EEE391 – CA1
Basics of Signals and Systems
Spring 2016-2017
Computer Assignment 1

Ecem İlgün - 21502157

a) 21502157 = *(d1d2d3d4d5d6d7d8)*
d1=2, d2=1, d3=5, d4=0,
d5=2, d6=1, d7=5, d8=7.

Frequency :
w0 = 21 rad/s.

Amplitudes:
A1 = 1, A2 = 5, A3 = 7.

Phase values:
φ1 = 21 degrees = 0.36651914292 radians
φ2 = 215 degrees = -145 degrees = -2.5307274154 radians (-0.80555555556π rad).
φ3 = 157 degrees = 2.7401669256 radians

b)
Program part i :

```
%i%
w0 = input('w0 (radian frequency) = ');
A1 = input('A1 (amplitude) =');
A2 = input('A2 =');
A3 = input('A3 =');
Phi1Degrees = input('phase shift1 (in degrees)  = ') ;
Phi2Degrees = input('phase shift2 (in degrees)  = ');
Phi3Degrees = input('phase shift3 (in degrees)  = ');
%Convert the numbers into radians
Phi1 = Phi1Degrees*pi/180;
Phi2 = Phi2Degrees*pi/180;
Phi3 = Phi3Degrees*pi/180;
```

In here, each input function gets an input from the user, and binds the value to the variables w0, A1, A2 etc.

```
>> ca1b
w0 (radian frequency) = 21
A1 (amplitude) =1
A2 =5
A3 =7
phase shift1 (in degrees)  = 21
phase shift2 (in degrees)  = -145
phase shift3 (in degrees)  = 157
```

Figure 1. Command window for program part i

In figure 1, The numbers to the right side of each '=' sign in each column are the numbers that are mentioned in homework assignment part a.

The writing to the left side of each '=' are the prompts that are highlighted as purple in the code segment above.

Program part ii:

```
%ii%
x1 = A1*exp(j*Phi1);
x2 = A2*exp(j*Phi2);
x3 = A3*exp(j*Phi3);
x = x1 + x2 + x3;
A = abs(x)
%angle(x) gives phi in radians
phi = (angle(x))*180/pi
```

MATLAB stores numbers of form A*exp(j*phi) as complex numbers in cartesian form which simplifies the addition process into simply "x1 + x2 + x3".

abs function gives the abstract value, which corresponds to the magnitude of a complex number.

angle function returns the corresponding angle of complex number in radians.

```
A =

    9.6084


phi =

  178.6546
```

Figure 2. Command window for program part ii

Program part iii:

```
%iii%
phi2=round(angle(x)*100)/100; %2 decimal point precision
if phi2 >= 0
    funct = ['x(t) = ',num2str(A),'*cos(',num2str(w0),'t+',num2str(phi2),')'];
else
    funct = ['x(t) = ',num2str(A),'*cos(',num2str(w0),'t',num2str(phi2),')'];
end
disp(funct);
```

In the code segment above, phase value is rounded up to 2 digits after decimal by round function.

If phase value is nonnegative (in form of 0, 2 etc.) a '+' sign added to indicate the addition of frequency and phase shift.

Otherwise, if the phase value is of from -1, -3 etc, no sign is added since the '-' sign of the value will take place as substraction sign.

```
x(t) = 9.6084*cos(21t+3.12)
fx >>
```

Figure 3. Command window for program part iii

Program part iv:

```
%iv%
p1 = [0;x1];
p2 = [0;x2];
p3 = [0;x3];
p = [0; x];
figure;
plot(real(p1), imag(p1),'LineWidth', 2); hold on;
plot(real(p2), imag(p2),'LineWidth', 2); hold on;
plot(real(p3), imag(p3),'LineWidth', 2); hold on;
plot(real(p), imag(p),'LineWidth', 5); hold on;
xlim([-30 30]);
ylim([-30 30]);
xlabel('real axis');
ylabel('imaginary axis');
grid on;
```

In the code segment, each vector is of the form [0; a], meaning each of them have origin as the start point and the complex number value as the end point.

Each vector is plotted seperately with different colors while the vector which has the ending point as the summation of the other vectors (therefore the other phasors) is plotted thicker than the others.

xlim and ylim functions are used to limit the upper and lower bounds of axes. Each axis corresponds the real / imaginary part as indicated in the homework description.

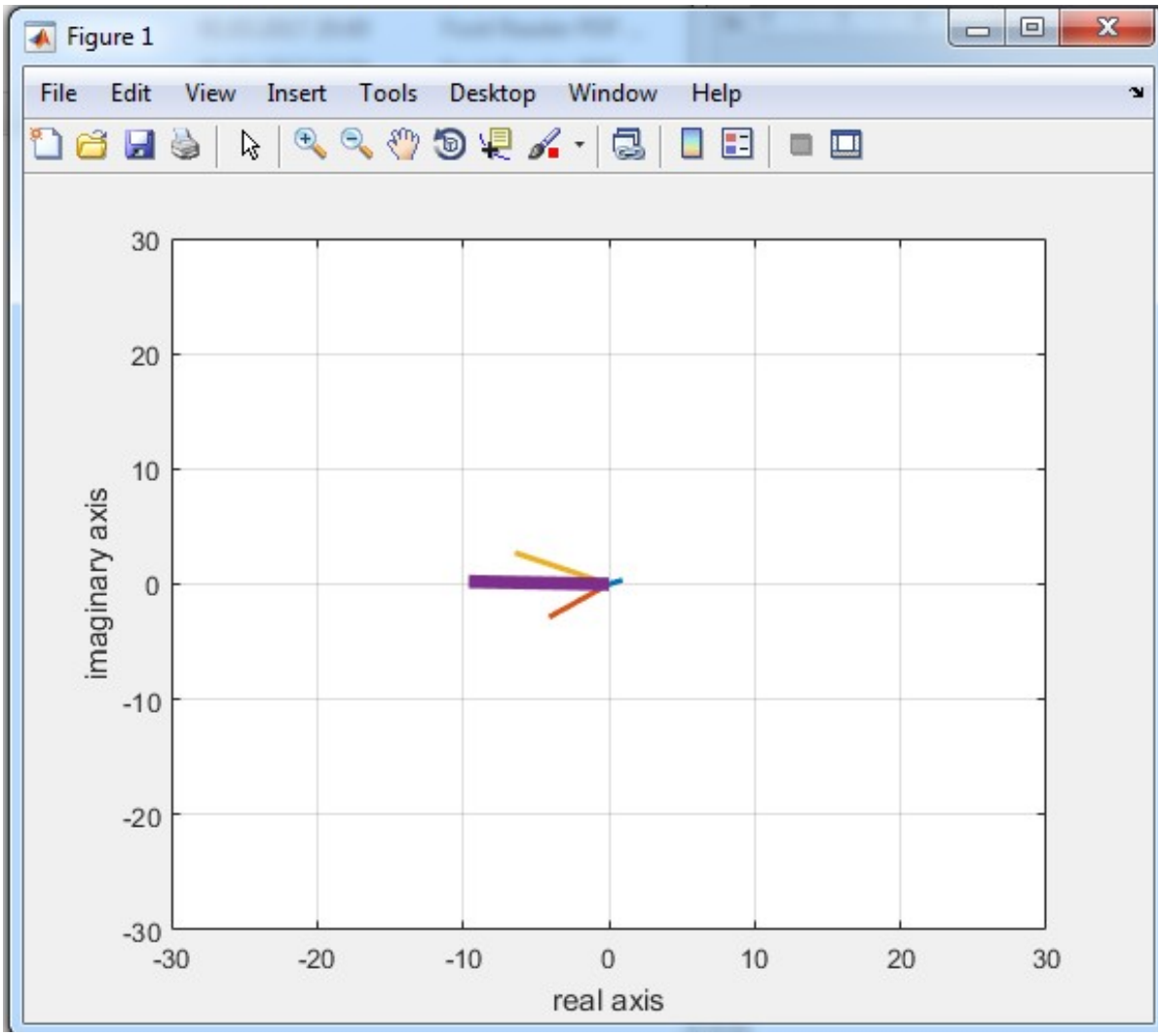

Figure 4. Resulting plot of phasors of program part iv

Program part v:

```
%v%
v1 = [0;x1];
v2 = [x1;(x2+x1)];
v3 = [(x1+x2);x];
v= [0; x];
figure;
plot(real(v1), imag(v1),'LineWidth', 2); hold on;
plot(real(v2), imag(v2),'LineWidth', 2); hold on;
plot(real(v3), imag(v3),'LineWidth', 2); hold on;
plot(real(v), imag(v),'LineWidth', 5); hold on;
xlabel('real axis');
ylabel('imaginary axis');
grid on;
```

In the code segment, the starting point of each vector is the end point of the previous vector, with the initial vector starting at origin. Therefore the vectors add up to each other to form the last vector, which is the summation of the phasors.

x and y axes are not manually limited to provide better fit and clearer looking plot.

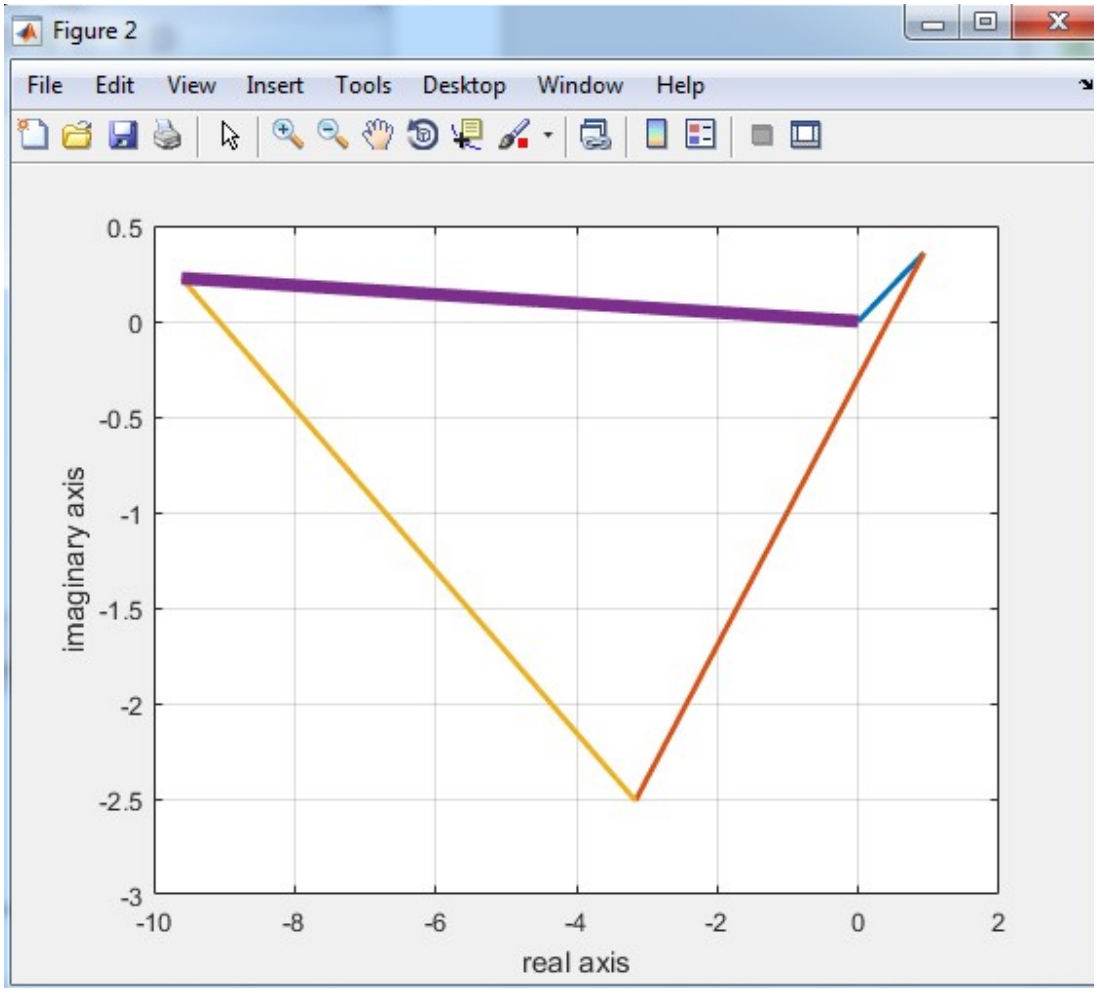

Figure 5. Resulting plot of phasors of program part iv

Appendix (Program):

```
%i%
w0 = input('w0 (radian frequency) = ');
A1 = input('A1 (amplitude) =');
A2 = input('A2 =');
A3 = input('A3 =');
Phi1Degrees = input('phase shift1 (in degrees)  = ') ;
Phi2Degrees = input('phase shift2 (in degrees)  = ');
Phi3Degrees = input('phase shift3 (in degrees)  = ');
%Convert the numbers into radians
Phi1 = Phi1Degrees*pi/180;
Phi2 = Phi2Degrees*pi/180;
Phi3 = Phi3Degrees*pi/180;

%ii%
x1 = A1*exp(j*Phi1);
x2 = A2*exp(j*Phi2);
x3 = A3*exp(j*Phi3);
x = x1 + x2 + x3;
A = abs(x)
%angle(x) gives phi in radians
phi = (angle(x))*180/pi

%iii%
phi2=round(angle(x)*100)/100; %2 decimal point precision
if phi2 >= 0
   funct = ['x(t) = ',num2str(A),'*cos(',num2str(w0),'t+',num2str(phi2),')'];
else
   funct = ['x(t) = ',num2str(A),'*cos(',num2str(w0),'t',num2str(phi2),')'];
end
disp(funct);

%iv%
p1 = [0;x1];
p2 = [0;x2];
p3 = [0;x3];
p = [0; x];
figure;
plot(real(p1), imag(p1),'LineWidth', 2); hold on;
plot(real(p2), imag(p2),'LineWidth', 2); hold on;
plot(real(p3), imag(p3),'LineWidth', 2); hold on;
plot(real(p), imag(p),'LineWidth', 5); hold on;
xlim([-30 30]);
ylim([-30 30]);
xlabel('real axis');
ylabel('imaginary axis');
grid on;

%v%
v1 = [0;x1];
```

```
v2 = [x1;(x2+x1)];
v3 = [(x1+x2);x];
v= [0; x];
figure;
plot(real(v1), imag(v1),'LineWidth', 2); hold on;
plot(real(v2), imag(v2),'LineWidth', 2); hold on;
plot(real(v3), imag(v3),'LineWidth', 2); hold on;
plot(real(v), imag(v),'LineWidth', 5); hold on;
xlabel('real axis');
ylabel('imaginary axis');
grid on;
```