**Classwork 6**

**Assignment 1:**

Define a function **GreenSqr(img, c, w)** which receives an image (**img**), a center location (**c**) and a number (**w**).

The function should return an image which has a green scale square in the middle with an edge size of **2\*w+1.**

Aside from input confirmation and return, you are not allowed to use more than a single code line, nor you are allowed to use built in functions.

**Assignment 2:**

Define a function **QuadGreenSqr(img, s)** which receives an image (**img**) and a list (**s**).

The function should return an image with green vertical lines in the row numbers listed in the list (**s**).

Aside from input confirmation and return, you are not allowed to use more than a single code line, nor you are allowed to use built in functions to draw the lines.

**Assignment 3:**

Define a function **ColorShift(img)** which receives an image (**img**).

The function should return an image in which all the values between 75 and 125 are doubled.

Aside from input confirmation and return, you are not allowed to use more than a single code line.

**Assignment 4:**

Copy the following function:

def myColorReplacement(img, read, write, amount, th):

if img.\_\_class\_\_ != np.ndarray:

return None

myImg = img.copy()

# ------------------------------------------------------------------

for m in range(read, read + amount):

for n in range(myImg.shape[1]):

if myImg[m,n,0] < th or myImg[m,n,2] < th:

myImg[m - read + write, n, 1] = 255

# ------------------------------------------------------------------

return myImg

Replace the code between the two dashed lines with a single code line.

**Assignment 5:**

Copy the following function:

def TwoLiner(img):

if img.\_\_class\_\_ != np.ndarray:

return None

im = img.copy()

D = im.shape

th = np.int64(np.round(D[0]/3.))

wi = np.int64(np.round(D[0]/30.))

if len(D) != 3 or D[0] < 30:

return None

#---------------------------------------------------------

for m in range(th - wi, th + wi + 1):

for n in range(D[1]):

im[m, n, 0] = 150

im[m, n, 1] = 255

im[m, n, 2] = 0

for m in range(2\*th - wi, 2\*th + wi + 1):

for n in range(D[1]):

im[m, n, 0] = 150

im[m, n, 1] = 255

im[m, n, 2] = 0

#----------------------------------------------------------

return im

Replace the code between the two dashed lines with a single code line.

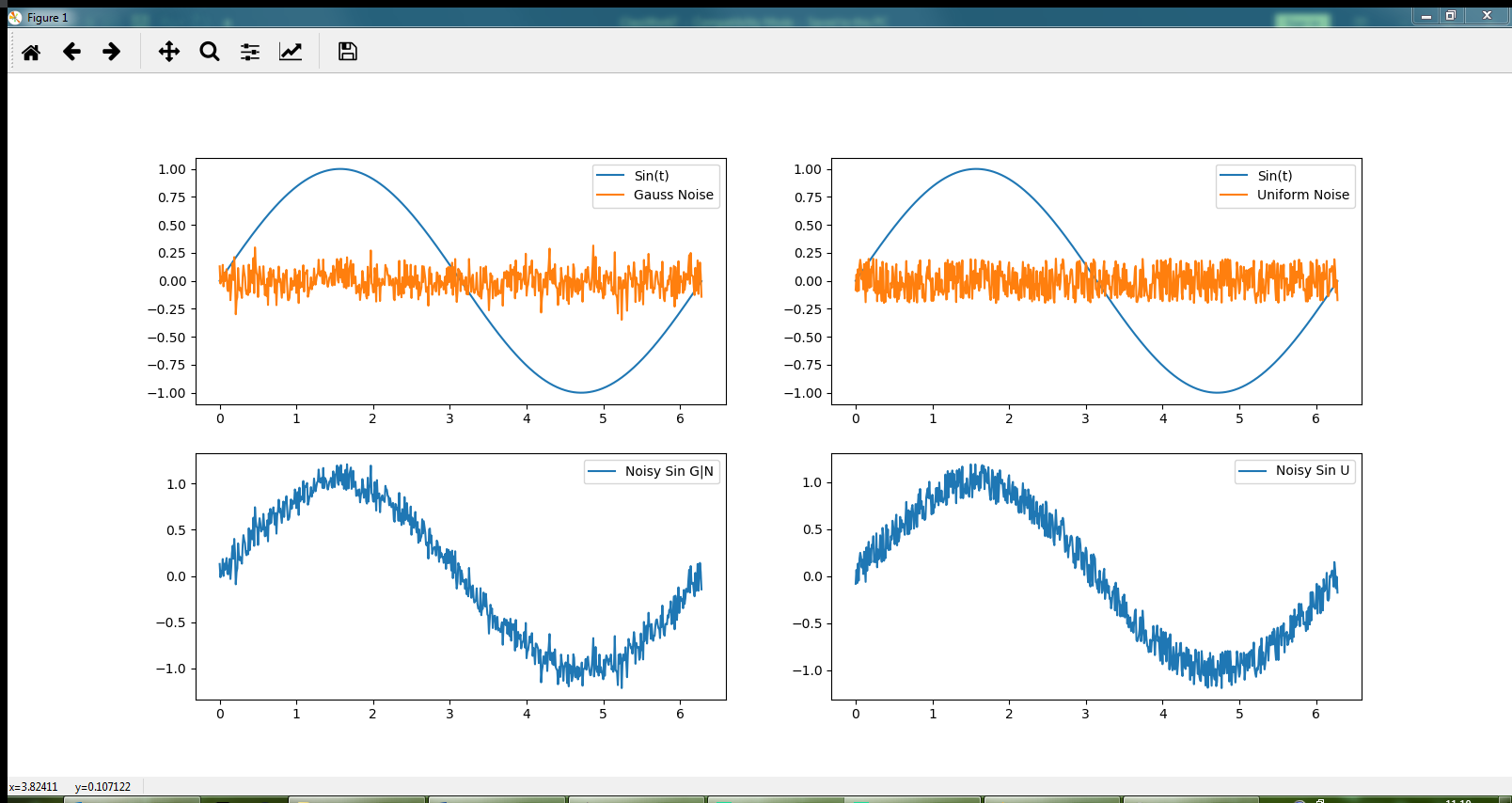
**Assignment 6:**

Define a function **NoisySin** which receives a time array and a character.

The function should return a noisy sine function, sine function and a noise array.

All of the returned signals must represent the above mentioned functions corresponding to the received time array. If the function doesn’t receive a character or receives a ‘g’ or a ‘n’, the noise should be of a Gaussian (Normal) distribution. If the character is ‘u’, the noise should be of Uniform distribution. Call examples: **NoisySin(time, Ntype)**, **NoisySin(time)**

Using this function present a graph as shown in the example:



**Assignment 7:**

Define a function **myMAF** which receives a signal and an order (default value is 1).

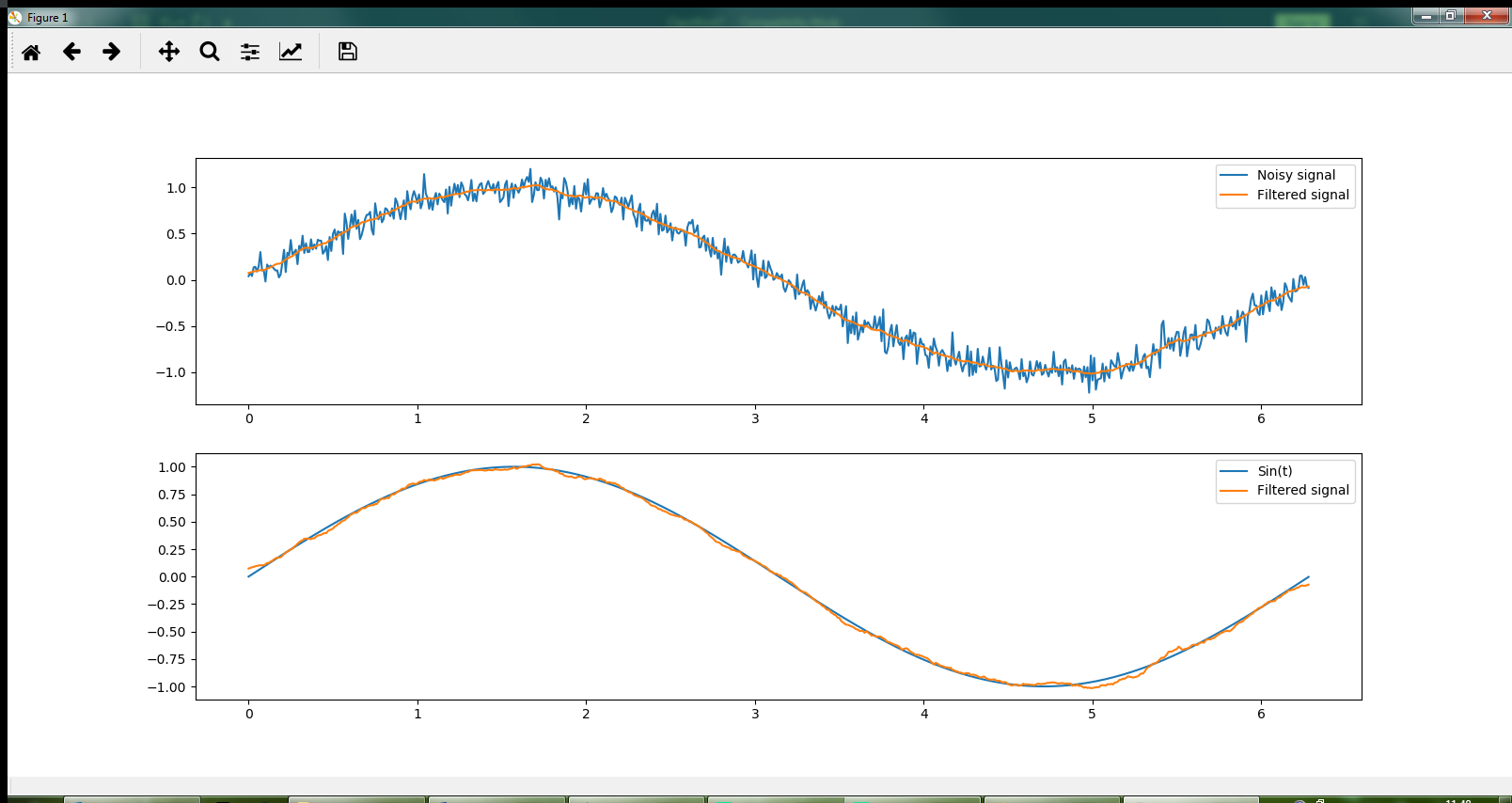
The function should return a filtered signal using moving average filter.

If the provided order is not a natural number, set it to default value.

Use Copy Padding for this assignment. Call examples: **myMAF(signal)**, **myMAF(signal, order)**

Test your function using the sine function enveloped in a gaussian noise with expectation 0 and variance 0.1. The sine function should be in range of sampled at 100 Hz.

Using this function present a graph as shown in the example:



**Assignment 8:**

Define a function **myMedFilt** which receives a signal and an order (default value is 1).

The function should return a filtered signal using moving average filter.

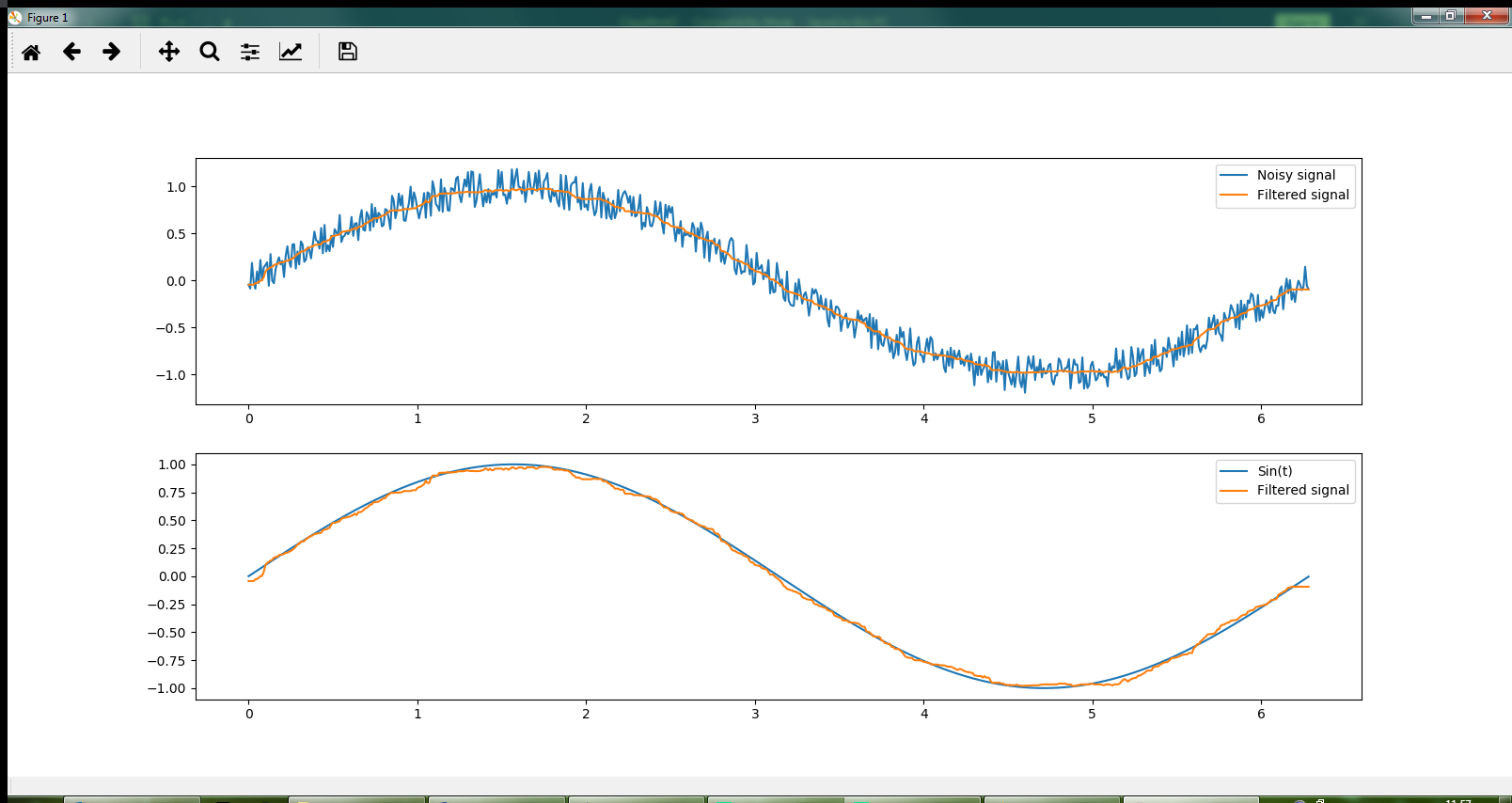
If the provided order is not a natural number, set it to default value.

Use Copy Padding for this assignment.

Call examples: **myMedFilt (signal)**, **myMedFilt (signal, order)**

Test your function using the sine function enveloped in a uniform noise in range of . The sine function should be in range of sampled at 100 Hz.

Using this function present a graph as shown in the example:



**Assignment 9:**

Define a procedure which loads two audio files, adjusts them to the same length and mixes the two sounds together while amplifying each one by a different coefficient.

Download the files from moodle (rain.wav, Walking.wav). One of the files is stereo and the other one is mono channel. Both files are in 16-bit representation and have sample rate of 48 kHz. Add the mono channel sound to each of the stereo channels sound outputs.

Save the resulted Audio file as a stereo channel file and present the graph for each of the records as mono:

