Homework 2

Nina Martinez Diers*
Bryn Mawr College Department of Physics
(Dated: February 16, 2024)

In this assignment, I practiced working with data to make a line graph and density plot.

1. EXERCISE 3.1: PLOTTING EXPERIMENTAL DATA

1.1. Introduction

The objective of this exercise was to practice plotting experimental data using a dataset about sunspots. This dataset contained data for the number of sunspots observed on the sun per month starting in January 1749[1].

1.2. Experiment

In order to visualize the sunspots data, I created a plot using pyplot that plotted all the sunspots data with respect to time. This was done using an algorithm that loaded the dataset into the program, then separated the columns of data into variables time in months and number of sunspots [1].

Because there is a high degree of fluctuation in the monthly number of sunspots, to get a better sense of the trend we performed a 5-month running average using the equation:

$$Y_k = \frac{1}{2r+1} \sum_{m=-r}^{r} y_{k+m} \tag{1}$$

where $r = 5$ to set a 5-month running average and y_k are the sunspot numbers.

In order to achieve this, I defined a function that calculated the average of all elements in a list according to Eq. 1. To generate the list being averaged, I added the first 5 elements of the sunspots data to a list y_k. The elements of y_k were averaged using the function, and that average was appended to the list Y_k, a list of the running average. Using a for loop over elements in the data sunspots, the element $sunspots[k]$ was appended to y_k, y_k was sliced to only include the last 5 elements in the list, and then the average of the new list y_k was appended to the running average Y_k. Below is the pseudocode used to plan the algorithm used:

Goal: calculate running average of all elements in a list
function RUNAVG(list)
  sum= 0
  for i in list **do**
    sum $+=$ list[i]
  end for
  avg = sum / (2*len(list)+1)
  return avg
  $Y_k = [0,0,0,0]$ make list of initial Y_k values. Because the running average uses previous 5 values, the running average for the first 4 months is not applicable so we just set the starting value to zero.
  for k in range(0,1000) **do**
    append list y_k with sunspots[k]
    if len(y_k) > 5 **then** $y_k = y_k$[1:6]
    end ifcalculate running average of sunspots
    if len(y_k) $== 5$ **then**
      call running average function
      append to list Y_k
    end if

The actual algorithm used evolved slightly in order to achieve what was outlined above:

r = 5 number of months
yk = [] list of sunspots to calculate average
Yk = [] list of running average
function AVG(list) calculate the average of all elements in a list
  sum = 0
  for i in range(0,r) **do**
    sum += list[i] calculate sum of all sunspots over 5 month period
  end for
  avg = sum / $(2 * len(list) + 1)$ divide sum by normalization constant to calculate average
  return avg
  Calculate Running Average of sunspots
  for k in range(0,1000) **do** only perform running average for the first 1000 months
    yk.append(sunspots[k]) append list k with sunspots[n]
    if len(yk) ¡ r **then** Because the running average uses previous 5 values, the running average for the first 4 months is not applicable so we just set the starting value to zero.
      Yk.append(0)
    end if
    if len(yk) $>$ r **then**
      yk = yk[1:r+1] if k contains 5 elements, keep only the last 5 (remove the first element)
    end if
    if len(yk) == r **then** when updated list is the correct length:
      Yk.append(Avg(yk)) Add element to the list

*Electronic address: nmartinezd@brynmawr.edu; URL: `Optionalhomepage`

with the new value for the running average

To modify the plot, the variables were modified to only include the first 1000 elements of the dataset and the running average of that data was calculated[1]. The plots of these data are shown in Fig. 1.

1.3. Results

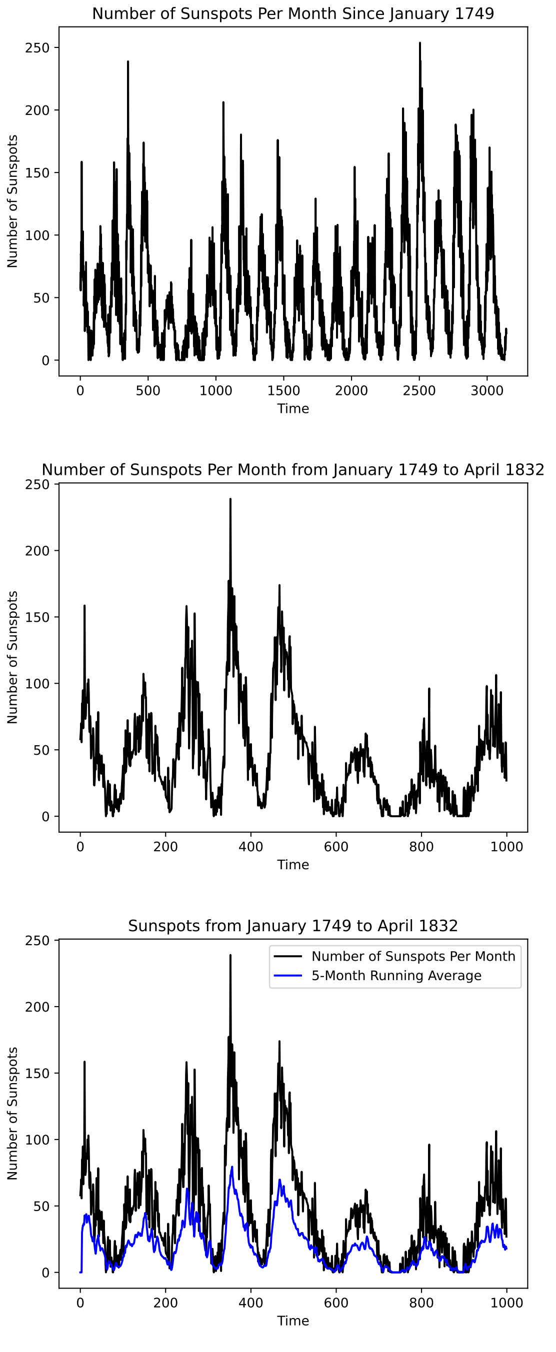

FIG. 1: Evolution of Sunspots Graphics

These data show that there are regular fluctuations in the number of sunspots observed on the sun. The maxima vary widely in altitude, and the minima hover close to zero. The average time between maxima is roughly 100 months.

2. EXERCISE 3.7: THE MANDELBROT SET

2.1. Introduction

The Mandelbrot set is a set of complex numbers discovered by Benoît Mandelbrot that repeats a pattern infinitely, also known as a fractal. A complex number $c = x + iy$ is determined to be a part of the Mandelbrot set if it always satisfies the condition:

$$|z'| < 2 \tag{2}$$

where z' is the result of iterating the equation:

$$z' = z^2 + c \tag{3}$$

Although technically one would have to iterate Eq. **??** infinitely to ensure that c is in the Mandelbrot set, we included c if Eq. 2 was satisfied over 100 iterations. The

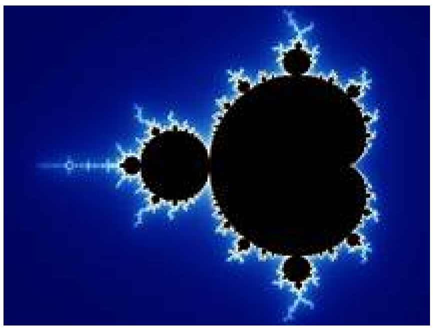

FIG. 2: A colorful depiction of the Mandelbrot set.[a]

[a]Image from Wikipedia page on the Mandelbrot Set: https://en.wikipedia.org/wiki/Mandelbrotset

shape of the Mandelbrot set is very distinct as shown in Fig. 2. The objective of this exercise was to calculate a subset of Mandelbrot numbers and graph them in a density map to visualize the pattern of the fractal.

2.2. Experiment

2.2.1. Generating the Mandelbrot Data

The pseudocode used to plan the Mandelbrot set calculation is depicted below:

- Use linspace to generate 100 evenly spaced points between -2 and 2 for x and y

- create 100x100 matrix of zeroes to populate with mandelbrot set numbers
- Use for loops to iterate over all (x,y) points. For each point, calculate c and check whether c is in the mandelbrot set =¿ if z ¿ 2, not mandelbrot; else mandelbrot
- If c is in the mandelbrot set, change (x,y) index in matrix from zero to one.
  - This will indicate that the value of c corresponding to that location in the matrix is in the mandelbrot set.
  - If c is not in the mandelbrot set, keep the zero in the matrix
  - If c is in the mandelbrot set, change the zero to a 1 in the matrix.
- This matrix shows for each xy point if the resulting c is in the mandelbrot set or not
  - It is a grid of ones that form the shape of the mandelbrot pattern
- make a density plot of the matrix data using plt.imshow(matrix)
- when that works, increase linspace to 1000 points to increase resolution

Using nested for loops to iterate over all values of x and y, a number $c = x + iy$ was generated. To check whether that number was in the Mandelbrot set, a function was called to return 0 when c was determined not to be in the Mandelbrot set or 1 when c was determined to be in the Mandelbrot Set. The output of the function was stored in the place in the matrix corresponding to the x- and y-values that generated c. This matrix formed the dataset that was plotted in the density map.

To define the function that checked whether c was in the Mandelbrot set, a while loop was used to iterate through Eq. 3. As long as $|z'|$ never surpassed 2 over the course of 100 iterations, the function returned a 1. If at any point $|z'|$ did surpass 2, the function returned a zero. The code defining the function is below:

```
def mdbCheck(c):
i = 0
z = 0
while i < 100:
    z = z**2 + c
    if abs(z) > 2:
        mdbNum = 0
        break
    i+ = 1
if abs(z) < 2:
    mdbNum = 1
return mdbNum
```

2.2.2. Creating the Density Plot

The algorithm used to create the plot is as follows:

```
import matplotlib.pyplot as plt
from matplotlib import cm
plt.imshow(mdbSet, cmap = cm.grayr, origin=”lower”)
ticks = [‘-2’, ‘-1.5’, ‘-1’, ‘-0.5’, ‘0’, ‘0.5’, ‘1’, ‘1.5’, ‘2’]
plt.xticks(np.linspace(0,1000,num=9,endpoint=True), ticks)
plt.yticks(np.linspace(0,1000,num=9,endpoint=True), ticks)
plt.xlabel(“X Values”)
plt.ylabel(“Y Values”)
plt.title(“The Mandelbrot Set”)
plt.savefig(“./MandelbrotSet.pdf”)
```

To create the density plot of the Mandelbrot data set, matplotlib.pyplot.imshow() was used. The color scheme grayr created a black and white color map. The r at the end of the color scheme name inverted the standard color gradient, which allowed the color of the Mandelbrot numbers to appear black and non-Mandelbrot numbers to show as white[2]. Additionally, because the matrix used to store data did not contain the x and y values that generated it, the locations of the tickmarks and the tick labels on the axes had to be relabeled using commands plt.xticks(location, ’labels’) and plt.yticks(location, ’labels’)[3]. The tick labels were reset to strings of half-step increments from -2 to 2, reflecting the ranges of x and y values used to generate the Mandelbrot set numbers. To define the appropriate number and location of tick marks, numpy.linspace() was used to generate 9 evenly spaced tickmarks including the endpoints of the data.

2.3. Results

The algorithm effectively recreated the Mandelbrot set pattern, as shown in Fig. 3. One problem I had was that the pdf output of the code maxed out in resolution, so even when I increased the resolution of the data, the resulting figure did not have improved resolution.

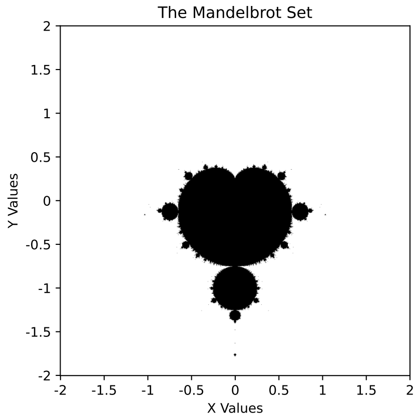

FIG. 3: Density plot of the Mandelbrot set created from x and y coordinates spanning $[-2, 2]$

[1] M. Newman, *Computational Physics* (2013), revised and expanded ed., ISBN 978-1-4801-4551-1.

[2] *Matplotlib: Colormap Reference* (????), URL https://matplotlib.org/stable/gallery/color/colormap_reference.html#reversed-colormaps.

[3] M. Wood, in *Python and Matplotlib Essentials for Physicists and Engineers* (Morgan & Claypool Publishers, 2015), IOP Concise Physics, ISBN 978-1-62705-620-5, URL doi:10.1088/978-1-6270-5620-5ch10.

Appendix A: Declaration of Collaborators

I did not collaborate with anyone on this assignment aside from consulting Dan Grin.

Appendix B: Survey Question

The homework took me at least 12 hours. The pseudocode/coding took about 4 hours each and then I spent 3 hours writing up the problem for each. I wasn't totally sure how to format the write up to include both problems, so I decided it made the most sense to have and Introduction/Experiment/Results section for each. Things I learned from the assignment were that I didn't know what sunspots were until this week, I learned a lot about running scripts and making plots locally on my computer, which I had never done before this class. I think the coding part of the homework set was just right, but I thought the write up tipped it over the edge into the "too much" category. I think part of that is that I don't feel like I knew what to include or how to structure it and that part will get easier and quicker over time.