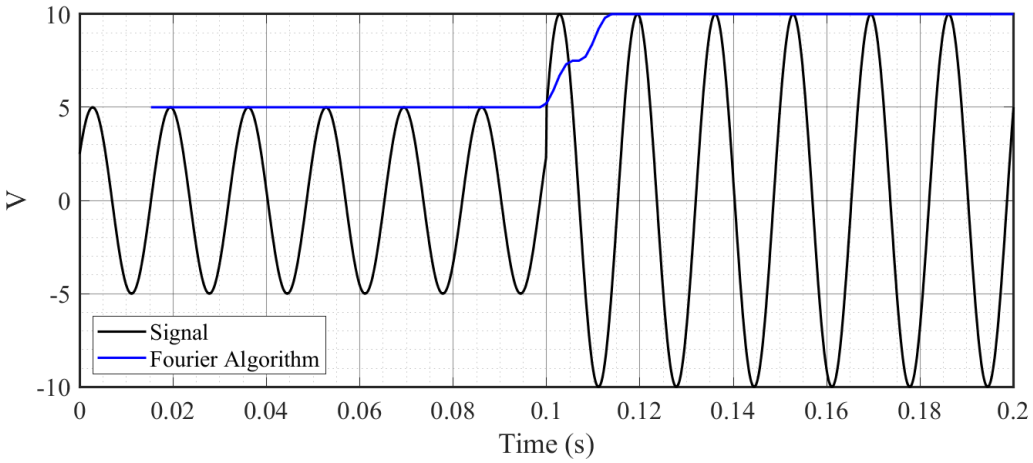
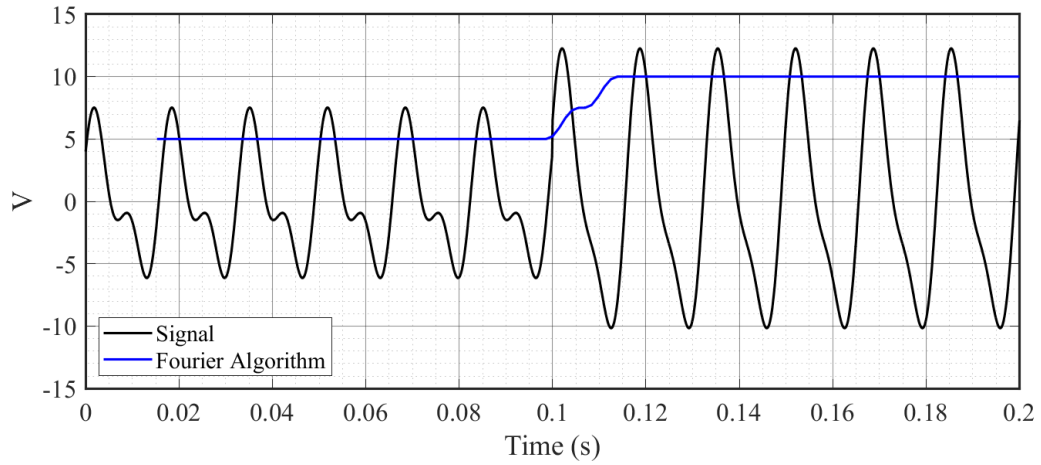
Homework 2

Submitted by Athul Jose P
WSU ID# 011867566

a) Plotted the fundamental signal with 12 samples per cycle and applied Fourier Algorithm

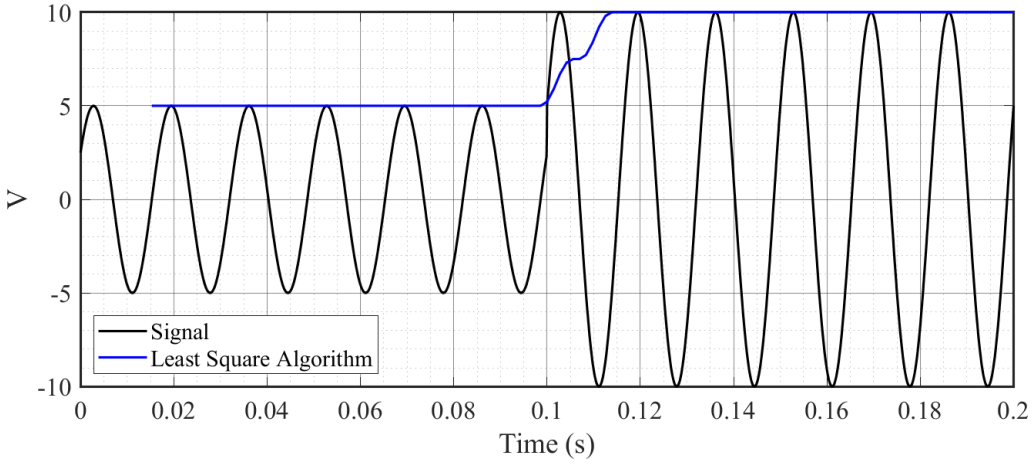


b) Plotted the fundamental signal with second harmonic and applied Fourier Algorithm

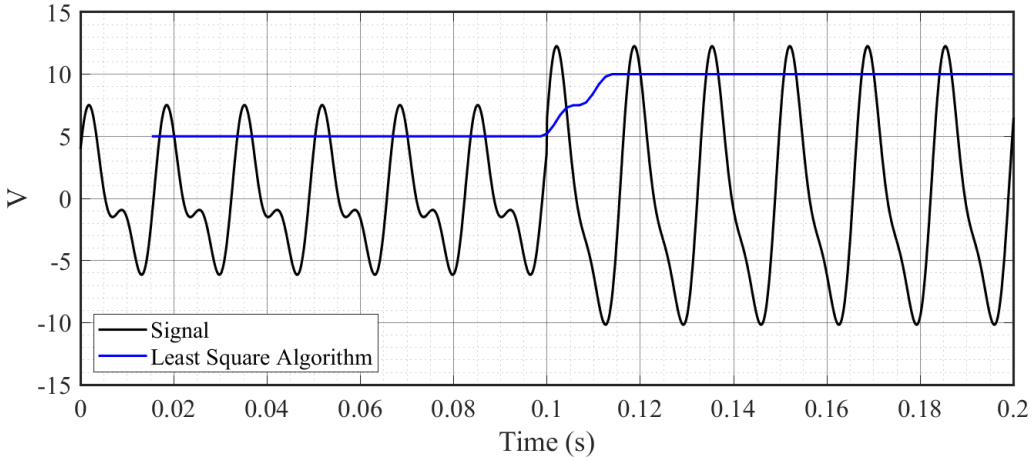


The results shows that Fourier Algorithm is excellent in calculating the fundamental amplitude of the applied signal. It is clear that the algorithm removes the second harmonic while calculating the signal amplitude. If both amplitudes are compared, similar results are obtained which means the extend of eliminating the second harmonic is successful.

c) a. Plotted the fundamental signal and applied Least Square Algorithm

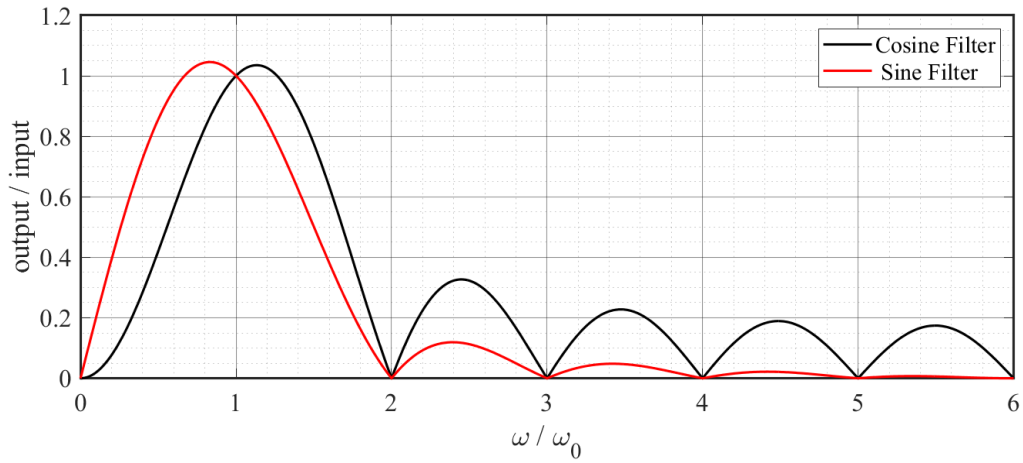


c) b. Plotted the fundamental signal with second harmonic and applied Least Square Algorithm



A similar kind of results is obtained in Least Square Algorithm. It is effectively eliminated the second harmonics and calculated the amplitude of the voltage signal. Both algorithms are good in calculating the fundamental amplitude of the given signal.

d) Drawn the frequency response of Cosine filter of Full-cycle Fourier Algorithm



The cosine filter of full-cycle Fourier algorithm gives a similar response like sine filter. However the magnitude of frequency components is higher in cosine filter. Both filters provide unity magnitude for fundamental frequency and all whole number harmonics are absent. However interharmonics are present in both filters.

MATLAB Code

1. Code for Fourier, Least Sqaure Algorithms

```
% Function for calculating Fourier Algorithm
% Fourier.m

function V_amp = Fourier(v,w,N)
del_T = (2*pi)/(w*N);
for i = N:length(v) % Loop for one window
    vcos = 0; vsin = 0;
    for j = 0:N-1
        vcos = vcos + (v(j+i-N+1) * sin((2*pi*j/N)));
        vsin = vsin + (v(j+i-N+1) * cos((2*pi*j/N)));
    end
```

```
    vcos = (2/N)*vcos; % cosine component
    vsin = (2/N)*vsin; % sine component
    V_amp(i-N+1) = sqrt(vcos^2+vsin^2); % amplitude
end
end

% Function for calculating Least Square Algorithm
% LSM.m

function V_amp = LSM(v,w,N)
del_T = (2*pi)/(w*N);
for i = 1:N % Calculating A matrix
    A(i,1) = sin((2*(i-1)-N+1)*w*del_T/2);
    A(i,2) = cos((2*(i-1)-N+1)*w*del_T/2);
end
A_inv = inv(A'*A)*A';
for i = N:length(v) % Loop for one window
    vcos = 0; vsin = 0;
    b = v(i-N+1:i);
    x = A_inv * b';
    vcos = x(1); % cosine component
    vsin = x(2); % sine component
    V_amp(i-N+1) = sqrt(vcos^2+vsin^2); % amplitude
end
end

% Main code
% main.m

clc
clear all; close all

f = 60; % frequency of signal
w = 2*pi*f; % angular frequency
N = 12; % number of samples
del_T = 1/(f*N); % sampling period
t = [0:0.0001:0.2]; % time series for plotting original signal
Ts = [0:del_T:0.2]; % time series for samples

for i = 1:length(t) % generation of signal only for plotting
    if t(i) < 0.1
        v1(i) = 5*sin((w*t(i))+(pi/6));
        v2(i) = 5*sin((w*t(i))+(pi/6)) + 3*sin((2*w*t(i))+(pi/6));
    else
        v1(i) = 10*sin((w*t(i))+(pi/6));
        v2(i) = 10*sin((w*t(i))+(pi/6))+ 3*sin((2*w*t(i))+(pi/6));
    end
end

for i = 1:length(Ts) % generation of samples
    if Ts(i) < 0.1
        vs1(i) = 5*sin((w*Ts(i))+(pi/6));
        vs2(i) = 5*sin((w*Ts(i))+(pi/6)) + 3*sin((2*w*Ts(i))+(pi/6));
    else
        vs1(i) = 10*sin((w*Ts(i))+(pi/6));
        vs2(i) = 10*sin((w*Ts(i))+(pi/6))+ 3*sin((2*w*Ts(i))+(pi/6));
    end
end
T_plot = Ts(N:length(Ts));
```

```
figure(1) % plot of signal with only fundamental frequency
plot(t,v1) % plotting original signal
hold on
V_Fourier = Fourier(vs1,w,N); %calling Fourier Algorithm
plot(T_plot,V_Fourier) % plotting amplitude
hold off

figure(2) % plot with second harmonics
plot(t,v2) % plotting original signal
hold on
V_Fourier = Fourier(vs2,w,N); %calling Fourier Algorithm
plot(T_plot,V_Fourier) % plotting amplitude
hold off

figure(3) % plot of signal with only fundamental frequency
plot(t,v1) % plotting original signal
hold on
V_LSM = LSM(vs1,w,N); %calling Least Square Algorithm
plot(T_plot,V_LSM) % plotting amplitude
hold off

figure(4) % plot with second harmonics
plot(t,v2) % plotting original signal
hold on
V_LSM = LSM(vs2,w,N); %calling Least Square Algorithm
plot(T_plot,V_LSM) % plotting amplitude
hold off
```

2. Code for Harmonic Response of Cosine Filter of Fourier Algorithm

```
clc
clear all; close all;

f = 60; % base frequency
w0 = 2 * pi * f; % base angular frequency
x = [0:0.01:6];
w = x * w0; % angular frequency
N = 12; % number of samples
dT = 1 / (f * N); % sampling period
Hc = 0;
Hs = 0;
for k=1:N % calculating response
    j = (2*k-N-1)/2;
    Hc = Hc+(2/N)*cos(j*w0*dT)*exp(1i*w*dT*j); % cosine filter
    Hs = Hs+(2/N)*sin(j*w0*dT)*exp(1i*w*dT*j); % sine filter
end
plot(x,abs(Hc),x,abs(Hs)) %plotting responses
```