Rüzgar Ayan 21801984

2020-2021 SPRING EEE391 MATLAB Assignment 1 Report

Part (a)

My student ID($d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8$) = 21801984. Then,

- $w_0 = (d_5 d_6)\, rad/s = 19\, rad/s$
- $A_1 = d_6 = 9$,
- $A_2 = d_7 = 8$
- $A_3 = d_8 = 4$
- $\phi_1 = (d_4 d_5 d_6)° = 19°$, already in the principal interval. In radians $\phi_1 = 0.3316$.
- $\phi_2 = (d_5 d_6 d_7)° = 198°$, converted into $\phi_2 = -162°$. In radians $\phi_2 = -2.8274$
- $\phi_3 = (d_6 d_7 d_8)° = 984°$, converted into $\phi_3 = -96°$. In radians $\phi_3 = -1.6755$.

Part (b)

When we run the program from the file called ”sumphasors.m”, it starts by asking the values from above and we enter them as follows:

```
Command Window
>> sumphasors
Enter omega_0: 19
Enter A_1: 9
Enter A_2: 8
Enter A_3: 4
Enter phi_1: 19
Enter phi_2: 198
Enter phi_3: 984
```

Then, to convert the phi values into principal invertal, I have used the following while loops that change the degrees by 360° until they are in the range (-180, 180].

```
%Convert into principal interval
for a = 1:3
    while phi_degrees(a) > 180
        phi_degrees(a) = phi_degrees(a) - 360;
    end
    while phi_degrees(a) < -180
        phi_degrees(a) = phi_degrees(a) + 360;
    end
end
```

Converting this into radians is straightforward as follows:

```
%Convert into radians
phi_radians = pi * phi_degrees / 180;
```

The radian values of phis are not printed into the console, but they are used in the computations to find the sum of the phasors.

Phasors given by the A and phi values are easily converted to the cartesian form by using the exp() function of MATLAB. Then, we just sum up the cartesian forms of the phasors as follows:

```
phasors = A .* exp(1i*phi_radians);

sum_of_phasors = sum(phasors);
real_part_sum = real(sum_of_phasors);
imag_part_sum = imag(sum_of_phasors);
```

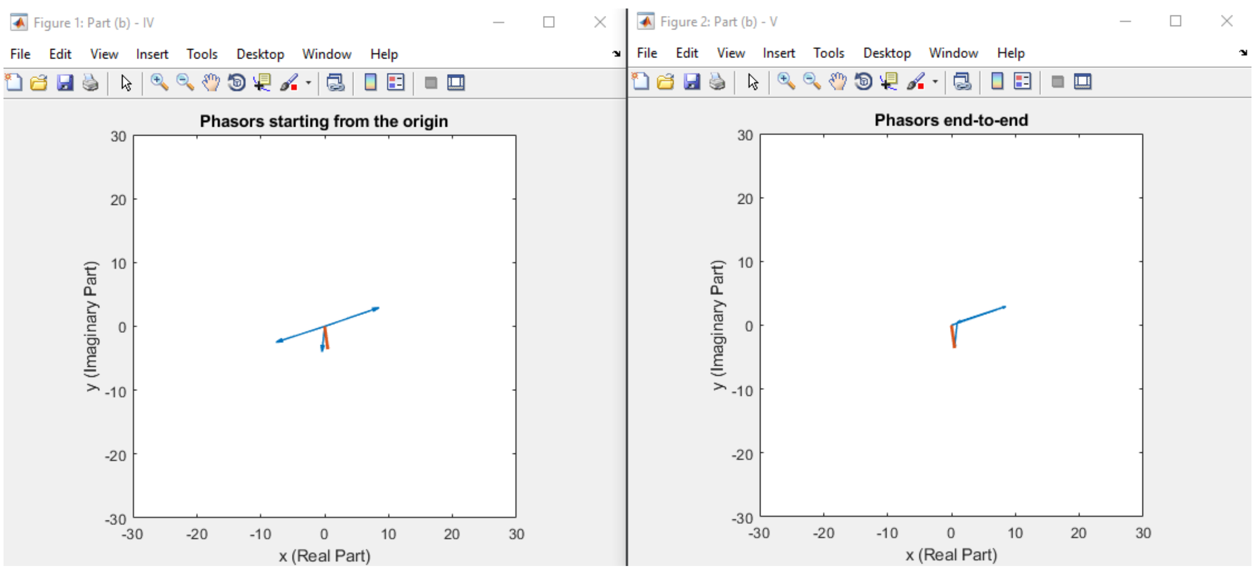
The last step is to convert this cartesian sum back to the polar form, and this is done using the regular formulas for conversion.

```
sum_A = sqrt(real_part_sum^2 + imag_part_sum^2);
sum_phi_radians = atan2(imag_part_sum, real_part_sum);
sum_phi_degrees = sum_phi_radians * 180 / pi;
```

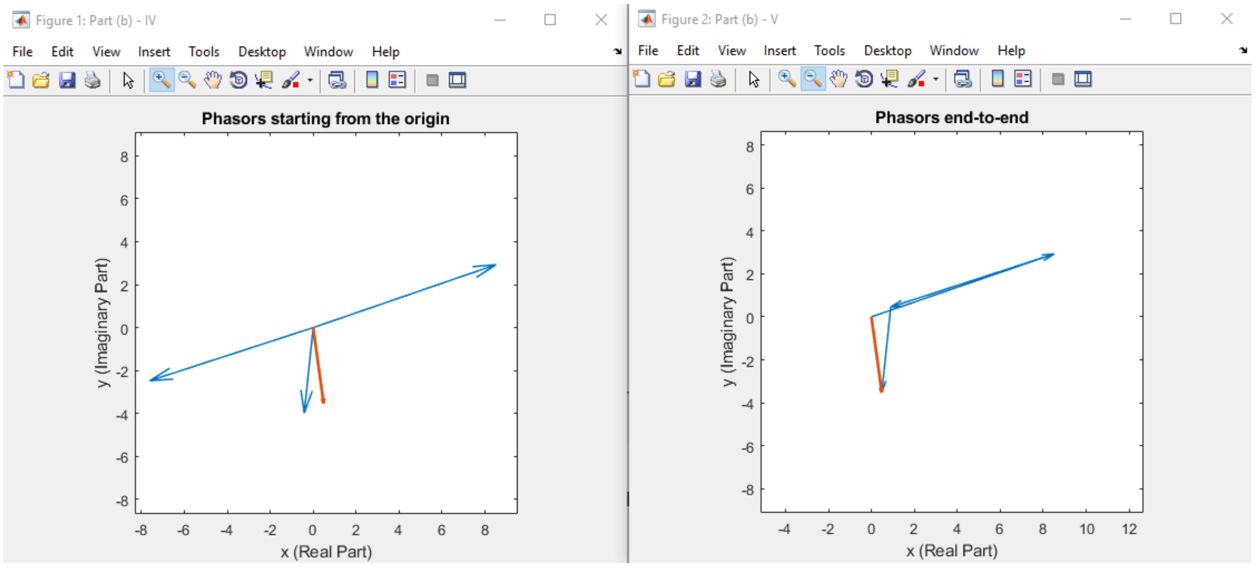
Finally, we just print the results into the console by formatting the numbers to show a specific number of decimal digits. The final results that we see in the console is as follows:

```
>> sumphasors
Enter omega_0: 19
Enter A_1: 9
Enter A_2: 8
Enter A_3: 4
Enter phi_1: 19
Enter phi_2: 198
Enter phi_3: 984

Resulting A = 3.5531
Resulting phi in degrees = -82.1855

Resulting sinusodial
     x(t)= 3.55 cos(19t-1.43)
fx >>
```

Other than the printed results, also two figure windows open and they show the following plots:



Since the assignment says that the ranges must be [-30, 30], figures look really small, to see more detail we can zoom in.



I have used the quiver function of MATLAB to draw these arrow-headed vectors. The sum phasor is shown with a slightly thicker line and in red color. After these, the program ends. Full MATLAB code can be found in the next page.

```matlab
%Input from user
omega_0 = input('Enter omega_0: ');

A = [0, 0, 0];
phi_degrees = [0, 0, 0];

A(1) = input('Enter A_1: ');
A(2) = input('Enter A_2: ');
A(3) = input('Enter A_3: ');
phi_degrees(1) = input('Enter phi_1: ');
phi_degrees(2) = input('Enter phi_2: ');
phi_degrees(3) = input('Enter phi_3: ');

%Convert into principal interval
for a = 1:3
    while phi_degrees(a) > 180
        phi_degrees(a) = phi_degrees(a) - 360;
    end
    while phi_degrees(a) <= -180
        phi_degrees(a) = phi_degrees(a) + 360;
    end
end

%Convert into radians
phi_radians = pi * phi_degrees / 180;

phasors = A .* exp(1i*phi_radians);

sum_of_phasors = sum(phasors);
real_part_sum = real(sum_of_phasors);
imag_part_sum = imag(sum_of_phasors);

sum_A = sqrt(real_part_sum^2 + imag_part_sum^2);
sum_phi_radians = atan2(imag_part_sum, real_part_sum);
sum_phi_degrees = sum_phi_radians * 180 / pi;

fprintf('\nResulting A = %.4f\n', sum_A);
fprintf('Resulting phi in degrees = %.4f\n', sum_phi_degrees);
if sum_phi_radians >= 0
    fprintf('\nResulting sinusodial \n\t x(t)= %.2f cos(%dt+%.2f)\n', sum_A, omega_0,
sum_phi_radians);
else
    fprintf('\nResulting sinusodial \n\t x(t)= %.2f cos(%dt-%.2f)\n', sum_A, omega_0,
abs(sum_phi_radians));
end

figure('Name', 'Part (b) - IV')
quiver(zeros(3,1),zeros(3,1),real(phasors.'),imag(phasors.'), 0, 'linewidth', 1.1);
hold on
quiver(0, 0, real_part_sum, imag_part_sum, 0, 'linewidth', 1);
axis equal
ylim([-30, 30]);
xlim([-30, 30]);
title('Phasors starting from the origin');
xlabel('x (Real Part)'), ylabel('y (Imaginary Part)');

startingPoints = [0 ; phasors(1); phasors(1) + phasors(2)];
endingPoints = [phasors(1) ; phasors(2); phasors(3)];

figure('Name', 'Part (b) - V')
quiver(real(startingPoints),imag(startingPoints),real(endingPoints),imag(endingPoints), 0,
'linewidth', 1.1);
hold on
quiver(0, 0, real_part_sum, imag_part_sum, 0, 'linewidth', 1);
axis equal
ylim([-30, 30]);
xlim([-30, 30]);
title('Phasors end-to-end');
xlabel('x (Real Part)'), ylabel('y (Imaginary Part)');
```