EEE321 - Signals & Systems

Lab Assignment II

Mohammed Abed (21903608)

Part 1:

- Code for the function SUMCS:

```
function [xs] = SUMCS(t, A, omega)

%SUMCS Computes xs as the linear sum of M complex exponentials sampled at

%time points given from t, with amplitude information in A and angular

%frequency information in omega.

%   t: 1×N vector that contains the time instants over which xs(t) is computed

%   A: 1×M complex-valued vector. i^{th} element is A_i

%   omega: 1×M vector. i^{th} element is ω_i

M = length(A);

xs = zeros(size(t));

for ii = 1:M

xs = xs + A(ii)*exp(1j*omega(ii)*t);

end

end
```

- Code for Computing Xs using the function SUMCS:

```
t = 0:0.001:1;

n = mod(21903608, 41); n=14

A = 3*rand(1,n) + 3j*rand(1,n);

omega = pi*rand(1,n);

y = SUMCS(t, A, omega);

%% Plots for Real and Imaginary Parts of X_s

figure

subplot(1,2,1)

plot(t,real(y),'r')

grid on

title("Real Part of X_s")

xlabel("t")

ylabel("Real(Xs)")

subplot(1,2,2)

plot(t, imag(y), 'b')

grid on

title("Imaginary Part of X_s")

xlabel("t")

ylabel("Imag(Xs)")

%% Plots for Magnitude and Phase of X_s
```

```
figure
subplot(1,2,1)
plot(t, abs(y))
title("Magnitude of X_s")
xlabel("t")
ylabel("|X_s|")
grid on
subplot(1,2,2)
plot(t, angle(y))
title("Phase of X_s")
xlabel("t")
ylabel("\angle{X_s}")
grid on
```

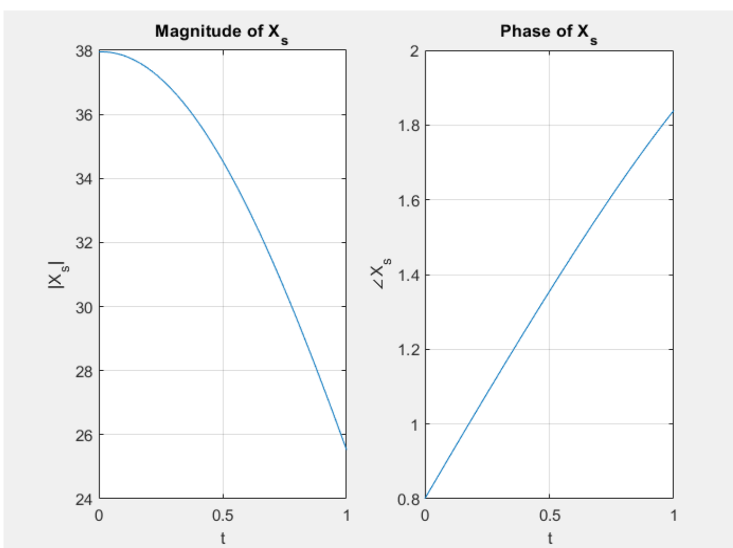

Figure 1: Magnitude And Phase of X_s

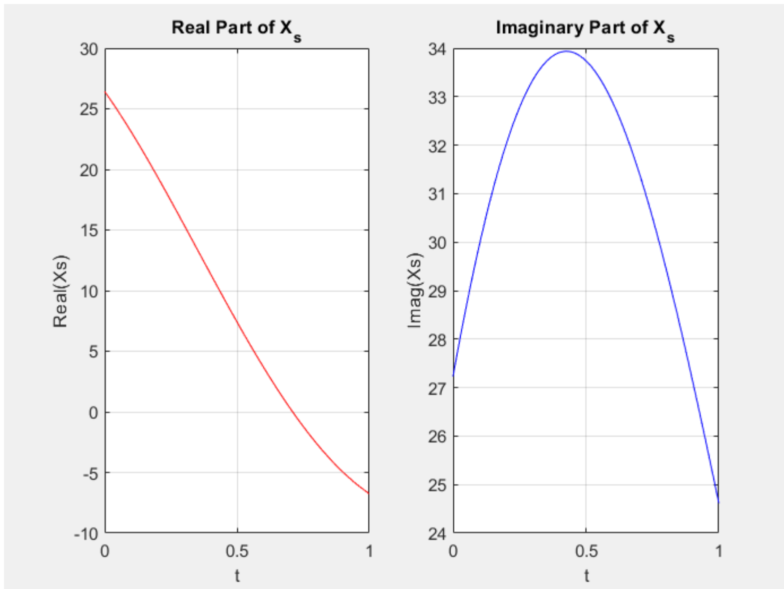

Figure 2: Real and Imaginary Parts of X_s

Part 2:

For this part, I attached a separate PDF file for graphing the function and solving the integral.

Part 3:

- Code for the function FSWave:

```
function [xt] = FSWave(t,K,T,W)

%The function Xt computes the Fourier Synthesis of the function x(t).

%In other words, the function computes X_telda(t) i.e.

% the approximation of the fourier series over -1.5T<t<-1.5T.

% t: denotes the time grid over which ˜ x(t ) is computed.

% xt: denotes the values of ˜ x(t ) computed over t.

% K, T and W: denote the parameters K, T and W that appear in Eq. 4 and Eq. 5.

s = 1;

for ii = -K:K

x_t = @(t)(1-2*t.^2).*exp(-1i*2*pi*ii*t/T);

x_k(s) = (1/T)*integral(x_t,-W/2,W/2);

s = s+1;

end

xt = SUMCS(t,x_k,2*pi*(-K:1:K)/T);

End
```

- Code for Computing X_{ti}'s using the function FSWave:

```
D11 = mod(21903608,11);

D5 = mod(21903608,5);

T = 2;

W=1;

K1 = 20 + D11;

t = [-5:0.001:5];

xt1 = FSWave(t,K,T,W);

figure;

title("The signal X_1_t");

subplot(2,1,1)

plot(t,real(xt1));

xlabel("t");

ylabel("Real Part of X_t_1");

grid on

subplot(2,1,2);

plot(t,imag(xt1))

xlabel("t");

ylabel("Imaginary Part of X_t_1");

grid on

K2 = 2 + D5;
```

```
K3 = 7 + D5;

K4 = 15 + D5;

K5 = 50 + D5;

K6 = 100 + D5;

xt2 = FSWave(t,K2,T,W);

xt3 = FSWave(t,K3,T,W);

xt4 = FSWave(t,K4,T,W);

xt5 = FSWave(t,K5,T,W);

xt6 = FSWave(t,K6,T,W);

figure;

plot(t,real(xt2));

xlabel("t");

ylabel("X_t_2");

grid on

title("The signal X_t_2 (K = 5)");

figure;

plot(t,real(xt3));

title("The signal X_t_3 (K = 10)");

xlabel("t");

ylabel("X_t_3");

grid on
```

```
figure;
plot(t,real(xt4));
title("The signal X_t_4 (K = 18)");
xlabel("t");
ylabel("X_t_4");
grid on
figure;
plot(t,real(xt5));
title("The signal X_t_5 (K = 53)");
xlabel("t");
ylabel("X_t_5");
grid on
figure;
plot(t,real(xt6));
title("The signal X_t_6 (K = 103)");
xlabel("t");
ylabel("X_t_6");
grid on
```

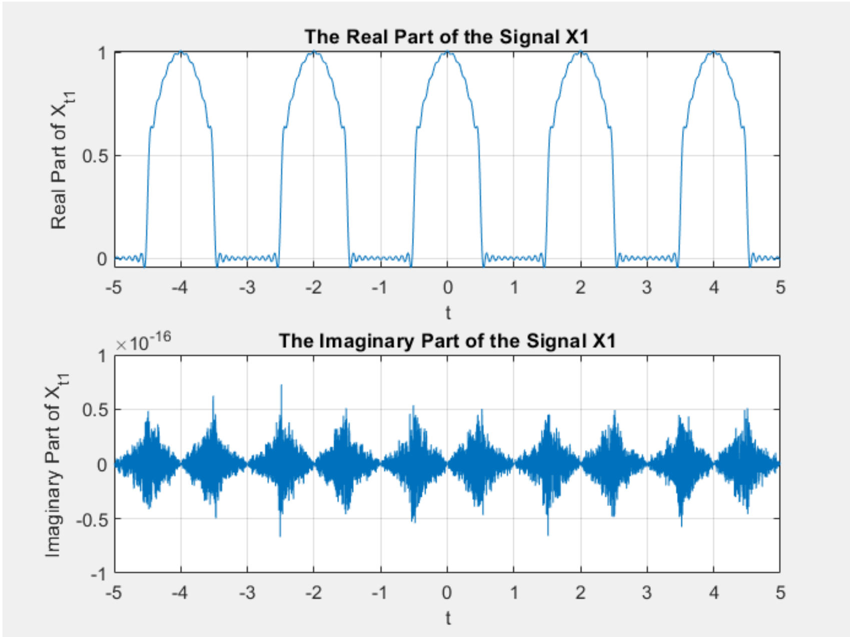

Figure 4: Real and Imaginary Parts of $X_1(t)$ [K=21]

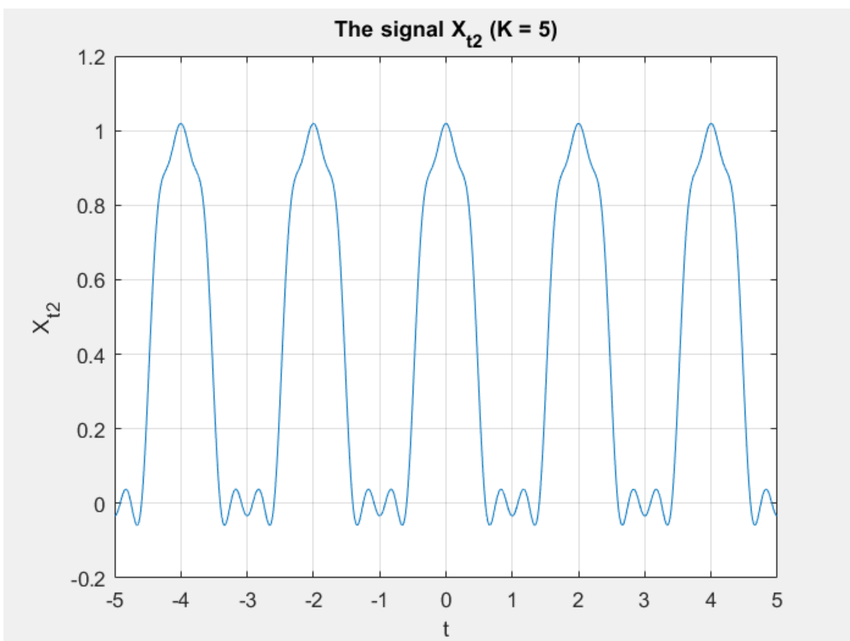

Figure 5: Signal $X_2(t)$ [K=5]

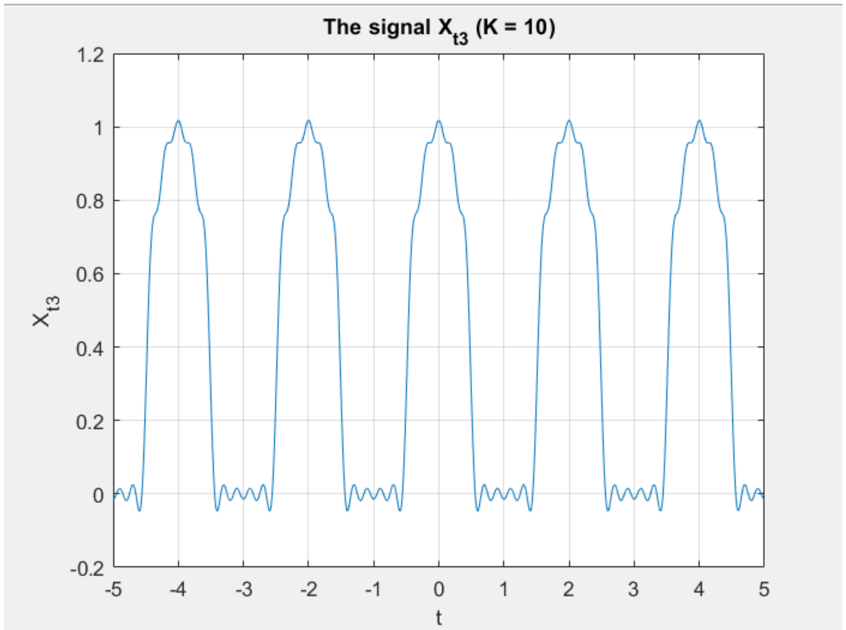

Figure 6: Signal X_3(t) [K=10]

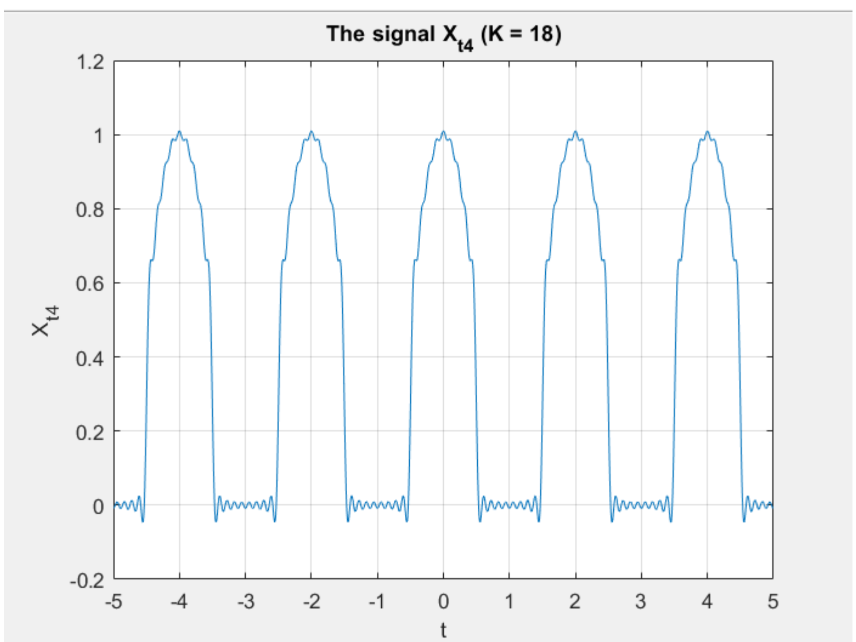

Figure 7: Signal X_4(t) [K=18]

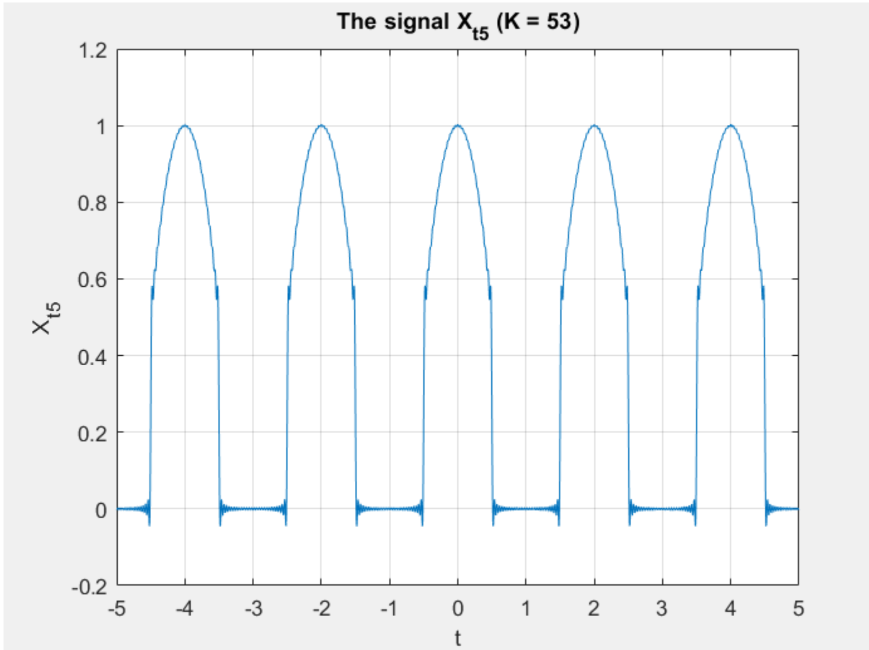

Figure 8: Signal $X_5(t)$ [K=53]

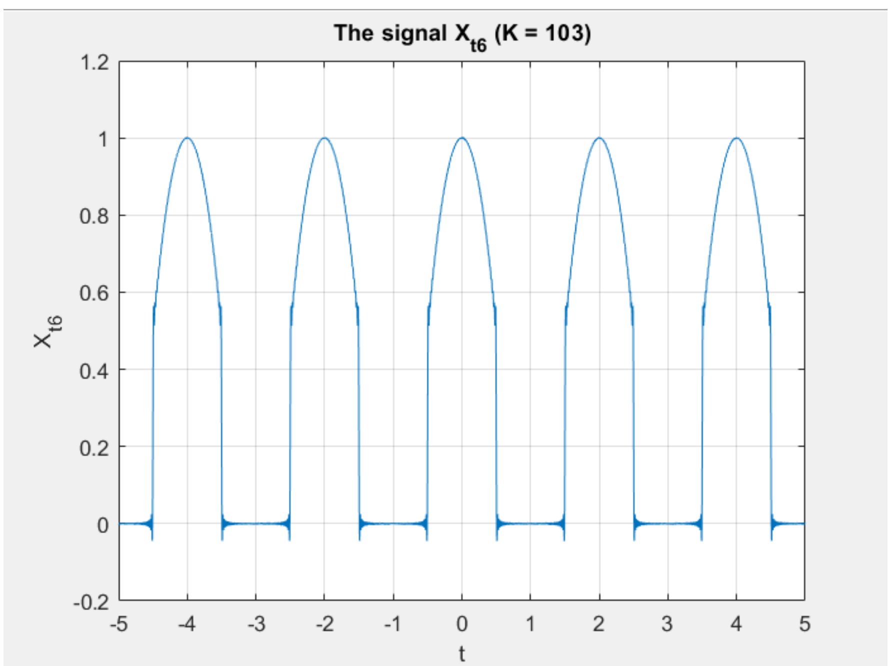

Figure 9: Signal $X_6(t)$ [K=103]

Part 4

a) This operation results in a reflection around the y-axis. For this particular function, there is no change since it is an even function. However, for other functions, the **FSWave** will be indexed differently (starting from K until -K instead of -K until K)

Code:

```
function [xt] = FSWave(t,K,T,W)

s = 1;

for ii = -K:K

x_t = @(t)(1-2*t.^2).*exp(-1i*2*pi*-ii*t/T);

x_k(s) = (1/T)*integral(x_t,-W/2,W/2);

s = s+1;

end

xt = SUMCS(t,x_k,2*pi*(-K:1:K)/T);

End
```

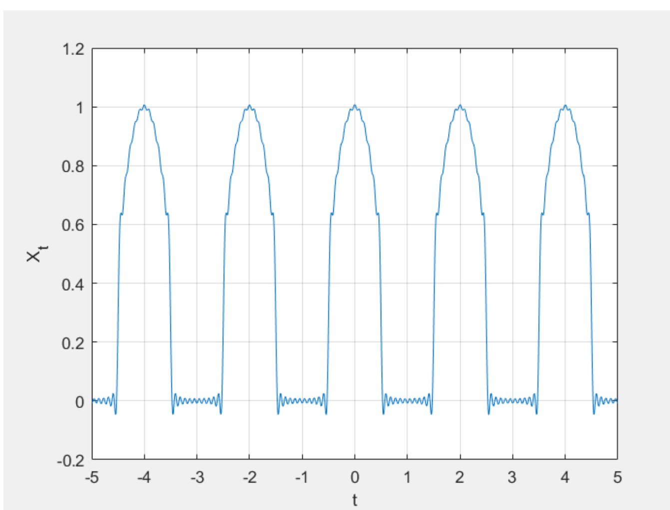

Figure 10: Signal X(t) [K=21]

b) This operation results in a shift to the right by $t_0 = 0.6$. To achieve that, the iteration of the loop was changed.

Code:

```
function [xt] = FSWave(t,K,T,W,t0)

s = 1;

for ii = -K:K

x_t = @(t)(1-2*t.^2).*exp(-1i*2*pi*ii*(t+t0)/T);

x_k(s) = (1/T)*integral(x_t,-W/2,W/2);

s = s+1;

end

xt = SUMCS(t,x_k,2*pi*(-K:1:K)/T);

end
```

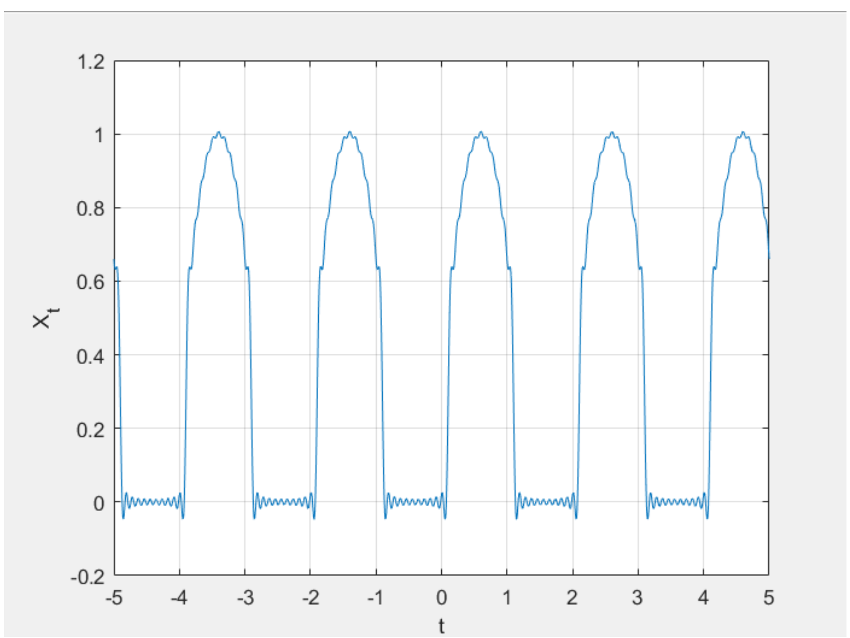

Figure 11: Signal X(t) [K=21] shifted to the right with $t_0 = 0.6$

c) This operation corresponds to taking a derivative with respect to t, and can be achieved by adding the term $jk2\pi/T$ to the loop.

Code:

```
function [xt] = FSWave(t,K,T,W)

s = 1;

for ii = -K:K

x_t = @(t)(1-2*t.^2).*(1i*ii*2*pi/T).*exp(-1i*2*pi*ii*t/T);

x_k(s) = (1/T)*integral(x_t,-W/2,W/2);

s = s+1;

end

xt = SUMCS(t,x_k,2*pi*(-K:1:K)/T);

End
```

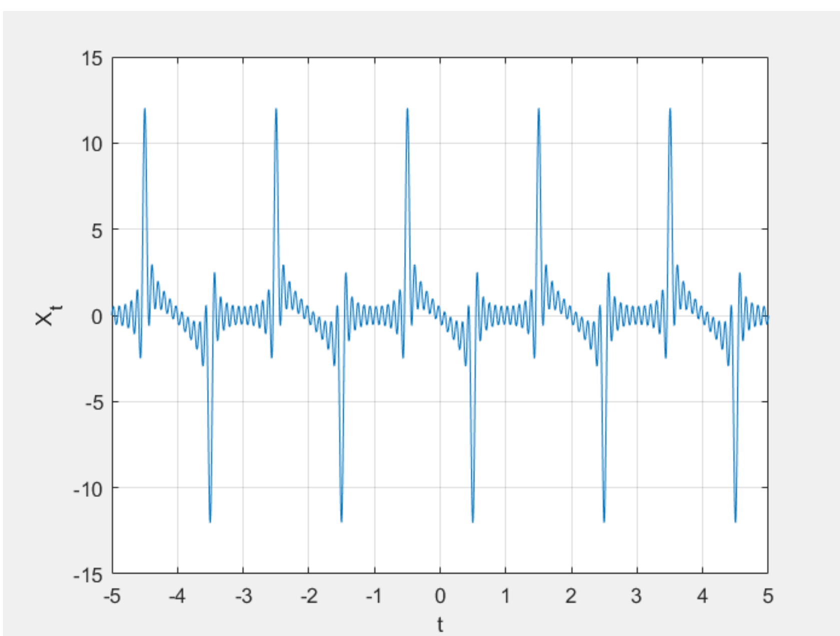

Figure 12: The Derivative of the Original Signal X_t

d) This operation corresponds to the conjugate symmetry property of real signal, and it is achieved by adding if statements to slice the domain of K.

Code:

```
function [xt] = FSWave(t,K,T,W)

s = 1;

for ii = -K:K

if K==0

        x_t = @(t)(1-2*t.^2).*exp(-1i*2*pi*ii*t/T);

end

if K < 0

        x_t = @(t)(1-2*t.^2).*exp(-1i*2*pi*(-(ii+1+K))*t/T);

end

if K > 0

        x_t = @(t)(1-2*t.^2).*(1i*ii*2*pi/T).*exp(-1i*2*pi*(ii+1-K)*t/T);

x_k(s) = (1/T)*integral(x_t,-W/2,W/2);

s = s+1;

end

xt = SUMCS(t,x_k,2*pi*(-K:1:K)/T);

end
```

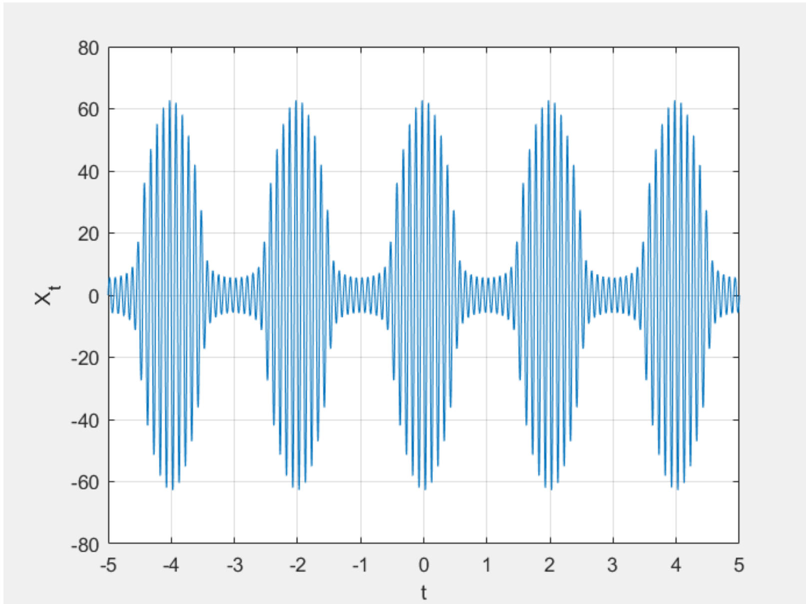

Figure 13: Conjugate SymmetricResult of the Signal X(t) [K=21]