Signals and systems lab sample codes:

NOTE: check for syntax/errors; do not replicate, try with your own logic of code

1. **Calculation of Energy and power of signals**

```
clc
x1 = @(t)(10*sin(10*pi*t));
f =@(x)exp(-2.*x);
x2 =@(t)((1+t).*(t>0 & t<2));
energy = integral(@(t)x1(t).^2,-pi,pi);

power = (integral(@(t)x1(t).^2,-pi,pi)/(2*pi));
```

2. **Addition and multiplication of signals**

```
%%  multiplication and addition of signals
clc
t1 = linspace(-1,5,2000);
x1 =@(t)(1.*(t>0 & t<1) + 2.*(t>1 & t<2) + (1.*(t>2 & t<3)));
x2 =@(t)((t).*(t>0 & t<1)+(1.*(t>1 & t<2))+((3-t).*(t>2 & t<3)));
x3 = @(t)(x1(t)+x2(t)); % addition of signals
x4 = @(t)(x1(t).*x2(t)); % multiplication of signals
subplot(2,2,1);
plot(t1,x1(t1));
subplot(2,2,2);
plot(t1,x2(t1));
subplot(2,2,3);
plot(t1,x3(t1));
subplot(2,2,4);
plot(t1,x4(t1));
```

3. **Sampling theorem verification**

```
% y(t) = 1+4cos(2000*pi*t)+ sin(1000*pi*t) is given to you & sampling it
using Nyquist criterion and reconstruct the original
clc;                        % Number of Samples
t= linspace(-20,20,1000);
n=0:1000;

x1= @(t) ones(1,length(t));
x2= @(t) 4*cos(2000*pi.*t);
x3= @(t) sin(1000*pi.*t);
y=  @(t) x1(t)+x2(t)+x3(t); % Total signal that has to be sampled

hold on
subplot(5,1,1);
plot(t,y(t));
title('Original Signal');
hold off

fm=1000; % Maximum frequency out of all the three signals.

% Case 1 : fs =2*fm (Nyquist Criterion)
fs1=2*fm;
ts1=1/fs1;
sample1=@(n) y(n.*(1/fs1));
```

```
hold on
subplot(5,1,2);
stem(n,sample1(n));
title('Sampled at fs =2*fm ');
hold off

% Case 2 : Sampling at fs>2*fm
fs2=6*fm;
sample2=@(n) y(n.*(1/fs2));
hold on
subplot(5,1,3);
stem(n,sample2(n));
title('Sampled at 6*fm');
hold off

% Case 3 : Sampling at fs< 2*fm

fs3=0.1*fm;
sample3=@(n) (y(n.*(1/fs3)));
hold on
subplot(5,1,4);
stem(n,sample3(n));
title('Sampled at 0.01*fm');
hold off

%Sample Reconstruction from Nquist Sampling Analysis
yrecnst=@(t)(0) ;
ynew = @(n) sample1(n).*(sinc(t-(n/fs1).*fs1));

for count=1:1000
    yrecnst= @(t) (yrecnst(t)+ ynew(count)) ;
end

hold on
subplot(5,1,5);
plot(t,yrecnst(t));
title('Reconstructed Signal y(t)');
hold off
```

4. **Obtaining Fourier series coefficients of given signal**

```
Trigonometric Fourier series of a given signal
t=linspace(-5,5,101);
T=2;                              %time period is 2 repeats after every 2sec
w0=(2*pi)/T;

u_t = @(t) (0*(t<0) + 1*(t>=0));            % defining the func.
x_t = @(t) (u_t(t+0.5) - u_t(t-0.5));
y_t = @(t) (x_t(t).*((t >= -0.5) & (t <= 0.5))) ;

a_o=(1/T)*integral(@(t)y_t(t),0,2*pi);

sum = zeros(1,101);

for n=1:200

    f=@(t) y_t(t).*cos(n*w0*t);
```

```
    a_n= (2/T).*integral(@(t)f(t),0,2*pi);       %writing fourier
coefficients
    g=@(t) y_t(t).*sin(n*w0*t);
    b_n= (2/T).*integral(@(t)g(t),0,2*pi);
    sum = sum + a_n .*cos(n*w0*t)+ b_n .*sin(n*w0*t);
    a_n1=@(t) (2/T).*integral(@(t)f(t),0,2*pi);
    c_n(n) = abs(a_n);
    d_n(n) = abs(b_n);

    subplot(3,1,3)
    stem(c_n);
    title('an');

    subplot(3,1,2)
    stem(d_n);
    title('bn');

end


X = (a_o)+sum;

subplot(3,1,1)
plot(X);
title('reconstructed signal');
```

5. Exponential Fourier series

Question 1:

```
clc
x =@(t)((1-abs(t)/0.5).*(t>=-0.5 & t<=0.5)+0.*(t<=-0.5 & t>=0.5));
C = zeros(1,50);
f = zeros(1,50);
phase = zeros(1,50);
Cn = @(n)(integral(@(t)(x(t).*exp(-1j*n*t)),0,2*pi));
z = Cn(0);
for count = 1:500
    C(count) = abs(Cn(count));
    phase(count) = angle(Cn(count));
    f(count) = count*1;
    z =@(t)(z(t) + (Cn(count).*exp(1j*1*count*t)));

end
subplot(3,1,1);
stem(f,C);
title('Magnitude spectrum');
subplot(3,1,2);
stem(f,phase);
title('Phase spectrum');
```

Exponential Fourier series: Question 2

```
clc
x =@(t)(1-abs(t)/0.5.*(abs(t)<=0.5) + 0.*(abs(t)>0.5));
C = zeros(1,50);
```

```
f = zeros(1,50);
freq = linspace(0,50,1000);
phase = zeros(1,50);
Cn = @(n)(integral(@(t)(x(t).*exp(-j*n*t)),0,2*pi));
z = Cn(0);
for count = 1:50
    C(count) = abs(Cn(count));
    phase(count) = angle(Cn(count));
    f(count) = count*1;
    z =@(t)(z(t) + (Cn(count).*exp(j*1*count*t)));

end
subplot(3,1,1);
stem(f,C);
subplot(3,1,2);
stem(f,phase);
```

6. Discrete convolution of a given signals

```
n = -10:50;
x1 =  (1).*(n>=0).*n + (0).*(n<0).*n;
x2 =  (1).*(n>=0).*exp(3*n) + (0).*(n<0).*exp(3*n);
X = x1;
Y = x2;
z = zeros(1,length(X)+length(Y)-1);
X = [X,zeros(1,length(Y)-1)];
Y = [Y,zeros(1,length(X)-1)];
z(1) = X(1)*Y(1);
for i = 2:(length(x1)+length(x2)-1)
    for j = 1:i
        z(i) = z(i) + X(j)*Y(i+1-j);
    end
end
subplot(3,1,1)
stem(x1)
subplot(3,1,2)
stem(x2)
subplot(3,1,3)
stem(z)
title('Convolution of discrete signals')
```

Also do discrete convolution of given discrete sequences, example, {-1, 2, 3, 4} and {2, 3, 4, 1}

7. Continuous convolution of given signals

```
t = linspace(-1,1,100);

x1 = @(t) 1.*exp(3.*t).*(t>=0);
subplot(2,2,1)
plot(x1(t))
title('x1(t)');

x2 = @(t) 1*t(t>=0);
subplot(2,2,2)
plot(x2(t))
```

```
title('x2(t)');

x3 = conv(x1(t),x2(t));
subplot(2,2,3)
plot(x3);
title('convolution using inbuilt function');

X = x1(t);
Y = x2(t);
z = zeros(1,length(X)+length(Y)-1);
X = [X,zeros(1,length(Y)-1)];
Y = [Y,zeros(1,length(X)-1)];
z(1) = X(1)*Y(1);
for i = 2:(length(x1(t))+length(x2(t))-1)
    for j = 1:i
        z(i) = z(i) + X(j)*Y(i+1-j);
    end
end
subplot(2,2,4)
plot(z)
title('convolution using definition')
```

8. **Find the DFT X(k) of the given discrete time sequence x(n) = {-1 , 3, 2, 1} Perform IDFT to get back the given discrete Fourier series x(n).**

9. **Perform time scaling, shifting, amplitude scaling operation on given signal**

10. **Determine the *natural response (zero input response,* it is due to the initial values of output alone) of the system described by the given equation**

11. **Determine the *forced response or zero state response* (it is due to the given input with zero initial output, but initial values of the input should be considered) of the system described by the given equation**

12. **Perform N point FFT for the given discrete time sequence of length 4, {-1, 2, 3, 2}**

Dr. B S Rao