

## Lab 2: Amplitude Modulation and Delays

*You are going to create a function that adds echoes to an inputted audio signal and then applies amplitude modulation to the signal with echoes.*

- 1) In cell 1: import the necessary libraries and audio files
- 2) In cells 2 through 5: Copy the following functions (one in each cell)
  - plotAudio2()
  - subplots()
  - makeEchoes()
  - amplitudeModulation()
- 3) In cell 6: create a function called echoAM() that inputs
  - an audio signal
  - the audio signal's sampling rate
  - a delay time (in seconds)
  - a list containing the amplitude for the delays (# of delays specified by the length of the list)
  - an amplitude modulation frequency
  - an amplitude modulation amplitude
  - a modulation index

The function should

- plot the original signal in the time domain using plotAudio2()
- use makeEchoes() to apply a delay according to the inputted values to the inputted signal
- plot the delay in the time domain using plotAudio2()
- use amplitudeModulation() to apply amplitude modulation according to the inputted to the delayed signal and to plot the Modulator Signal, Carrier Signal, and Product Signal as subplots in a single plot
- return the modified signal

Remember the basic template for creating functions

```
In [1]: 1 def printText(string):
        2     print(string)

In [2]: 1 printText('Hello world!')
Hello world!
```

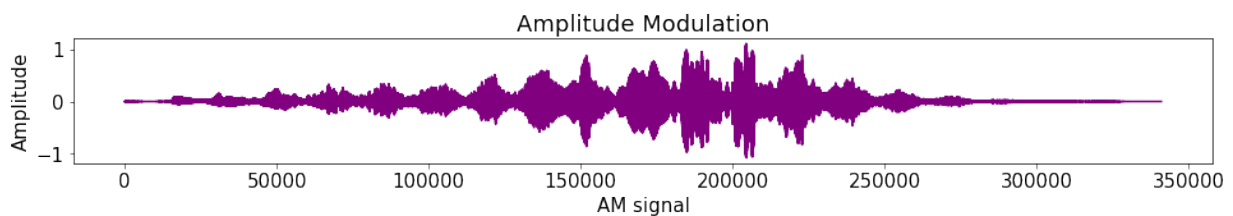
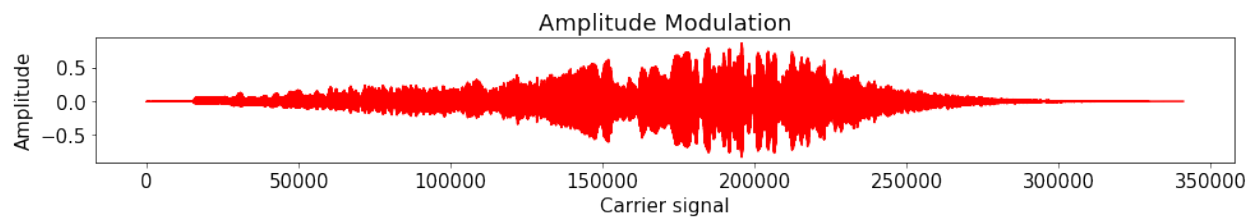
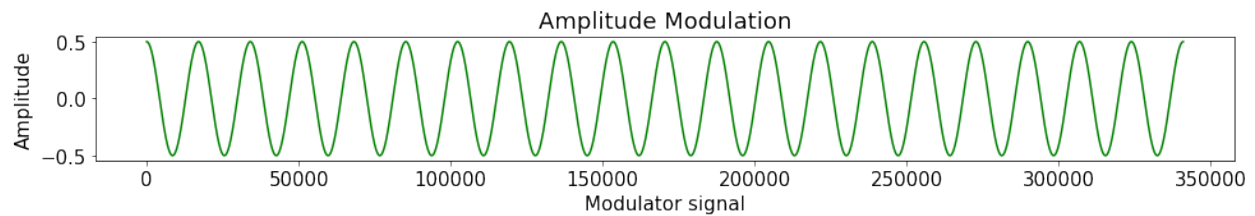
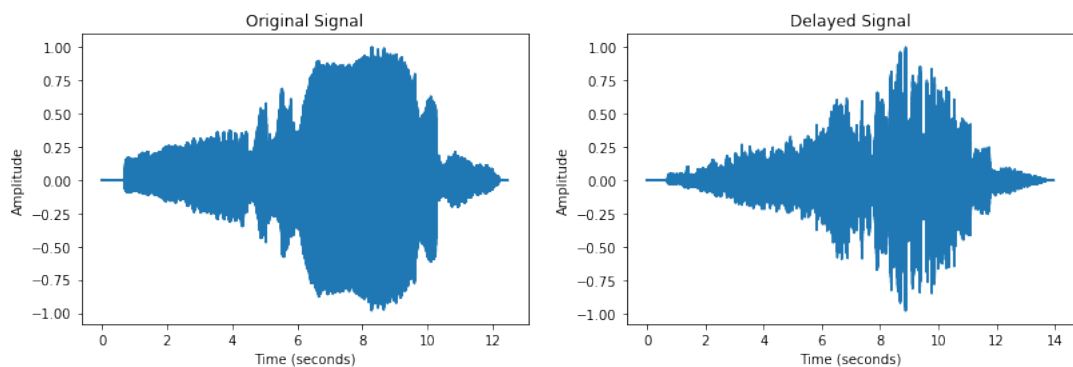
- 4) In cell 7: use librosa.load() to open avm.wav and use IPython.display.Audio() to play it  
sig , sr = librosa.load('avm.wav')

5) In cell 8: run the function with the following arguments and use `IPython.display.Audio()` to play the modulated signal (named `modSignal1`)

```
delay = 0.5  
echoes = [1,1,1]  
modulatorFreq = 20  
modulatorAmp = 0.5  
modIndex = 1
```

```
modSignal1 = echoAM(sig,sr,delay,echoes,modulatorFreq,modulatorAmp,modIndex)
```

This should generate the following plots (note that the first two plots will be above and below each other, not side by side)

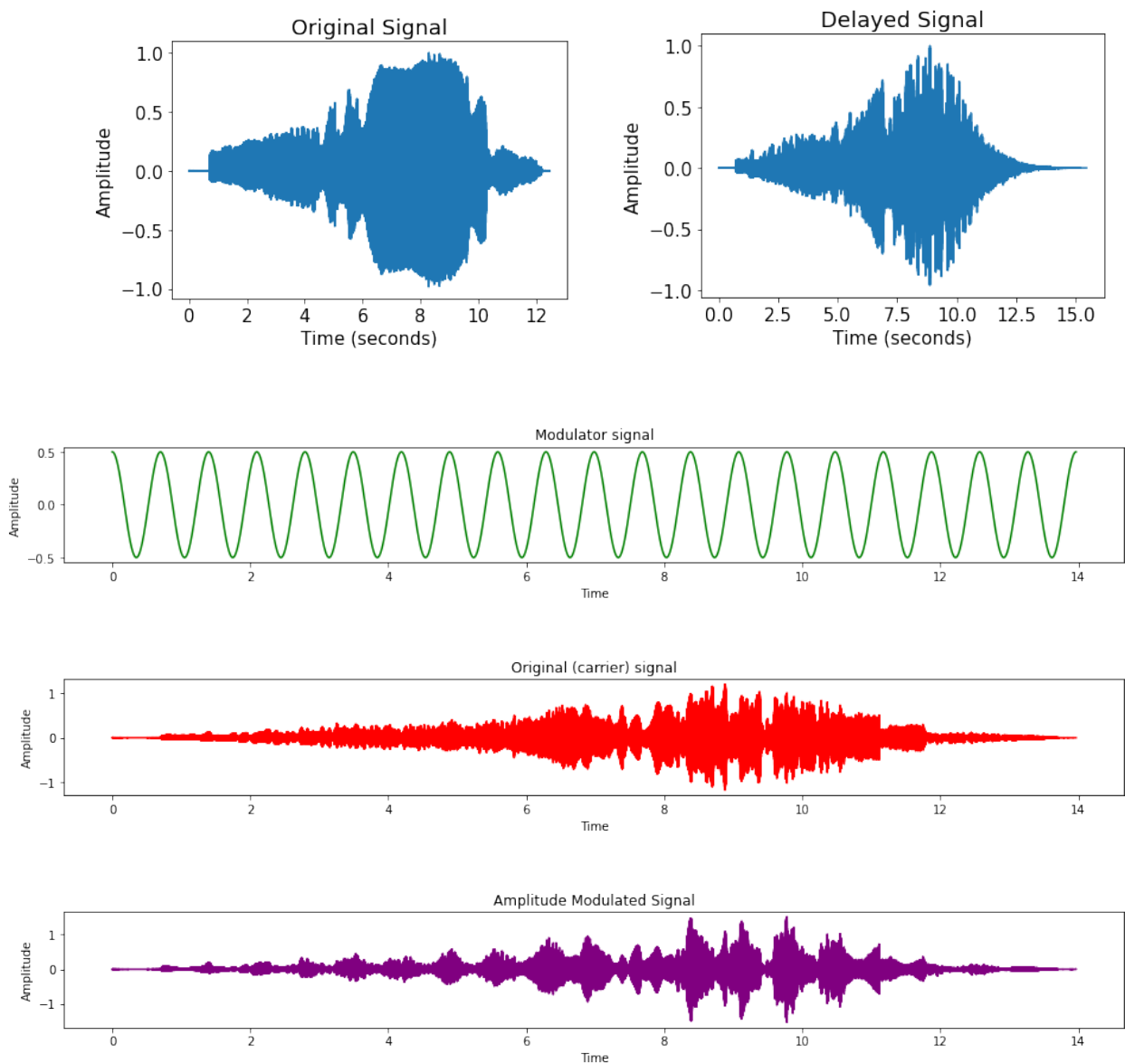


6) In cell 9: run the function with the following arguments and use `IPython.display.Audio()` to play the modulated signal (named `modSignal2`)

```
delay = 0.5  
echoes = [1,0.5,0.25,0.125,0.0625,0.03125]  
modulatorFreq = 20  
modulatorAmp = 0.5  
modIndex = 1
```

```
modSignal2 = echoAM(sig,sr,delay,echoes,modulatorFreq,modulatorAmp,modIndex)
```

This should generate the following plots (note that the first two plots will be above and below each other, not side by side)

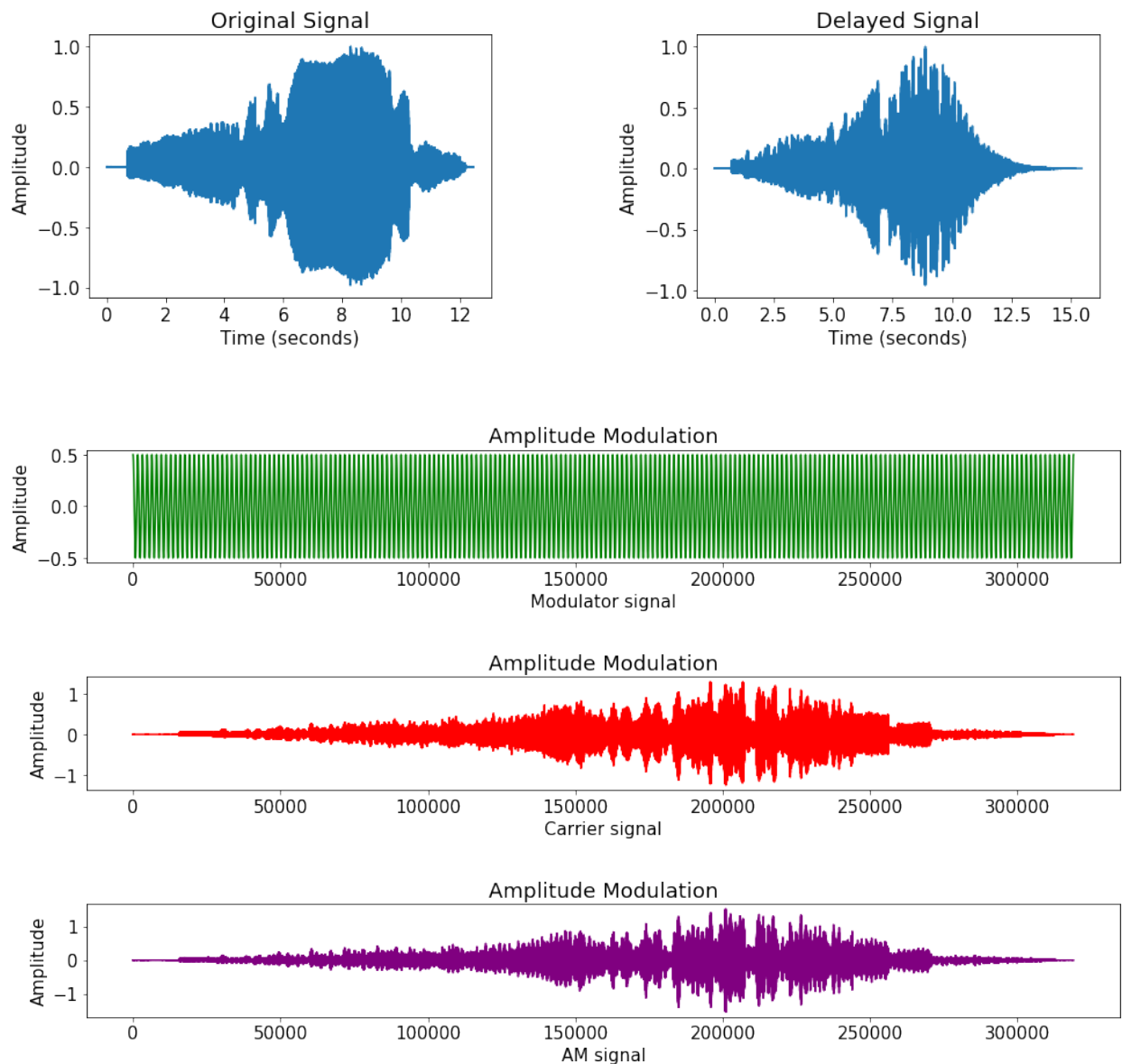


7) In cell 10: run the function with the following arguments and use `IPython.display.Audio()` to play the modulated signal (named `modSignal3`)

```
delay = 0.5  
echoes = [1,1,1,1]  
modulatorFreq = 200  
modulatorAmp = 0.5  
modIndex = 0.5
```

```
modSignal3 = echoAM(sig,sr,delay,echoes,modulatorFreq,modulatorAmp,modIndex)
```

This should generate the following plots (note that the first two plots will be above and below each other, not side by side)

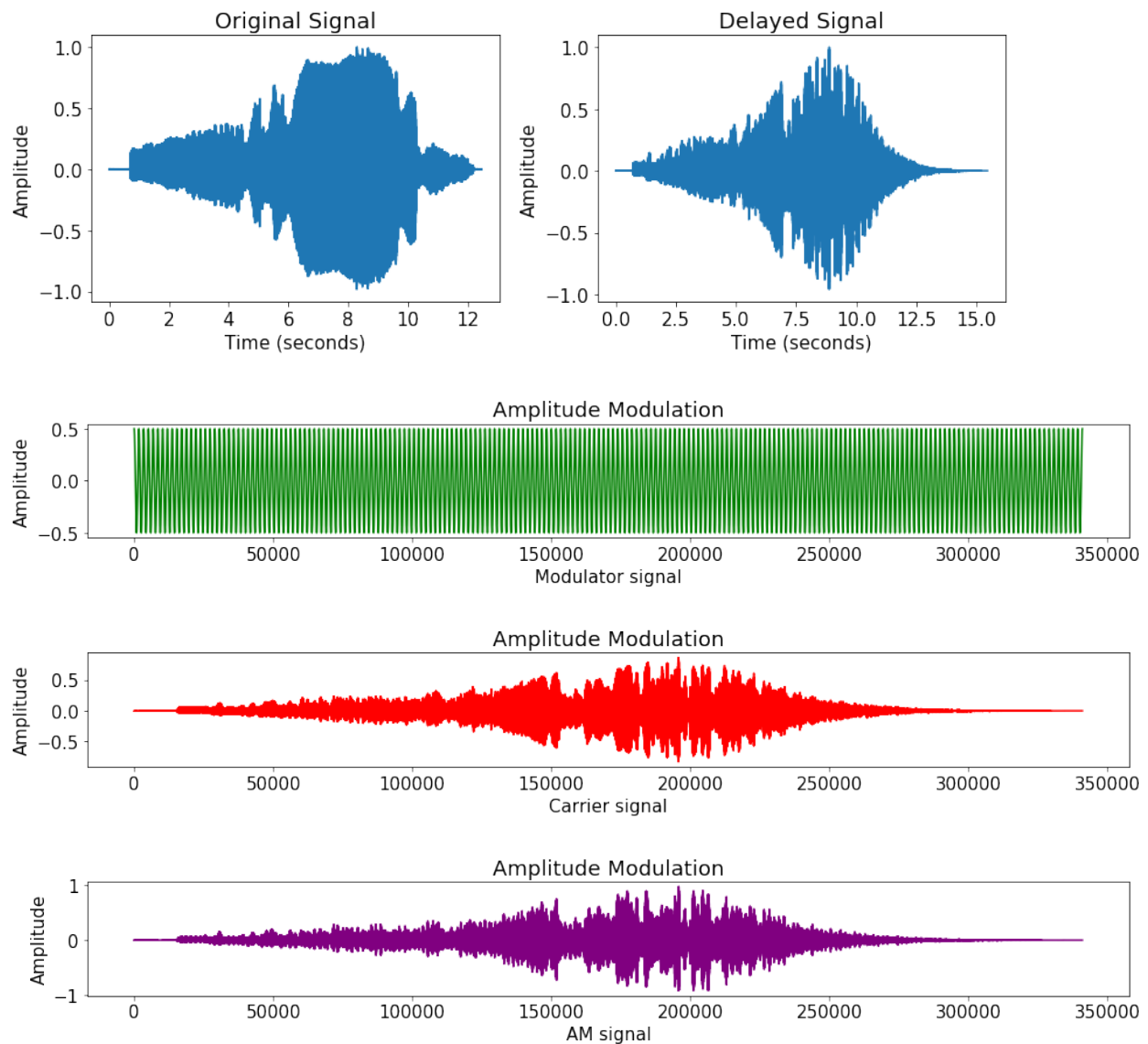


8) In cell 11: run the function with the following arguments and use `IPython.display.Audio()` to play the modulated signal (named `modSignal4`)

```
echoes = [1,0.5,0.25,0.125,0.0625,0.03125]
delay = 0.5
modulatorFreq = 200
modulatorAmp = 0.5
modIndex = 0.5
```

```
modSignal4 = echoAM(sig,sr,delay,echoes,modulatorFreq,modulatorAmp,modIndex)
```

This should generate the following plots (note that the first two plots will be above and below each other, not side by side)



9) In cells 12 and 13 run the function twice more, using your own settings for the inputted variables. You can also run the function with other audio files if you like.

10) Name the file labAssignment2-*name* (where *name* is your last name) and sync to your icm2023-*name* repository on the Github server and submit a link to the file on Blackboard

### Workflow suggestion

Rather than writing the entire function in Step 3 (Cell 6) all at once. I recommend writing it stages. To do so, you'll also have to complete Steps 4 and 5 (Cells 7 and 8) and make modified calls to the echoAM function (see below).

#### Plot original signal

- echoAM(sig,sr)

#### Create a delayed signal

- echoAM(sig,sr,delay,echoes)

#### Plot delayed signal

- echoAM(sig,sr,delay,echoes)

#### Create amplitude modulated signal

- echoAM(sig,sr,delay,echoes,modulatorFreq,modulatorAmp,modIndex)

#### Return modified signal

- modSignal1 = echoAM(sig,sr,delay,echoes,modulatorFreq,modulatorAmp,modIndex)

The following image gives you an overview of the signal flow in the echoAM function

