Ouermi timbwaoga Aime Judicael

course: Phys 410

Assignment I Report

Introduction

The purpose of this lab was and Introduction to data manipulation using libraries and different tools available. We were suppose to use some raw data and different techniques to output and Image.

Approach

My initial plan was to developpe a module for each of the section of the assignment. I wanted to do so by using the basic library of Python. Because I wanted to use this assignment as an opportunity to also learn python. I then started building classes and function to process the data and convert into image. While in the process I realised that given the timeframe I have there will not be enough time for to design my framework and the assignment done.

After such realisation I decided to use libraries to do the assignment.
I collaborated with some classmates and did some research on what would be the best way to complete the assignment. My approach was still based on modular implementation and using different libraries and tools.

Result

1.

I was quite easy to produce a 2D array, but a bit challenging to to produce a 2D that will be used in part to produce the image. This was because I started off not using the libraries. I as soon I started using the that, I was able to construct a 2D array. I included

cvs and exel spreadsheet of the array values ranging from x = 500 to 600 and y = 500 to 600.

2.

In order to construct the image I used the constructed array in part 1 and plt.imshow() function from the library matplotlib. This give that fig1. shown bellow. As we can observe this picture is not quite smooth. In order to smooth it out we will apply different filtering techniques.

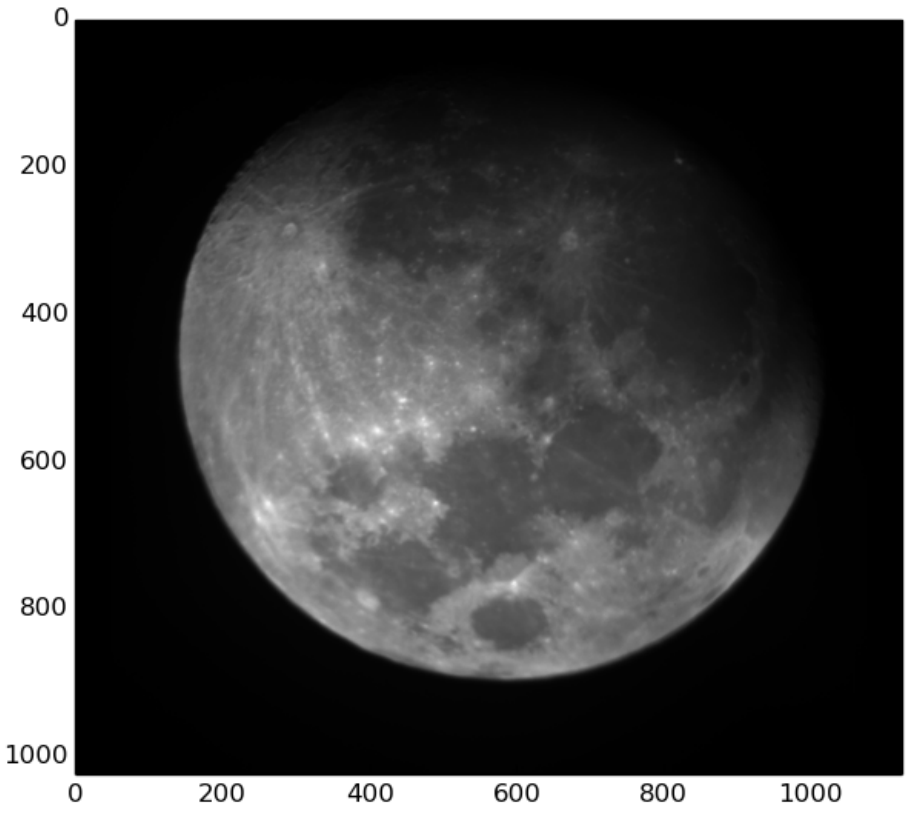

fig1 : raw data

3.

In this part we are ask to pave the center on the image produce in 1. This is for us to explore array manipulation and image manipulation. We can implement this by

identifying the indices we need to cover the region that need to pave and pave it. We obtain the image bellow.

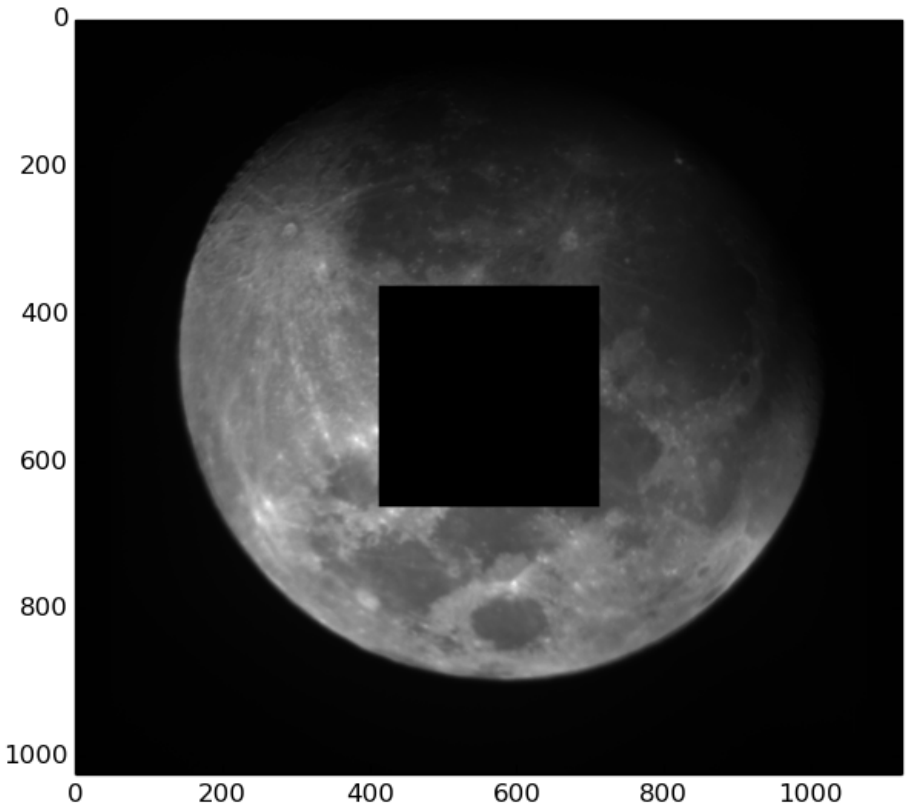

fig2

4.

In this section we are exploring how applying the a smoothing technique that takes the average change the picture and how to actually apply such smoothing techniques. It is hard to notice but this slightly improves the quality of the image.

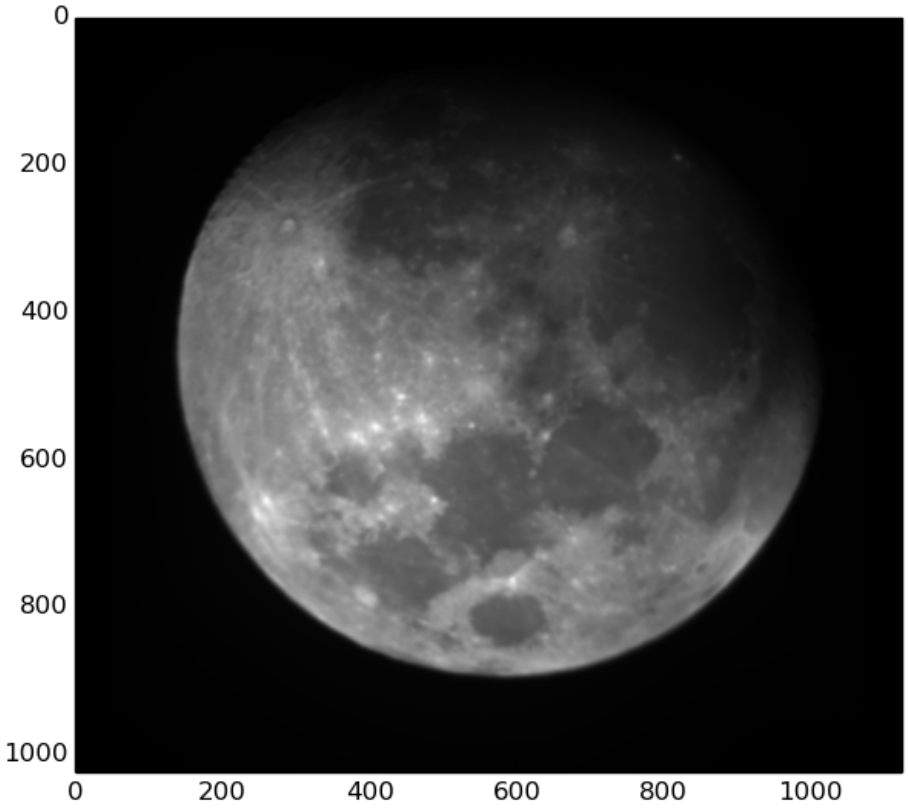

fig3: 5X5 average

5.

We are using a different techniques here that takes into acCount a weighted average. As in the previous section this used to optimize the quality of the image produced. The weighted techniques is often better because it takes into account the specific weight of each neighboring pixel when doing the calculation. Here we applied on 3 by 3.

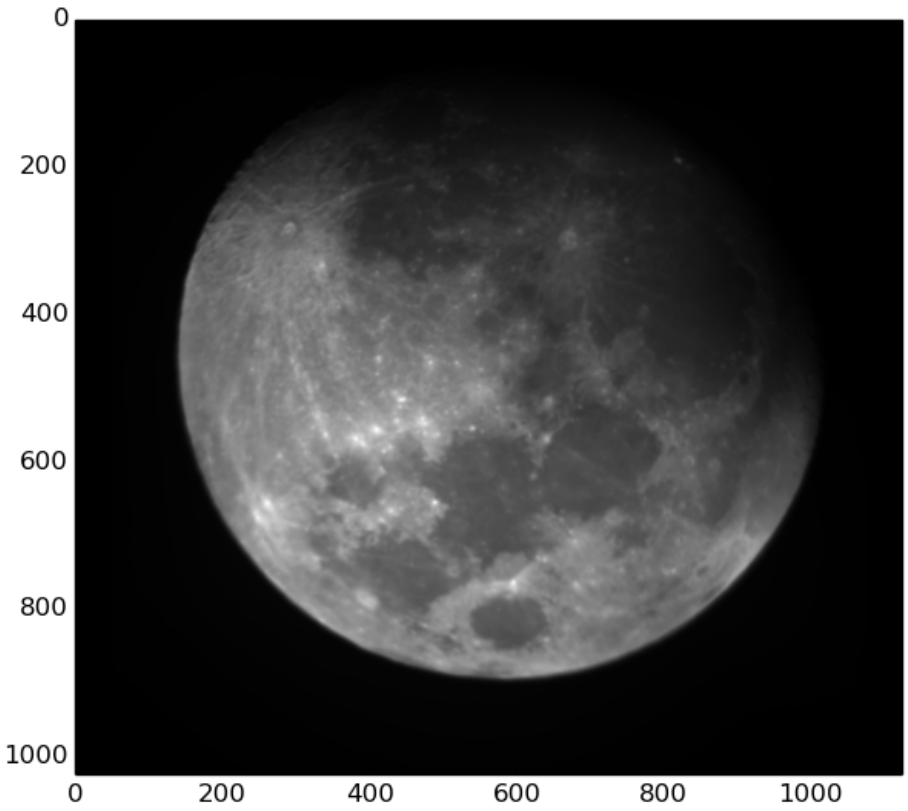

fig5.

6.

He we are applying the laplace transform (provided in lecture.) to improve the quality of our output image. the laplace transform is really good at identifying edges. Once applied can clearly identify where the edges are as u can see in fig6 below

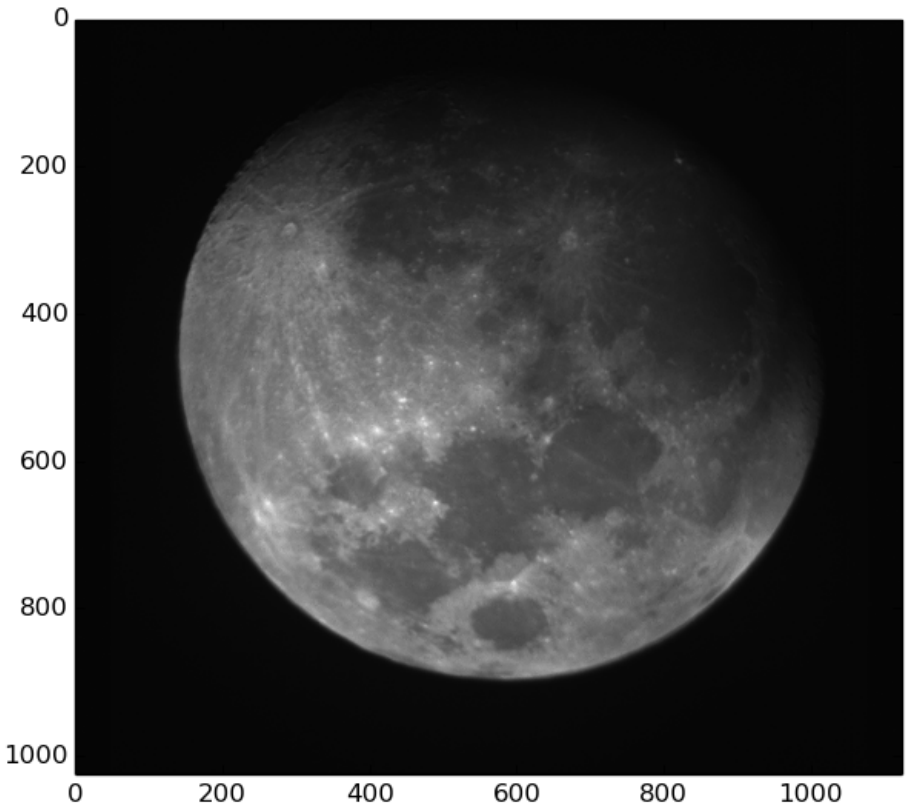

fig6.

Observation

The different libraries I used also Add built in tools that I could use to the smoothing. however they might have not produced the same result. If have more time the plan was to use those and compare them to what I have.

This assignment allow as to appreciate how important it is to find the correct tools in order to properly explore the data.

CODE:

```
import numpy as np
import matplotlib.pyplot as plt
```

```
import matplotlib.cm as cm
import csv

x,y,temp = np.loadtxt('moon.txt').T #Transposed for easier unpacking
nrows, ncols = 1024, 1124

# conversion to 2D array
grid = temp.reshape((nrows, ncols))
gridprint = grid[499:600:,499:600];
fl = open('gridprint.csv', 'w')
writer = csv.writer(fl)
writer.writerow([])
for values in gridprint:
        writer.writerow(values)
fl.close()
raw_plot = plt.imshow(grid, extent=(x.min(), x.max(), y.max(), y.min()), cmap=cm.gray)
#plt.size(300, 300)
plt.show(raw_plot)

# copy inital 2d array
paved = np.zeros((1024, 1124))
for i in range(1024):
        for j in range(1124):
   paved[i, j] = grid[i, j]

# region that will be paved
paved[362:662, 412:712] = 0
```

```
paved_plot = plt.imshow(paved, extent=(x.min(), x.max(), y.max(), y.min()),
cmap=cm.gray)
plt.show(paved_plot)

# array to hold the somthed data
smoth_array = np.zeros((1024, 1124))



# averaging funtion
def somthe_function(array):
        average = (np.sum(array)) / 25
        return average

for i in range(1024):
        for j in range(1124):
         if i < 3 or i > 1020 or j < 3 or j > 1120 :
        smoth_array[i, j] = grid[i, j]
   else:
        smoth_array[i, j] = somthe_function(grid[i:i+5, j:j+5])

average_plot = plt.imshow(smoth_array, extent=(x.min(), x.max(), y.max(), y.min()),
cmap=cm.gray)
plt.show(average_plot)



#weighted smoth array
wsmoth_array = np.zeros((1024, 1124))
weight_array = np.array([[1/16., 2/16., 1/16.], [2/16., 4/16., 2/16.], [1/16., 2/16., 1/16.]],
float)
```

```
def weighted_smothe_function(array):
        sum_weigthed_average = np.sum(array * weight_array)
        return sum_weigthed_average

# applying weighet smothing
for i in range(1024):
        for j in range(1124):
         if i < 2 or i > 1021 or j < 2 or j > 1121 :
        wsmoth_array[i, j] = grid[i, j]
    else:
        wsmoth_array[i, j] = weighted_smothe_function(grid[i:i+3, j:j+3])

weighted_plot = plt.imshow(wsmoth_array, extent=(x.min(), x.max(), y.max(), y.min()),
cmap=cm.gray)
plt.show(weighted_plot)


# laplatian weighted smothing
lap = np.array([[0, 1, 0], [1, -4, 1], [0, 1, 0]], float)
lapsmothe_array = np.zeros((1024,1124))
def lap_smothe_function(array):
        #larray = array*lap
        sumlarray = np.sum(array * lap)
        return sumlarray

for i in range(1024):
        for j in range(1124):
        if i < 2 or i > 1021 or j < 2 or j > 1121:
```

```
        lapsmothe_array[i,j]=grid[i,j]
        else:
        lapsmothe_array[i,j]=lap_smothe_function(grid[i:i+3:,j:j+3:])

lap_plot = plt.imshow(grid-lapsmothe_array, extent=(x.min(), x.max(), y.max(), y.min()),
cmap=cm.gray)

plt.show(lap_plot)
```