HW3

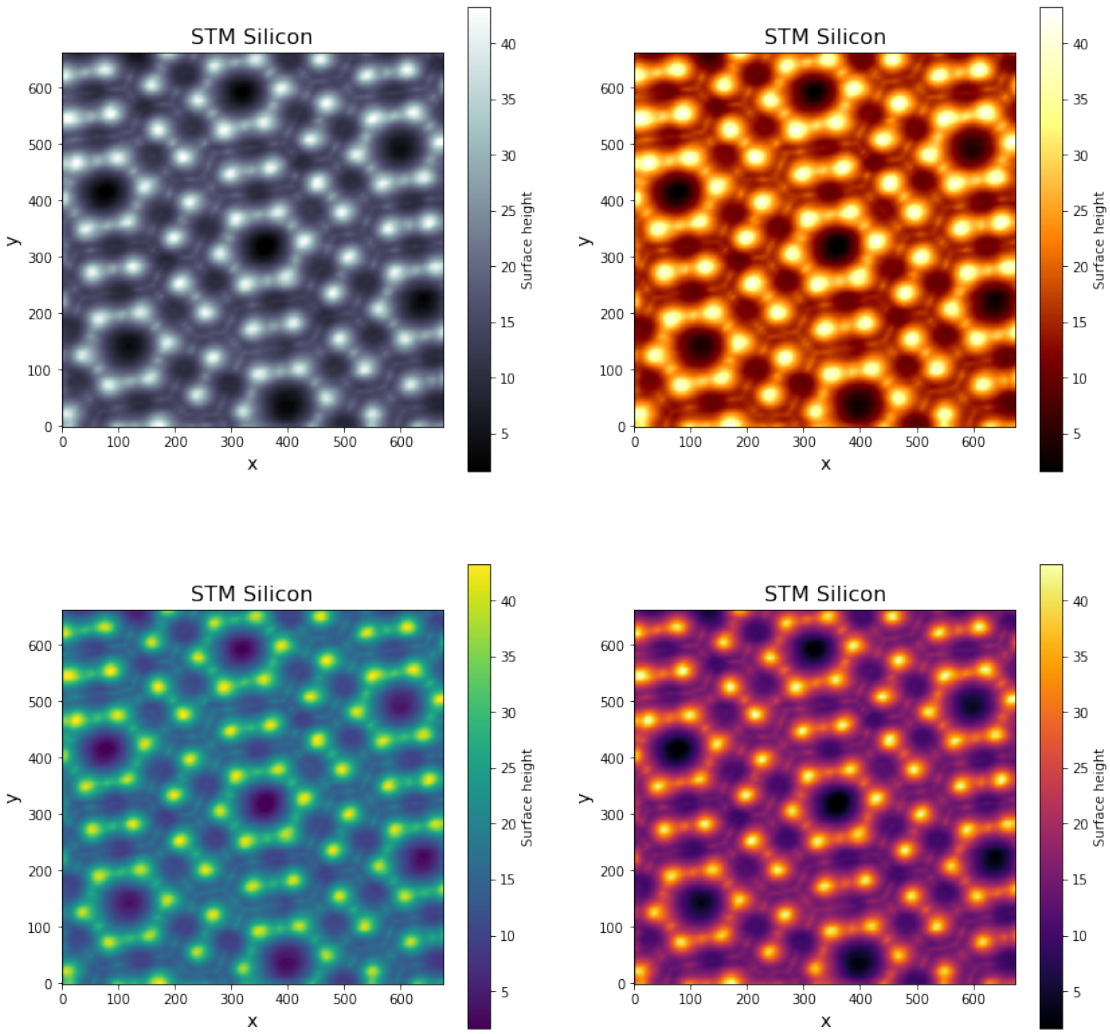
February 14, 2017

1 Homework 3

1.1 Problem 1

```
In [3]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from math import *

        #Loading data from 'sunspots.txt' file as an array.
        STMData=np.loadtxt("stm.txt")

        #Density plots
        plt.figure(figsize=(14, 14))

        #Supplot 1
        plt.subplot(221)
        plt.imshow(STMData, origin='lower', cmap='bone')
        plt.xlabel("x",fontsize = 14)
        plt.ylabel("y",fontsize = 14)
        plt.title("STM Silicon",fontsize = 16)
        plt.colorbar(label="Surface height")

        #Supplot 2
        plt.subplot(222)
        plt.imshow(STMData, origin='lower', cmap='afmhot')
        plt.xlabel("x",fontsize = 14)
        plt.ylabel("y",fontsize = 14)
        plt.title("STM Silicon",fontsize = 16)
        plt.colorbar(label="Surface height")

        #Supplot 3
        plt.subplot(223)
        plt.imshow(STMData, origin='lower', cmap='viridis')
        plt.xlabel("x",fontsize = 14)
        plt.ylabel("y",fontsize = 14)
        plt.title("STM Silicon",fontsize = 16)
        plt.colorbar(label="Surface height")
```

```
#Supplot 4
plt.subplot(224)
plt.imshow(STMData, origin='lower', cmap='inferno')
plt.xlabel("x",fontsize = 14)
plt.ylabel("y",fontsize = 14)
plt.title("STM Silicon",fontsize = 16)
plt.colorbar(label="Surface height")

plt.show()
```



1.2 Problem 2

1.2.1 Part (a)

```
In [1]: from math import *
        import numpy as np

        #Definition of a function which takes as an input an integer n
        #and outputs the nth number of the Catalan sequence.
        def CatalanSequenseFucntion(n):
            if n==0:
                return 1
            else:
                return(4.*n-2.)/(n+1.)*CatalanSequenseFucntion(n-1.)
        print("The 100th number of the Catalan sequense is: {0}".\
        format(CatalanSequenseFucntion(100)))
```

```
The 100th number of the Catalan sequense is: 8.965199470901317e+56
```

1.2.2 Part (b)

```
In [2]: from math import *
        import numpy as np

        #Definition of a function which takes as an input two non negative
        #integers n and m and outputs their greatest common divisor.
        def g(m,n):
            if n == 0:
                return m
            else:
                return g(n,m%n)
        print("The greatest common divisor of 108 and 192 is : {0}".format(g(108,192)))
```

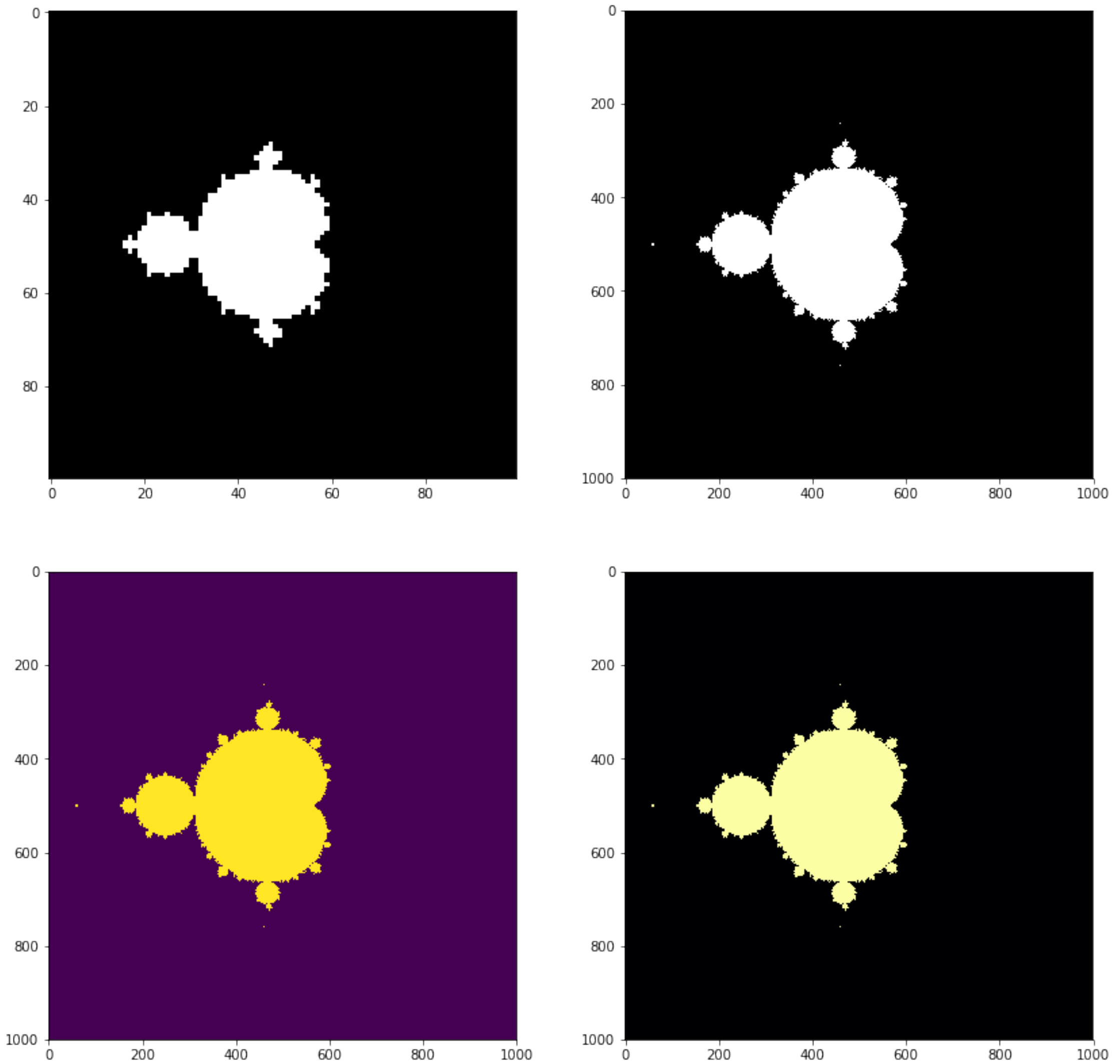
```
The greatest common divisor of 108 and 192 is : 12
```

1.3 Problem 3

1.3.1 Part (a,b)

```
In [4]: import matplotlib.pyplot as plt
        import numpy as np

        #Resolutions
        LowRes=100
        HighRes=1000

        #Definiton of a function that singles out the elements of the Mandelbrot set
        def MandelbrotFunction(c):
```

```python
    z=0
    for l in range(100):
        z=z**2+c
        if np.abs(z)>2:
            return 0
    return 1

#Mandelbrot set corresponding to low resolution.
MandelbrotSetLR=np.zeros((LowRes,LowRes))
Xlr=np.linspace(-2,2,LowRes)
Ylr=np.linspace(-2,2,LowRes)
for l in range(LowRes):
        for m in range(LowRes):
            c=complex(Ylr[m],Xlr[l])
            MandelbrotSetLR[l,m]=MandelbrotFunction(c);

#Mandelbrot set corresponding to low resolution.
MandelbrotSetHR=np.zeros((HighRes,HighRes))
Xhr=np.linspace(-2,2,HighRes)
Yhr=np.linspace(-2,2,HighRes)
for l in range(HighRes):
        for m in range(HighRes):
            c=complex(Yhr[m],Xhr[l])
            MandelbrotSetHR[l,m]=MandelbrotFunction(c);


#Density plots
plt.figure(figsize=(14, 14))

#Supplot 1
plt.subplot(221)
plt.imshow(MandelbrotSetLR, cmap='bone')

#Supplot 2
plt.subplot(222)
plt.imshow(MandelbrotSetHR, cmap='afmhot')

#Supplot 3
plt.subplot(223)
plt.imshow(MandelbrotSetHR, cmap='viridis')

#Supplot 4
plt.subplot(224)
plt.imshow(MandelbrotSetHR, cmap='inferno')

plt.show()
```


0 20 40 60 80
0 20 40 60 80
0 200 400 600 800 1000
0 200 400 600 800 1000
0 200 400 600 800 1000
0 200 400 600 800 1000
0 200 400 600 800 1000
0 200 400 600 800 1000