Fourier Transform

Continuous Fourier Transform

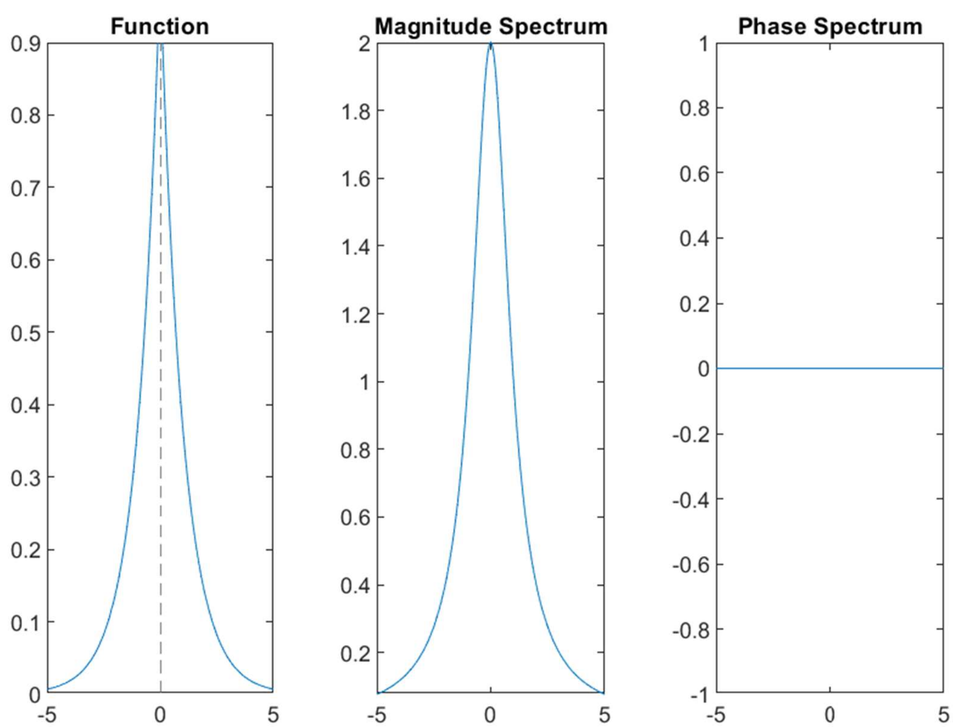
Code:

```
a=sym(1);
t = 0 : 0.002 : 1;
syms x;
syms t;

f = exp(-a * abs(x));
subplot(1,3,1);
fplot((f));
title('Function');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(1,3,2);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(1,3,3);
fplot(P);
title('Phase Spectrum');
```

Result:



Discrete Fourier Transform

Code:

```
a = input('Enter Sequence of [a]: ');
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
disp('The Fourier Transform is');
disp(y(k));
subplot(211);
stem(k, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(212);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> DiscreteFourier

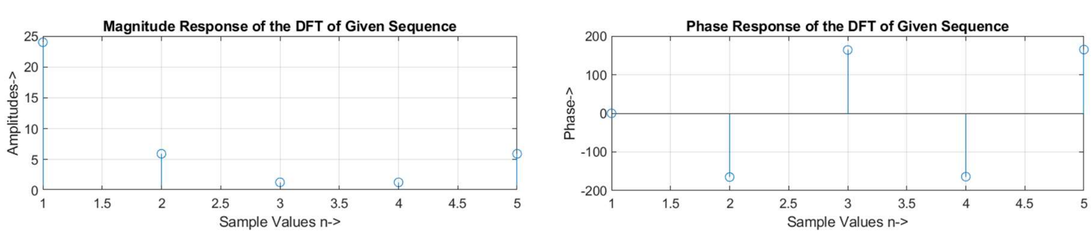
Enter Sequence of [a]: [2 5 7 6 4]

The Length of the Sequence of [a] is:

5

The Fourier Transform is

24.0000 + 0.0000i -5.7361 - 1.5388i -1.2639 + 0.3633i -1.2639 - 0.3633i -5.7361 + 1.5388i



Basic Signals

1. Unit Step Signal

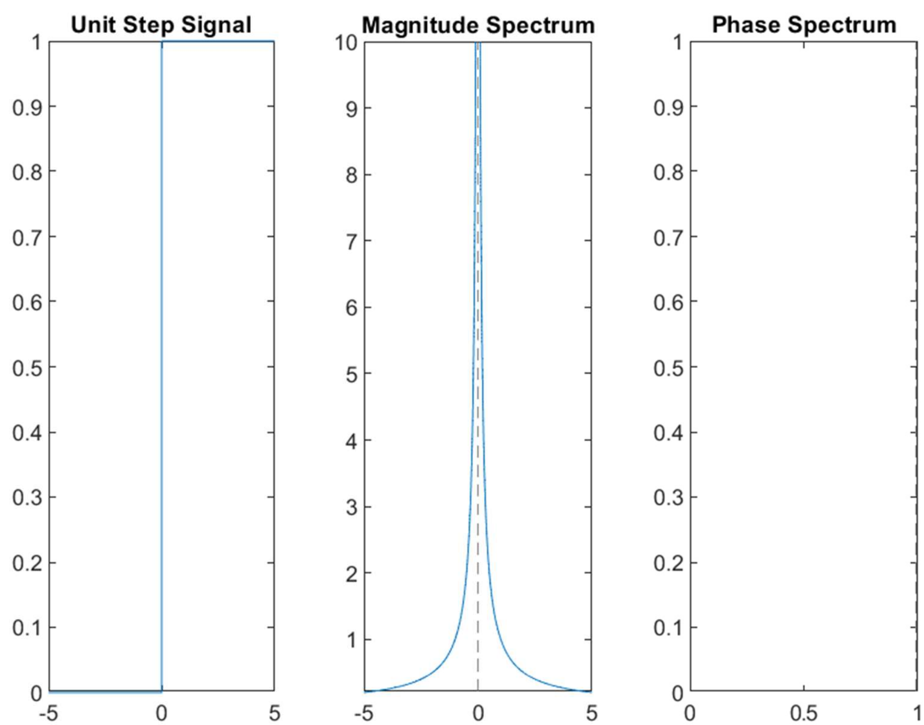
a. Using Built in Function, Continuous

Code:

```
%Continuous Fourier Transform Unit Step Signal
t = 0 : 0.002 : 1;
syms t;
f = heaviside(t);
subplot(131);
fplot(f);
title('Unit Step Signal');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-tan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Without Built-in Function, Discrete

Code:

```
%discrete unit step signal Fourier transform
N = -5 : 5;
a = [zeros(1, 5) ones(1, 6)];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = -5 : 5;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Unit Step Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> disunit

The Length of the Sequence of [a] is: 11

The Fourier Transform is

Columns 1 through 5

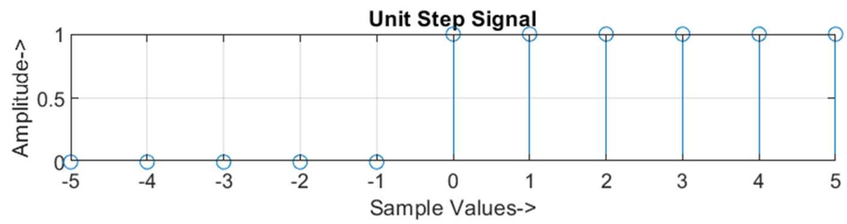
6.0000 + 0.0000i -1.4595 + 3.1958i   0.3413 + 0.3938i -1.1549 + 0.3391i -0.0846 + 0.5883i

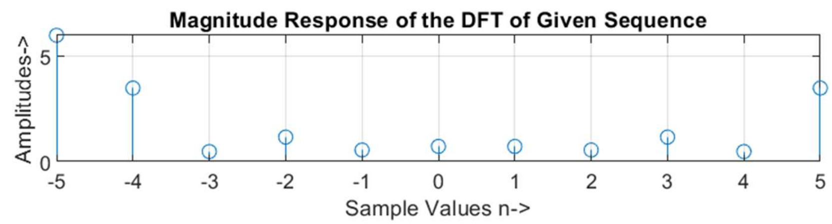
Columns 6 through 10

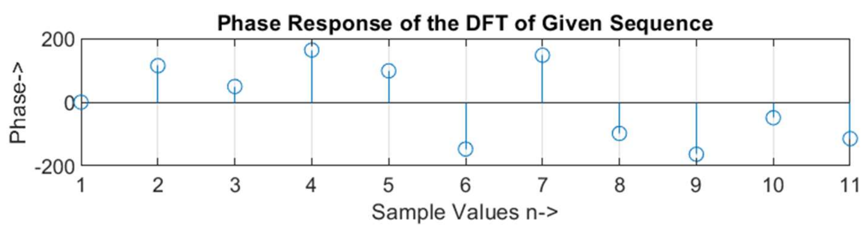
-0.6423 - 0.4128i -0.6423 + 0.4128i -0.0846 - 0.5883i -1.1549 - 0.3391i   0.3413 - 0.3938i

Column 11

-1.4595 - 3.1958i







c. Without Built-In Function, Continuous

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)heaviside(t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

```
>> unit

-(exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i - 1i))/(100*pi)


(imag((exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i - 1i))/pi)^2/10000 + real((exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i - 1i))/pi)^2/10000)^(1/2)
```

```
-atan(imag((exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i - 1i))/pi)/real((exp(-
pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i - 1i))/pi))
```

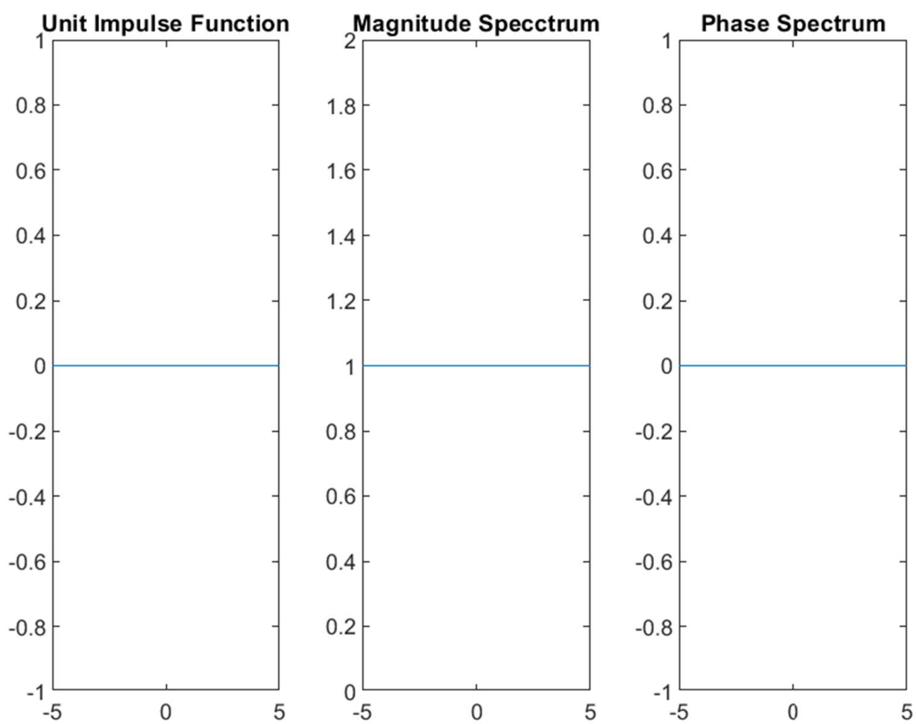
2. Impulse Signal

a. Continuous, With Built-in Function

Code:

```
t = -1 : 0.01 : 1;
syms t;
f = dirac(t);
subplot(131);
fplot(f);
title('Unit Impulse Function');
r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Specctrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
%discrete unit impulse signal fourier transform
N = -5 : 5;
```

```
a = [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = -5 : 5;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Unit Impulse Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> disimp

The Length of the Sequence of [a] is:

11

The Fourier Transform is

Columns 1 through 5

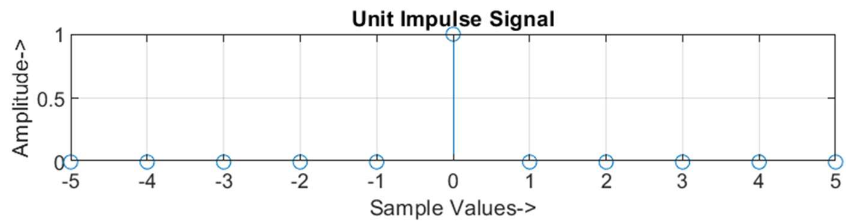
1.0000 + 0.0000i -0.9595 - 0.2817i   0.8413 + 0.5406i -0.6549 - 0.7557i   0.4154 + 0.9096i

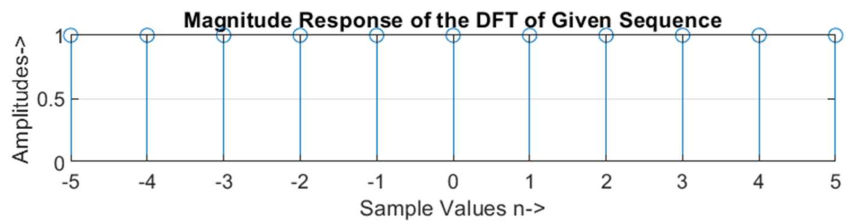
Columns 6 through 10

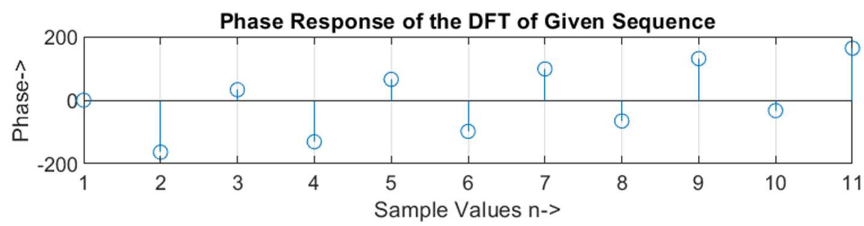
-0.1423 - 0.9898i -0.1423 + 0.9898i   0.4154 - 0.9096i -0.6549 + 0.7557i   0.8413 - 0.5406i

Column 11

-0.9595 + 0.2817i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)dirac(t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

>> imp

sign(t)/2

(imag(sign(t))^2/4 + real(sign(t))^2/4)^(1/2)

-atan(imag(sign(t))/real(sign(t)))

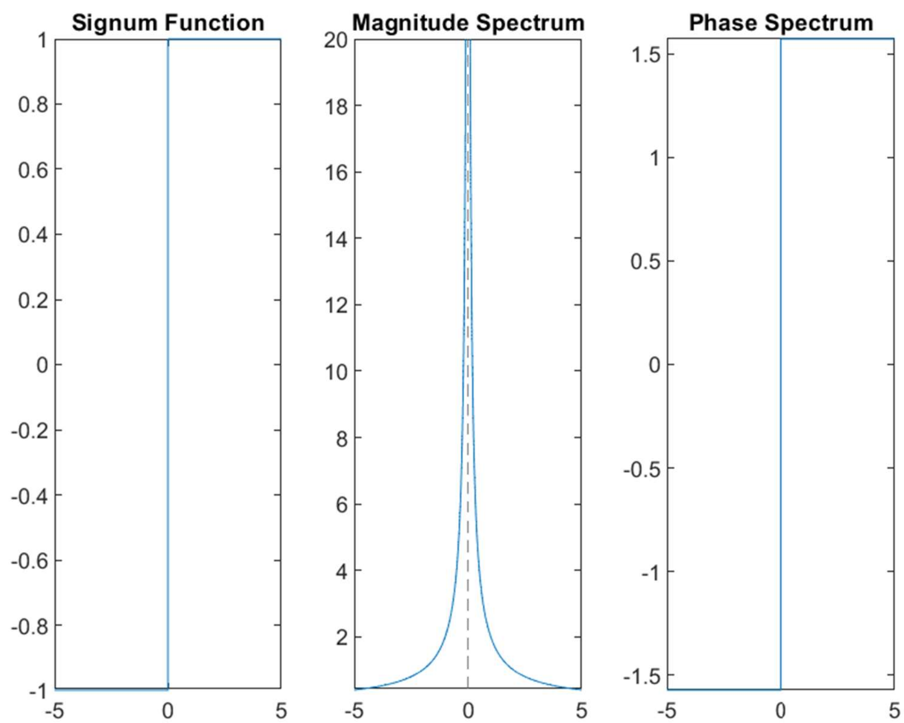
3. Signum Function

a. Continuous, With Built-in Function

Code:

```
t = -2 : 0.002 : 2;
f = sign(t);
subplot(131);
fplot(f);
title('Signum Function');
r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
%discrete unit step signal fourier transform
N = -5 : 5;
a = [-1, -1, -1, -1, -1, 0, 1, 1, 1, 1, 1];
```

```
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = -5 : 5;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Signum Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> dissign

The Length of the Sequence of [a] is:

11

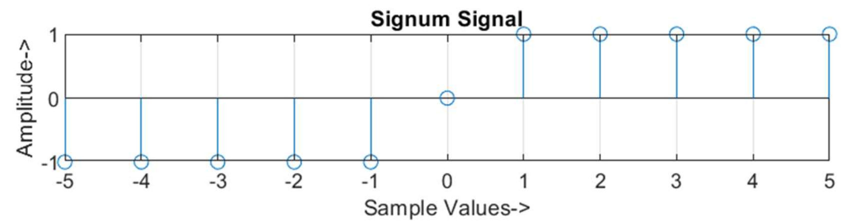
The Fourier Transform is

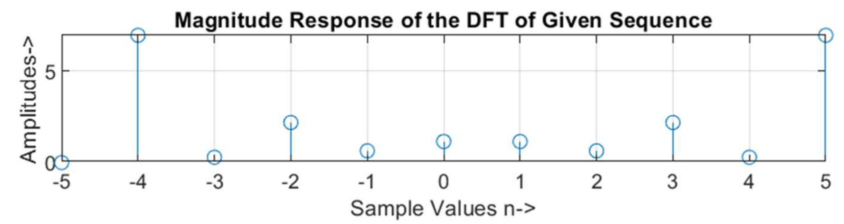
Columns 1 through 6

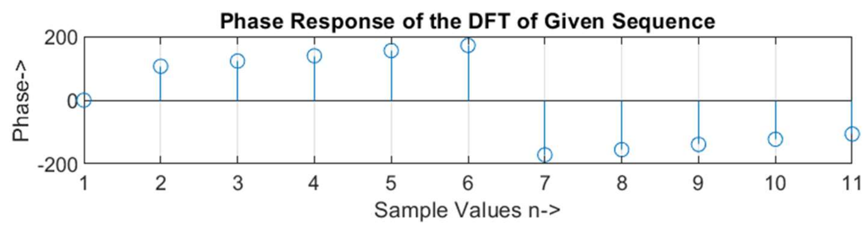
0.0000 + 0.0000i -1.9595 + 6.6734i -0.1587 + 0.2470i -1.6549 + 1.4339i -0.5846 + 0.2670i -1.1423 + 0.1642i

Columns 7 through 11

-1.1423 - 0.1642i -0.5846 - 0.2670i -1.6549 - 1.4339i -0.1587 - 0.2470i -1.9595 - 6.6734i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)sign(t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

>> sign

*(exp(-pi*t*100i)*sign(t)*1i)/(100*pi)*

*(imag((exp(-pi*t*100i)*sign(t)*1i)/(100*pi))^2 + real((exp(-pi*t*100i)*sign(t)*1i)/(100*pi))^2)^(1/2)*

*-atan(imag((exp(-pi*t*100i)*sign(t)*1i)/(100*pi))/real((exp(-pi*t*100i)*sign(t)*1i)/(100*pi)))*

4. Exponential Signal

a. Continuous, With Built-in Function

Code:

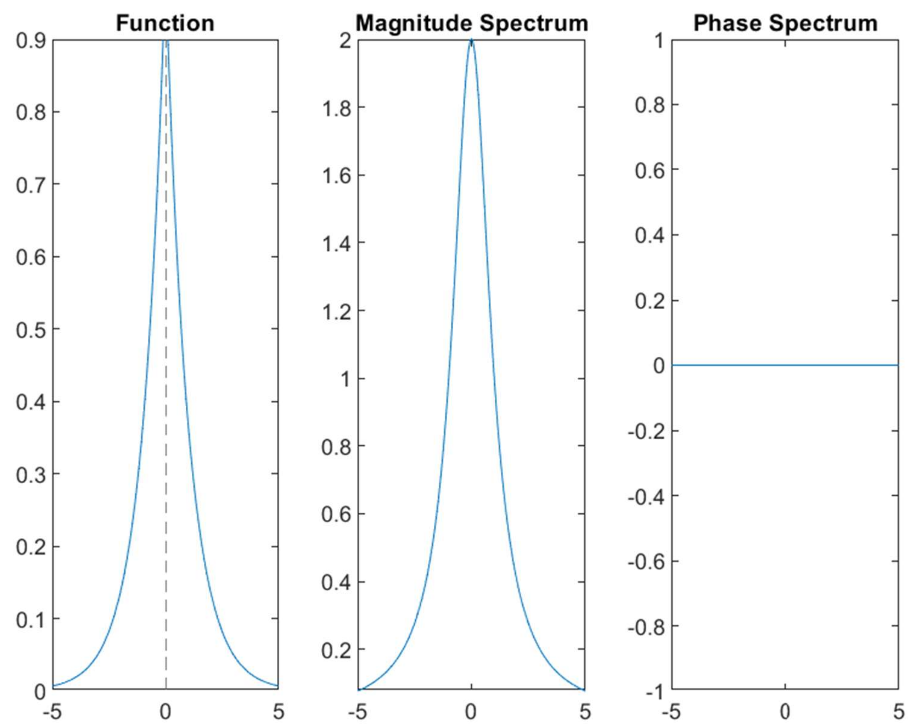
```
a=sym(1);

t = 0 : 0.002 : 1;
syms x;
syms t;

f = exp(-a * abs(x));
subplot(1,3,1);
fplot((f));
title('Function');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(1,3,2);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(1,3,3);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
%discrete unit step signal fourier transform
N = -5 : 5;
a = [exp(N)];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = -5 : 5;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Exponential Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> disexp

The Length of the Sequence of [a] is:

11

The Fourier Transform is

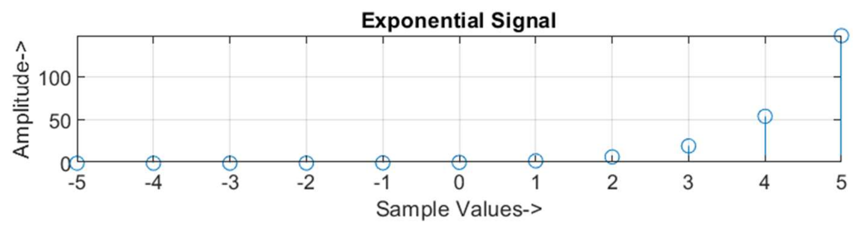
*1.0e+02 **

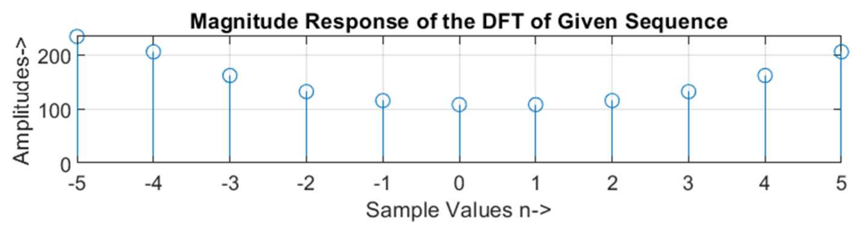
Columns 1 through 6

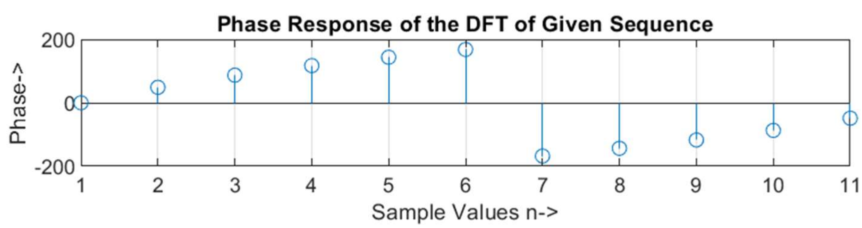
2.3478 + 0.0000i 1.3605 + 1.5538i 0.0850 + 1.6271i -0.6106 + 1.1846i -0.9386 + 0.6936i -1.0699 + 0.2271i

Columns 7 through 11

-1.0699 - 0.2271i -0.9386 - 0.6936i -0.6106 - 1.1846i   0.0850 - 1.6271i   1.3605 - 1.5538i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)exp(-t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

```
>> exp

-(exp(-t)*exp(-pi*t*100i))/(1 + pi*100i)

(imag((exp(-t)*exp(-pi*t*100i))/(1 + pi*100i))^2 + real((exp(-t)*exp(-pi*t*100i))/(1 + pi*100i))^2)^(1/2)

-atan(imag((exp(-t)*exp(-pi*t*100i))/(1 + pi*100i))/real((exp(-t)*exp(-pi*t*100i))/(1 + pi*100i)))
```

5. Unit Ramp Signal

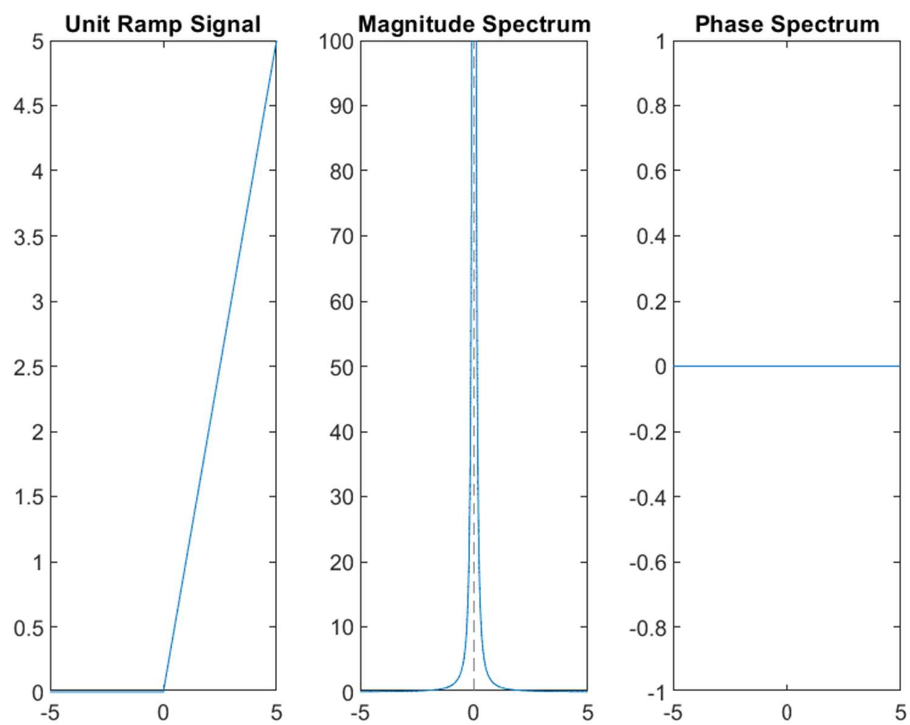
a. Continuous, With Built-in Function

Code:

```
%Continuous Fourier Transform Unit Step Signal
t = -1 : 0.002 : 1;
syms t;
f = t * heaviside(t);
subplot(131);
fplot(f);
title('Unit Ramp Signal');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
%discrete unit step signal fourier transform
N = 0 : 5;
a = [N];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = 0 : 5;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Unit Ramp Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

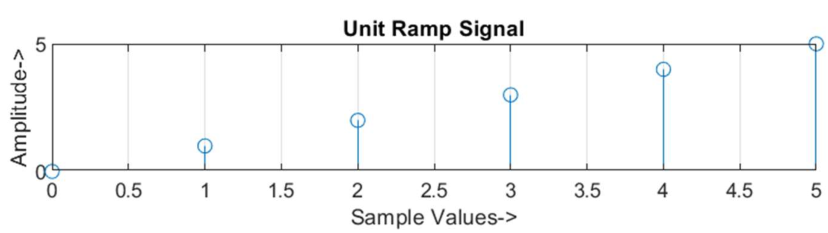
>> disramp

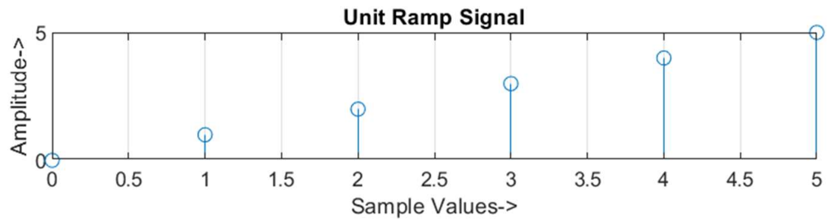
The Length of the Sequence of [a] is:

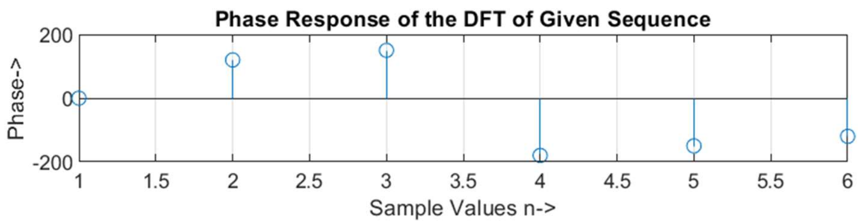
6

The Fourier Transform is

15.0000 + 0.0000i -3.0000 + 5.1962i -3.0000 + 1.7321i -3.0000 - 0.0000i -3.0000 - 1.7321i -3.0000 - 5.1962i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)heaviside(t) * t;
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

>> ramp

*(exp(-pi*t*100i)*heaviside(t)*(1 + pi*t*100i - exp(pi*t*100i)))/(10000*pi^2)*

*(imag((exp(-pi*t*100i)*heaviside(t)*(1 + pi*t*100i - exp(pi*t*100i)))/pi^2)^2/100000000 + real((exp(-pi*t*100i)*heaviside(t)*(1 + pi*t*100i - exp(pi*t*100i)))/pi^2)^2/100000000)^(1/2)*

*-atan(imag((exp(-pi*t*100i)*heaviside(t)*(1 + pi*t*100i - exp(pi*t*100i)))/pi^2)/real((exp(-pi*t*100i)*heaviside(t)*(1 + pi*t*100i - exp(pi*t*100i)))/pi^2))*

6. Parabolic Signal

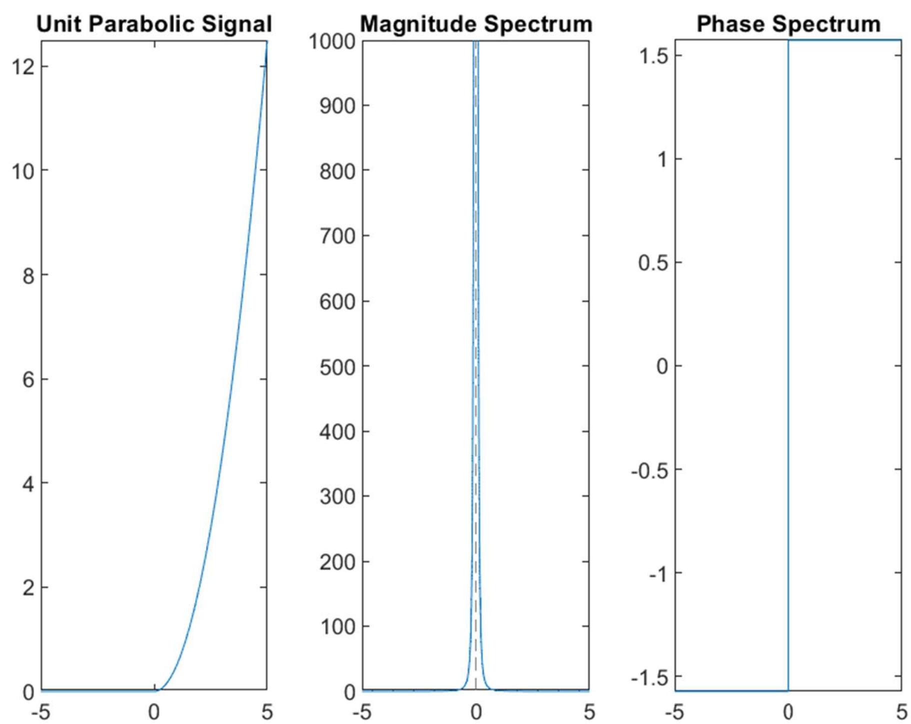
a. Continuous, With Built-in Function

Code:

```
t = 0 : 0.002 : 1;
syms t;
f = ((t ^ 2) / 2) * heaviside(t);
subplot(131);
fplot(f);
title('Unit Parabolic Signal');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
N = 0 : 5;
f = ((N .^ 2) / 2);
a = f;
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = 0 : 5;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Unit Parabolic Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

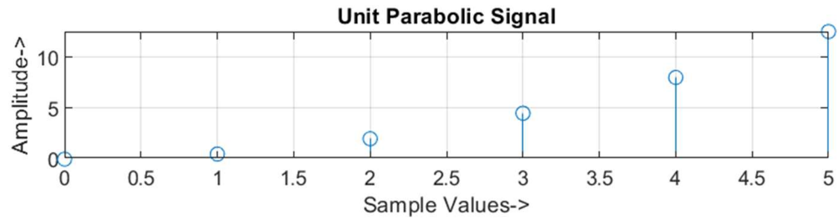
>> dispar

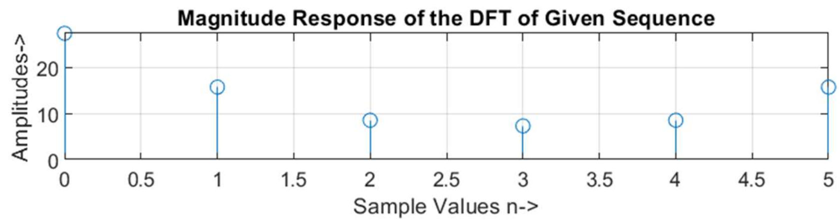
The Length of the Sequence of [a] is:

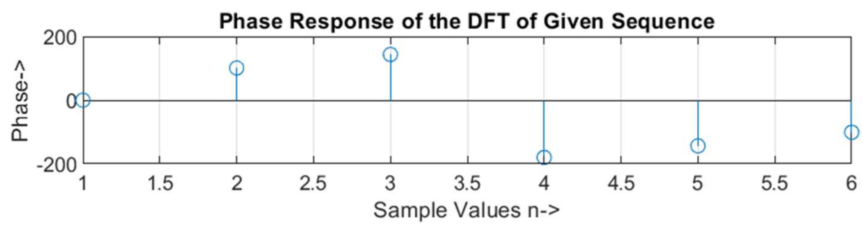
6

The Fourier Transform is

27.5000 + 0.0000i -3.0000 +15.5885i -7.0000 + 5.1962i -7.5000 - 0.0000i -7.0000 - 5.1962i -3.0000 -15.5885i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)heaviside(t) * ((t ^ 2) / 2);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

>> par

*(exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i + 100*pi*t + t^2*pi^2*5000i - 1i))/(1000000*pi^3)*

*(imag((exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i + 100*pi*t + t^2*pi^2*5000i - 1i))/pi^3)^2/1000000000000 + real((exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i + 100*pi*t + t^2*pi^2*5000i - 1i))/pi^3)^2/1000000000000)^(1/2)*

*-atan(imag((exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i + 100*pi*t + t^2*pi^2*5000i - 1i))/pi^3)/real((exp(-pi*t*100i)*heaviside(t)*(exp(pi*t*100i)*1i + 100*pi*t + t^2*pi^2*5000i - 1i))/pi^3))*

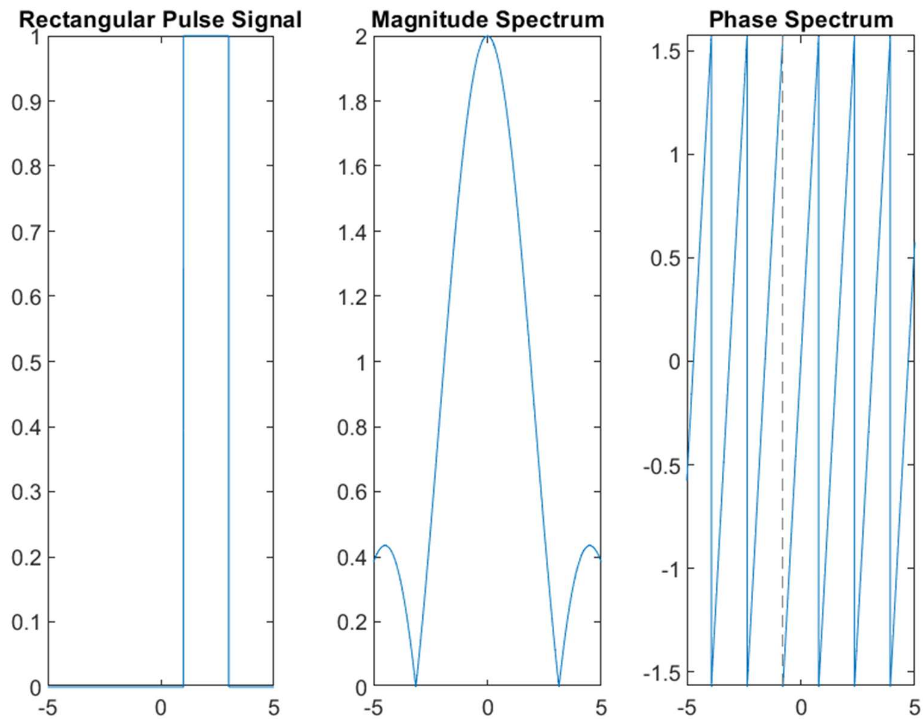
7. Rectangular Impulse Signal
a. Continuous, With Built-in Function

Code:

```
t = 0 : 0.01 : 4;
syms t;
a = sym(1);
b = sym(3);
f = rectangularPulse(a, b, t);
subplot(131);
fplot(f);
title('Rectangular Pulse Signal');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
N = 0 : 10;
a = [0, 0, 0, 3, 3, 3, 3, 3, 0, 0, 0];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = 0 : 10;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Rectangular Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
```

```
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> disrect

The Length of the Sequence of [a] is:

11

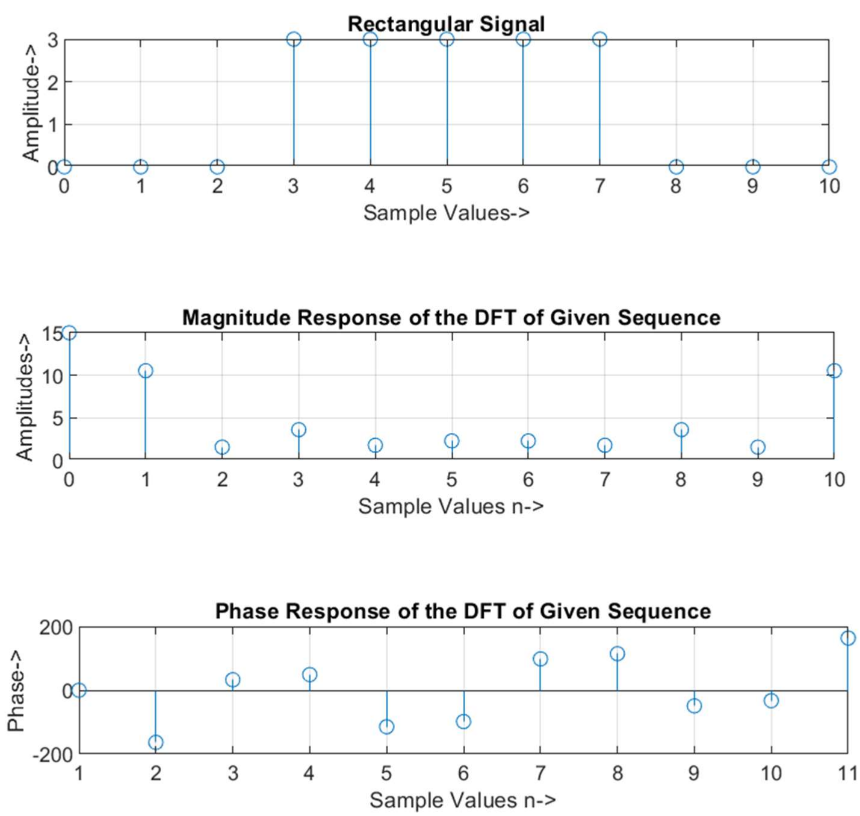
The Fourier Transform is

Columns 1 through 6

15.0000 + 0.0000i -10.1131 - 2.9695i 1.3152 + 0.8452i 2.3646 + 2.7289i -0.7407 - 1.6219i - 0.3260 - 2.2672i

Columns 7 through 11

-0.3260 + 2.2672i -0.7407 + 1.6219i 2.3646 - 2.7289i 1.3152 - 0.8452i -10.1131 + 2.9695i



c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
b = sym(3);
x = @(t) * rectangularPulse(a, b, t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

>> rect

*int(exp(-pi*t*100i)*rectangularPulse(1, 3, t), t)*

*(int(imag(exp(-pi*t*100i))*rectangularPulse(1, 3, t), t)^2 + int(real(exp(-pi*t*100i))*rectangularPulse(1, 3, t), t)^2)^(1/2)*

*-atan(int(imag(exp(-pi*t*100i))*rectangularPulse(1, 3, t), t)/int(real(exp(-pi*t*100i))*rectangularPulse(1, 3, t), t))*

8. Triangular Impulse Signal

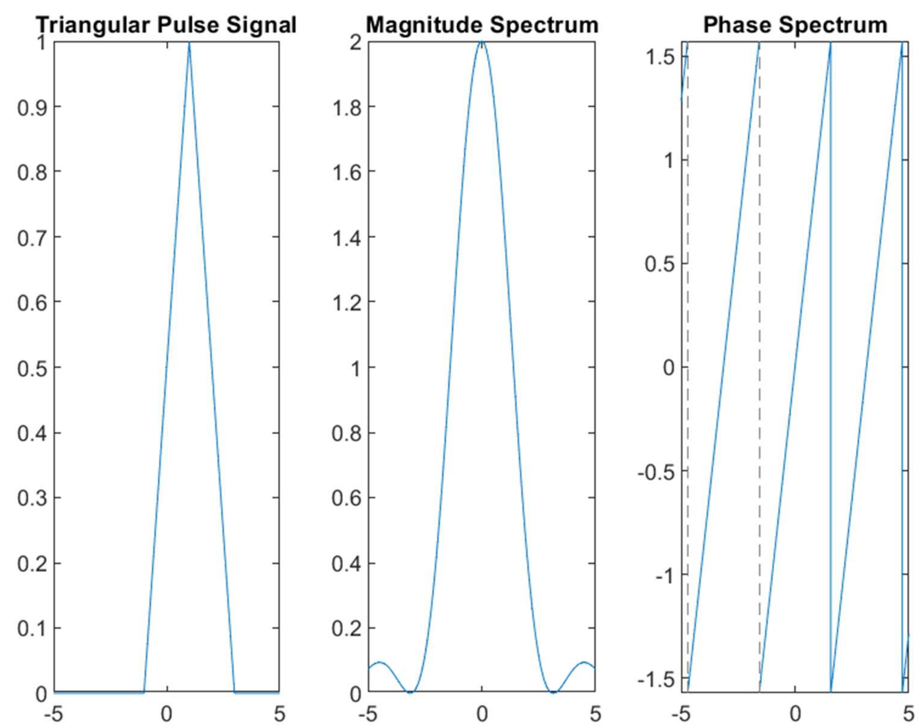
a. Continuous, With Built-in Function

Code:

```
t = 0 : 0.01 : 5;
a = sym(1);
b = sym(2);
c = sym(3);
f = triangularPulse(a, b, c, t);
subplot(131);
fplot(f);
title('Triangular Pulse Signal');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
N = 0 : 10;
a = [0, 0, 0, 1, 2, 3, 2, 1, 0, 0, 0];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = 0 : 10;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Triangular Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> distri

The Length of the Sequence of [a] is:

11

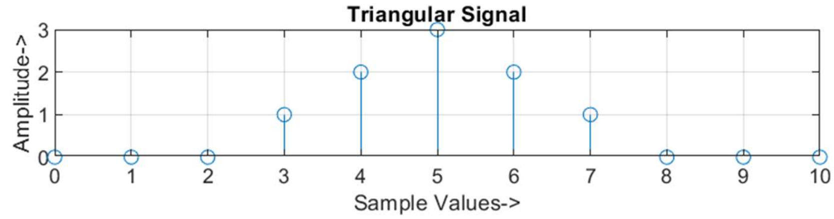
The Fourier Transform is

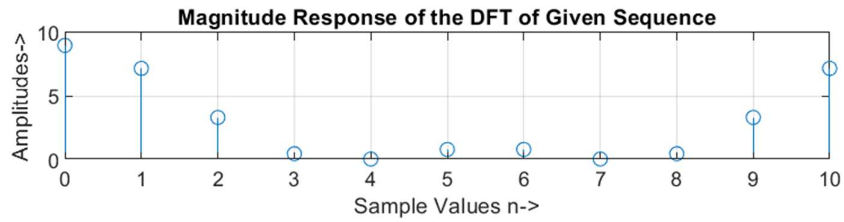
Columns 1 through 6

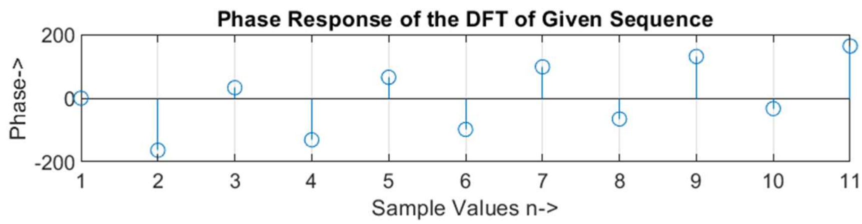
9.0000 + 0.0000i -6.9044 - 2.0273i 2.8198 + 1.8122i -0.3351 - 0.3868i 0.0398 + 0.0873i -0.1202 - 0.8359i

Columns 7 through 11

-0.1202 + 0.8359i 0.0398 - 0.0873i -0.3351 + 0.3868i 2.8198 - 1.8122i -6.9044 + 2.0273i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
b = sym(2);
c = sym(3);
x = @(t)triangularPulse(a, b, c, t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

>> tri

*int(exp(-pi*t*100i)*triangularPulse(1, 2, 3, t), t)* *(int(imag(exp(-pi*t*100i))*triangularPulse(1, 2, 3, t), t)^2 + int(real(exp(-pi*t*100i))*triangularPulse(1, 2, 3, t), t)^2)^(1/2)*

*-atan(int(imag(exp(-pi*t*100i))*triangularPulse(1, 2, 3, t), t)/int(real(exp(-pi*t*100i))*triangularPulse(1, 2, 3, t), t))*

9. Sinusoidal Signal

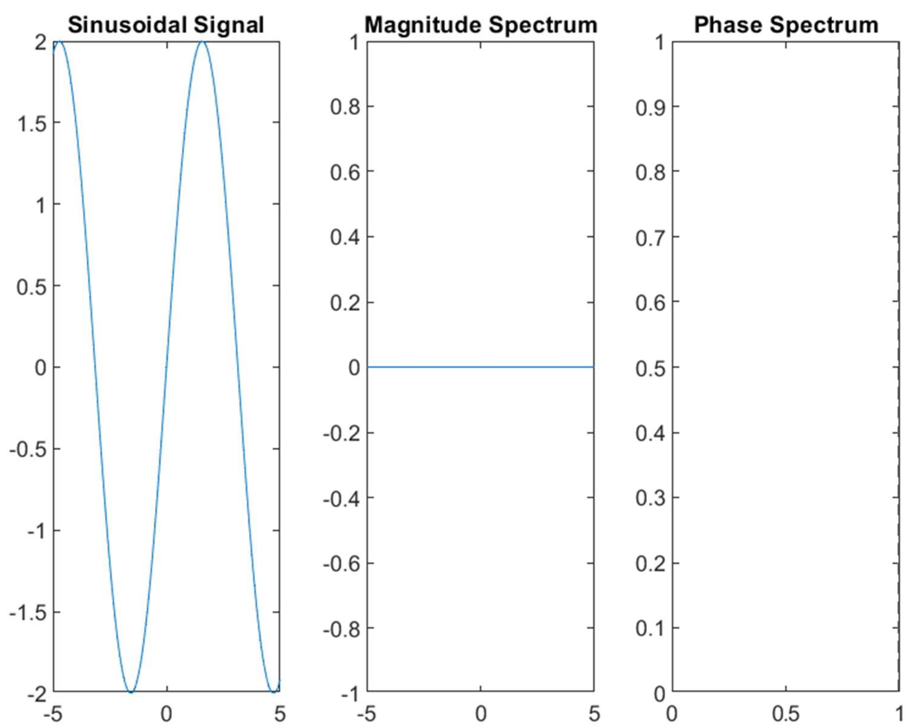
a. Continuous, With Built-in Function

Code:

```
t = -1 : 0.01 : 1;
a = sym(1);
b = sym(2);
syms t;
f = b * sin(a * t);
subplot(131);
fplot(f);
title('Sinusoidal Signal');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
N = -10 : 10;
a = [sin(N)];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = -10 : 10;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Sinusoidal Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> dissin

The Length of the Sequence of [a] is:

21

The Fourier Transform is

Columns 1 through 6

0.0000 + 0.0000i -0.0826 + 0.5480i -0.4691 + 1.5209i -3.4941 + 7.2556i 2.8004 - 4.1074i 1.5343 - 1.6536i

Columns 7 through 12

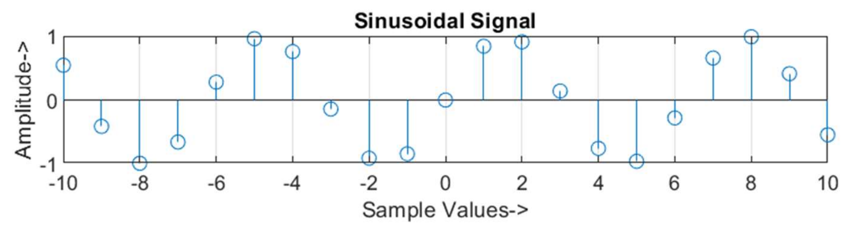
1.2372 - 0.9867i 1.1131 - 0.6427i 1.0507 - 0.4124i 1.0182 - 0.2324i 1.0041 - 0.0752i 1.0041 + 0.0752i

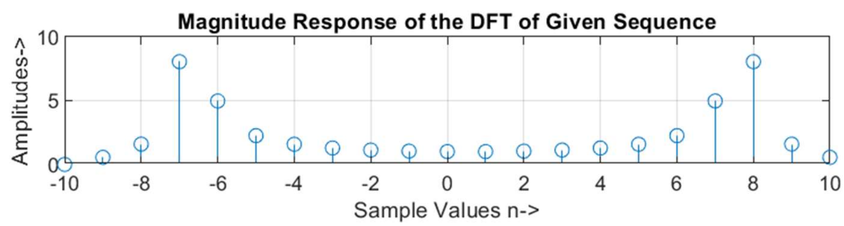
Columns 13 through 18

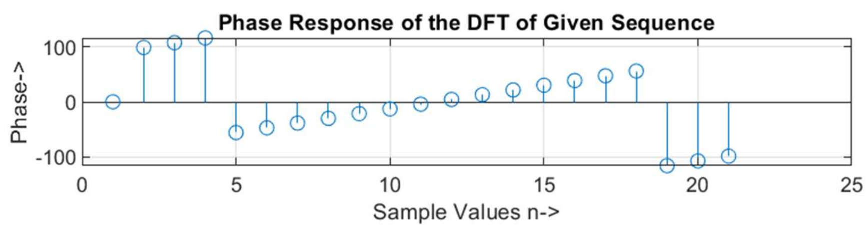
1.0182 + 0.2324i   1.0507 + 0.4124i   1.1131 + 0.6427i   1.2372 + 0.9867i   1.5343 + 1.6536i   2.8004 + 4.1074i

Columns 19 through 21

-3.4941 - 7.2556i -0.4691 - 1.5209i -0.0826 - 0.5480i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)sin(t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

```
>> sin

(exp(-pi*t*100i)*(cos(t) + pi*sin(t)*100i))/(10000*pi^2 - 1)


(imag((exp(-pi*t*100i)*(cos(t) + pi*sin(t)*100i))/(10000*pi^2 - 1))^2 + real((exp(-pi*t*100i)*(cos(t) + pi*sin(t)*100i))/(10000*pi^2 - 1))^2)^(1/2)

-atan(imag((exp(-pi*t*100i)*(cos(t) + pi*sin(t)*100i))/(10000*pi^2 - 1))/real((exp(-pi*t*100i)*(cos(t) + pi*sin(t)*100i))/(10000*pi^2 - 1)))
```

10. Sinc Function

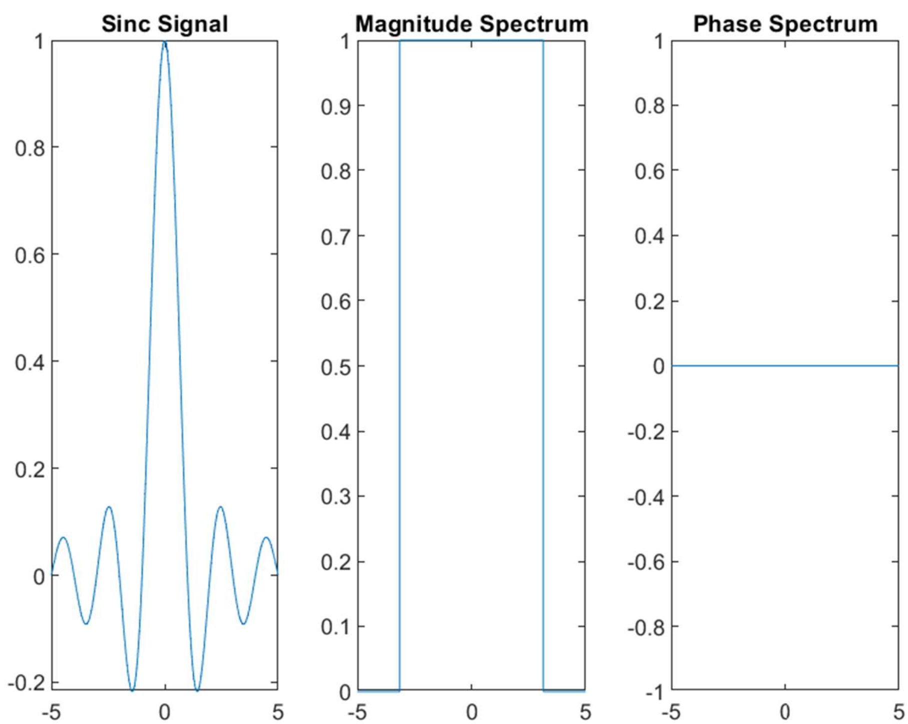
a. Continuous, With Built-in Function

Code:

```
t = -2 : 0.01 : 2;
syms t;
f = sin(pi * t) / (pi * t);
subplot(131);
fplot(f);
title('Sinc Signal');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
N = -10 : 10;
```

```
a = [sinc(N)];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = -10 : 10;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Sinc Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

>> dissinc

The Length of the Sequence of [a] is:

21

The Fourier Transform is

Columns 1 through 6

1.0000 + 0.0000i -0.9888 - 0.1490i   0.9556 + 0.2948i -0.9010 - 0.4339i   0.8262 + 0.5633i -0.7331 - 0.6802i

Columns 7 through 12

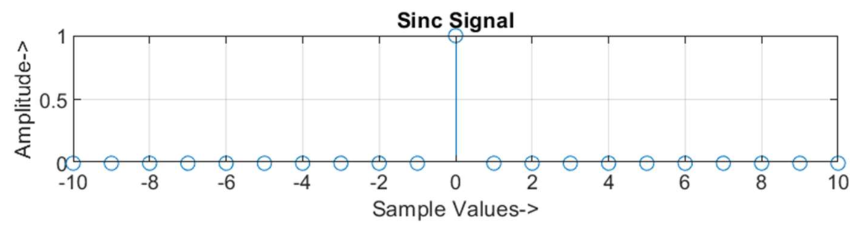
0.6235 + 0.7818i -0.5000 - 0.8660i   0.3653 + 0.9309i -0.2225 - 0.9749i   0.0747 + 0.9972i   0.0747 - 0.9972i

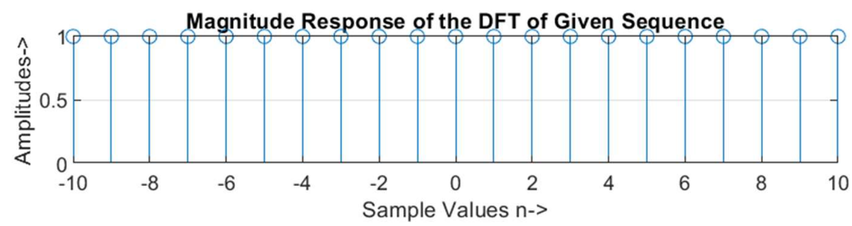
Columns 13 through 18

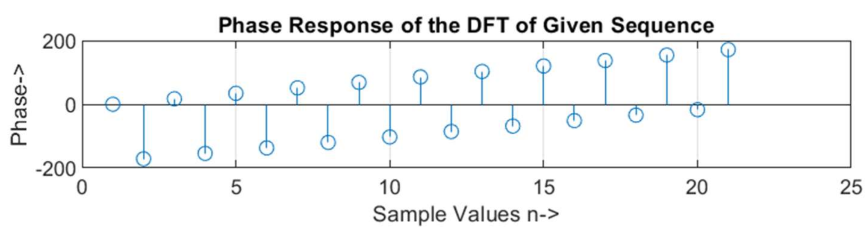
-0.2225 + 0.9749i   0.3653 - 0.9309i -0.5000 + 0.8660i   0.6235 - 0.7818i -0.7331 + 0.6802i   0.8262 - 0.5633i

Columns 19 through 21

-0.9010 + 0.4339i   0.9556 - 0.2948i -0.9888 + 0.1490i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)sinc(t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

>> sinc

*- (ei(-pi*t*99i)*1i)/(2*pi) + (ei(-pi*t*101i)*1i)/(2*pi)*

((imag(ei(-pi*t*99i))/(2*pi) - imag(ei(-pi*t*101i))/(2*pi))^2 + (real(ei(-pi*t*99i))/(2*pi) - real(ei(-pi*t*101i))/(2*pi))^2)^(1/2)

atan((real(ei(-pi*t*99i))/(2*pi) - real(ei(-pi*t*101i))/(2*pi))/(imag(ei(-pi*t*99i))/(2*pi) - imag(ei(-pi*t*101i))/(2*pi)))

11. Sampling Signal

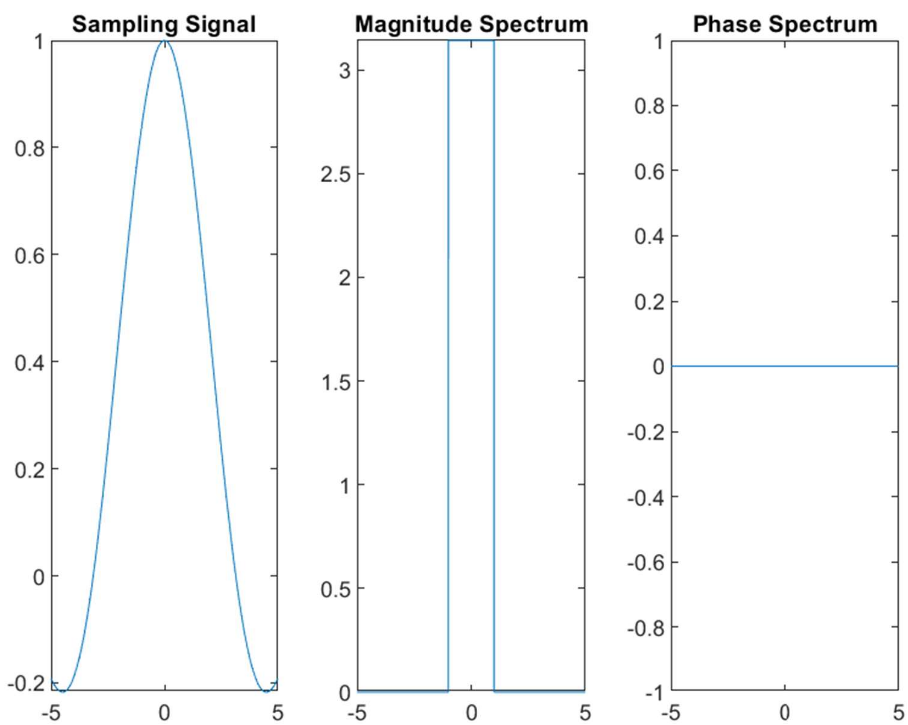
a. Continuous, With Built-in Function

Code:

```
t = -2 : 0.01 : 2;
syms t;
a = sym(1);
f = sin(a * t) / (a * t);
subplot(131);
fplot(f);
title('Sampling Signal');

r = fourier(f);
xr = real(r);
xi = imag(r);
X = sqrt((xr ^ 2) + (xi ^ 2));
subplot(132);
fplot(X);
title('Magnitude Spectrum');
P = (-atan(xi / xr));
subplot(133);
fplot(P);
title('Phase Spectrum');
```

Result:



b. Discrete, Without Built-in Function

Code:

```
N = -5 : 5;
a = [sinc(1/pi * N)];
N = length(a);
disp('The Length of the Sequence of [a] is: ');
disp(N);
for k = 1 : N
    y(k) = 0;
    for i = 1 : N
        y(k) = y(k) + a(i) * exp((-2 * pi * 1i / N) * ((i - 1) * (k - 1)));
    end
end

k = 1 : N;
M = -5 : 5;
subplot(311);
stem(M, a);
grid;
xlabel('Sample Values->');
ylabel('Amplitude->');
title('Sampling Signal');
disp('The Fourier Transform is');
disp(y(k));
subplot(312);
stem(M, abs(y(k)));
grid;
xlabel('Sample Values n->');
ylabel('Amplitudes->');
title('Magnitude Response of the DFT of Given Sequence');
subplot(313);
stem(angle(y(k)) * (180 / pi));
grid;
xlabel('Sample Values n->');
ylabel('Phase->');
title('Phase Response of the DFT of Given Sequence');
```

Result:

The Length of the Sequence of [a] is:

11

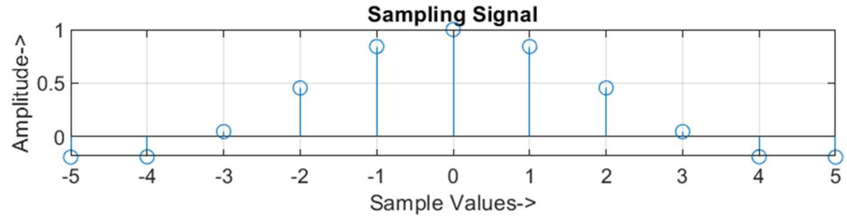
The Fourier Transform is

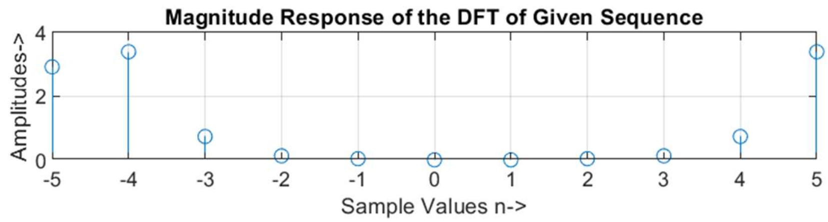
Columns 1 through 6

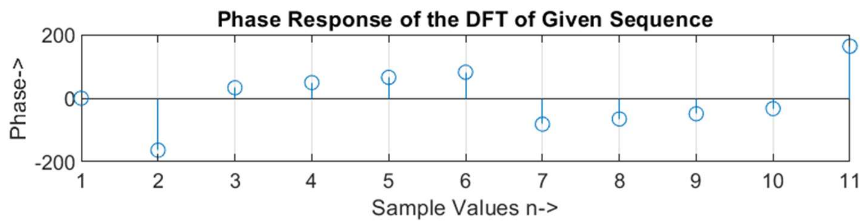
2.9243 + 0.0000i -3.2584 - 0.9568i   0.6264 + 0.4025i   0.0917 + 0.1058i   0.0213 + 0.0467i   0.0020 + 0.0139i

Columns 7 through 11

0.0020 - 0.0139i   0.0213 - 0.0467i   0.0917 - 0.1058i   0.6264 - 0.4025i -3.2584 + 0.9568i







c. Continuous, Without Built-in Function

Code:

```
syms t;
w = 2 * pi * 50;
a = sym(1);
x = @(t)sinc((1 / pi) * t);
y = @(t)x(t) * exp(-1j * w * t);
z = int(y(t), t);
xr = real(z);
xi = imag(z);
X = sqrt((xr ^ 2) + (xi ^ 2));
P = (-atan(xi / xr));
disp(z);
disp(X);
disp(P);
```

Result:

>> samp

-

(ei((pi*t*1795705689808975741i)/18014398509481984)*9007199254740992i)/(573416113922265

9*pi) + (ei(-
(pi*t*1807174012087421059i)/18014398509481984)*9007199254740992i)/(5734161139222659*p
i)

(((9007199254740992*imag(ei(-
(pi*t*1807174012087421059i)/18014398509481984)))/(5734161139222659*pi) -
(9007199254740992*imag(ei(-
(pi*t*1795705689808975741i)/18014398509481984)))/(5734161139222659*pi))^2 +
((9007199254740992*real(ei(-
(pi*t*1807174012087421059i)/18014398509481984)))/(5734161139222659*pi) -
(9007199254740992*real(ei(-
(pi*t*1795705689808975741i)/18014398509481984)))/(5734161139222659*pi))^2)^(1/2)

atan(((9007199254740992*real(ei(-
(pi*t*1807174012087421059i)/18014398509481984)))/(5734161139222659*pi) -
(9007199254740992*real(ei(-
(pi*t*1795705689808975741i)/18014398509481984)))/(5734161139222659*pi))/((9007199254740
992*imag(ei(-(pi*t*1807174012087421059i)/18014398509481984)))/(5734161139222659*pi) -
(9007199254740992*imag(ei(-
(pi*t*1795705689808975741i)/18014398509481984)))/(5734161139222659*pi)))