EMAT10007 – Introduction to Computer Programming

Exercises – Week 9. Matplotlib

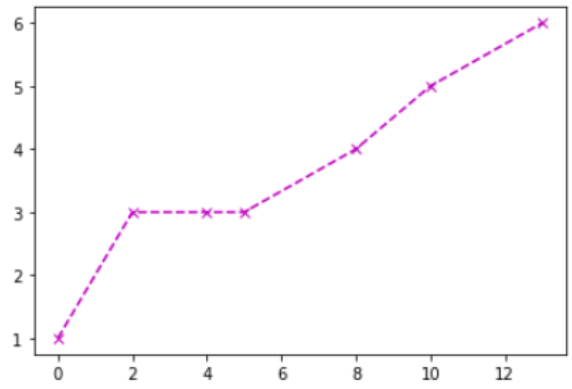
9.1 Plotting

Essential Questions

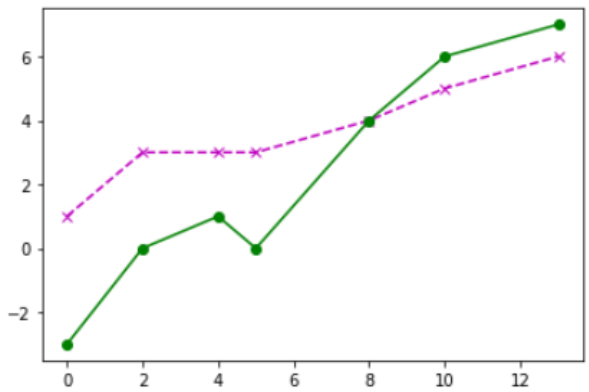
Note: A widely-used way to import the matplotlib module is by adding the following line at the start of your code: `import matplotlib.pyplot as plt`. Any function belonging to the matplotlib module can then be accessed by writing, for example, `plt.plot()`.

Exercise 1 - Line and scatter graphs

1. Create two lists of integers named `x` and `y` with the following values:
   `x = [0,2,4,5,8,10,13]`
   `y = [1,3,3,3,4,5,6]`
   Plot a line graph of `y` against `x` .
   Hint: Remember to use `plt.show()` to display the graph.
2. Modify your code so the resulting graph looks like the graph below:



3. On the same axes, plot the graph of `f` against `x` so it looks like the graph below:
   `f = [-3,0,1,0,4,6,7]`



4. Now alter your graph so it has the following
    - title: `Plot of y,f vs x`
    - x axis label: `x`
    - legend indicating which line is `y` data and which is `f`
5. Save your plot as a .pdf file
6. Plot the function x^n for the values $n = 2, 3, 4$ where the values of x, on the horizontal axis are all integers in the range 0 to 10, using a single figure (set of axes) and a separate line for each value of n.
7. Display the graphs of the function x^n for the values $n = 2, 3, 4$. This time, display the graphs for each value of n on separate subplots.

Exercise 2 - Bar charts histograms and importing data

1. Create a function named `diceRolls` that takes the number of 6-sided dice rolls, `n` as an input argument and return the results of the dice rolls.
   Hint: Use the numpy `np.random.randint()` function.
2. Modify the `diceRolls` function so it saves the results in a file diceRolls.csv.
   Hint: Remember to import any modules you need at the start of your code.
3. Call the `diceRolls` function, choosing a value for `n`. Then import the dice rolls from your diceRolls.csv file and save it as a numpy array, `data`. Print `data`.
4. Visualise the data in the `data` array. First, try using `plt.plot()`. This shows the results of each dice roll, but is slightly confusing to look at.
5. Now, let's try to represent the distribution of dice rolls using a histogram. Use `plt.hist()` and `plt.show()` to visualise the histogram.
   Hint: Try using the different optional arguments (e.g. `align`) to change the appearance of the plot (https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist.html)
6. Add a title and x and y axis labels to your histogram. Try running your code with n = 10, 100, 1,000 and 10,000 to see if/how the distribution of dice rolls changes.
7. Save your plot as a .png file
8. Import data from sample_student_data.txt and plot a bar chart of the weight of each student.
9. Import data from sample_student_data.txt and plot a histogram of the height of all students.

Advanced Questions

The file 'douglas_data.csv' contains a data set of recorded parameters for a sample of wooden beams. Open the file using a spreadsheet program or text editor and look at the data. Which rows and columns contain non-numeric data?

(A) Import the data from 'douglas_data.csv':

- using `np.loadtxt`
- using using the default data type (`float`)
- specifying the correct delimiter

- excluding the rows and columns containing non-numeric data.

Note: The use of scientific notation can be suppressed by:

```
np.set_printoptions(suppress=True)
```

(B) Select the first 10 rows of the array to create a new array.
The data in the last column is in units N/mm^2.
Convert the data in this column to units N/m^2.

(C) The area of each beam in the data set is 0.01 m^2.
The density of each beam (kg/m^3) is given in the fifth column.
The height of each beam (cm) is given in the sixth column.
Add a new column to the array that contains mass of each beam (kg) using:

mass = area $\times$ height $\times$ density

(D) Create two sub-plots:

- a scatter graph of bend strength against knot ratio
- a bar chart showing the mass of each sample

(E) Import data from sample_student_data.txt and plot a histogram of the height of all female students.

9.2 Curve Fitting

Essential Questions

Exercise 3 - Fitting polynomials

1. Fit third degree polynomial function the x,y data given.

```
x = [1, 6, 3, 4, 10, 2, 7, 8, 9, 5]
y = [2, 4, 5, 4, 13, 3, 4, 8, 12, 4]
```

Find the root mean square error for the fitted data

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\varepsilon_i^2}$$

2. Generate two numpy arrays of 20 random floating point number ranging from 1 to 10, named `x` and `y`. Sort the lists so they are both in ascending order.
Hint: Use `np.random.uniform()` and `np.sort()`.
3. Display the data from the sorted `x` and `y` functions as a scatter plot.

4. Use `polyfit` to determine the coefficients of a second degree polynomial (`deg = 2`) fit to your `x` and `y` data. Then use `poly1d` to generate a new array of fitted data, `yfit`.
5. Show the fitted data as a line graph on the same axes as the raw data.
6. Modify your code so the polynomial fit is now of the 5th degree.
7. How does this affect the fit to your data?
   Compare the root mean square error (RMSE) of the fitted data for the 2nd and 5th degree polynomial.

Exercise 4 - Fitting arbitrary functions

1. Import data in from 'signal_data.csv'.
2. Fit a function of the form $y = a\sin(x + b)$ to the data.(i.e. find constants a and b).
3. Plot the raw and fitted data on the same graph.

Advanced Questions

The file 'douglas_data.csv' contains a data set of recorded parameters for a sample of wooden beams.

(A) Import the data from 'douglas_data.csv':

- using `np.loadtxt`
- using using the default data type (`float`)
- specifying the correct delimiter
- excluding the rows and columns containing non-numeric data.

Note: The use of scientific notation can be suppressed by:

```
np.set_printoptions(suppress=True)
```

(B) Plot a scatter graph of bend strength against knot ratio.

(C) Can you think of a mathematical function that looks like it could fit this data? Define the mathematical function as a Python function and use `curve_fit` to fit a function to the data.
Hint: Remember to import `curve_fit` from `scipy.optimize`.

(D) Use the Python function you defined to generate some fitted data.

(E) Plot the fitted data as a line plot on the same axes as the raw data. Label the axes.

(F) Show the equation of the fitted line in the figure legend.

(G) What is the root mean square error (RMSE) of the fitted function?