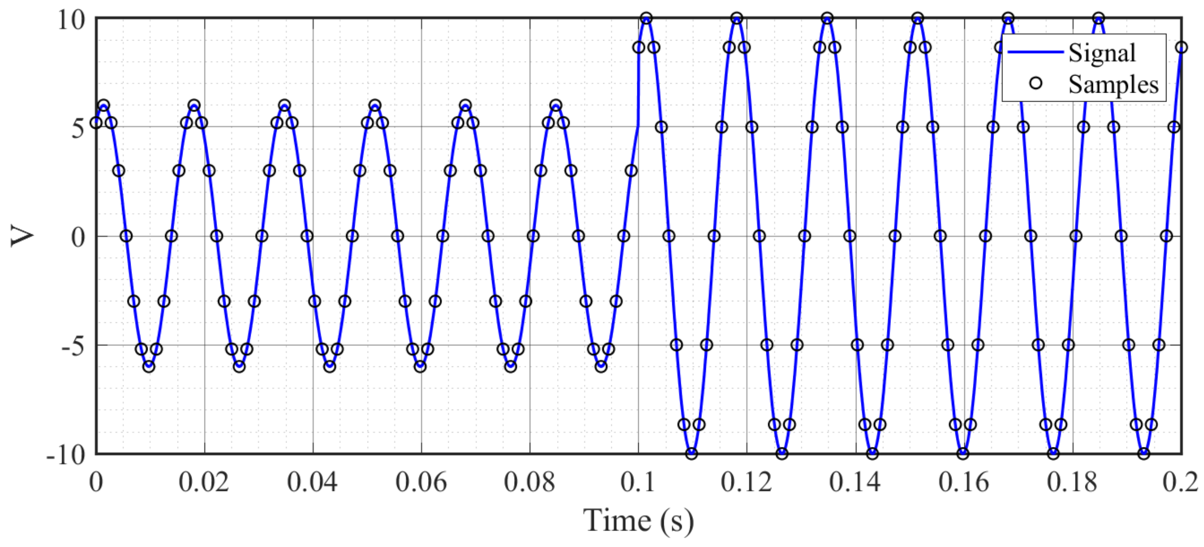
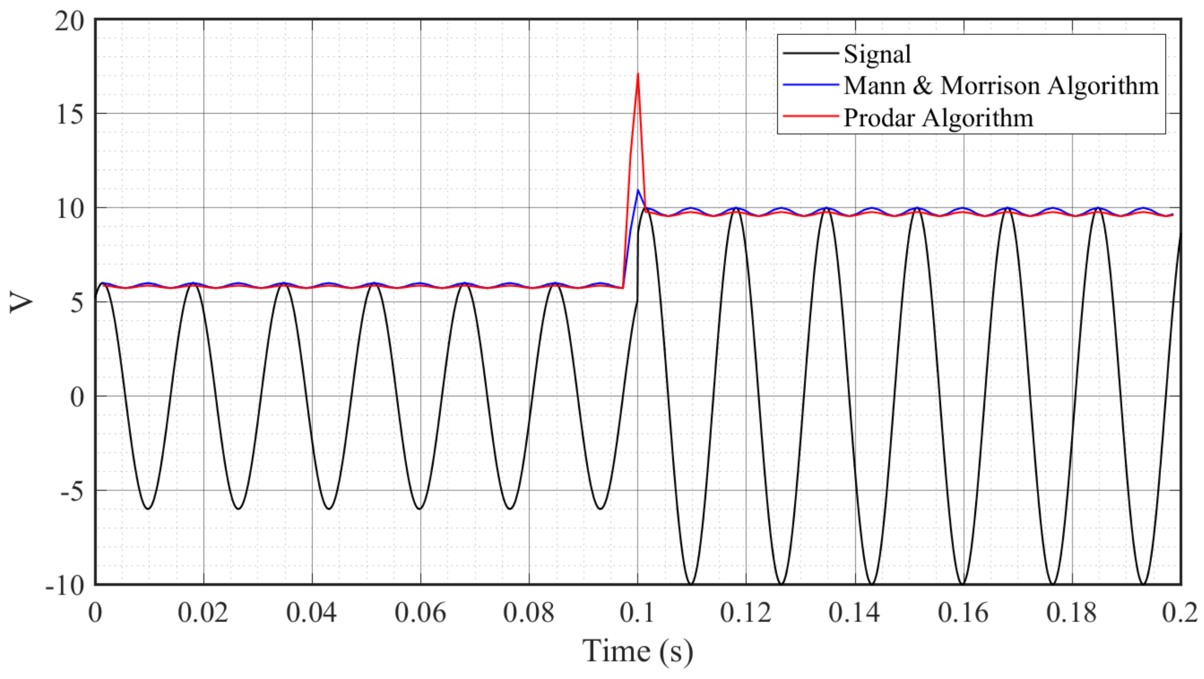
Homework 1

Submitted by Athul Jose P
WSU ID# 011867566

a) Plotted the signal with 12 samples per cycle

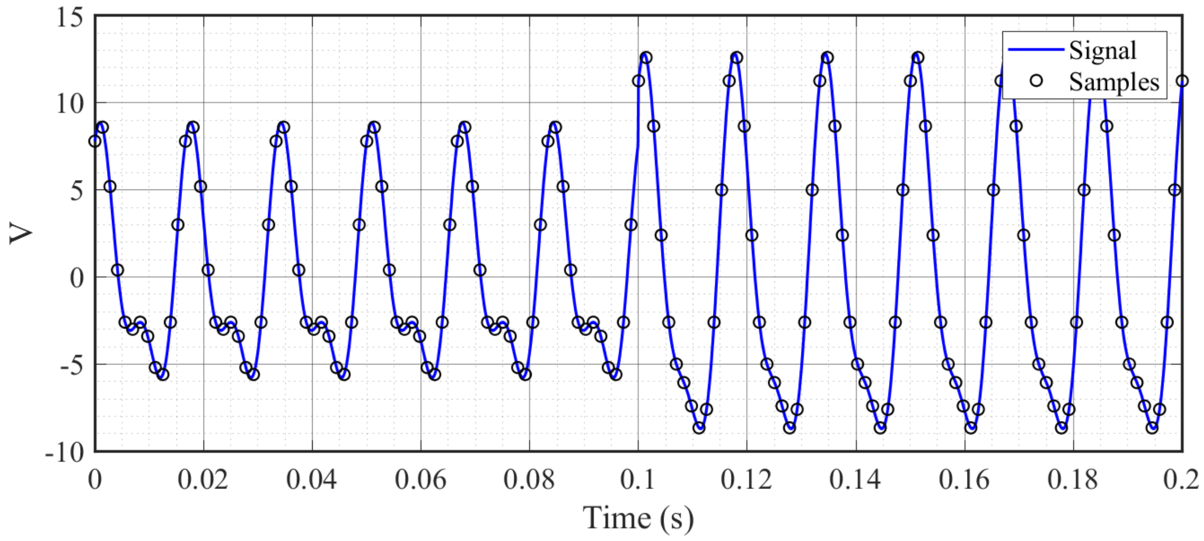


b&c) Calculated the amplitude using Mann & Morrison Algorithm and Prodar Algorithm

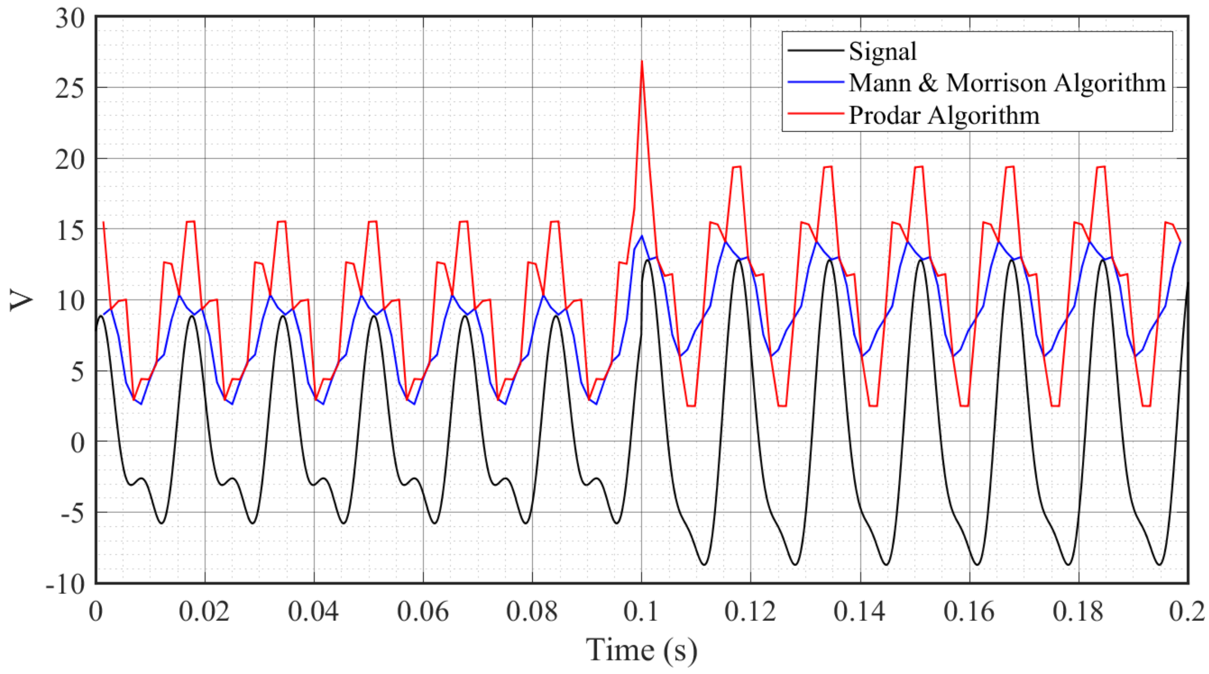


Both Mann & Morrison Algorithm and Prodar Algorithm estimates the magnitude of the given signal. An oscillatory waveform is observed in both algorithms. However, the oscillations in Prodar is less than Mann & Morrison Algorithm. In Prodar algorithm, there is a large spike observed when the signal goes through a transition. This indicates Prodar is vulnerable to sudden variations/higher harmonics.

d) Added the second harmonic and plotted the signal with 12 samples per cycle.



Calculated the amplitude using Mann & Morrison Algorithm and Prodar Algorithm



Addition of second harmonic shows the sensitivity of the algorithm to harmonics. It is evident that both algorithms fails to give amplitude estimates. From the diagram, it is clear that Prodar Algorithm is more vulnerable to harmonics than Mann & Morrison Algorithm

<u>MATLAB Code</u>

```
% Function for calculating Mann & Morrison Algorithm
% Mann_Morris.m

function V_amp = Mann_Morris(v,w,del_T)
for i = 2:length(v)-1
    vcos = (v(i+1)-v(i-1))/(2*w*del_T); % cosine component
    vsin = v(i); % sine component
    V_amp(i-1) = sqrt(vcos^2+vsin^2); %amplitude
end
end
```

```
% Function for calculating Prodar Algorithm
% Prodar.m
function V_amp = Prodar(v,w,del_T)
for i = 2:length(v)-1
    vcos = (v(i+1)-v(i-1))/(2*w*del_T); % cosine part
    vsin = (v(i-1)-(2*v(i))+v(i+1))/((w*del_T)^2); % sine part
    V_amp(i-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) = 6*sin((w*t(i))+(pi/3));
        v2(i) = 6*sin((w*t(i))+(pi/3)) + 3*sin((2*w*t(i) )+(pi/3));
    else
        v1(i) = 10*sin((w*t(i))+(pi/3));
        v2(i) = 10*sin((w*t(i))+(pi/3))+ 3*sin((2*w*t(i) )+(pi/3));
    end
end

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

figure(1) % plot of signal with only fundamental frequency
plot(t,v1) % plotting original signal
hold on
V_Mann = Mann_Morris(vs1,w,del_T);%calling Mann&Morrison Algorithm
plot(T_plot,V_Mann) % plotting amplitude
V_Prod = Prodar(vs1,w,del_T); %calling Prodar Algorithm
plot(T_plot,V_Prod) % plotting amplitude
hold off

figure(2) % plot with second harmonics
plot(t,v1) % plotting original signal
hold on
V_Mann = Mann_Morris(vs2,w,del_T);%calling Mann&Morrison Algorithm
plot(T_plot,V_Mann) % plotting amplitude
V_Prod = Prodar(vs2,w,del_T); %calling Prodar Algorithm
plot(T_plot,V_Prod) % plotting amplitude
hold off
```