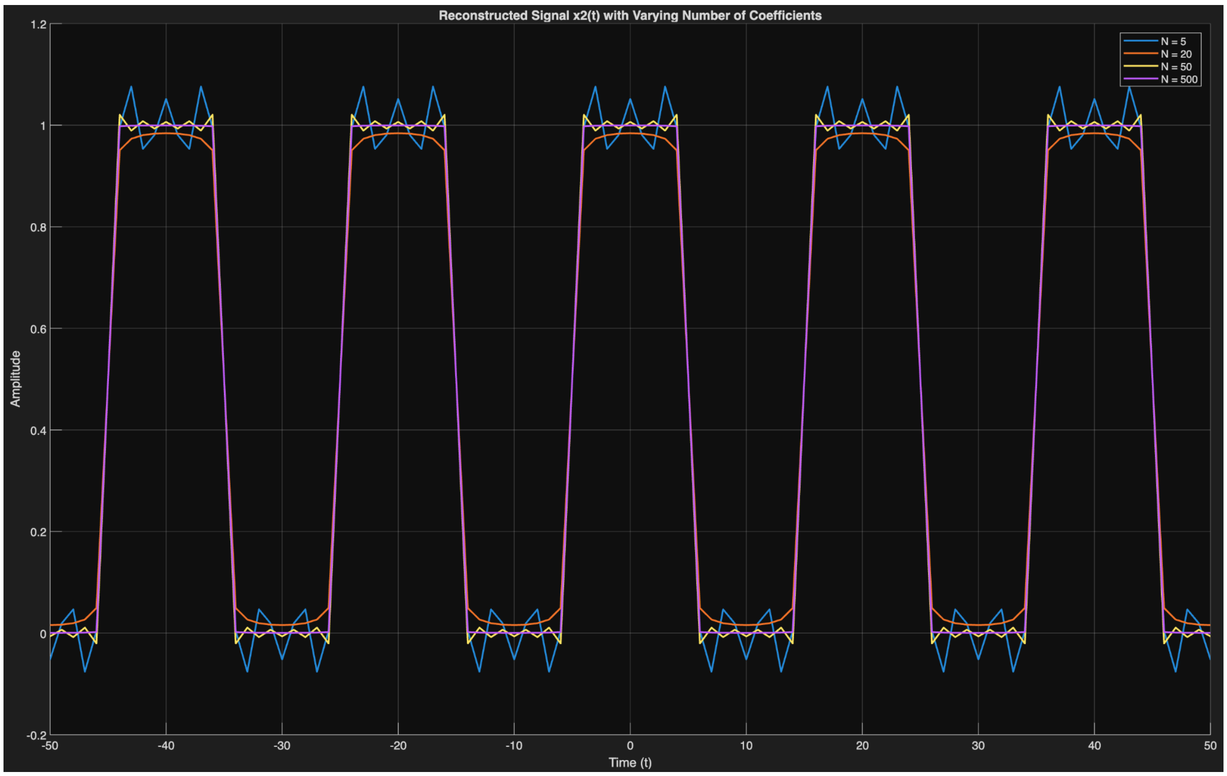


Figure 5: Reconstructed signal x2(t) showing Gibbs Phenomenon. This is the screenshot of the 'x2 Reconstruction' tab.]

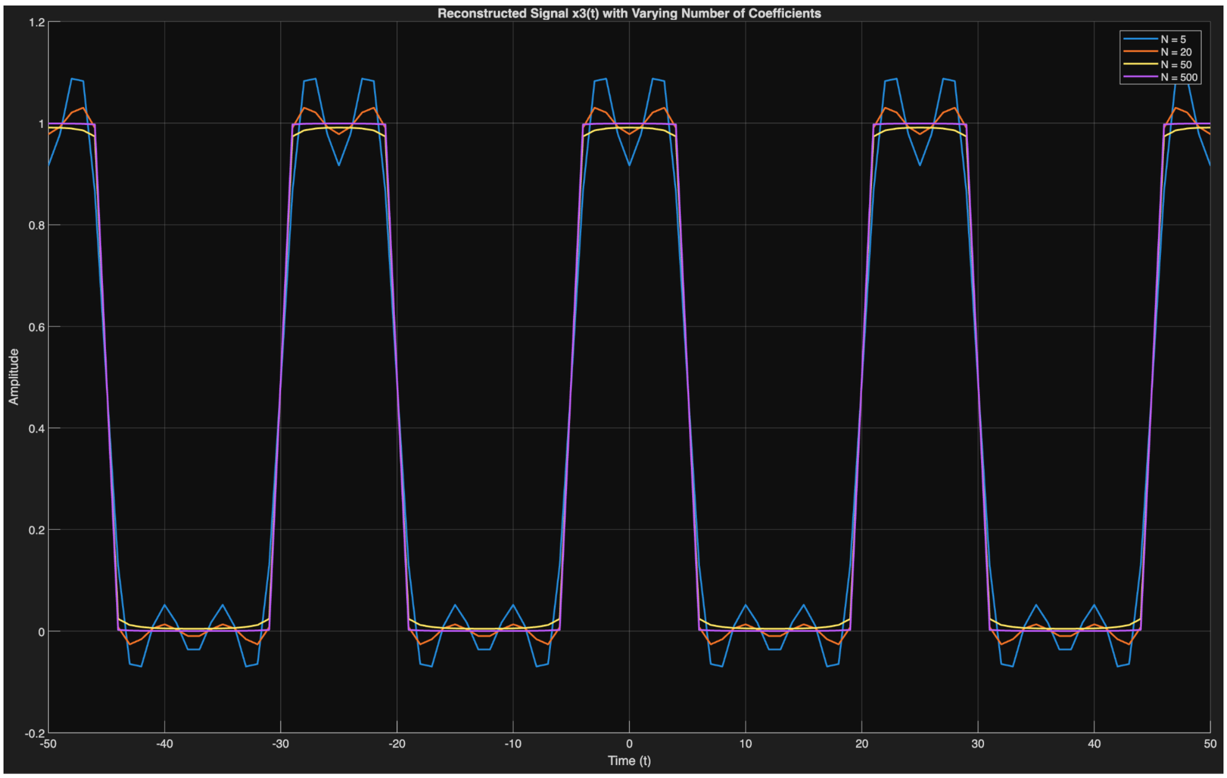
Analysis: Figure 5 provides a clear illustration of signal reconstruction and the Gibbs phenomenon.

With N=5, the reconstruction is a poor, wave-like approximation.

As N increases to 20 and 50, the signal begins to resemble a square wave more closely, with sharper edges.

With N=500, the approximation is very close to the ideal square wave. However, at the points of discontinuity (the corners), distinct overshoots and ringing are visible. This is the Gibbs phenomenon, an artifact of truncating the Fourier series for a signal with a jump discontinuity.

Reconstruction of x3(t):



[Figure 6: Reconstructed signal x3(t). This is the screenshot of the 'x3 Reconstruction' tab.]

Analysis: The reconstruction of x3(t) shows the same qualitative behavior as x2(t), including the Gibbs phenomenon at the discontinuities.

4. Discussion

(Problem B.1)

The fundamental frequency of a periodic signal is determined by the reciprocal of its fundamental period T_0, that is $f_0 = \frac{1}{T_0}$, or equivalently in angular form $\omega_0 = \frac{2\pi}{T_0}$.

$x_1(t) = cos(\frac{3\pi}{10}t) + \frac{1}{2}cos(\frac{\pi}{10}t)$

$\omega_0 = \frac{\pi}{10}$

$x_2(t)$ → A periodic rectangular pulse with period $T_0 = 20$

$\omega_0 = \frac{2\pi}{20} = \frac{\pi}{10}$

$x_3(t)$ → A periodic rectangular pulse with period $T_0 = 40$

$\omega_0 = \frac{2\pi}{40} = \frac{\pi}{20}$

(Problem B.2)

The primary difference between the fourier coefficients is in the distribution and amplitude pattern. For $x_1(t)$, the signal is a sum of two cosine components. Consequently, its Fourier series consists of only four nonzero coefficients, located symmetrically at n = ± 1 and n = ± 3. Each coefficient is real-valued, corresponding to pure cosine harmonics, and all have zero phase because the signal is even and purely real.

In contrast, $x_2(t)$ being a periodic rectangular pulse, contains infinitely many harmonics. Only odd harmonics are nonzero, and their magnitudes decrease proportionally to 1/n. The alternating signs of Dn cause the phase to switch between 0 and π, reflecting the alternating polarity of the harmonic components.

(Problem B.3)

Both $x_2(t)$ and $x_3(t)$ are the same rectangular pulse, having a height of 1 and width of 10, but their periods are different, with it being 20 and 40 respectively. This changes their Fourier coefficients in two ways. All the coefficients of $x_3(t)$ are about half those of $x_2(t)$ and $x_3(t)$ has zeroes every 4th harmonic because its period is twice as long.

(Problem B.4)

From the figure, $x_4(t)$ equals +0.5 on the pulse and -0.5 elsewhere, with the same timing as $x_2(t)$. Hence $x_4(t) = x_2(t) - 0.5$

Because D_0 is the average value over a period, constants affect only D_0:

$D_0[x_4] = D_0[x_2] - 0.5$

But $D_0[x_2] = \tau/T = 10/20 = 0.5$.Therefore

$D_0[x_4] = 0.5 - 0.5 = 0$

Effect of Truncation and Gibbs Phenomenon (Problem B.5):

As the number of Fourier coefficients (N) used for reconstruction increases, the approximation of the original signal improves. For smooth signals like x1(t), the reconstruction quickly becomes perfect. For signals with discontinuities like x2(t), increasing N makes the flat parts of the signal flatter and the transitions steeper. However, it does not eliminate the overshoot at the discontinuity. The Gibbs phenomenon dictates that this overshoot will always be present and will be approximately 9% of the jump height, regardless of how many coefficients are used. Increasing N only narrows the width of the overshoot region.

Perfect Reconstruction (Problem B.6):

The number of Fourier coefficients required for perfect reconstruction depends on whether the signal is band-limited.

- $x1(t)$ is band-limited; its frequency content does not extend beyond $n=\pm3$. Therefore, it requires only a finite number of coefficients (specifically, 4 non-zero Dn values) for perfect reconstruction.
- $x2(t)$ and $x3(t)$ are not band-limited; their sinc spectra extend to infinity. Therefore, they require an infinite number of coefficients for perfect reconstruction.

Storing Coefficients vs. Storing a Signal (Problem B.7):

Storing Fourier coefficients instead of time-domain samples is a viable scenario and is the basis of many compression techniques (like JPEG for images and MP3 for audio).

Advantages: It can be extremely efficient for signals that are sparse in the frequency domain (like $x1(t)$ or signals with only a few dominant frequencies). For such signals, storing a few coefficients can be far more memory-efficient than storing thousands of time-domain samples.

Disadvantages: For complex, non-sparse signals, the number of significant coefficients can be very large, potentially requiring more storage than the original samples. Furthermore, accessing the signal requires a computational step (the synthesis sum), which adds latency compared to simply reading stored samples.

5. Conclusion

This laboratory assignment provided a practical and insightful exploration of Fourier series. It was demonstrated that any periodic signal can be represented as a sum of harmonically related complex exponentials. The characteristics of a signal in the time domain (e.g., its shape, period, and continuity) were shown to have a direct and predictable impact on its spectrum in the frequency domain. The process of signal reconstruction highlighted the power of the Fourier series but also its fundamental limitations when dealing with discontinuities, as evidenced by the persistent Gibbs phenomenon.

6. Appendix: MATLAB Code

```
% ELE 532 Signals & Systems I - Laboratory Assignment 3

% Fourier Series Analysis


% --- Environment Initialization ---

clear all;

close all;

clc;
```

```
%%
======================================================================
===

%  Create Main Figure and Tab Group

%
======================================================================
====


mainFig = figure('Name', 'ELE 532 Lab 3 Results - Fourier Series Analysis', 'NumberTitle', 'off',
'WindowState', 'maximized');

tabGroup = uitabgroup('Parent', mainFig);

fprintf('Main figure window created.\n\n');


%%
======================================================================
===

%  Problem A.4: Generate and Plot Spectra

%
======================================================================
====


fprintf('Executing Problem A.4: Generating and Plotting Spectra...\n');


n_ranges = {(-5:5), (-20:20), (-50:50), (-500:500)};

signal_types = {'x1', 'x2', 'x3'};


for i = 1:length(signal_types)
```

```
    signal_name = signal_types{i};

    tab_spec = uitab('Parent', tabGroup, 'Title', [signal_name ' Spectra']);


    for j = 1:length(n_ranges)

        n_vec = n_ranges{j};

        Dn = calculate_Dn(n_vec, signal_name);


        subplot(2, 2, j, 'Parent', tab_spec);

        stem(n_vec, abs(Dn), 'filled');

        title(['Magnitude Spectrum |D_n| for N = ' num2str(max(n_vec))]);

        xlabel('n (harmonic index)');

        ylabel('Magnitude');

        grid on;

    end

end

fprintf('All spectra for x1, x2, and x3 have been plotted.\n\n');


%%
=====================================================================
===

%  Problem A.6: Reconstruct Signals from Fourier Series

%
=====================================================================
====

fprintf('Executing Problem A.6: Reconstructing Signals...\n');
```

```
t = -300:1:300;

for i = 1:length(signal_types)
    signal_name = signal_types{i};
    tab_recon = uitab('Parent', tabGroup, 'Title', [signal_name ' Reconstruction']);
    ax_recon = axes('Parent', tab_recon);

    hold(ax_recon, 'on');

    legend_entries = {};

    for j = 1:length(n_ranges)
        n_vec = n_ranges{j};
        Dn = calculate_Dn(n_vec, signal_name);
        x_reconstructed = reconstruct_signal(Dn, n_vec, signal_name, t);

        plot(ax_recon, t, real(x_reconstructed), 'LineWidth', 1.5);
        legend_entries{j} = ['N = ' num2str(max(n_vec))];
    end

    hold(ax_recon, 'off');
    title(ax_recon, ['Reconstructed Signal ' signal_name '(t) with Varying Number of
Coefficients']);
```

```
    xlabel(ax_recon, 'Time (t)');

    ylabel(ax_recon, 'Amplitude');

    legend(ax_recon, legend_entries);

    grid(ax_recon, 'on');

    xlim(ax_recon, [-50, 50]);

end

fprintf('All signals for x1, x2, and x3 have been reconstructed and plotted.\n');

fprintf('\nLab 3 script execution complete.\n');


%%
======================================================================
===

%  Helper Functions

%
======================================================================
====


function Dn = calculate_Dn(n_range, signal_type)

    Dn = zeros(size(n_range));


    switch signal_type

        case 'x1'

            Dn(n_range == 1) = 0.25;

            Dn(n_range == -1) = 0.25;

            Dn(n_range == 3) = 0.5;
```

```
            Dn(n_range == -3) = 0.5;

        case 'x2'
            Dn(n_range == 0) = 0.5;
            n_nonzero = n_range(n_range ~= 0);
            Dn(n_range ~= 0) = 0.5 * sin(pi * n_nonzero / 2) ./ (pi * n_nonzero / 2);

        case 'x3'
            Dn(n_range == 0) = 0.4;
            n_nonzero = n_range(n_range ~= 0);
            Dn(n_range ~= 0) = 0.4 * sin(pi * 2 * n_nonzero / 5) ./ (pi * 2 * n_nonzero / 5);
    end
end

function x_rec = reconstruct_signal(Dn, n_range, signal_type, t)
    if strcmp(signal_type, 'x1') || strcmp(signal_type, 'x2')
        w0 = pi/10;
    else % signal_type == 'x3'
        w0 = 2*pi/25;
    end

    x_rec = zeros(size(t));
```

```
    for i = 1:length(n_range)
        n = n_range(i);
        D_n = Dn(i);
        x_rec = x_rec + D_n * exp(1j * n * w0 * t);
    end
end
```