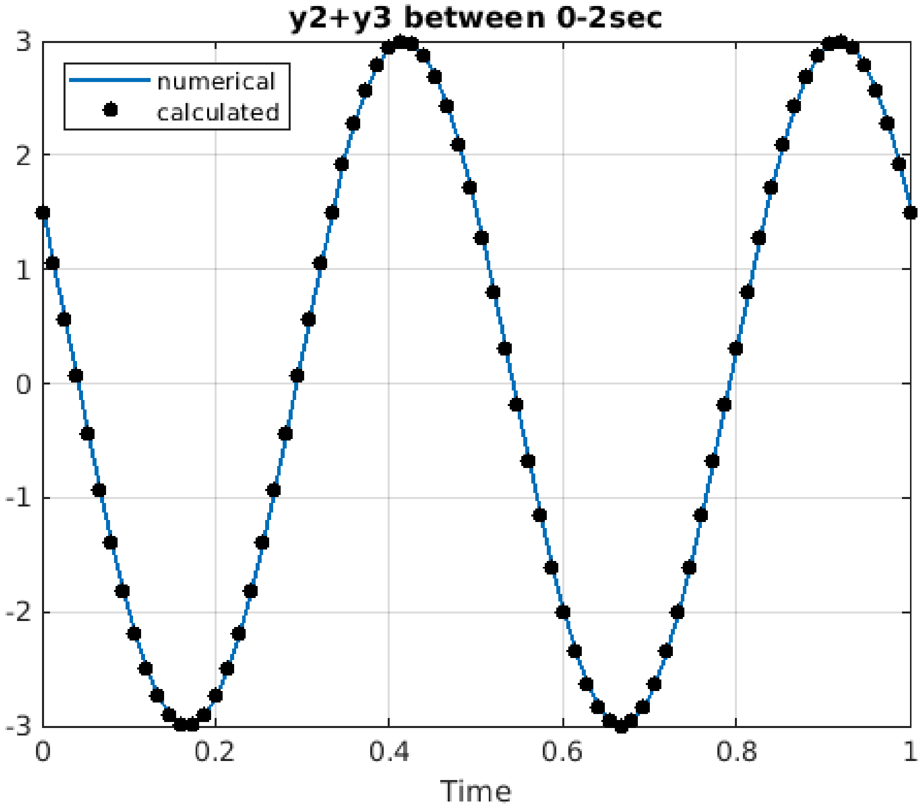
```
% BME671L
% Lab #3: for, getframe, movie, input

% Your name: Ravitashaw Bathla

% LABEL ALL AXIS WHERE APPROPRIATE
```

Q1: Remember Q8-10 in Lab 2 involved the sum of 2 sinusoids with the same frequency. y2 + y3 can be written in the form A*cos(w*t + phi). Use phasor addition to determine the exact values for A, w, and phi (include units). How close were your numerical answers from last week? Recall: y2: A = 5, f = 2 Hz, time shift = 1/6 s y3: A = 8, f = 2 Hz, time shift = -1/12 s

```
y2 = @(t) 5*cos(4*pi*(t-1/6));
y3 = @(t) 8*cos(4*pi*(t+1/12));

% YOUR ANSWER:
% y2+y3 = 3*cos(4*pi*t+pi/3)
% A = 3
% w = 4*pi rad/s
% phi = pi/3 rad
% The answers for Amplitude and phi very approximately very close. If
 the
% numbers are rounded up slightly, the answers match very well.
```

Q2: Double check your work. On the same figure plot y2 + y3 with enough points so that the curve appears smooth. Then use the time vector below to plot your simplified representation as black dots on top of the sinusoid. Change 'Linewidth' to 1.5 and the 'MarkerSize' to 20. Add a legend labeling the curve 'numerical' and the dots 'calculated'. Have the legend automatically appear in the upper left corner.

```
tt = 0:0.05:1;
figure(1), clf
tt = 0:1/75:2;
y_23 = @(t) 3*cos(4*pi*t+(pi/3));
x23 = y_23(tt);
plot(tt, x23, 'Linewidth',1.5, 'DisplayName', 'numerical')
title('y2+y3 between 0-2sec')
ylim([-3 3])
xlim([0, 1])
grid on
xlabel('Time')

x2_3 = y_23(tt);
hold on;
plot(tt, x2_3, '.k', 'MarkerSize', 20, 'DisplayName', 'calculated');
title('y2+y3 between 0-2sec');
ylim([-3 3])
xlim([0, 1])
grid on
xlabel('Time')
legend("Location","northwest")
```



Q3: z(t) is defined as z(t) = X*exp(j*w*t) with the parameters below. What is the phase shift of z(t)?

```
X = -2.5 + 1j*0;      % phasor at t = 0
w = 4*pi;             % radian frequency (rad/s)
fs = 48;              % sampling frequency (Hz)

% YOUR ANSWER:
% The phase shift of z(t) is pi
```

Q4: Compute frequency (f), period (T), sampling interval (ts), and the number of samples taken in one period (N). Include units in the comments.

```
    % frequency: 2 Hz
    % period: 0.5 sec
    % sampling interval: 0.0208 sec
    % N: 25
```

Q5: Define an anonymous complex function that evaluates z(t)

```
z = @(t) X * exp(1j*w*t);
```

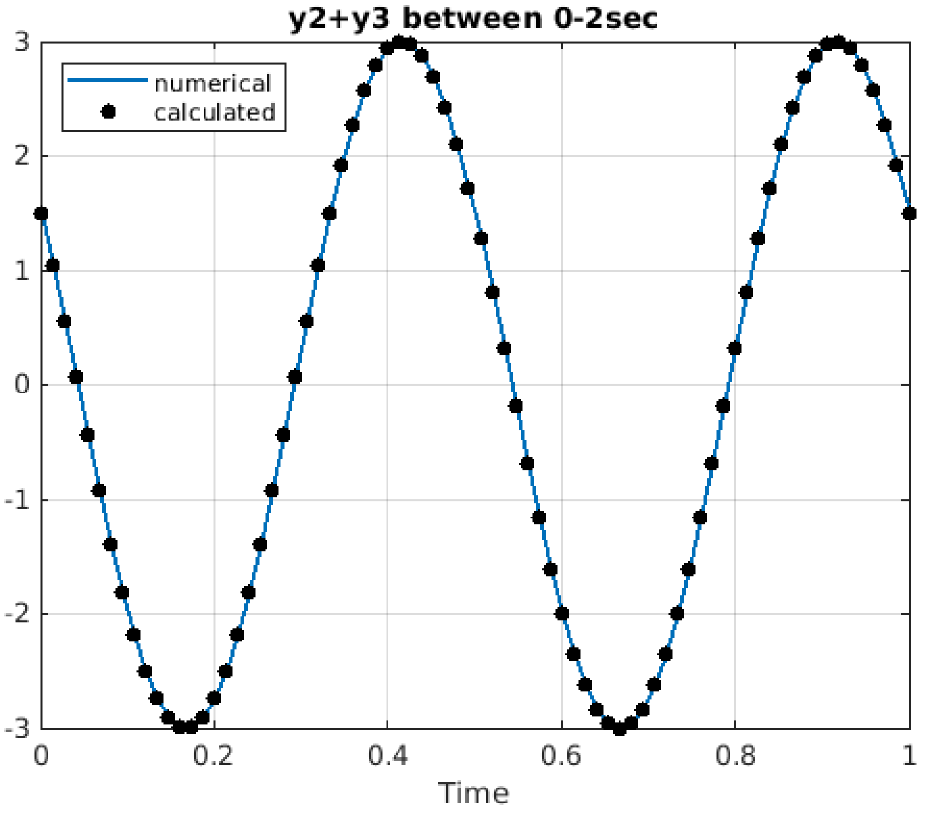
Q6: Next we will create a movie of the rotating phasor corresponding to z(t) along with a plot of the corresponding real signal. To begin, define a complex function zcirc(theta) that describes a circle with the radius equal to the amplitude of z(t). Create a corresponding vector, th, that contains the associated angles with a small enough increment that the circle has a smooth appearance.

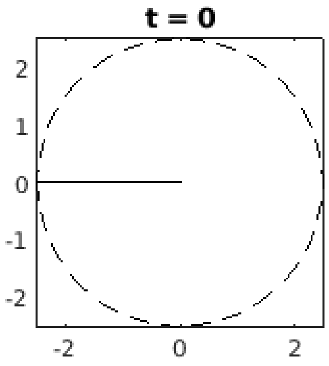
```
zcirc = @(theta) abs(X)*exp(1j*theta);
```

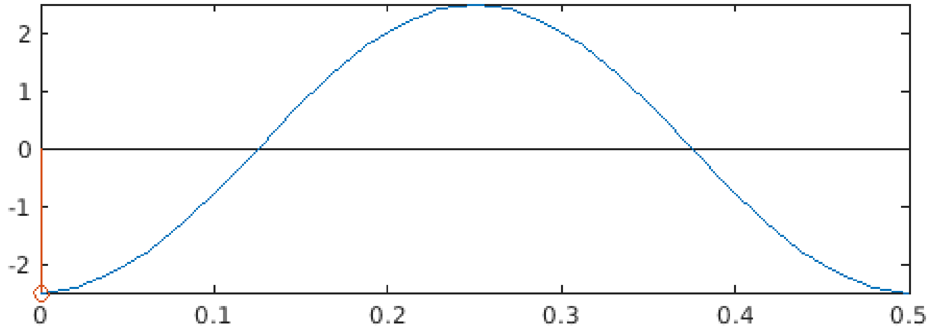
```
th = -pi:2*pi/fs:pi;
```

Create the first frame of the movie. Create a figure with two plots. In the top plot show: * the circle plotted with a dashed black line * a phasor corresponding to the current time (t = 0) * a title that includes a counter displaying the current time * axis labels, square axis (use the 'axis' command) In the bottom plot show: * z(t) from 0 to T plotted with enough points that the curve looks smooth * the current measurements of z(t) marked using the stem command * x-axis labeled time

```
figure(2), clf
subplot(2,1,1)

z_val = zcirc(th);

plot(real(z_val),imag(z_val),'--k');
axis square, axis tight;
axis([-2.5 2.5 -2.5 2.5]);

title(['t = ', num2str(0)]);
hold on;
plot([0 real(z_val(1))],[0 imag(z_val(1))], 'k-');

tth = 0:1/fs:0.5;
subplot(2,1,2)

plot(tth, real(z(tth)));
hold on;
stem(tth(1), real(z(tth(1))));
hold off;

% ********************************************************************
%   SHOW YOUR FIRST FRAME TO THE TA TO RECEIVE CREDIT FOR THE LAB
% ********************************************************************
```

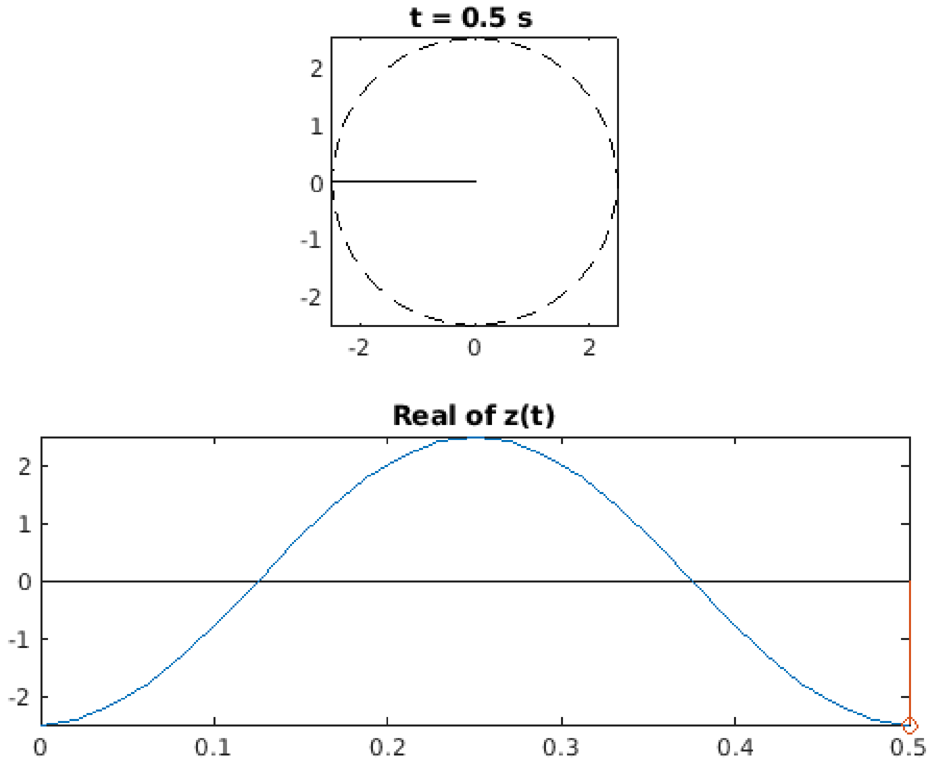

y2+y3 between 0-2sec
numerical
calculated
3
2
1
0
-1
-2
-3
0
0.2
0.4
0.6
0.8
1
Time


t = 0
2
1
0
-1
-2
-2
0
2


2
1
0
-1
-2
0
0.1
0.2
0.3
0.4
0.5

Q7: Now, use the 'for' loop to generate the frames of the movie. The movie should have 24 frames for time instances ts apart. For each frame there will be two plots. Top plot will show: * the circle and phasor corresponding to the current time * a title that includes a counter displaying the current time * axis labels, square axis (use the 'axis' command) Bottom plot will show: * z(t) from 0 to T plotted with enough points that the curve looks smooth * the current measurements of z(t) marked using the stem command * x-axis labeled time During every loop: * save the current frame using 'getframe' command * at the end of the loop, release the figure so that the next frame can be generated. HINT: if you put the pause statements in the loop you can pause the execution until any key is pressed.

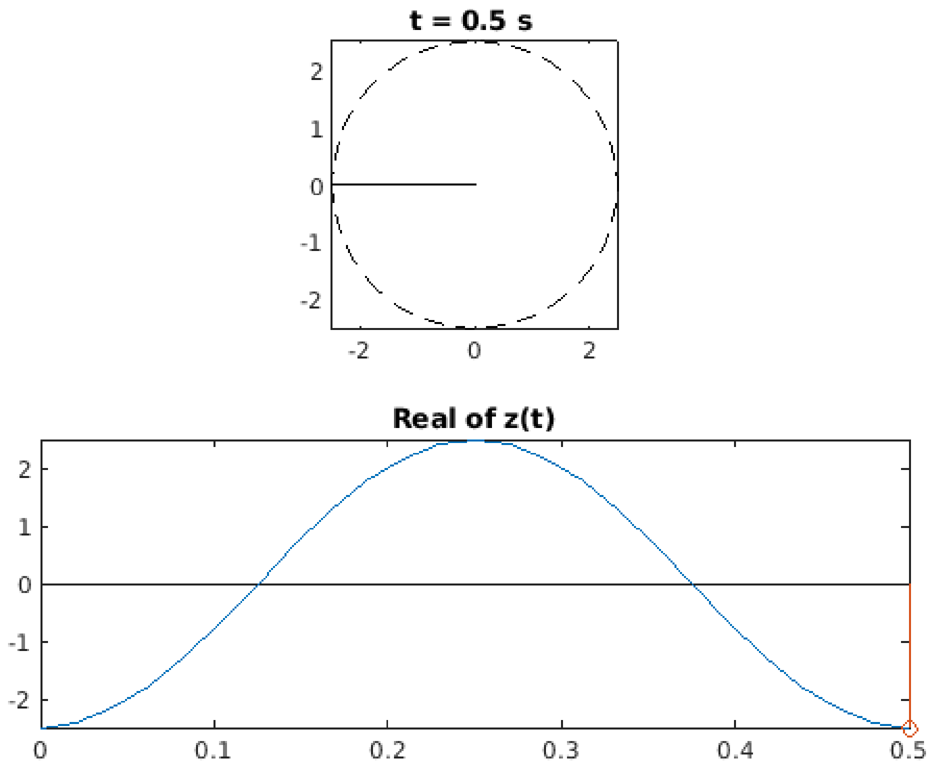
```
tth = 0:1/fs:0.5;
for j=1:length(th)
    subplot(2,1,1)

    z_val = zcirc(th);

    plot(real(z_val),imag(z_val),'--k')
    t_l = (th(j)+pi)/w;
    axis square, axis tight;
    axis([-2.5 2.5 -2.5 2.5]);
    hold on
    plot([0 real(zcirc(th(j)))],[0 imag(zcirc(th(j)))], 'k-');
    title(['t = ', num2str(t_l), ' s ']);
    hold off;

    subplot(2,1,2)
    plot(tth, real(z(tth)));
    title('Real of z(t)');
    hold on;
    stem(t_l, real(z(t_l)));

    M(j) = getframe(gcf);
    hold off;
end
```



Q8: Play movie for k cycles of z(t): * use 'input' command to ask 'How many cycles should be played?' * use 'movie' command to play the movie in figure 2 at the speed of 6 frames/s and for the requested number of cycles. * use 'axis off' so that the movie appears in the correct part of the figure

PUBLISHING: hard code the number of cycles played to 1 when you publish as a PDF to avoid an error.

```
figure(2), clf
axis off;
prompt = 'Enter number of cycles to play ';
% cycles = input(prompt)
cycles = 1;
fps = 6;
movie(gcf,M,cycles,fps);
```



When you are done:

```
% * Make sure to show the indicated result/figure to the TA before
 11:59PM Friday to receive credit
% * upload your script to Sakai
%   * upload a pdf containing your script and outputs

return
```

Published with MATLAB® R2020a