Experiment No. 1 Date: 15/10/22

Aim: To simulate Amplitude Modulation (AM) & Demodulation using Python.

Objectives:

1. To develop a simulation model for time-domain representation of signals.

2. To plot & observe the AM waveform for two-tone modulation.

3. To plot & observe the AM spectrum for two-tone modulation.

4. To develop a simulation model for frequency response of the ideal discrete Low Pass Filter.

5. To plot & observe the demodulated waveform based on the coherent detection scheme.

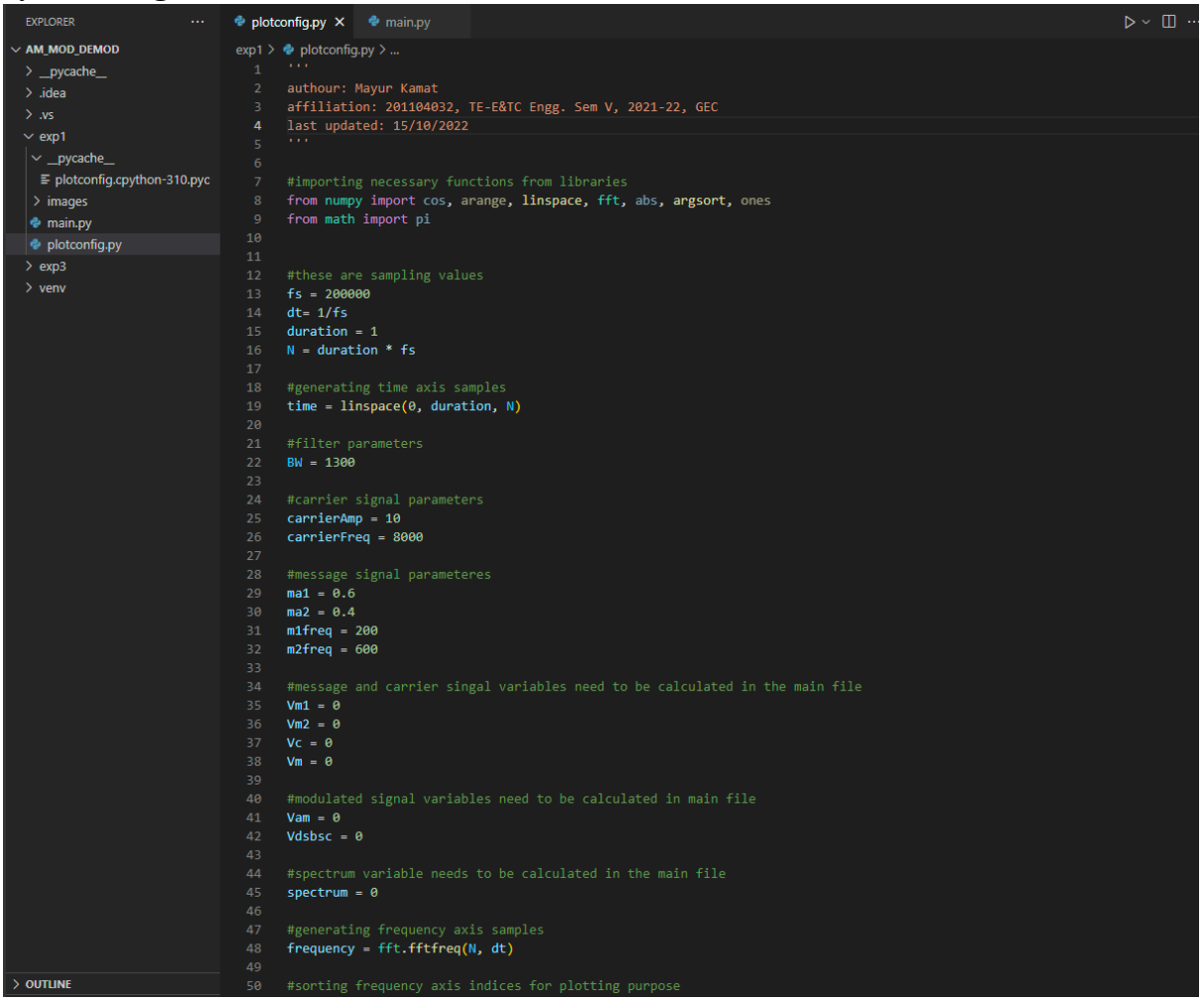
Resources/Specifications:

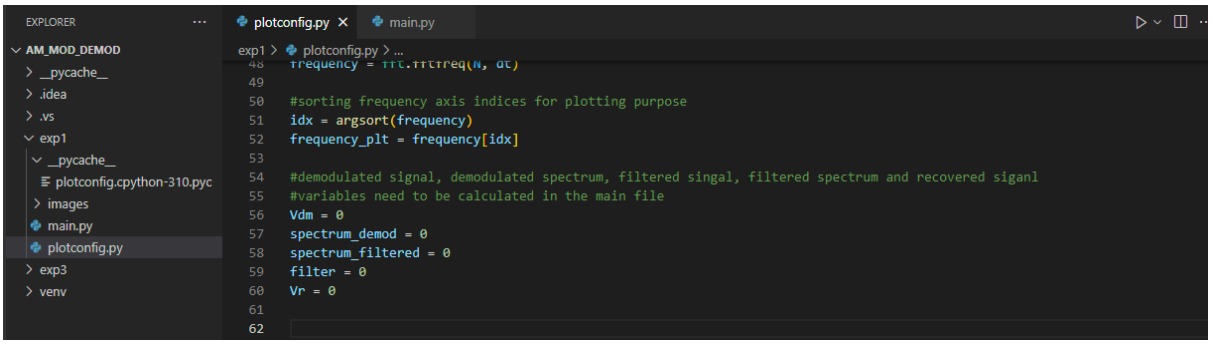
1. Desktop/Laptop System
2. Python 3 and necessary libraries

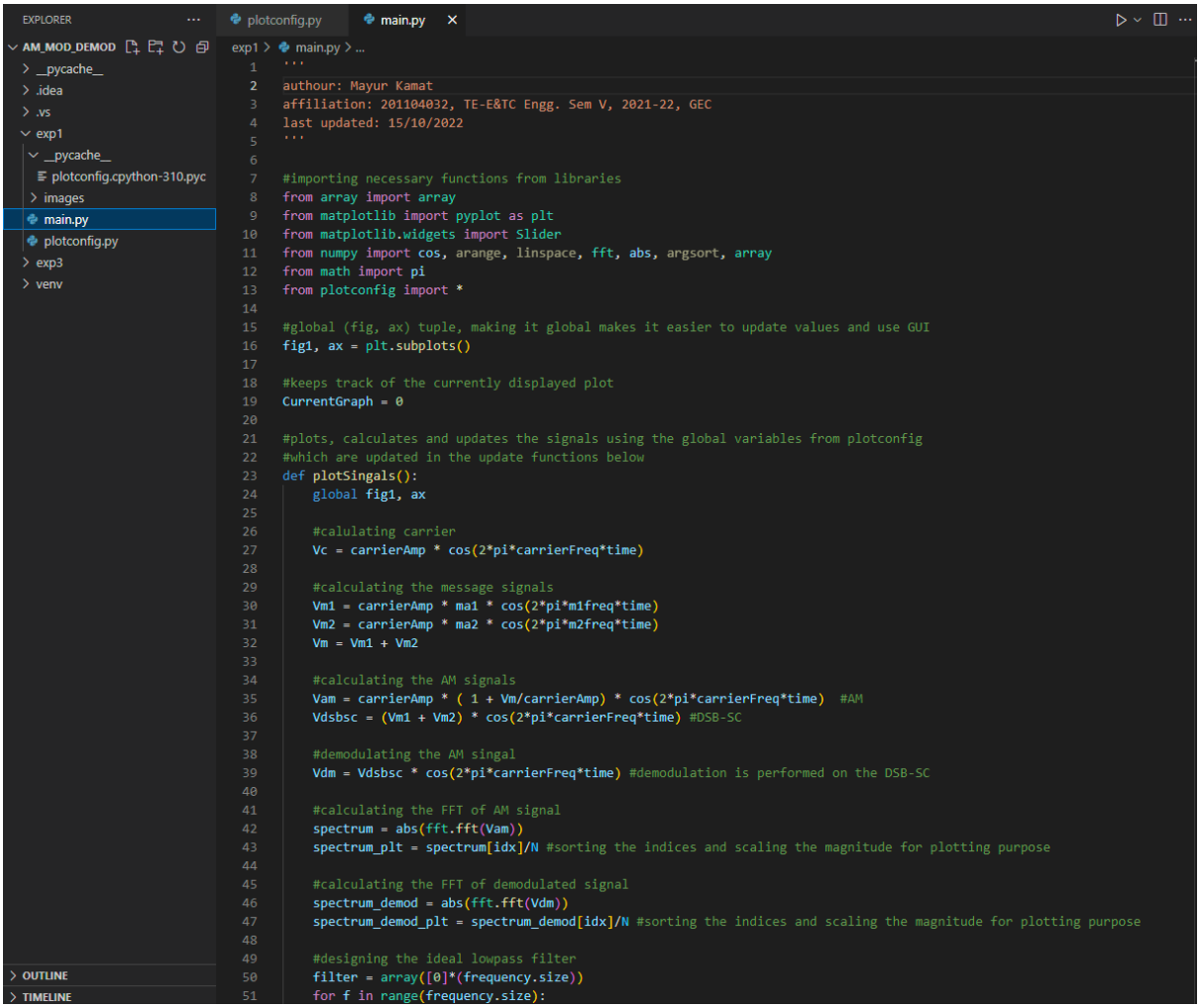
Algorithm:

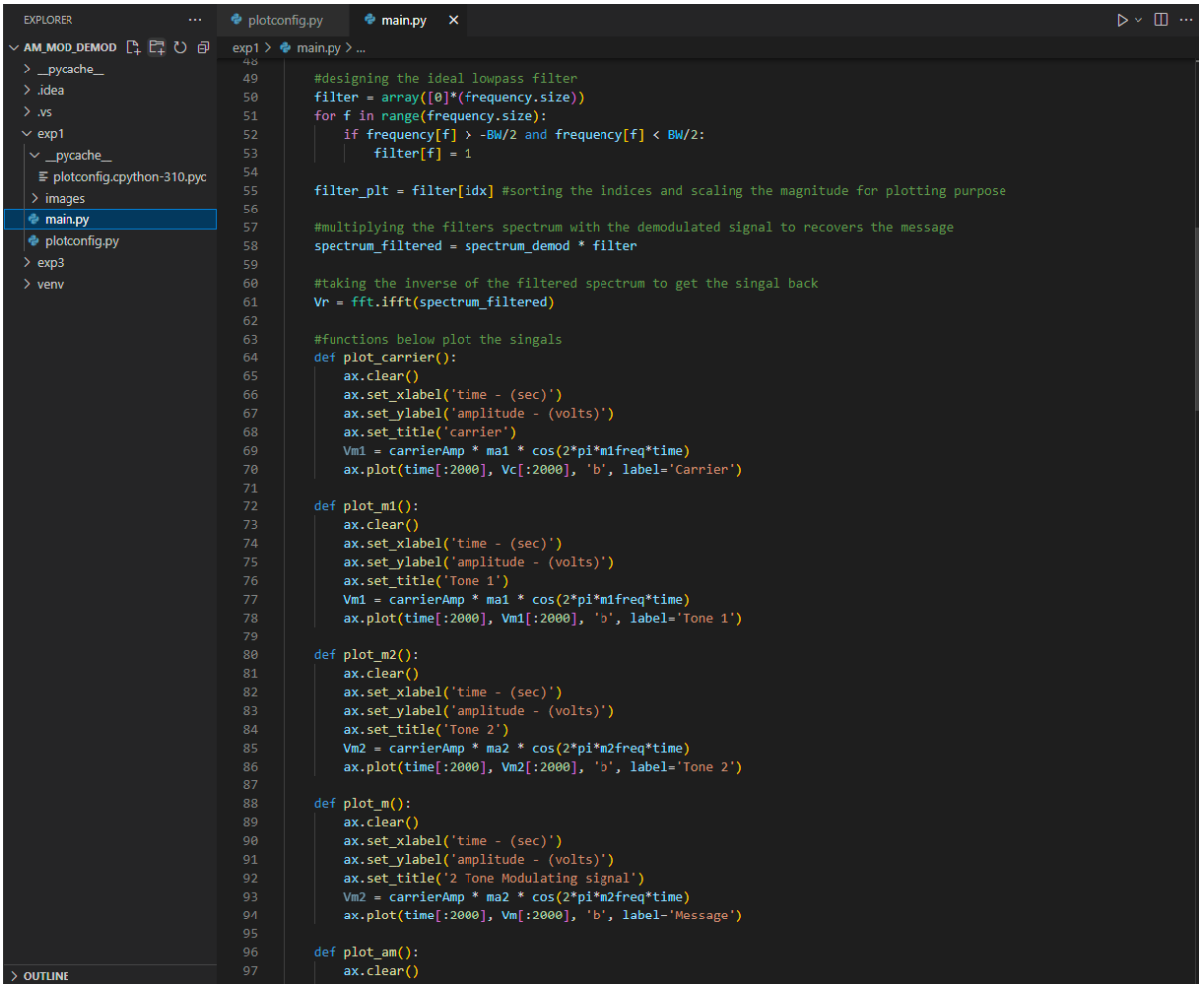
1. Define the amplitude and frequency parameters for modulating signal (two-tone) and carrier signal. Also define sampling frequency and create the array for time indices.
2. Define the AM expression based on Step 1.
3. Plot and observe the time-domain waveforms for the modulating, carrier and AM signals.
4. Implement the Discrete Fourier Transform (DFT) operation on the AM signal.
5. Shift, scale, plot and observe the AM spectrum by defining appropriate frequency indices.
6. Implement the Coherent Detection scheme using a suitable mathematical model to obtain the demodulated signal.
7. Define the ideal Discrete Low Pass Filter (LPF) response, and extract the modulating signal spectrum by multiplying the LPF response with the DFT of the demodulated signal.
8. Find the inverse transform of the spectrum extracted in Step 7 and plot the demodulated signal waveform. Compare the same with the modulation signal waveform in Step 3.

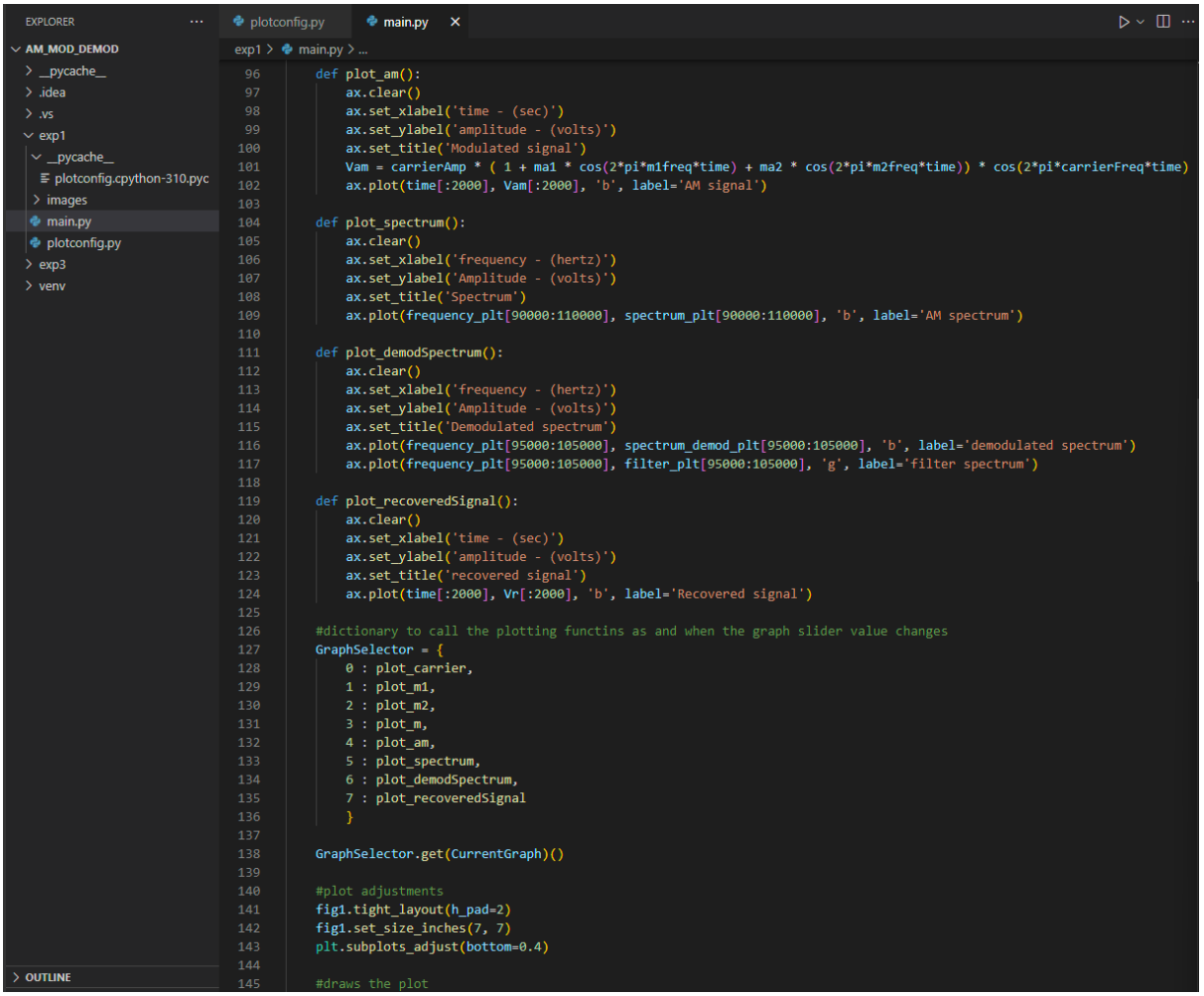
Python Program:

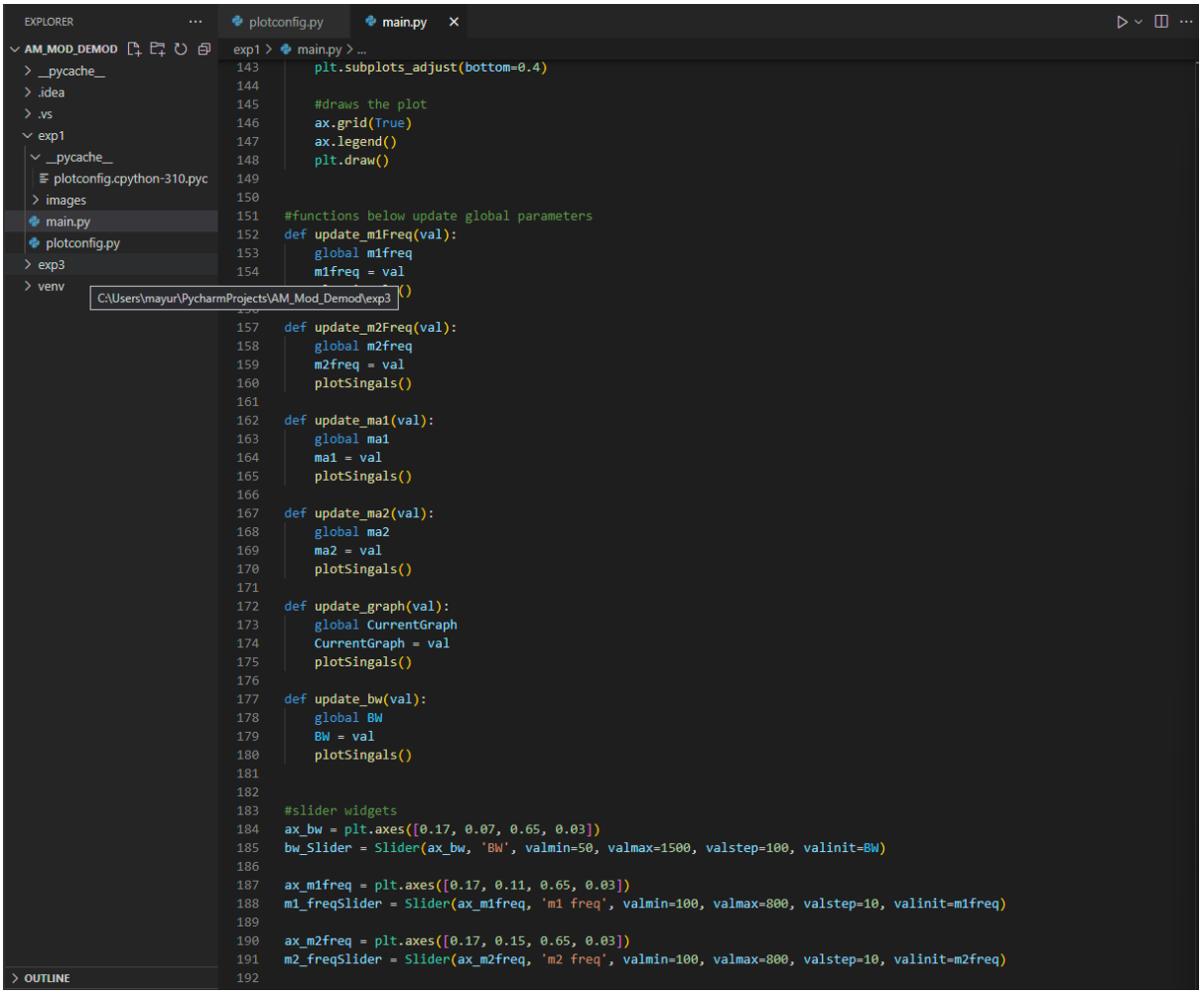


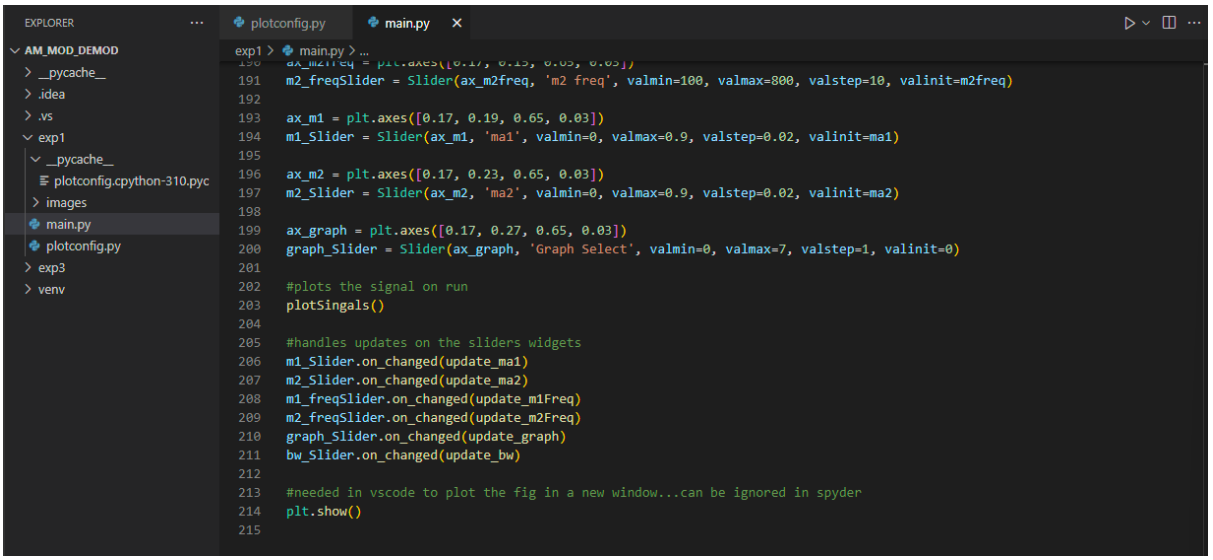



EXPLORER
plotconfig.py
main.py
AM_MOD_DEMOD
__pycache__
.idea
.vs
exp1
__pycache__
plotconfig.cpython-310.pyc
images
main.py
plotconfig.py
exp3
venv
OUTLINE
TIMELINE
exp1 > main.py > ...
'''
authour: Mayur Kamat
affiliation: 201104032, TE-E&TC Engg. Sem V, 2021-22, GEC
last updated: 15/10/2022
'''
#importing necessary functions from libraries
from array import array
from matplotlib import pyplot as plt
from matplotlib.widgets import Slider
from numpy import cos, arange, linspace, fft, abs, argsort, array
from math import pi
from plotconfig import *
#global (fig, ax) tuple, making it global makes it easier to update values and use GUI
fig1, ax = plt.subplots()
#keeps track of the currently displayed plot
CurrentGraph = 0
#plots, calculates and updates the signals using the global variables from plotconfig
#which are updated in the update functions below
def plotSingals():
global fig1, ax
#calulating carrier
Vc = carrierAmp * cos(2*pi*carrierFreq*time)
#calculating the message signals
Vm1 = carrierAmp * ma1 * cos(2*pi*m1freq*time)
Vm2 = carrierAmp * ma2 * cos(2*pi*m2freq*time)
Vm = Vm1 + Vm2
#calculating the AM signals
Vam = carrierAmp * (1 + Vm/carrierAmp) * cos(2*pi*carrierFreq*time) #AM
Vdsbsc = (Vm1 + Vm2) * cos(2*pi*carrierFreq*time) #DSB-SC
#demodulating the AM singal
Vdm = Vdsbsc * cos(2*pi*carrierFreq*time) #demodulation is performed on the DSB-SC
#calculating the FFT of AM signal
spectrum = abs(fft.fft(Vam))
spectrum_plt = spectrum[idx]/N #sorting the indices and scaling the magnitude for plotting purpose
#calculating the FFT of demodulated signal
spectrum_demod = abs(fft.fft(Vdm))
spectrum_demod_plt = spectrum_demod[idx]/N #sorting the indices and scaling the magnitude for plotting purpose
#designing the ideal lowpass filter
filter = array([0]*(frequency.size))
for f in range(frequency.size):


EXPLORER
AM_MOD_DEMOD
__pycache__
.idea
.vs
exp1
__pycache__
plotconfig.cpython-310.pyc
images
main.py
plotconfig.py
exp3
venv
OUTLINE
plotconfig.py
main.py
exp1 > main.py > ...
#designing the ideal lowpass filter
filter = array([0]*(frequency.size))
for f in range(frequency.size):
if frequency[f] > -BW/2 and frequency[f] < BW/2:
filter[f] = 1
filter_plt = filter[idx] #sorting the indices and scaling the magnitude for plotting purpose
#multiplying the filters spectrum with the demodulated signal to recovers the message
spectrum_filtered = spectrum_demod * filter
#taking the inverse of the filtered spectrum to get the singal back
Vr = fft.ifft(spectrum_filtered)
#functions below plot the singals
def plot_carrier():
ax.clear()
ax.set_xlabel('time - (sec)')
ax.set_ylabel('amplitude - (volts)')
ax.set_title('carrier')
Vm1 = carrierAmp * ma1 * cos(2*pi*m1freq*time)
ax.plot(time[:2000], Vc[:2000], 'b', label='Carrier')
def plot_m1():
ax.clear()
ax.set_xlabel('time - (sec)')
ax.set_ylabel('amplitude - (volts)')
ax.set_title('Tone 1')
Vm1 = carrierAmp * ma1 * cos(2*pi*m1freq*time)
ax.plot(time[:2000], Vm1[:2000], 'b', label='Tone 1')
def plot_m2():
ax.clear()
ax.set_xlabel('time - (sec)')
ax.set_ylabel('amplitude - (volts)')
ax.set_title('Tone 2')
Vm2 = carrierAmp * ma2 * cos(2*pi*m2freq*time)
ax.plot(time[:2000], Vm2[:2000], 'b', label='Tone 2')
def plot_m():
ax.clear()
ax.set_xlabel('time - (sec)')
ax.set_ylabel('amplitude - (volts)')
ax.set_title('2 Tone Modulating signal')
Vm2 = carrierAmp * ma2 * cos(2*pi*m2freq*time)
ax.plot(time[:2000], Vm[:2000], 'b', label='Message')
def plot_am():
ax.clear()


EXPLORER
AM_MOD_DEMOD
__pycache__
.idea
.vs
exp1
__pycache__
plotconfig.cpython-310.pyc
images
main.py
plotconfig.py
exp3
venv
OUTLINE
plotconfig.py
main.py
exp1 > main.py > ...
96 def plot_am():
97 ax.clear()
98 ax.set_xlabel('time - (sec)')
99 ax.set_ylabel('amplitude - (volts)')
100 ax.set_title('Modulated signal')
101 Vam = carrierAmp * (1 + ma1 * cos(2*pi*m1freq*time) + ma2 * cos(2*pi*m2freq*time)) * cos(2*pi*carrierFreq*time)
102 ax.plot(time[:2000], Vam[:2000], 'b', label='AM signal')
103
104 def plot_spectrum():
105 ax.clear()
106 ax.set_xlabel('frequency - (hertz)')
107 ax.set_ylabel('Amplitude - (volts)')
108 ax.set_title('Spectrum')
109 ax.plot(frequency_plt[90000:110000], spectrum_plt[90000:110000], 'b', label='AM spectrum')
110
111 def plot_demodSpectrum():
112 ax.clear()
113 ax.set_xlabel('frequency - (hertz)')
114 ax.set_ylabel('Amplitude - (volts)')
115 ax.set_title('Demodulated spectrum')
116 ax.plot(frequency_plt[95000:105000], spectrum_demod_plt[95000:105000], 'b', label='demodulated spectrum')
117 ax.plot(frequency_plt[95000:105000], filter_plt[95000:105000], 'g', label='filter spectrum')
118
119 def plot_recoveredSignal():
120 ax.clear()
121 ax.set_xlabel('time - (sec)')
122 ax.set_ylabel('amplitude - (volts)')
123 ax.set_title('recovered signal')
124 ax.plot(time[:2000], Vr[:2000], 'b', label='Recovered signal')
125
126 #dictionary to call the plotting functins as and when the graph slider value changes
127 GraphSelector = {
128 0 : plot_carrier,
129 1 : plot_m1,
130 2 : plot_m2,
131 3 : plot_m,
132 4 : plot_am,
133 5 : plot_spectrum,
134 6 : plot_demodSpectrum,
135 7 : plot_recoveredSignal
136 }
137
138 GraphSelector.get(CurrentGraph)()
139
140 #plot adjustments
141 fig1.tight_layout(h_pad=2)
142 fig1.set_size_inches(7, 7)
143 plt.subplots_adjust(bottom=0.4)
144
145 #draws the plot


EXPLORER
plotconfig.py
main.py
AM_MOD_DEMOD
exp1 > main.py > ...
__pycache__
.idea
.vs
exp1
__pycache__
plotconfig.cpython-310.pyc
images
main.py
plotconfig.py
exp3
venv
C:\Users\mayur\PycharmProjects\AM_Mod_Demod\exp3
OUTLINE
143 plt.subplots_adjust(bottom=0.4)
144
145 #draws the plot
146 ax.grid(True)
147 ax.legend()
148 plt.draw()
149
150
151 #functions below update global parameters
152 def update_m1Freq(val):
153 global m1freq
154 m1freq = val
157 def update_m2Freq(val):
158 global m2freq
159 m2freq = val
160 plotSingals()
161
162 def update_ma1(val):
163 global ma1
164 ma1 = val
165 plotSingals()
166
167 def update_ma2(val):
168 global ma2
169 ma2 = val
170 plotSingals()
171
172 def update_graph(val):
173 global CurrentGraph
174 CurrentGraph = val
175 plotSingals()
176
177 def update_bw(val):
178 global BW
179 BW = val
180 plotSingals()
181
182
183 #slider widgets
184 ax_bw = plt.axes([0.17, 0.07, 0.65, 0.03])
185 bw_Slider = Slider(ax_bw, 'BW', valmin=50, valmax=1500, valstep=100, valinit=BW)
186
187 ax_m1freq = plt.axes([0.17, 0.11, 0.65, 0.03])
188 m1_freqSlider = Slider(ax_m1freq, 'm1 freq', valmin=100, valmax=800, valstep=10, valinit=m1freq)
189
190 ax_m2freq = plt.axes([0.17, 0.15, 0.65, 0.03])
191 m2_freqSlider = Slider(ax_m2freq, 'm2 freq', valmin=100, valmax=800, valstep=10, valinit=m2freq)
192


EXPLORER
plotconfig.py
main.py
AM_MOD_DEMOD
exp1 > main.py > ...
__pycache__
.idea
.vs
exp1
__pycache__
plotconfig.cpython-310.pyc
images
main.py
plotconfig.py
exp3
venv
191 m2_freqSlider = Slider(ax_m2freq, 'm2 freq', valmin=100, valmax=800, valstep=10, valinit=m2freq)
192
193 ax_m1 = plt.axes([0.17, 0.19, 0.65, 0.03])
194 m1_Slider = Slider(ax_m1, 'ma1', valmin=0, valmax=0.9, valstep=0.02, valinit=ma1)
195
196 ax_m2 = plt.axes([0.17, 0.23, 0.65, 0.03])
197 m2_Slider = Slider(ax_m2, 'ma2', valmin=0, valmax=0.9, valstep=0.02, valinit=ma2)
198
199 ax_graph = plt.axes([0.17, 0.27, 0.65, 0.03])
200 graph_Slider = Slider(ax_graph, 'Graph Select', valmin=0, valmax=7, valstep=1, valinit=0)
201
202 #plots the signal on run
203 plotSingals()
204
205 #handles updates on the sliders widgets
206 m1_Slider.on_changed(update_ma1)
207 m2_Slider.on_changed(update_ma2)
208 m1_freqSlider.on_changed(update_m1Freq)
209 m2_freqSlider.on_changed(update_m2Freq)
210 graph_Slider.on_changed(update_graph)
211 bw_Slider.on_changed(update_bw)
212
213 #needed in vscode to plot the fig in a new window...can be ignored in spyder
214 plt.show()
215

Plots:

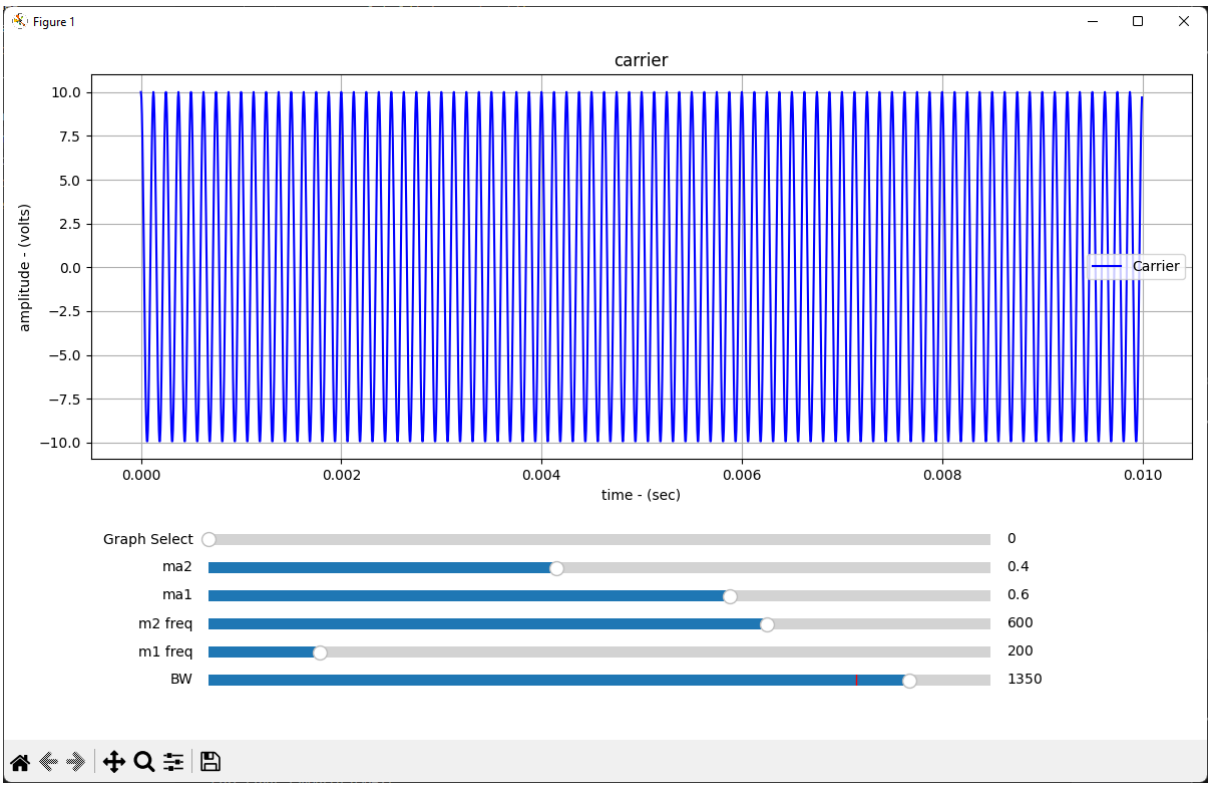


Figure 1
carrier
amplitude - (volts)
time - (sec)
Carrier
Graph Select 0
ma2 0.4
ma1 0.6
m2 freq 600
m1 freq 200
BW 1350

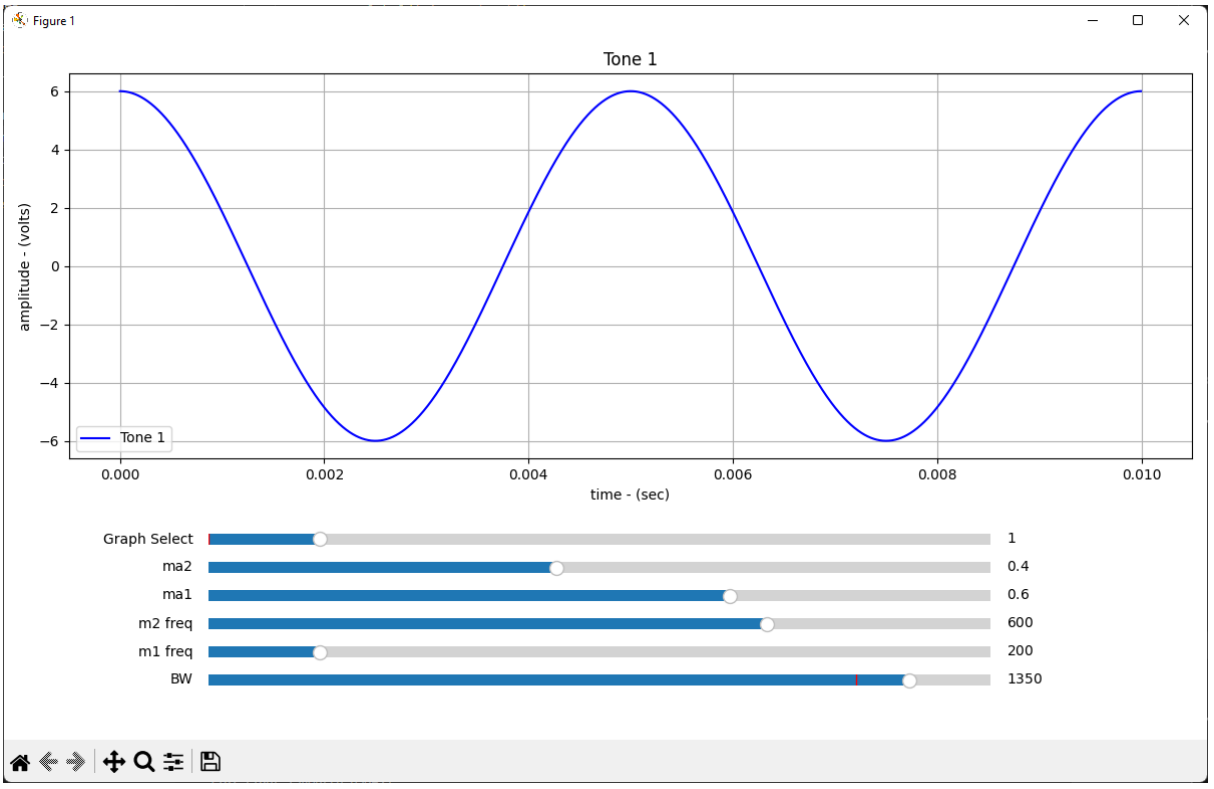


Figure 1
Tone 1
amplitude - (volts)
time - (sec)
Tone 1
Graph Select 1
ma2 0.4
ma1 0.6
m2 freq 600
m1 freq 200
BW 1350

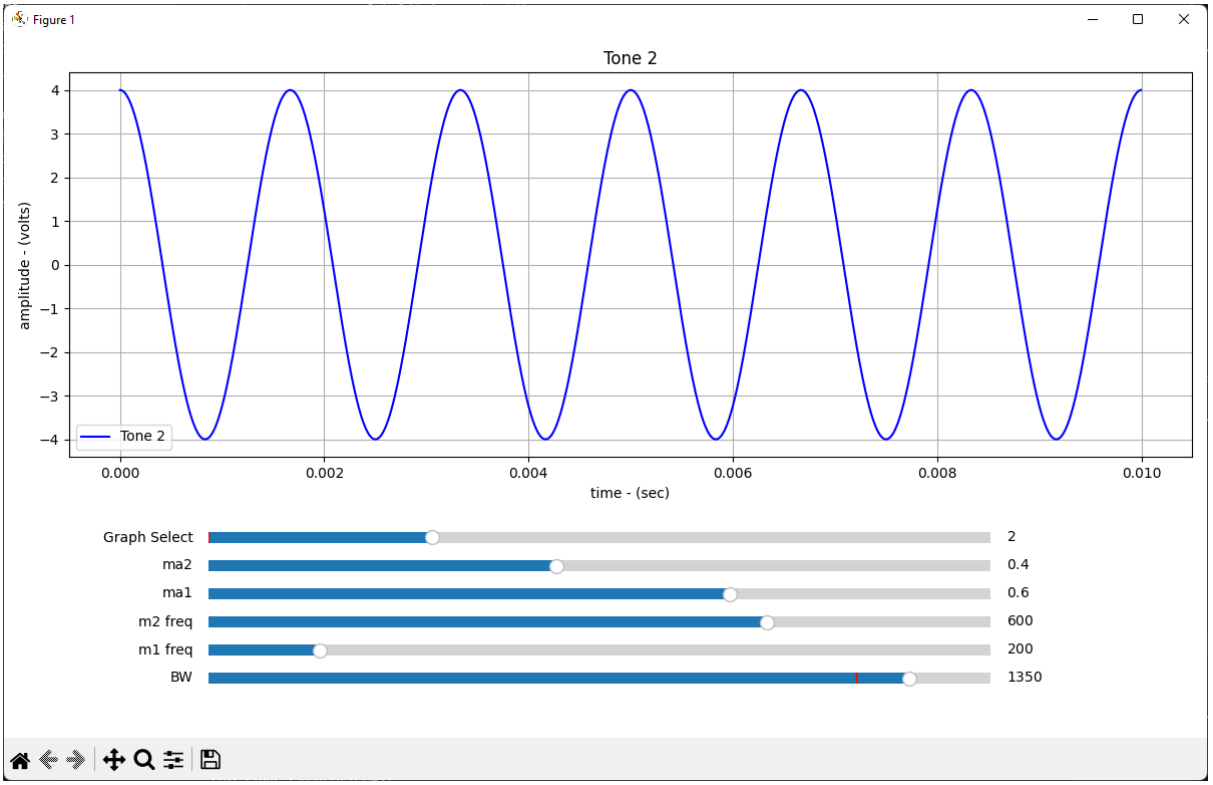


Figure 1
Tone 2
amplitude - (volts)
time - (sec)
Tone 2
Graph Select 2
ma2 0.4
ma1 0.6
m2 freq 600
m1 freq 200
BW 1350

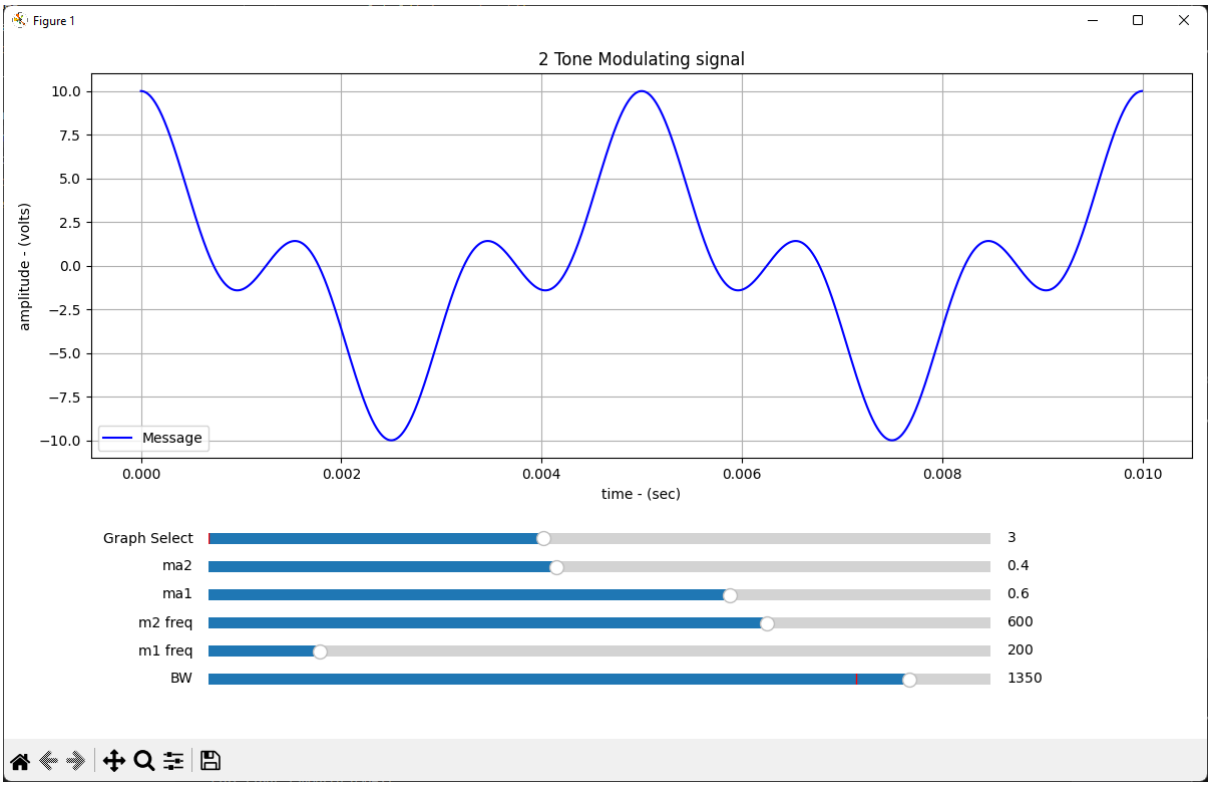


Figure 1
2 Tone Modulating signal
amplitude - (volts)
time - (sec)
Message
Graph Select 3
ma2 0.4
ma1 0.6
m2 freq 600
m1 freq 200
BW 1350

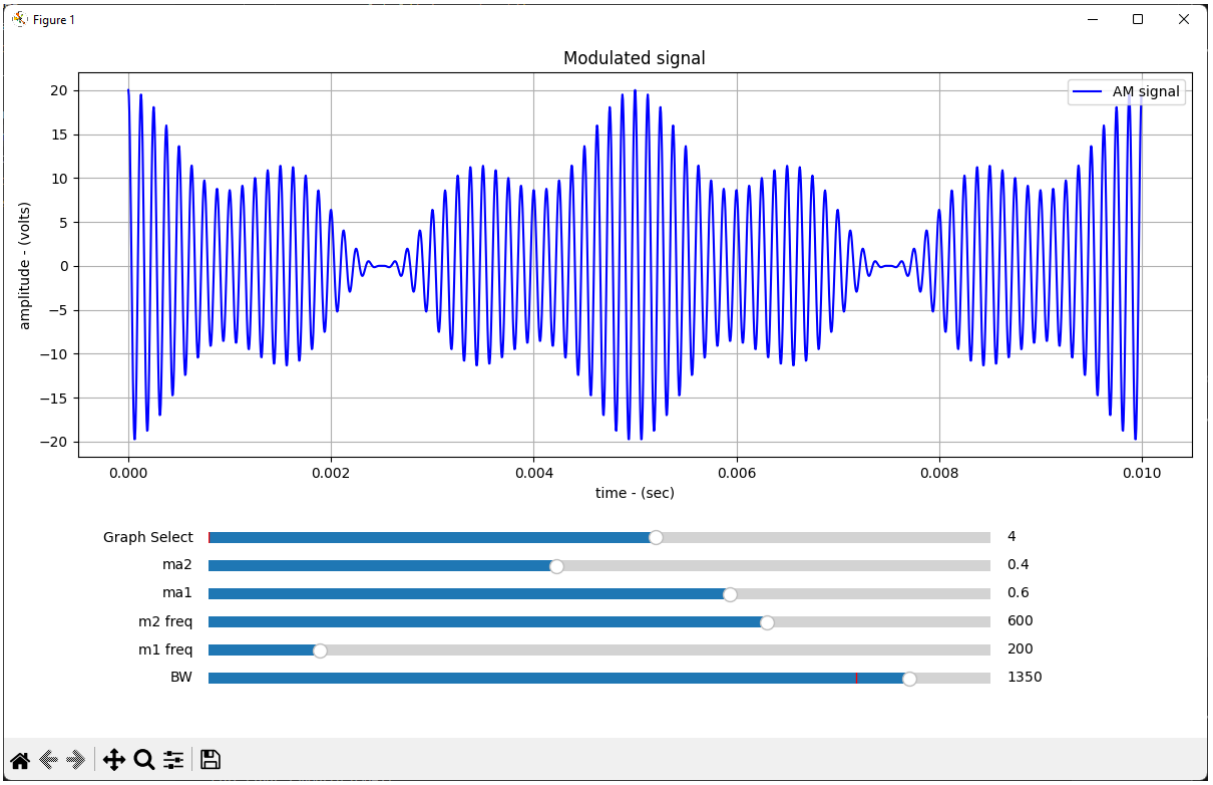


Figure 1
Modulated signal
AM signal
amplitude - (volts)
time - (sec)
Graph Select 4
ma2 0.4
ma1 0.6
m2 freq 600
m1 freq 200
BW 1350

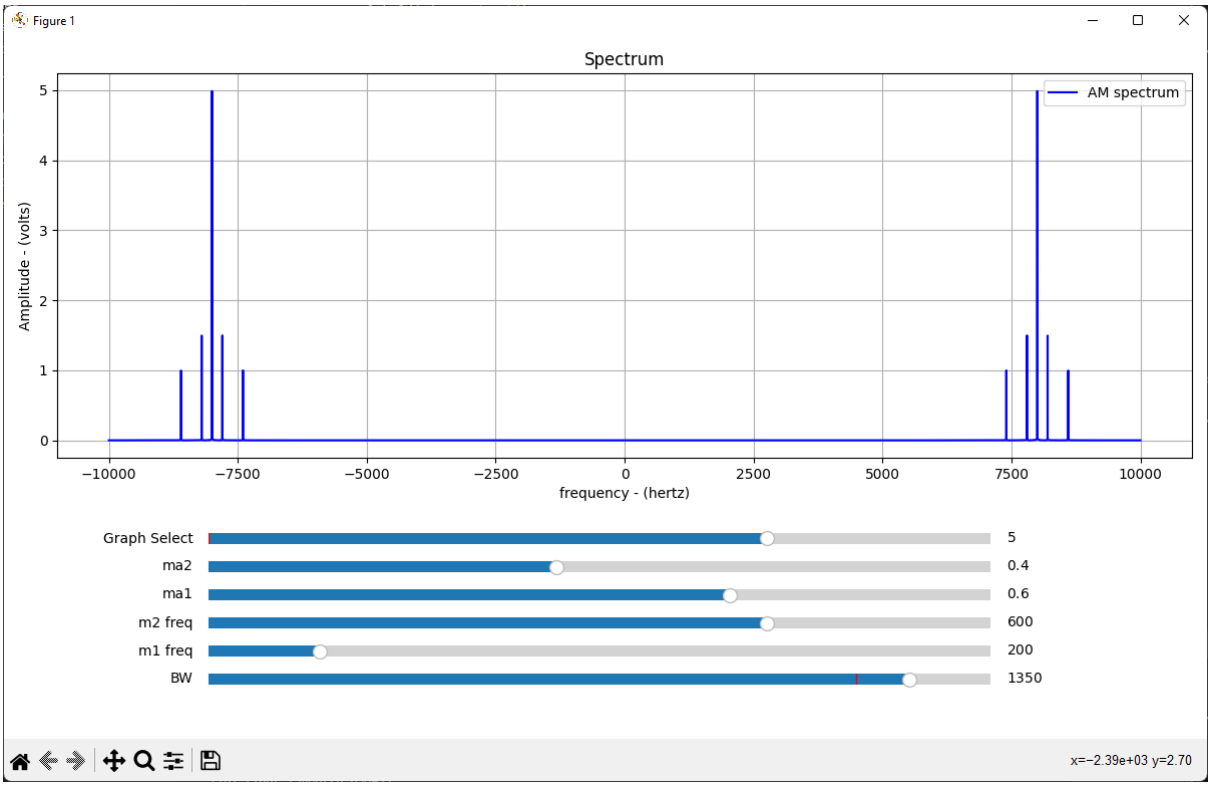


Figure 1
Spectrum
AM spectrum
Amplitude - (volts)
frequency - (hertz)
Graph Select 5
ma2 0.4
ma1 0.6
m2 freq 600
m1 freq 200
BW 1350
x=−2.39e+03 y=2.70

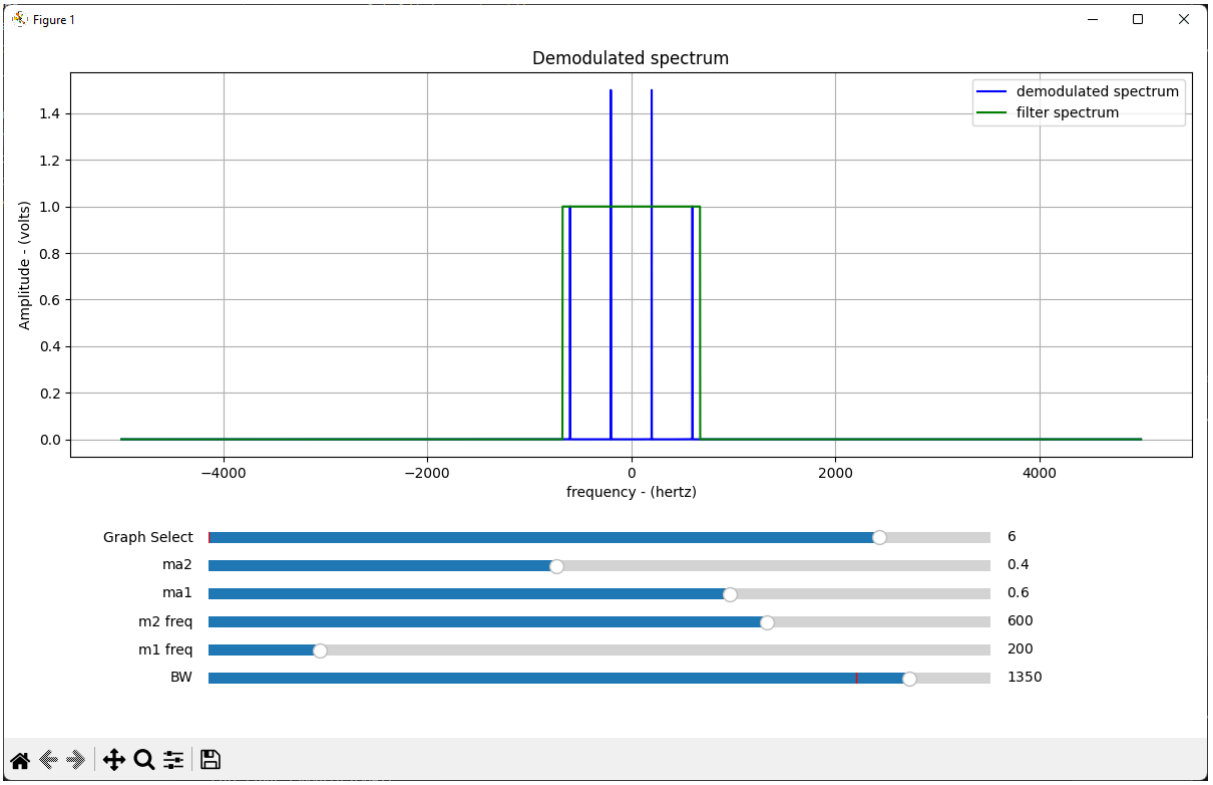


Figure 1
Demodulated spectrum
demodulated spectrum
filter spectrum
Amplitude - (volts)
frequency - (hertz)
Graph Select 6
ma2 0.4
ma1 0.6
m2 freq 600
m1 freq 200
BW 1350

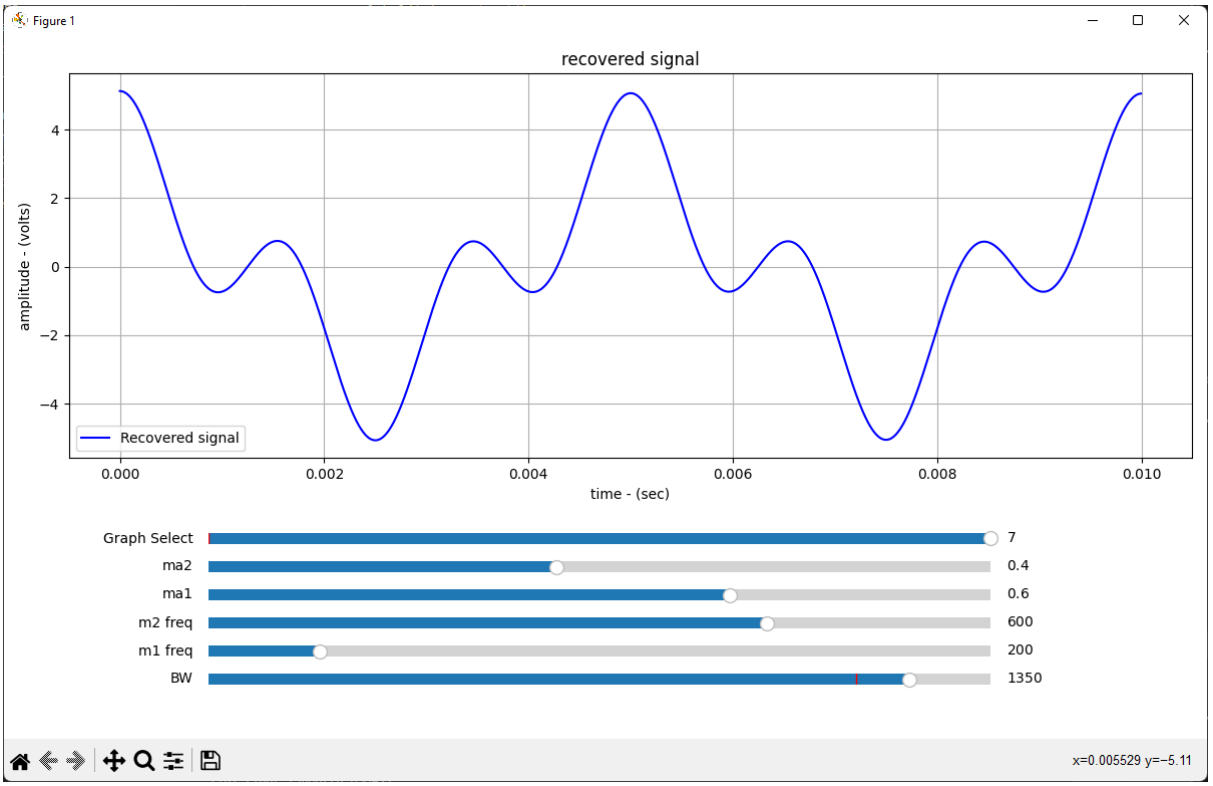


Figure 1
recovered signal
Recovered signal
amplitude - (volts)
time - (sec)
Graph Select 7
ma2 0.4
ma1 0.6
m2 freq 600
m1 freq 200
BW 1350
x=0.005529 y=−5.11

Parameter Settings:

Modulating Signal 1: Sinusoid with $V_{m1} = 0.6 * 10 = 6$ Volts , $f_{m1} = 200$ Hz

Modulating Signal 2: Sinusoid with $V_{m2} = 0.4 * 10 = 4$ Volts , $f_{m2} = 600$ Hz .

Carrier Signal: Sinusoid with $V_c = 10$ Volts , $f_c = 8000$ Hz.

LPF Cut-off Frequency = BW/2 = 1350/2 = 675 Hz.

Conclusion:

Signature of the Instructor