Matlab CA1

First Part :

دانشکده فنی دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

پروژه اول درس ریاضیات مهندسی

طراحان

محمدامین کشمیری

سروش مس‌فروش

مقدمه

هدف از این تمرین آشنایی مقدمتی دانشجویان با محیط متلب و آنالیز فوریه می باشد.
این تمرین در سه بخش طراحی شده است.

- آشنایی با متلب

بخش اول غالبا شامل مفاهیم اولیه کار در محیط متلب شامل کار با ماتریس ها و رسم چند نمودار ساده می باشد.

- سری فوریه

در این قسمت شما باید تابعی بنویسید که سری فوریه تابع دلخواه را محاسبه نماید و در ادامه به رسم سری فوریه و خود تابع و مقایسه آنها می‌پردازید، همچنین در انتها از شما خواسته می‌شود که سری فوریه یک تابع خاص را محاسبه کرده و نتیجه را با حل دستی تطابق دهید.

- تبدیل فوریه

در این قسمت برای درک بهتر تبدیل فوریه، به صورت گام به گام، با رسم یک تابع در حوزه زمان و سپس انتقال آن به حوزه فرکانس را به کمک تبدیل فوریه بررسی می‌کنید، و رسم آن می‌پردازید.

در ادامه برای بررسی کاربردی تبدیل فوریه و درک بهتر مفهوم فرکانس نمونه‌برداری یک فایل صوتی در اختیار شما قرار گرفته است و انجام یک سری عملیات از شما مطالبه می‌گردد.

۱ آشنایی با MATLAB

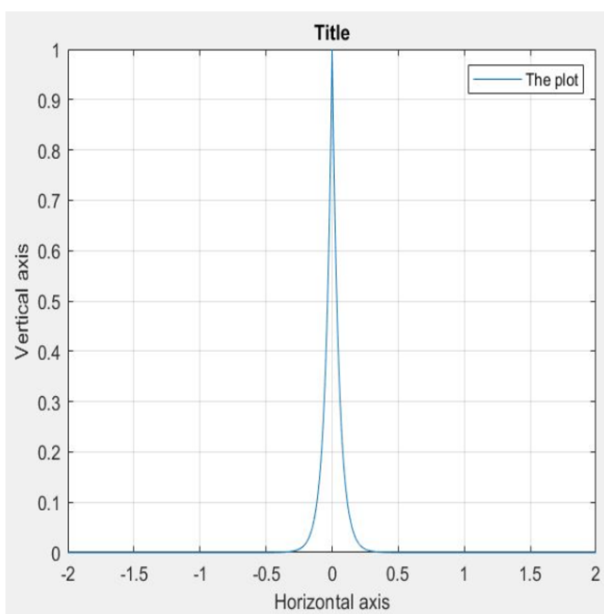
ابتدا برای این که در طی پروژه کمتر درگیر Syntax Error شوید، مراحل رسم دو تابع در MATLAB بررسی می‌شود.

مثال اول

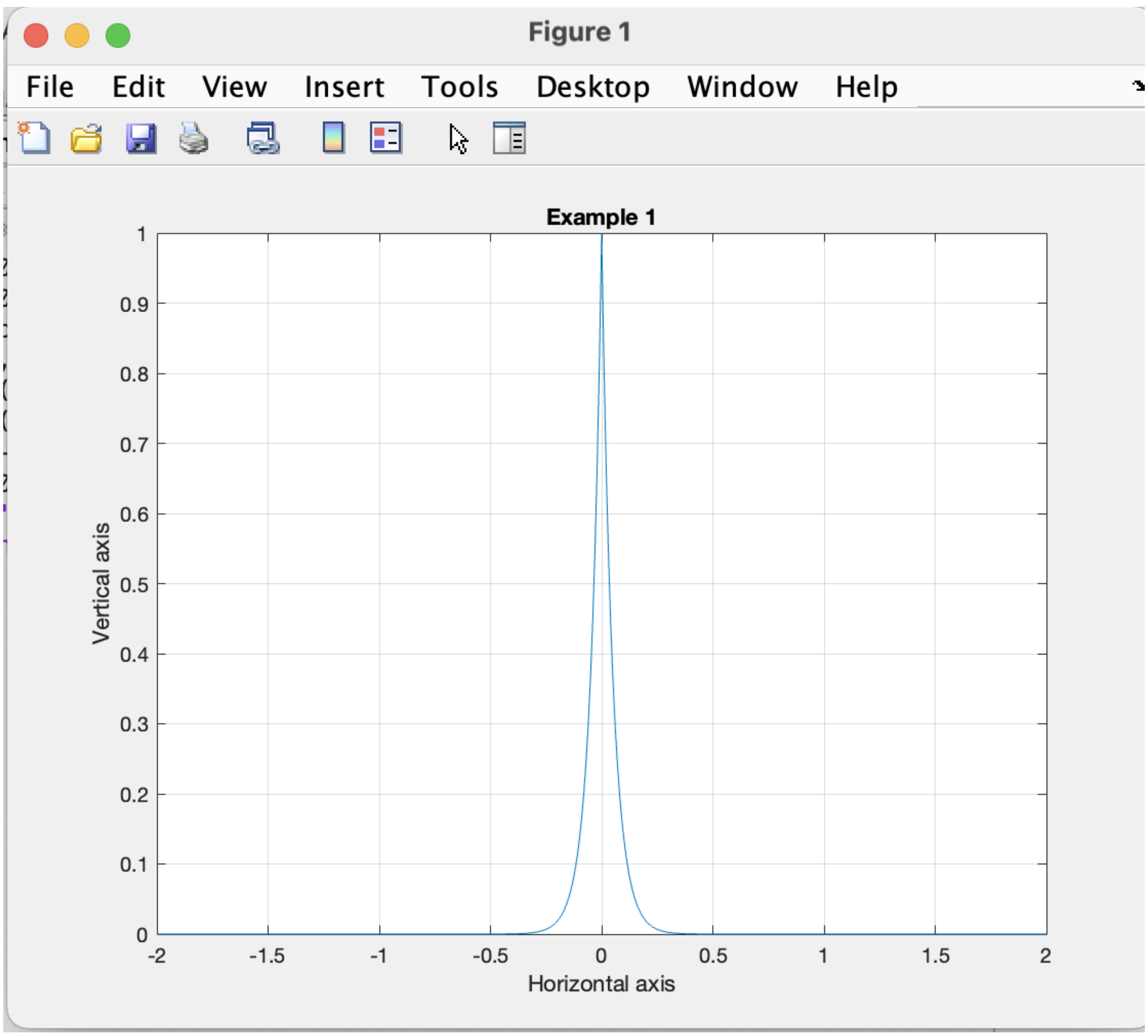
در این بخش رسم تابع، $x(t) = e^{-20|t|}$ بررسی می‌شود.

```
%Sample plotting
fs=100;
t=-30:1/fs:30;
x = exp(-20.*abs(t));
plot(t,x);
xlabel('Horizontal axis')
ylabel('Vertical axis')
xlim([-2 2])
ylim([0 1])
title('Title')
grid on
```

شکل ۱: کد مثال اول



```
fs = 100;
t = -30:1/fs:30;
x = exp(-20.*abs(t));
plot(t, x);
xlabel('Horizontal axis');
ylabel('Vertical axis');
xlim([-2, 2]);
ylim([0, 1]);
title('Example 1');
grid on;
```



Part 1 :

۱.۱ رسم‌نمودار

با توجه به توضیحات ارائه شده، نمودار توابع زیر را رسم کنید.

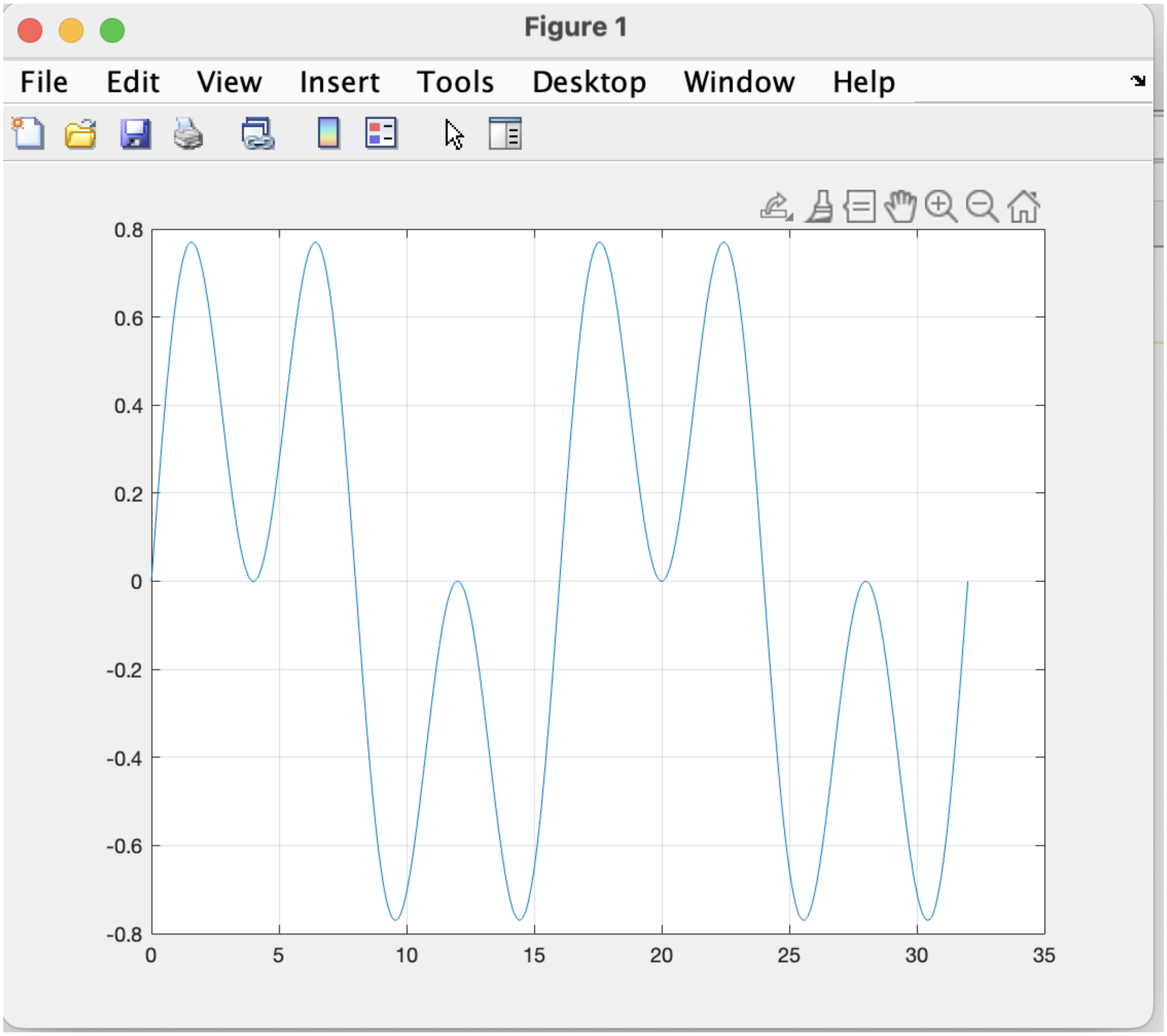
- $\cot\left(\frac{\pi t}{4}\right)\sin\left(\frac{\pi t}{8}\right)$
- $sgn\left(\frac{1}{t^2}\right)$
-

$$\begin{cases} -1, \quad t<-3 \\ 3ramp(t), \quad -3<t<3 \\ e^{-2.5t}, \quad t>3 \end{cases}$$

we have 3 plts :

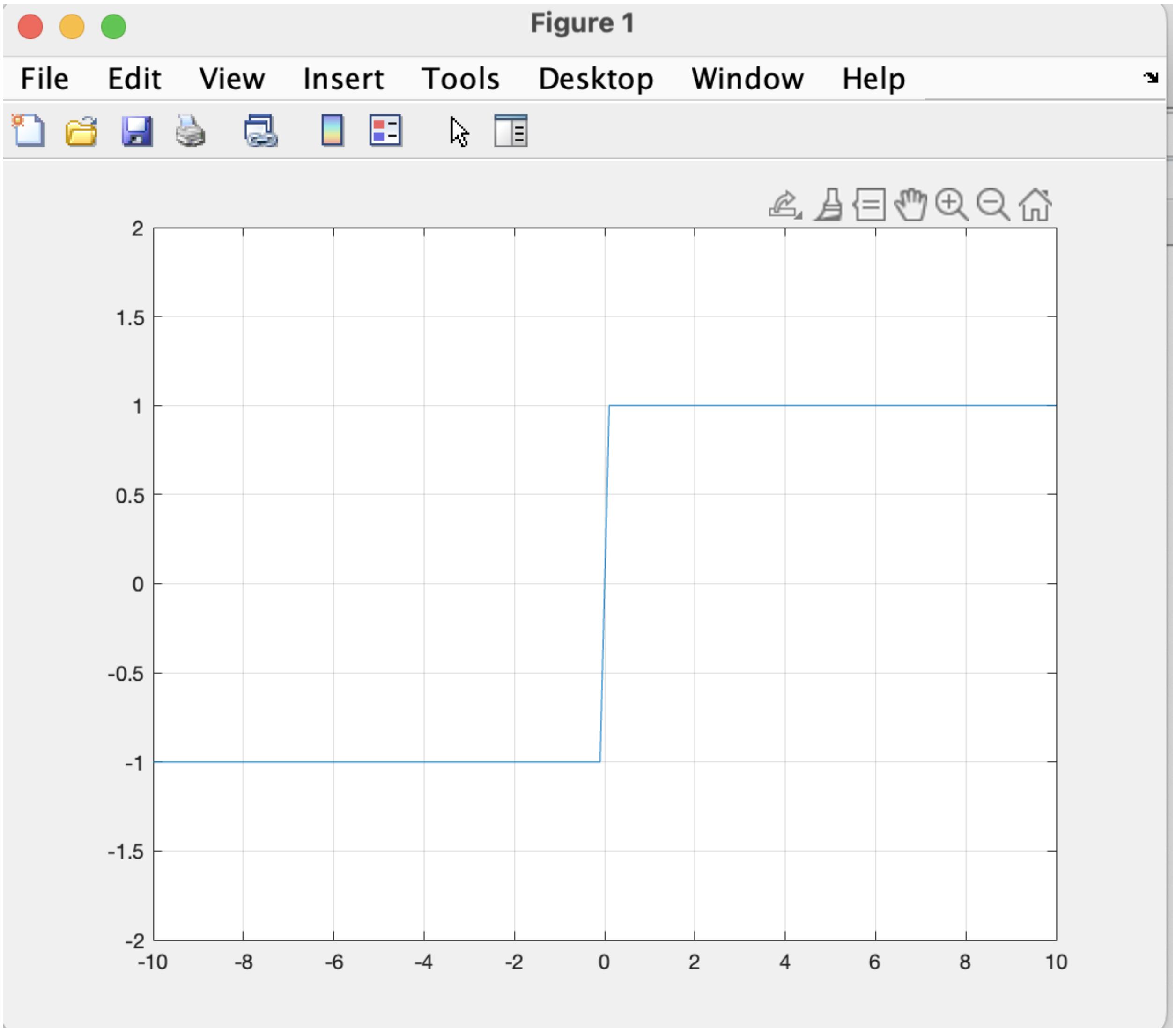
first :

```
x = linspace(0, 32,1000)
plot(x, sin(pi*x/4).*cos(pi*x/8))
grid on
xlim([0,32])
```



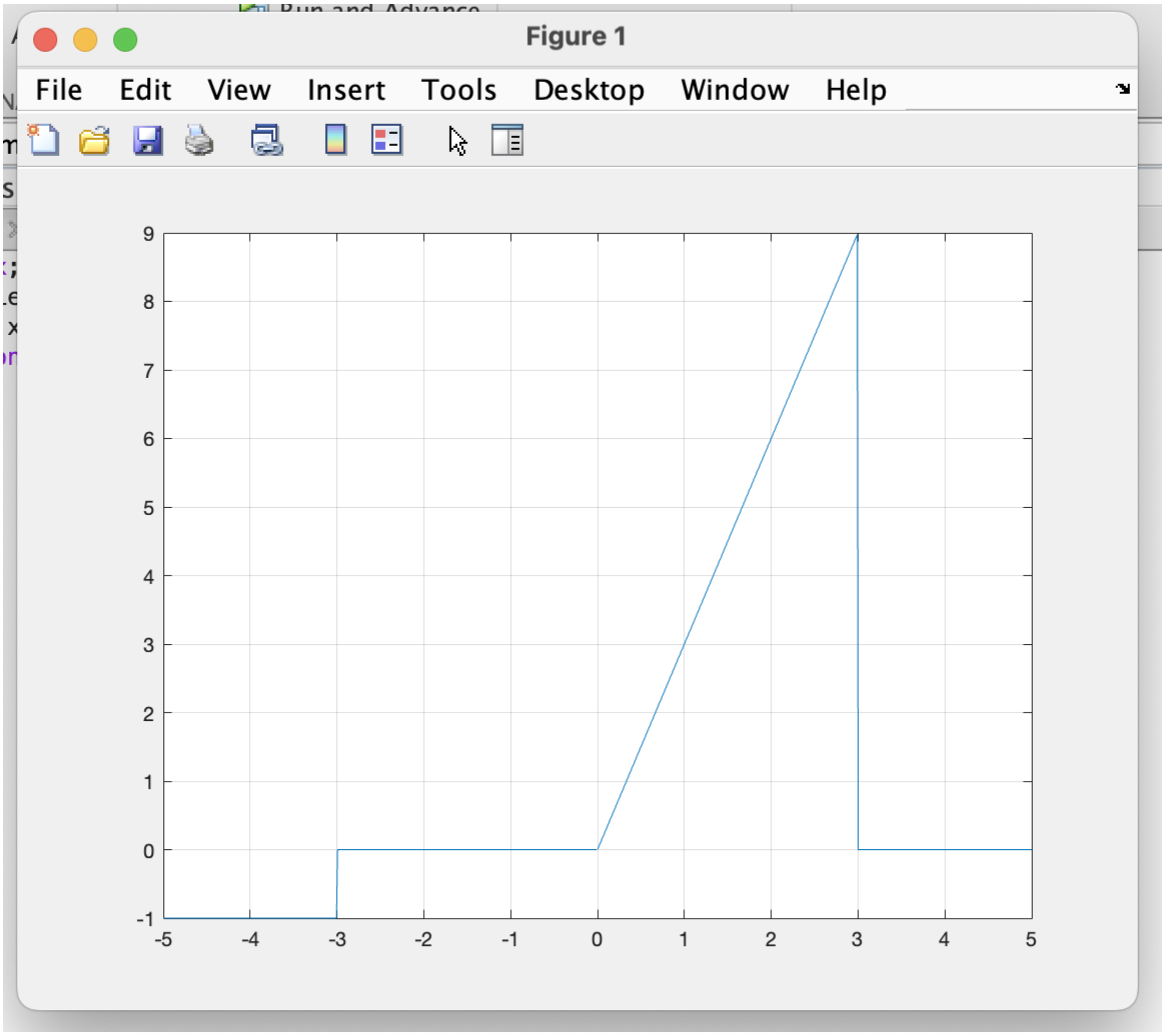
socond :

```
x = linspace(-10, 10,100)
plot(x, sign(x))
grid on
xlim([-10,10])
ylim([-2,2])
```



thered :

```
syms x;
y = piecewise(x<-3,-1,-3<x<0,0,0<x<3,3*x,x>3,exp(-2.5*x));
fplot(x,y);
grid on
```



Part 2 :

۲ سری فوریه

در این قسمت به بررسی سری فوریه با MATLAB می‌پردازیم.

۱.۲ محاسبه سری فوریه

تابعی بنویسید که سری فوریه تابعی به فرم $f(x) = x^{\alpha}$ را محاسبه نماید، ورودی های تابع به صورت زیر خواهد بود.

- Num

  تعداد جملات سری فوریه.

- P

  تناوب‌های مدنظر برای نمایش تابع.

- α

  توان چندجمله‌ای.

- Nshow

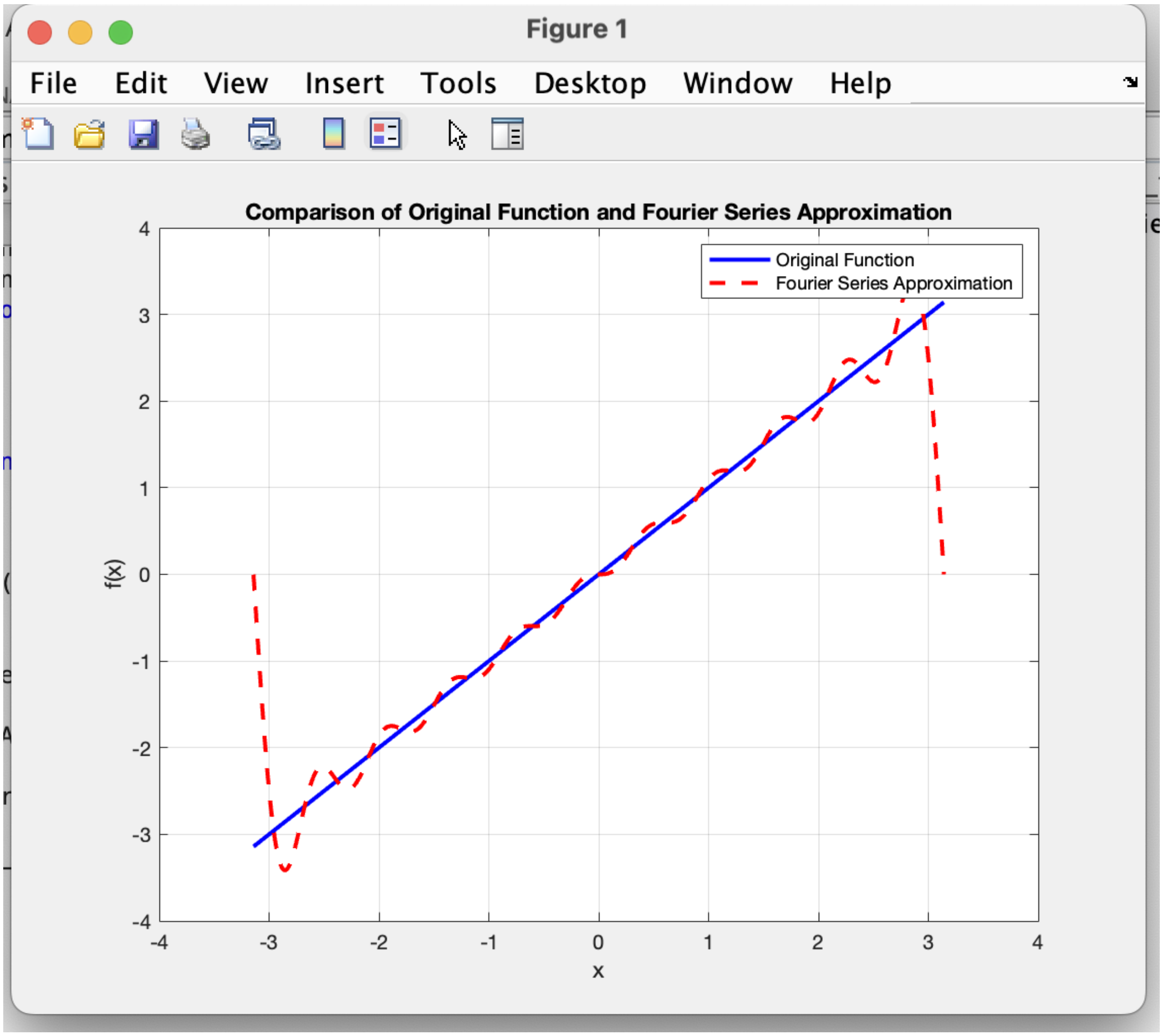
  تعداد جملات سری فوریه جهت نمایش هنگام خروجی گرفتن.

that function for fourier transform :

```
function [A0, An, Bn] = compute_fourier_series(f, T, num_terms)
    syms x;
    f_sym = f(x);
    A0 = (2/T) * int(f_sym, x, -T/2, T/2);
    A0 = double(A0);
    An = zeros(1, num_terms);
    Bn = zeros(1, num_terms);
    for k = 1:num_terms
        An(k) = (2/T) * int(f_sym * cos(k*x), x, -T/2, T/2);
        An(k) = double(An(k));
        Bn(k) = (2/T) * int(f_sym * sin(k*x), x, -T/2, T/2);
        Bn(k) = double(Bn(k));
```

```
    end
end
```

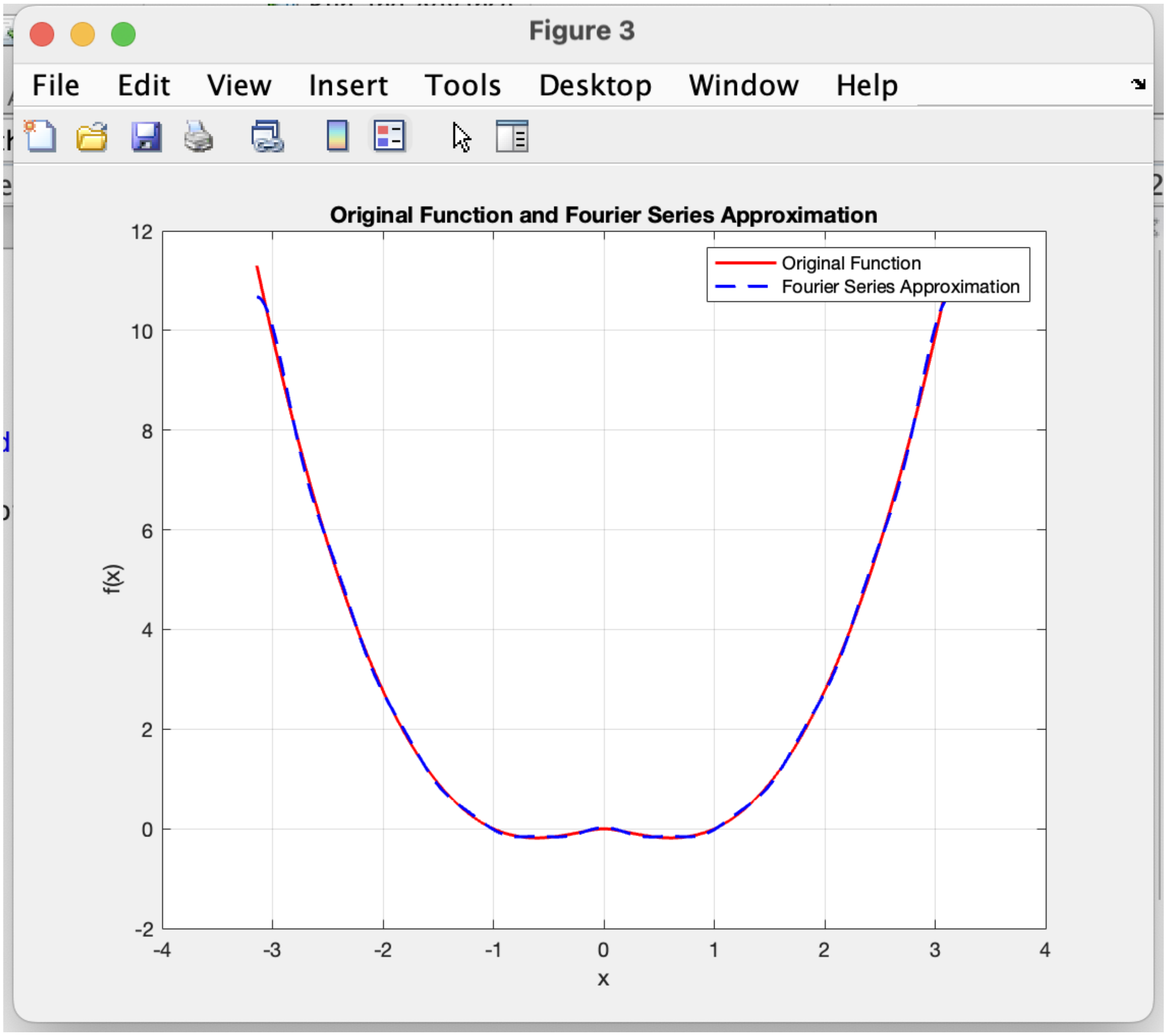
Example of this :




```
f = @(x) x;

T = 2 * pi;
num_terms = 10;

[A0, An, Bn] = compute_fourier_series(f, T, num_terms);
```

```
f_fourier = @(x) A0/2;
for k = 1:num_terms
    f_fourier = @(x) f_fourier(x) + An(k) * cos(k * x) + Bn(k) *
end

x_values = linspace(-pi, pi, 1000);
f_values = arrayfun(f, x_values);
f_fourier_values = arrayfun(f_fourier, x_values);

figure;
plot(x_values, f_values, 'b', 'LineWidth', 2);
hold on;
plot(x_values, f_fourier_values, 'r--', 'LineWidth', 2);
legend('Original Function', 'Fourier Series Approximation');
xlabel('x');
ylabel('f(x)');
title('Comparison of Original Function and Fourier Series Approx
grid on;
hold off;
```

Part 2_2




```
function plot_fourier_series()
    alpha = 1;
    beta = 2;
    T = 2 * pi;
    num_terms = 10;
    Nshow = 5;

    f = @(x) x.^beta .* log(alpha .* x);

    [A0, An, Bn] = compute_fourier_series(f, T, num_terms);
```

```
    fprintf('A0 = %.4f\n', A0);
    fprintf('An coefficients:\n');
    disp(An(1:Nshow));
    fprintf('Bn coefficients:\n');
    disp(Bn(1:Nshow));

    syms x;
    fourier_series = A0 / 2;
    for k = 1:num_terms
        fourier_series = fourier_series + An(k) * cos(2 * pi * 
    end

    f_series = matlabFunction(fourier_series);

    x_vals = linspace(-pi, pi, 1000);
    y_vals_original = f(x_vals);
    y_vals_series = f_series(x_vals);

    figure;
    plot(x_vals, y_vals_original, 'r', 'LineWidth', 1.5);
    hold on;
    plot(x_vals, y_vals_series, 'b--', 'LineWidth', 1.5);
    xlabel('x');
    ylabel('f(x)');
    title('Original Function and Fourier Series Approximation');
    legend('Original Function', 'Fourier Series Approximation');
    grid on;
    hold off;
end

function [A0, An, Bn] = compute_fourier_series(f, T, num_terms)
    syms x;
    f_sym = f(x);
    A0 = (2/T) * int(f_sym, x, -T/2, T/2);
    A0 = double(A0);
    An = zeros(1, num_terms);
```

```
    Bn = zeros(1, num_terms);
    for k = 1:num_terms
        An(k) = (2/T) * int(f_sym * cos(2 * pi * k * x / T), x,
        An(k) = double(An(k));
        Bn(k) = (2/T) * int(f_sym * sin(2 * pi * k * x / T), x,
        Bn(k) = double(Bn(k));
    end
end

plot_fourier_series();
```

part 3_3

```
function plot_fourier_series_comparison()
    T = 2 * pi;
    f = @(x) x.^2;
    num_terms_list = [10, 50, 100];
    colors = ['r', 'g', 'b'];
    x_vals = linspace(-pi, pi, 1000);
    y_vals_original = f(x_vals);

    for i = 1:length(num_terms_list)
        num_terms = num_terms_list(i);
        [A0, An, Bn] = compute_fourier_series(f, T, num_terms);

        syms x;
        fourier_series = A0 / 2;
        for k = 1:num_terms
            fourier_series = fourier_series + An(k) * cos(2 * pi
        end

        f_series = matlabFunction(fourier_series);
        y_vals_series = f_series(x_vals);
```

```
        figure;
        plot(x_vals, y_vals_original, 'k', 'LineWidth', 1.5);
        hold on;
        plot(x_vals, y_vals_series, '--', 'LineWidth', 1.5, 'Co
        xlabel('x');
        ylabel('f(x)');
        title(['Fourier Series Approximation with ', num2str(nu
        legend('Original Function', ['Fourier Series (', num2st
        grid on;
        hold off;
    end
end

function [A0, An, Bn] = compute_fourier_series(f, T, num_terms)
    syms x;
    f_sym = f(x);
    A0 = (2/T) * int(f_sym, x, -T/2, T/2);
    A0 = double(A0);
    An = zeros(1, num_terms);
    Bn = zeros(1, num_terms);
    for k = 1:num_terms
        An(k) = (2/T) * int(f_sym * cos(2 * pi * k * x / T), x,
        An(k) = double(An(k));
        Bn(k) = (2/T) * int(f_sym * sin(2 * pi * k * x / T), x,
        Bn(k) = double(Bn(k));
    end
end

plot_fourier_series_comparison();
```

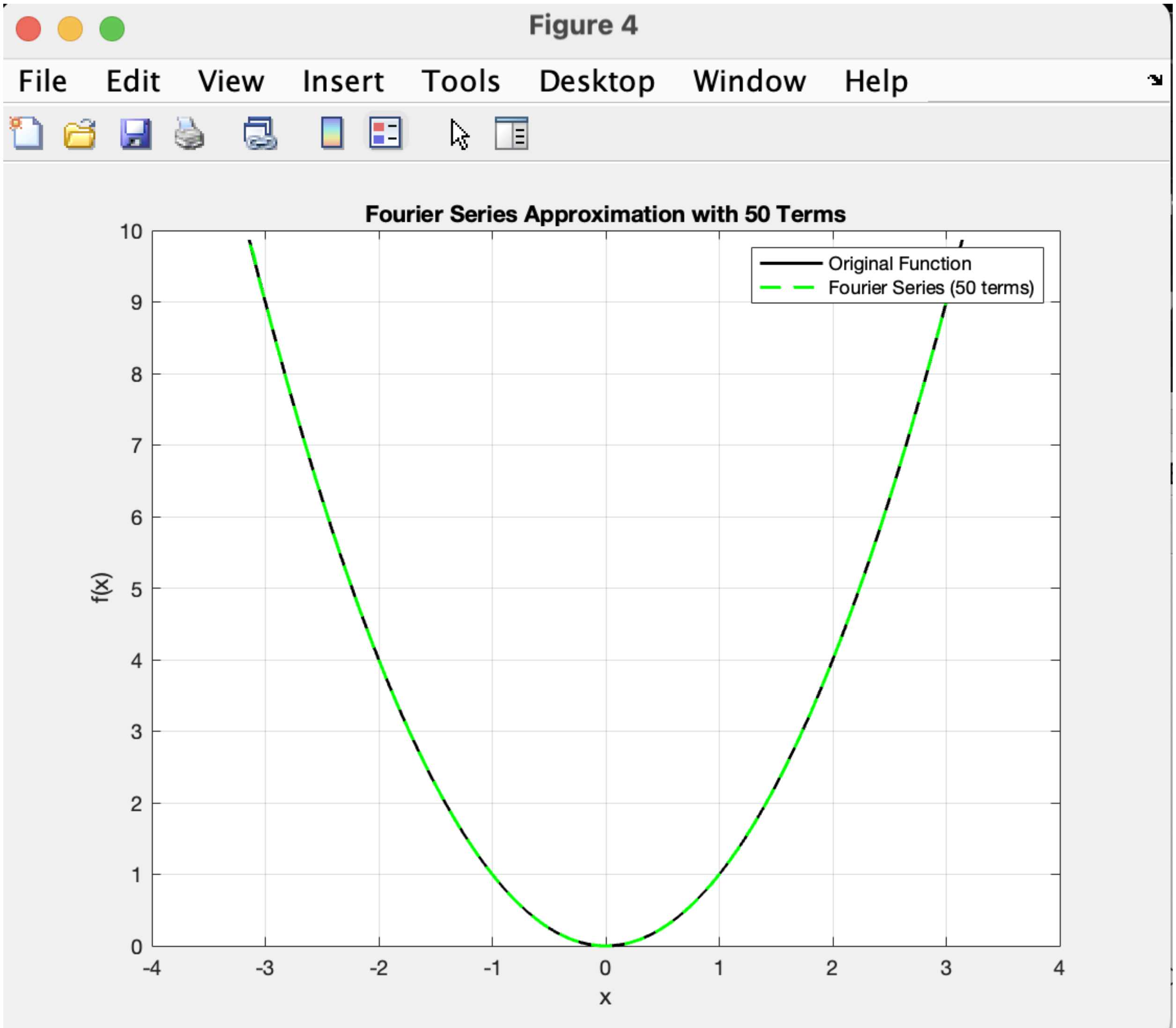


Figure 4
File Edit View Insert Tools Desktop Window Help
Fourier Series Approximation with 50 Terms
Original Function
Fourier Series (50 terms)
f(x)
x
10
9
8
7
6
5
4
3
2
1
0
-4
-3
-2
-1
0
1
2
3
4

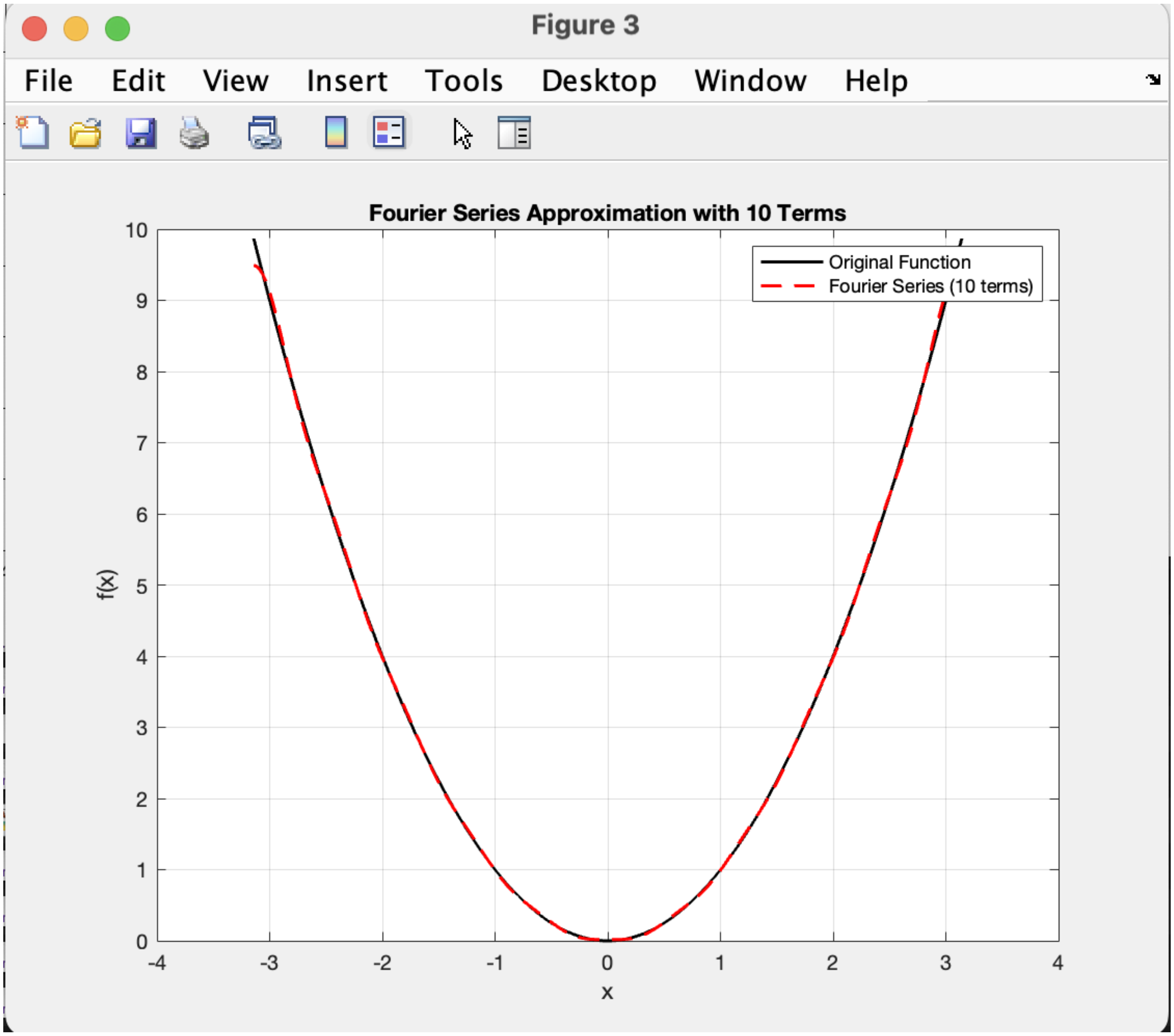


Figure 3
File Edit View Insert Tools Desktop Window Help
Fourier Series Approximation with 10 Terms
Original Function
Fourier Series (10 terms)
f(x)
x
10
9
8
7
6
5
4
3
2
1
0
-4
-3
-2
-1
0
1
2
3
4

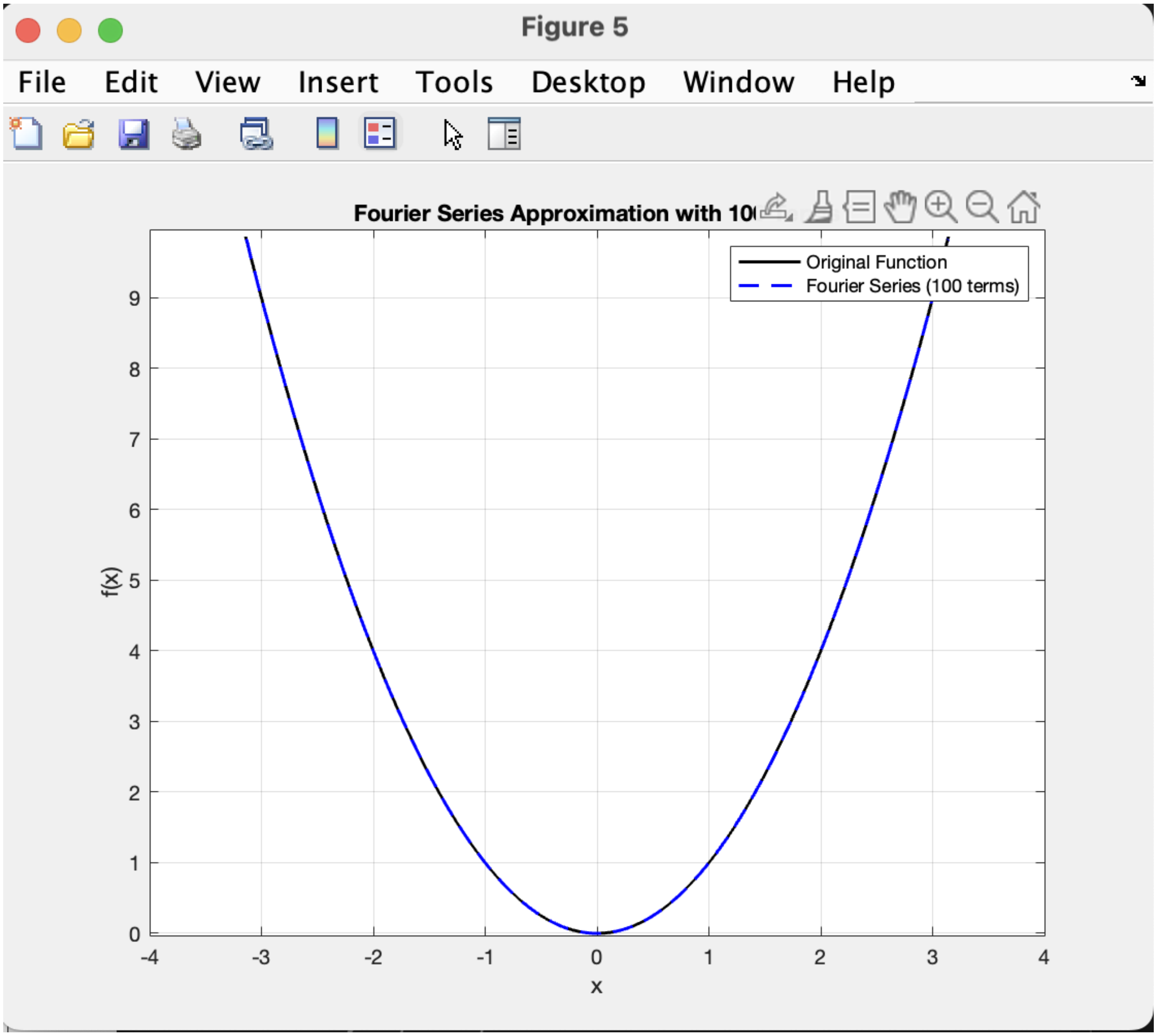


Figure 5
File Edit View Insert Tools Desktop Window Help
Fourier Series Approximation with 10
Original Function
Fourier Series (100 terms)
f(x)
x
9
8
7
6
5
4
3
2
1
0
-4
-3
-2
-1
0
1
2
3
4

part 4_2:

$$a_0 = \frac{1}{2\pi}\int_{-\pi}^{\pi} x^2 \mathrm{dx} = \frac{1}{6\pi} \times 2\pi^3 = \frac{\pi^2}{3}$$

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} x^2 \cos(\mathrm{nx})\mathrm{dx} = \frac{(n^2x^2-2)\sin(nx)+2nx\cos(nx)}{\pi n^3}\Big|_{-\pi}^{\pi} = \frac{2\left(\overbrace{(\pi^2n^2-2)\sin(\pi n)}^{0}+2\pi n\cos(\pi n)\right)}{\pi n^3}$$

$$= \frac{4}{n^2}\overbrace{\cos(\pi n)}^{(-1)^n} = \frac{4(-1)^n}{n^2}$$

$$b_n = 0(\mathrm{even})$$

$$\Longrightarrow f(x) = \frac{\pi^2}{3} + 4\sum_{n=1}^{\infty}\frac{(-1)^n}{n^2}\cos(\mathrm{nx})$$

$$x = \pi \Longrightarrow \frac{1}{2}(\pi^2 + (-\pi)^2) = \frac{\pi^2}{3} + 4\sum_{n=1}^{\infty}\frac{(-1)^n}{n^2}\cos(n\pi) = \frac{\pi^2}{3} + 4\sum_{n=1}^{\infty}\frac{(-1)^n}{n^2}(-1)^n = \frac{\pi^2}{3} + 4\sum_{n=1}^{\infty}\frac{1}{n^2}$$

$$\Longrightarrow \sum_{n=1}^{\infty}\frac{1}{n^2} = \frac{1}{4}\times\left(\pi^2 - \frac{\pi^2}{3}\right) = \frac{\pi^2}{6} \simeq 1.644$$

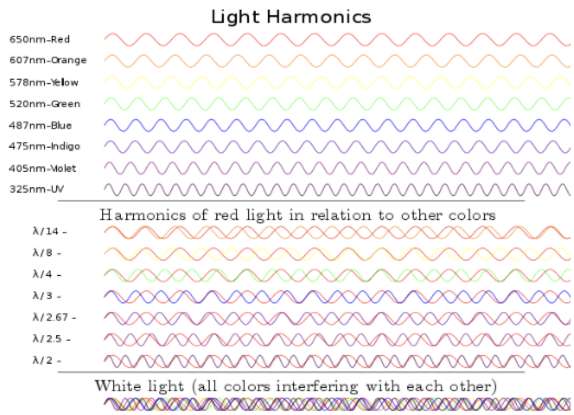
```
function plot_fourier_series_comparison()
    T = 2 * pi;
    f = @(x) x.^2;
    num_terms = 100;
    x_val = pi;

    [A0, An, Bn] = compute_fourier_series(f, T, num_terms);

    syms x;
    fourier_series = A0 / 2;
    for k = 1:num_terms
        fourier_series = fourier_series + An(k) * cos(2 * pi * k
    end
```

```
    f_series = matlabFunction(fourier_series);
    y_val_series = f_series(x_val);
    y_val_series = y_val_series/6

    manual_series = (pi^2 / 3 + 4 * sum(((-1).^(1:num_terms)) ./

    fprintf('Computed Fourier Series Value at x = %f: %f\n', x_v
    fprintf('Manual Fourier Series Value at x = %f: %f\n', x_va
end

function [A0, An, Bn] = compute_fourier_series(f, T, num_terms)
    syms x;
    f_sym = f(x);
    A0 = (2/T) * int(f_sym, x, -T/2, T/2);
    A0 = double(A0);
    An = zeros(1, num_terms);
    Bn = zeros(1, num_terms);
    for k = 1:num_terms
        An(k) = (2/T) * int(f_sym * cos(2 * pi * k * x / T), x,
        An(k) = double(An(k));
        Bn(k) = (2/T) * int(f_sym * sin(2 * pi * k * x / T), x,
        Bn(k) = double(Bn(k));
    end
end

plot_fourier_series_comparison();
```

```
Computed Fourier Series Value at x = 3.141593: 1.638301
Manual Fourier Series Value at x = 3.141593: 1.638301
```

۵.۲ آنالیز هارمونیک در سری فوریه

آنالیز هارمونیک شاخه‌ای از ریاضیات است که مرتبط با نمایش توابع به صورت برآیندی از امواج پایه بوده و به مطالعه و نمایش مفاهیم سری فوریه و تبدیل فوریه می‌پردازد. اخیرا، این شاخه کاربردهای گسترده‌ای در نظریه اعداد، پردازش سیگنال، مکانیک کوانتومی، و علوم اعصاب دارد.



به روند یافتن ضرایب سری فوریه برای تابع با کمک مقادیر عددی آنالیز هارمونیک گفته می‌شود، روابط پایه مورد نیاز در این قسمت به شرح زیر است.

$$f(x) = \frac{A_0}{2} + \sum_{n=1}^{\infty} \left(A_n \cos(nx) + B_n \sin(nx)\right),$$

$$A_0 = \frac{2\sum f(x)}{n}, \quad A_n = \frac{2\sum f(x)\cos(nx)}{n}, \quad B_n = \frac{2\sum f(x)\sin(nx)}{n}$$

لازم به ذکر است که در بسط سری فوریه، $(A_1 \cos(x) + B_1 \sin(x))$ را هارمونیک اول، $(A_2 \cos(2x) + B_2 \sin(2x))$ را هارمونیک دوم و $(A_n \cos(nx) + B_n \sin(nx))$ را هارمونیک n ام می‌نامیم.

Part 5_2 :

۱.۵.۲ شبیه‌سازی آنالیز هارمونیک

در این قسمت، شما باید تابعی بنویسید که با گرفتن نقاط x و مقدار تابع $f(x)$ در آن نقطه، n هارمونیک اول سری فری را با روش بیان شده،محاسبه کرده، نمایش داده و رسم کند.

ریاضیات مهندسی — پروژه اول

در نهایت قطعه کد خود را برای ورودی نمونه زیر و تا ۴ هارمونیک آزمایش کنید.

x	0	$\frac{\pi}{3}$	$\frac{2\pi}{3}$	π	$\frac{4\pi}{3}$	$\frac{5\pi}{3}$	2π
$f(x)$	1	1.4	1.9	1.7	1.5	1.2	1

خروجی کد شما برای جدول ارائه شده به صورت زیر خواهد بود:

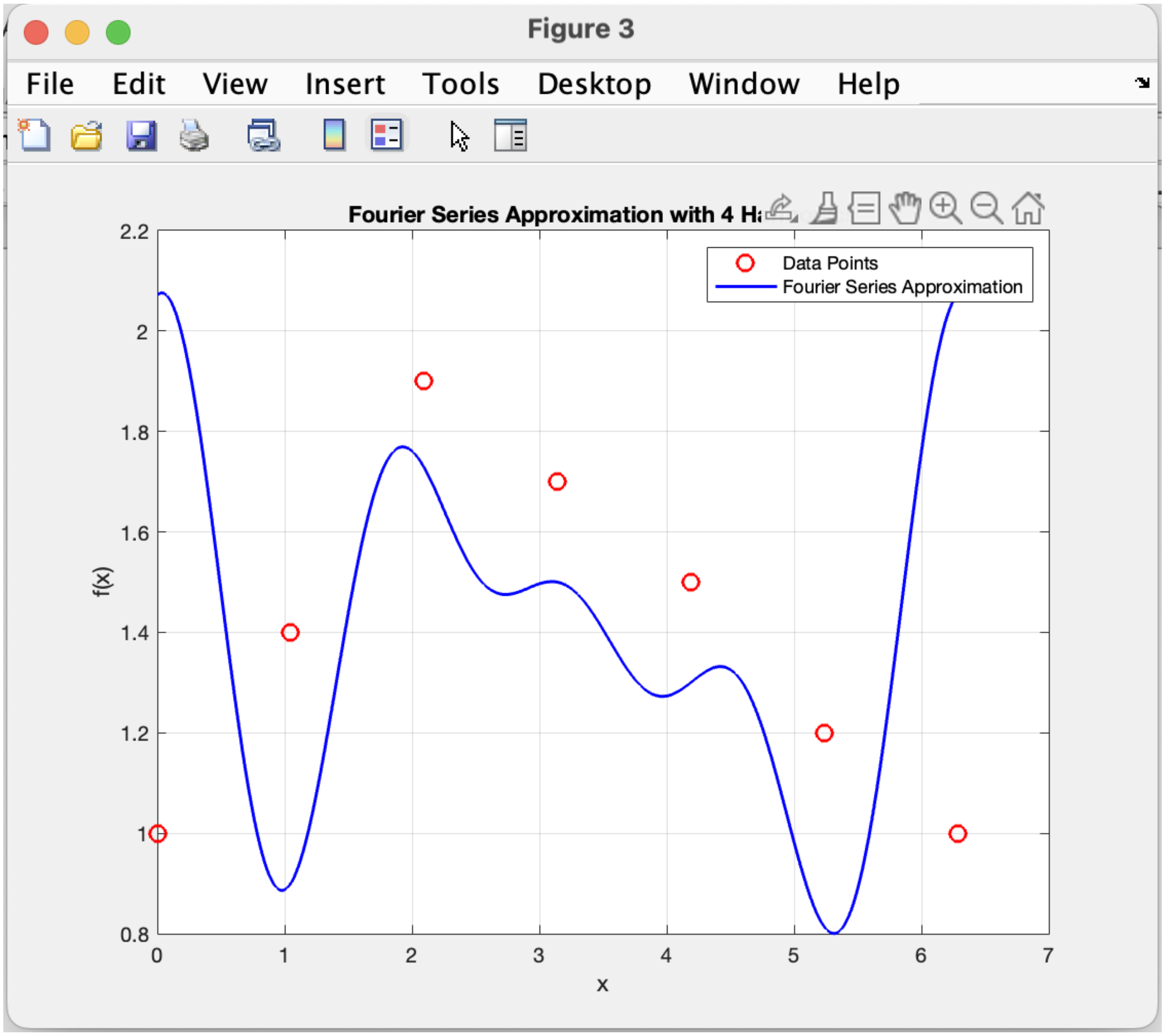
$$f(x) = 1.3857 - 0.2\cos(x) + 1.0392\sin(x) + 0.7\cos(2x) - 0.1732\sin(2x)$$
$$+ 0.7333\cos(3x) + \dots$$

```
function harmonic_analysis(x, f_values, num_harmonics)
    T = 2 * pi;
    n = length(x);
    A0 = (2 / n) * sum(f_values);
```

```
An = zeros(1, num_harmonics);
Bn = zeros(1, num_harmonics);

for k = 1:num_harmonics
    cos_kx = cos(k * x);
    sin_kx = sin(k * x);

    An(k) = (2 / n) * sum(f_values .* cos_kx);
    Bn(k) = (2 / n) * sum(f_values .* sin_kx);
end

syms x_sym;
fourier_series = A0 / 2;
for k = 1:num_harmonics
    fourier_series = fourier_series + An(k) * cos(k * x_sym
end

disp(['f(x) = ' num2str(A0 / 2)]);
for k = 1:num_harmonics
    if An(k) ~= 0
        disp([' + ' num2str(An(k)) ' cos(' num2str(k) 'x)']
    end
    if Bn(k) ~= 0
        disp([' + ' num2str(Bn(k)) ' sin(' num2str(k) 'x)']
    end
end

f_series = matlabFunction(fourier_series);
x_vals = linspace(min(x), max(x), 1000);
y_vals_series = f_series(x_vals);

figure;
plot(x, f_values, 'ro', 'MarkerSize', 8, 'LineWidth', 1.5);
hold on;
plot(x_vals, y_vals_series, 'b', 'LineWidth', 1.5);
```

```
    xlabel('x');
    ylabel('f(x)');
    title(['Fourier Series Approximation with ', num2str(num_ha
    legend('Data Points', 'Fourier Series Approximation');
    grid on;
    hold off;
end

x_sample = [0, pi/3, 2*pi/3, pi, 4*pi/3, 5*pi/3, 2*pi];
f_values_sample = [1, 1.4, 1.9, 1.7, 1.5, 1.2, 1];

harmonic_analysis(x_sample, f_values_sample, 4);
```

```
f(x) = 1.3857
 + -0.028571 cos(1x)
 + 0.14846 sin(1x)
 + 0.2 cos(2x)
 + -0.049487 sin(2x)
 + 0.31429 cos(3x)
 + -1.1547e-16 sin(3x)
```



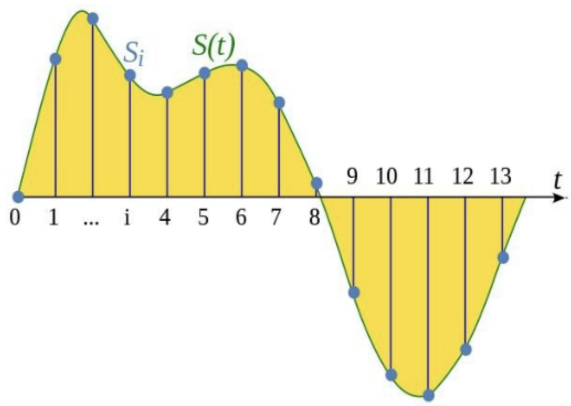
in this part we work with audio files :

۳ تبدیل فوریه

در این قسمت به بررسی حوزه فرکانس چند تابع می‌پردازیم و سپس مفاهیم و کاربردهای تبدیل فوریه را در کار با Audio بررسی می‌کنیم.

۱.۳ فرکانس نمونه برداری

تصویر زیر را در نظر بگیرید.



تابع پیوسته‌زمان $S(t)$ را در نظر بگیرید، از این تابع در بازه‌های زمانی Δt ثانیه نمونه‌برداری می‌شود و تابع گسسته‌زمان S_i به دست می‌آید. فاصله زمانی بین هر دو نمونه را نرخ نمونه‌برداری می‌نامیم و فرکانس نمونه‌برداری به صورت زیر تعریف می‌گردد.

$$Sampling\ Freq = \frac{1}{Sampling\ Rate}$$

۲.۳ بررسی حوزه زمان و فرکانس چند تابع

تابع $\cos(\pi t)$ را در نظر بگیرید.

- ابتدا تابع $\cos(\pi t)$ را در حوزه زمان در 2 دوره تناوب رسم کنید.
  - تذکر: در این بخش برای ترسیم توابع فرکانس نمونه برداری را $f_s = 1000$ در نظر بگیرید.
- تبدیل فوریه این تابع را با کمک دستور fft و fftshift حساب کنید.
  - تذکر: در help متلب در مورد دستور های fft و fftshift مطالعه کنید و لزوم استفاده از fftshift برای گرفتن خروجی دقیق را تحقیق نمایید.
- تبدیل فوریه این تابع را به صورت تئوری محاسبه کنید و نتیجه این بخش را با خروجی MATLAB مطابقت دهید.

حال تمامی مراحل بالا را برای دو تابع $f(x) = 1$ و $f(x) = \delta(x)$ تکرار کنید.

Part 2_3:

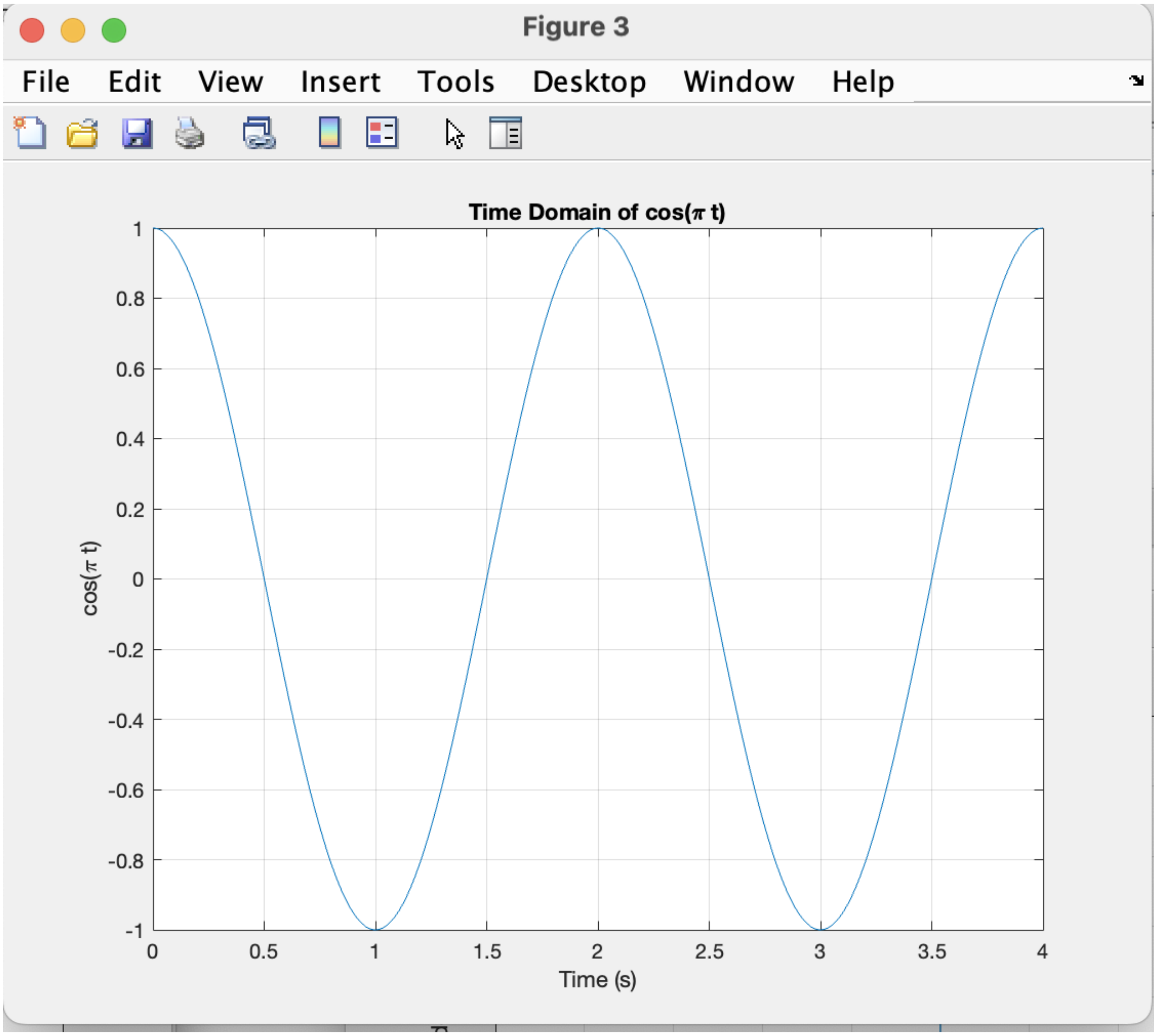
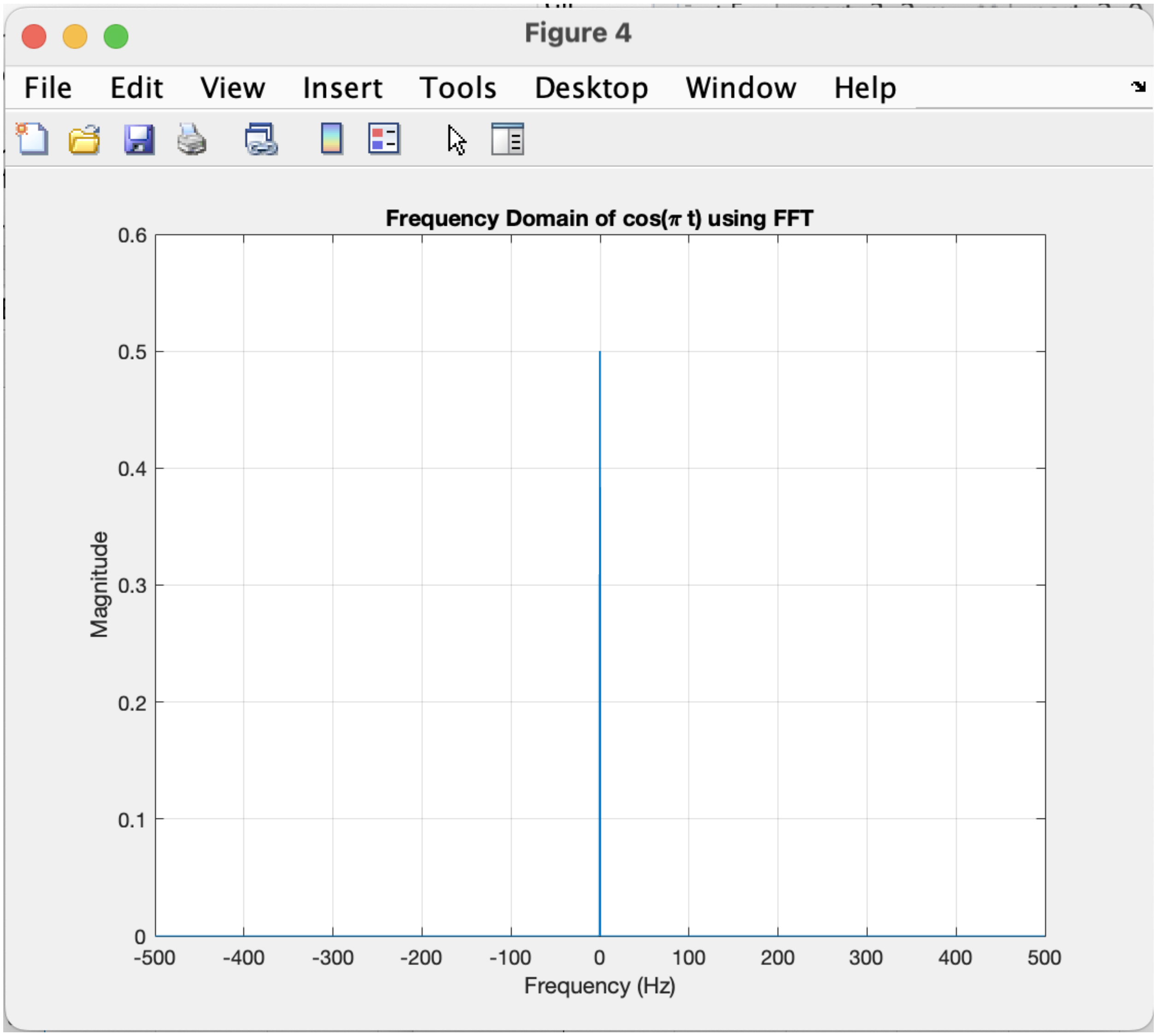


Figure 3
File
Edit
View
Insert
Tools
Desktop
Window
Help
Time Domain of cos(π t)
cos(π t)
Time (s)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.5
1
1.5
2
2.5
3
3.5
4




```
fs = 1000;
T = 2;
t = 0:1/fs:2*T;

f_cos = cos(pi * t);

figure;
plot(t, f_cos);
xlabel('Time (s)');
ylabel('cos(\pi t)');
title('Time Domain of cos(\pi t)');
```

```
grid on;

N = length(f_cos);
f_cos_fft = fft(f_cos);
f_cos_fft_shifted = fftshift(f_cos_fft);
f = (-N/2:N/2-1)*(fs/N);

figure;
plot(f, abs(f_cos_fft_shifted)/N);
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('Frequency Domain of cos(\pi t) using FFT');
grid on;
```

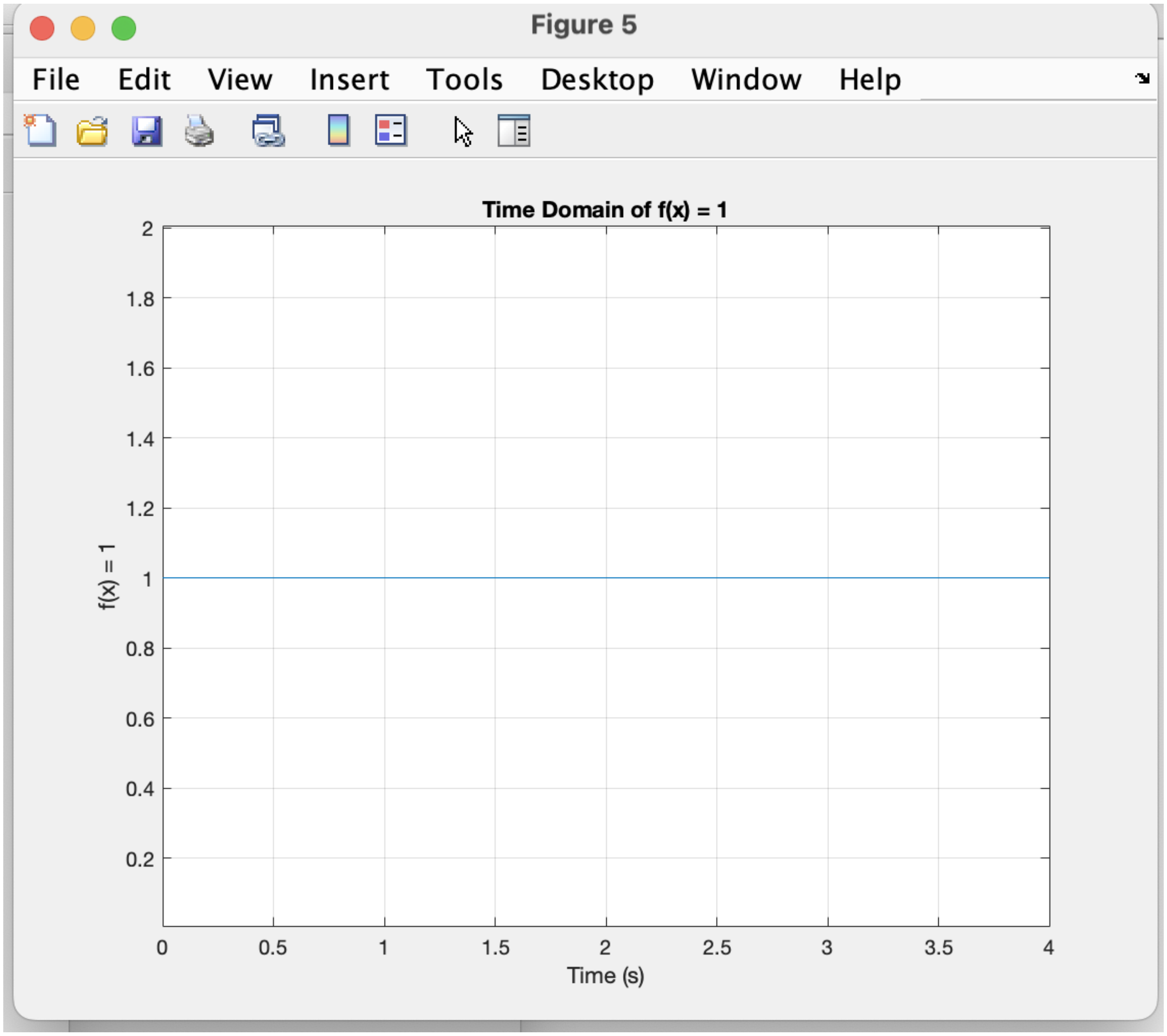
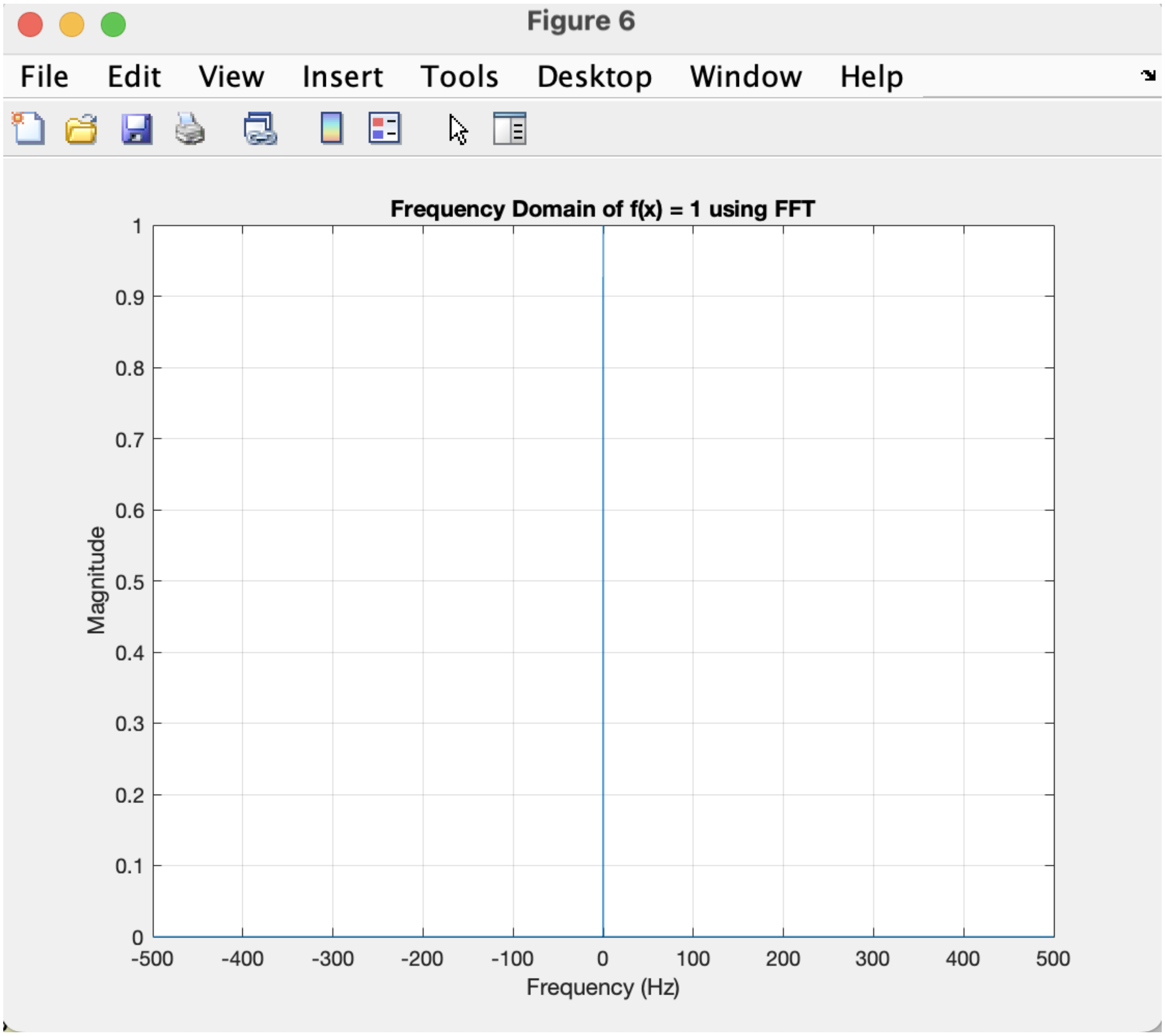


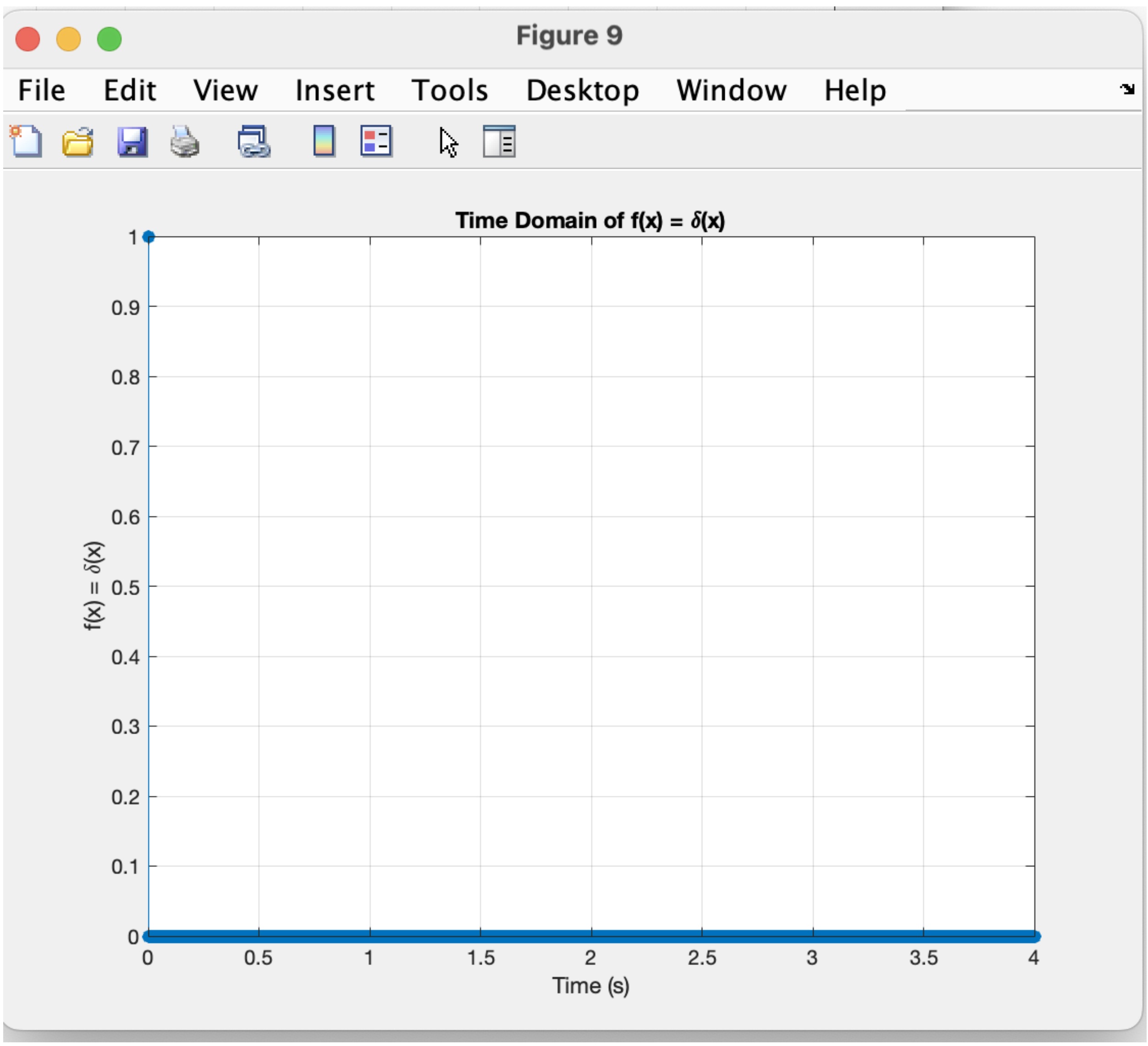
Figure 5
File Edit View Insert Tools Desktop Window Help
Time Domain of f(x) = 1
f(x) = 1
Time (s)
0 0.5 1 1.5 2 2.5 3 3.5 4
0.2 0.4 0.6 0.8 1 1.2 1.4 1.6 1.8 2

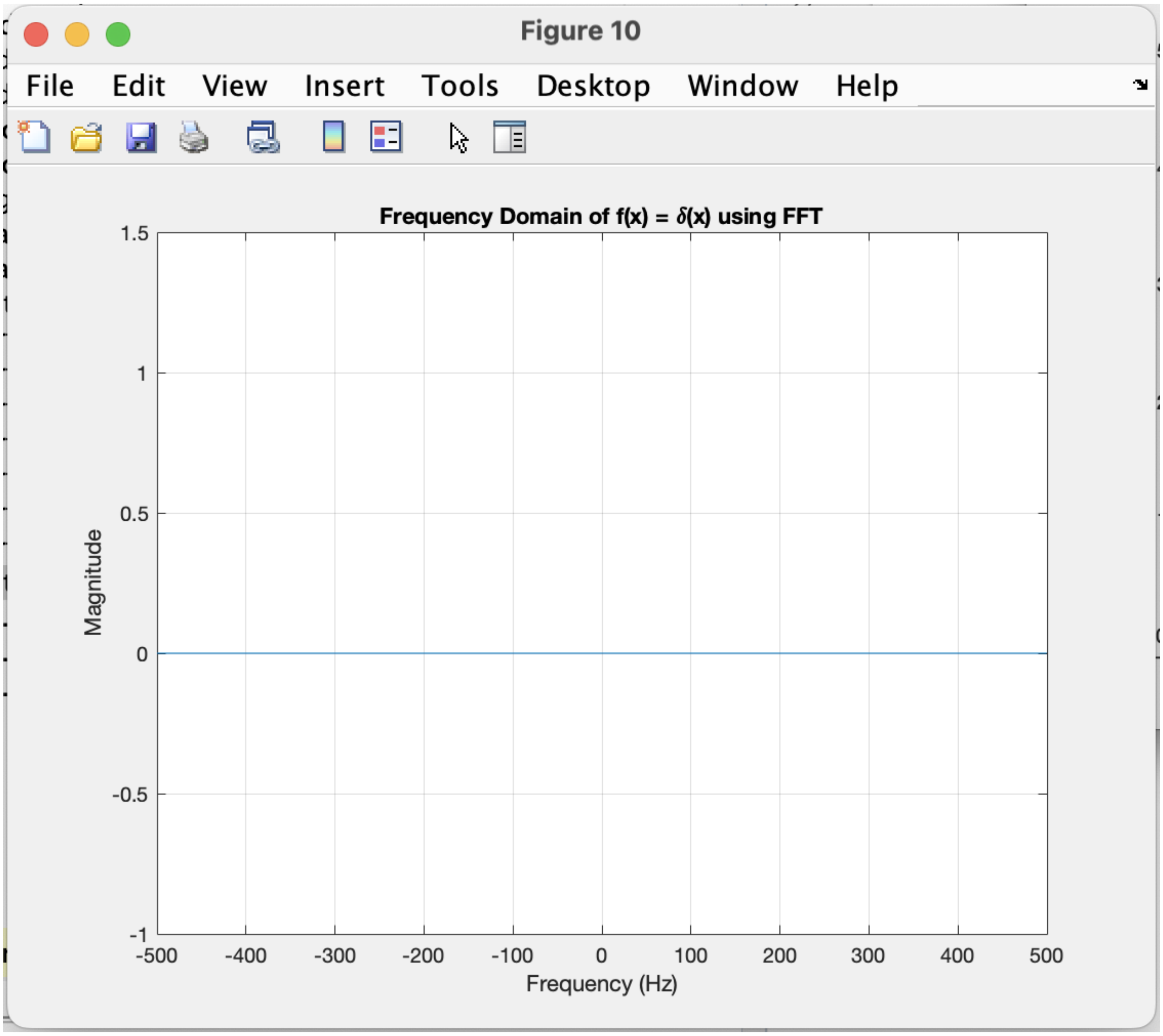



```
f_const = ones(size(t));

figure;
plot(t, f_const);
xlabel('Time (s)');
ylabel('f(x) = 1');
title('Time Domain of f(x) = 1');
grid on;

f_const_fft = fft(f_const);
f_const_fft_shifted = fftshift(f_const_fft);
```

```
figure;
plot(f, abs(f_const_fft_shifted)/N);
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('Frequency Domain of f(x) = 1 using FFT');
grid on;
```

for delta we have this :






```
f_delta = zeros(size(t));
f_delta(t == 0) = 1;

figure;
stem(t, f_delta, 'filled');
xlabel('Time (s)');
ylabel('f(x) = \delta(x)');
title('Time Domain of f(x) = \delta(x)');
grid on;

f_delta_fft = fft(f_delta);
```

```
f_delta_fft_shifted = fftshift(f_delta_fft);

figure;
plot(f, abs(f_delta_fft_shifted)/N);
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('Frequency Domain of f(x) = \delta(x) using FFT');
grid on;
```