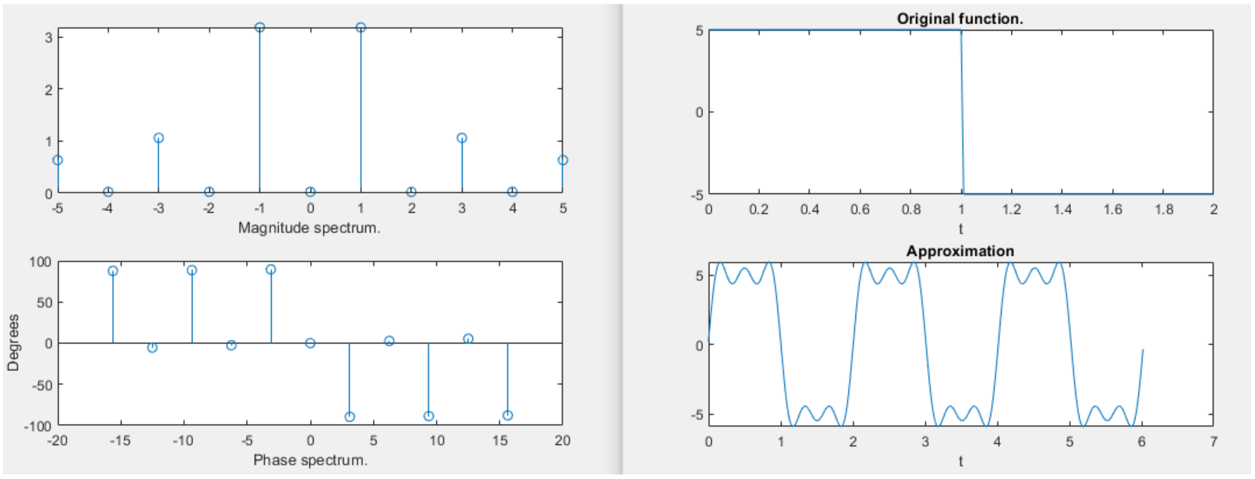
Frank Isidore Gomez
5/3/21
ID: 1650550
figomez
ECE 103L

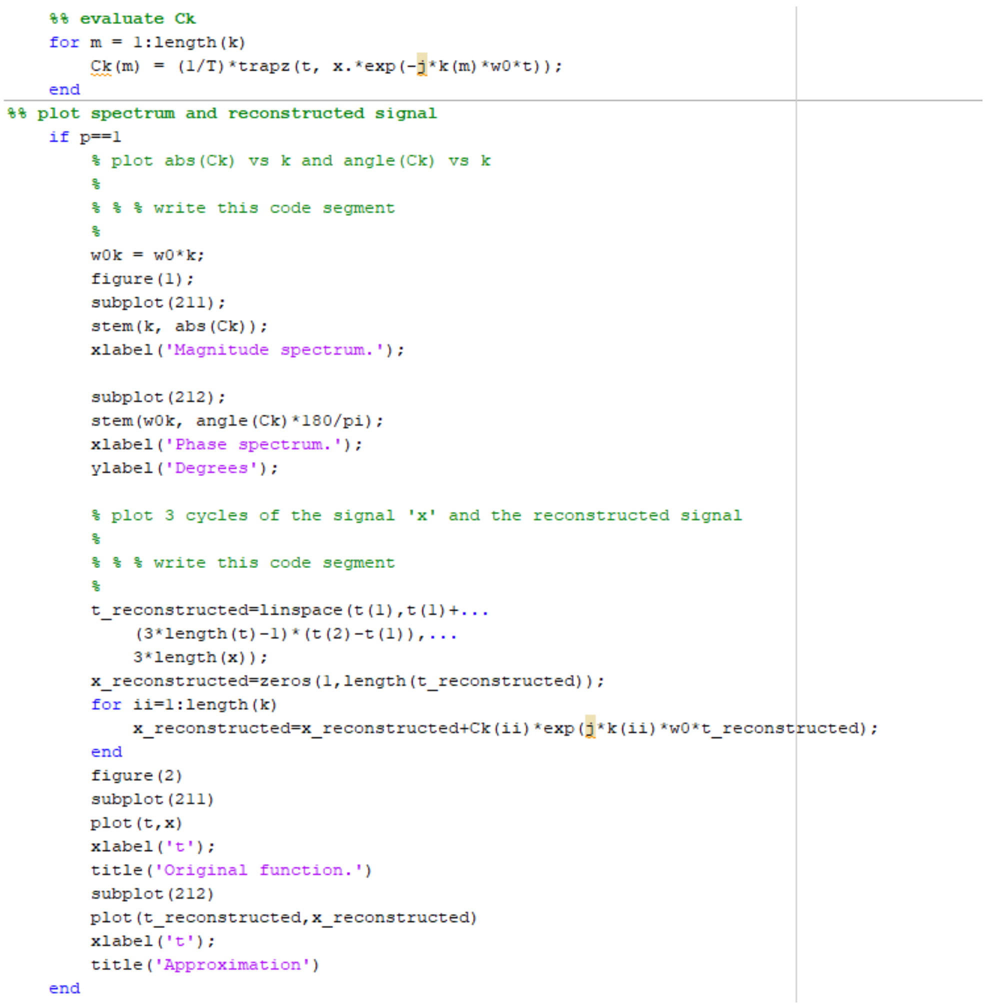
Lab 4 - Fourier Series in Matlab

1. During lab 4, we have seen numerical implementation of Fourier Series for periodic signals. As first part of this assignment, you need to write a Matlab function that would take an array representing a single period of a signal (`x`), corresponding time array (`t`), and return the Fourier Series coefficients (`Ck`) in exponential form. The function should also be able to take two (2) optional input arguments: number of Fourier coefficients (`Nk`) and plot option (`p`). Use the template '`fourier_series_exp.m`' for this problem.

1.



fs_numerical, by fourier_series_exp.m



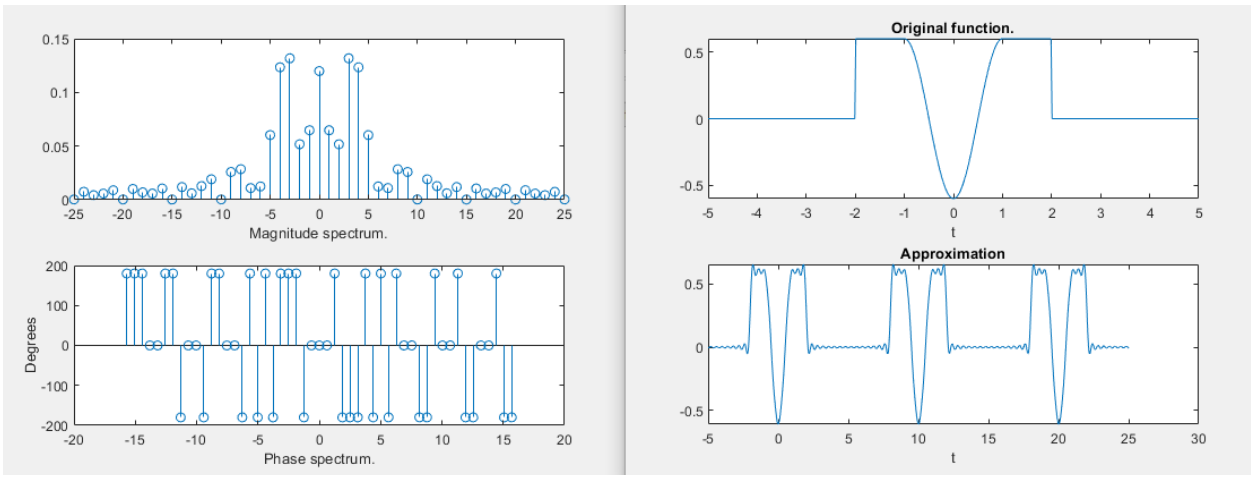
```
    %% evaluate Ck
    for m = 1:length(k)
        Ck(m) = (1/T)*trapz(t, x.*exp(-j*k(m)*w0*t));
    end
%% plot spectrum and reconstructed signal
    if p==1
        % plot abs(Ck) vs k and angle(Ck) vs k
        %
        % % % write this code segment
        %
        w0k = w0*k;
        figure(1);
        subplot(211);
        stem(k, abs(Ck));
        xlabel('Magnitude spectrum.');

        subplot(212);
        stem(w0k, angle(Ck)*180/pi);
        xlabel('Phase spectrum.');
        ylabel('Degrees');

        % plot 3 cycles of the signal 'x' and the reconstructed signal
        %
        % % % write this code segment
        %
        t_reconstructed=linspace(t(1),t(1)+...
            (3*length(t)-1)*(t(2)-t(1)),...
            3*length(x));
        x_reconstructed=zeros(1,length(t_reconstructed));
        for ii=1:length(k)
            x_reconstructed=x_reconstructed+Ck(ii)*exp(j*k(ii)*w0*t_reconstructed);
        end
        figure(2)
        subplot(211)
        plot(t,x)
        xlabel('t');
        title('Original function.')
        subplot(212)
        plot(t_reconstructed,x_reconstructed)
        xlabel('t');
        title('Approximation')
    end
```

Same as last time, the first segment actually took quite a bit to do. We were supposed to receive the period of a function, and the length of the period, and then construct the fourier series coefficient of said function, and then use the fourier series to *reconstruct* the original function. Most of the code was given to us, so once I understood exactly what a fourier series coefficient was, it was pretty straightforward.

2. A signal $x = 0.6\left\{u(t+2) - (\cos(\pi t) + 1)\left[u(t+1) - u(t-1)\right] - u(t-2)\right\}$ with a period $-5 \le t \le 5$ controls the location of the light source in an optical scanner. Plot the signal for the interval $-5 \le t \le 25$, its spectrum ($|C_k|$ vs ω and $\angle C_k$ vs ω), and reconstructed time domain signal using 51 Fourier Series coefficients. Use the function you have written in problem 1 for solving this problem.

2.

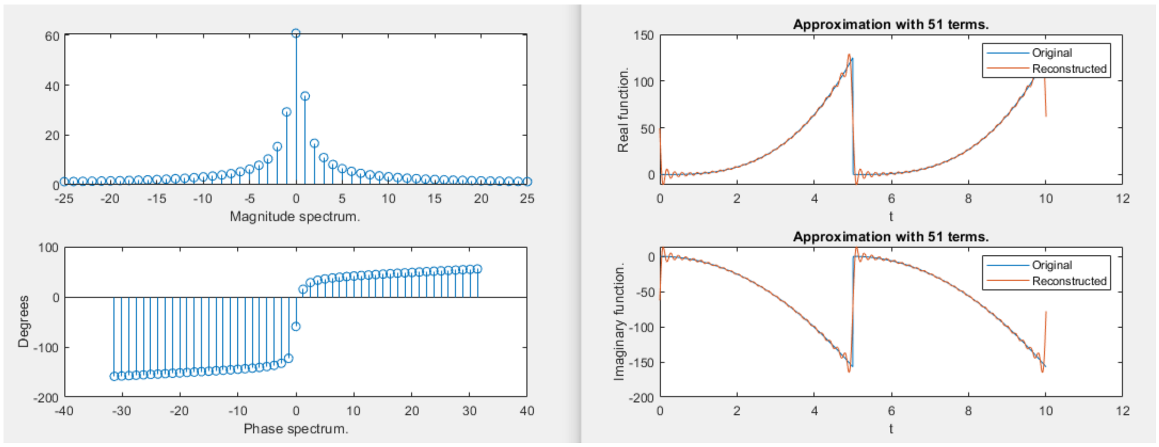
```
t = -5:0.01:5;
x = 0.6.*(heaviside(t+2) - (cos(pi*t) + 1).*(heaviside(t+1) - heaviside(t-1)) - heaviside(t-2));
fourier_series_exp(x,t,51,1);
```



This part was really simple, all we had to do was plug in a piecewise function from -5 to 5 into our function from part 1. Still, it was useful, as I learned yet another way to declare piecewise functions by use of the heaviside function. Seems the most useful to me.

3. So far all the signals we have handled in this course are real signals. However, we can also use complex numbers to represent signals (complex signals). Let's consider a single period of a periodic signal $z(t) = t^3 - j2\pi t^2$, $0 < t \leq 5$. Calculate 51 Fourier Series coefficients (C_k) for this signal and reconstruct the time domain signal $\hat{z}(t)$ using these Fourier Series coefficients. Plot the spectrum ($|C_k|$ vs ω and $\angle C_k$ vs ω) and the real and imaginary part of $z(t)$ and $\hat{z}(t)$ for an interval of $0 \leq t \leq 10$. You can modify the Matlab file '`fs_numerical.m`' which was used during the lab for solving this problem. Following are the sample plots for your reference.

3.



```
t = 0:0.01:5;
z = t.^3 - j*2*pi.*t.^2;
q = fourier_series_exp(z, t, 51, 1);
```

```
figure(2)
subplot(211)
x_extended=repmat(x,1,2);
t_extended=linspace(t(1),t(1)+(length(x_extended)-1)*(t(2)-t(1)),length(x_extended));
plot(t_extended,x_extended)
hold on;
plot(t_reconstructed,x_reconstructed)
hold off;
legend('Original', 'Reconstructed');
xlabel('t');
ylabel('Real function.')
title('Approximation with 51 terms.')
subplot(212)
plot(t_extended,imag(x_extended))
hold on
plot(t_reconstructed,imag(x_reconstructed))
hold off
legend('Original', 'Reconstructed');
xlabel('t');
ylabel('Imaginary function.')
title('Approximation with 51 terms.')
```

Same as the last problem, we were to use our earlier function to display the fourier series coefficient. New thing this time is the imaginary component of the function, something I thought would be difficult to implement. But, no, it was actually extremely simple. I included a copy of part 1's function at the bottom of this module, made it display the same approximation twice, and made the second function display the imaginary part by making it plot imag(the function). So, we learned how to graph imaginary functions here.