MA305 – Classwork #10
Least Squares Fit of a Straight Line (with NumPy)

The file *'data10.txt'* (posted at your course Canvas) contains the following three data sets'

X_1	Y_1	X_2	Y_2	X_3	Y_3
10.0	8.04	10.0	9.14	10.0	7.46
8.0	6.95	8.0	8.14	8.0	6.77
13.0	7.58	13.0	8.74	13.0	12.74
9.0	8.81	9.0	8.77	9.0	7.11
11.0	8.33	11.0	9.26	11.0	7.81
14.0	9.96	14.0	8.10	14.0	8.84
6.0	7.24	6.0	6.13	6.0	6.08
4.0	4.26	4.0	3.10	4.0	5.39
12.0	10.84	12.0	9.13	12.0	8.15
7.0	4.82	7.0	7.26	7.0	6.42
5.0	5.68	5.0	4.74	5.0	5.73

1. Start a script in Python (importing *NumPy* and *Matplotlib*) and do the following:

a. Read the data file *'data10.txt'* into a matrix `M`.

```python
#!/usr/bin/env python3
import numpy as np
import matplotlib.pyplot as plt

M=np.loadtxt('data10.txt', skiprows=3)
```

b. Extract the first two columns of `M` to arrays: `x` and `y`

```python
  x=M[:,0]
  y=M[:,1]
```

c. Plot the data sets.

```python
plt.plot(x,y,'o')  # point specifications: '.', '+', 'x', 'd'
plt.show()
```

d. Use `polyfit` to fit `(x,y)` to a straight line.

```python
p1=np.polyfit(x,y,1) # 1=linear, 2=quadratic, 3=cubic etc
```

Print the value of `p1`. It should give you the slope m and the intercept c of the line of best fit $y = mx + c$.

e. Define 50 equally spaced points `xx` in the range of `x` values and using `polyval` evaluate the fit at these points. Write it to `z`.

```python
xx=np.linspace(min(x),max(x),50)
z=np.polyval(p1,xx)
```

f. Plot the points (with 'o' specification), and the line (with '-'), on the same window. Save the plot as *'figure1.png'*.

```
fig=plt.figure()
plt.plot(x,y,'o', x,z,'-')
plt.legend(['data', 'line of best fit'], loc='best')
plt.title('Least squares fit of a straight line for data set A')
plt.xlabel('$X_1$')
plt.ylabel('$Y_1$')
plt.show()
fig.savefig('figure1.png')
```

g. Repeat the process (b-f) for the remaining two data sets.

2. *Submit the Python code and the 3 figures in .pdf format for the three data sets through your course Canvas.*

Extra Credit Homework **Due: Monday, 11/20/2023**

---

3. Least Squares fitting with Gnuplot.

a. Start up `gnuplot` and read about fit (`help fit`). It looks complicated but it is, in fact, very simple.

b. Plot the data (x_i, y_i) from *'data10.txt'* with points, lines, linespoints options.
`gnuplot> plot 'data10.txt' using 1:2 with points pt 7 ps 2`
`gnuplot> plot 'data10.txt' using 3:4 with lines lw 2`
`gnuplot> plot 'data10.txt' using 5:6 with linespoints lw 2 pt 5 ps 2`

c. Fit the data (x_1, y_1) from *'data10.txt'* to a line $y = mx + c$.
`gnuplot> f(x)=m*x+c ; fit f(x) 'dat8.txt' using 1:2 via m, c`

d. Record the values of m, c, *the sum of squares of residuals*, and *the root mean square (rms) of residuals*.

e. Plot the data (x_1, y_1) and the fit $f(x)$ on the same window and save the plot as a *.png file.
`gnuplot> plot (f(x)) lw 2 title 'line of best fit', 'data10.txt' using 1:2 with points pt 7 ps 2 title 'data'`
`gnuplot> set title 'Least squares fit of a straight line for data set A'`
`gnuplot> set xlabel 'x1'; set ylabel 'y1'; replot`
Note: To save the plot as a *.png file (equivalent to *.jpg) in Gnuplot, type:
`gnuplot> set terminal png; set output 'figure1.png'`
`gnuplot> replot`
You can set the screen display back again by typing:
`gnuplot> set terminal x11`

f. Similarly, fit the data to a quadratic `g(x)=a*x**2+b*x+c` and plot them. Would you say the quadratic fits the data pretty well? Better than the straight line?

h. Try fitting the data to an exponential `h(x)=a*exp(b*x)`.
What is the error now (sum of squares of residuals)?
Does the fit look better in the plot?

i. Convert the *.png* files of the figures obtained in (e) and (f) into *.pdf* files using the Unix command `convert` and **submit the plots**.
`>> convert -quality 1000 figure1.png figure1.pdf`

---

4. Least Squares fitting in Matlab.

a. Start up Matlab and read the help pages on `dlmread`, `polyfit` and `polyval`.

b. Start up Matlab session.
`>> diary 'hw10script.txt'` (similar to Unix `script` to create 'typescript' file)

c. Read the data file 'lab10.dat' into a matrix:
`>> M = dlmread('data10.dat',' ',3)` (delimiter white space, skip 3 lines)

d. Extract the 1st column of M to a vector x, and the 2nd column to a vector y. Plot the points (with 'o' as LineSpec).
`plot(x,y,'o')`

e. Use `polyfit` to fit x, y to a straight line.
`p1=polyfit(x,y,1)` use 2 for quadratic, 3 for cubic

f. Define 50 equally spaced points in the range of x values and using `polyval` evaluate the fit at these points. Write it to z.
```
xx=linspace(min(x),max(x),50)
z=polyval(p1,xx)
```

g. Plot the points, and the line, on the same plot. Does it seem to fit well?
`plot( x, y, 'o', xx, z, '-', 'LineWidth', 2 )`

h. Use `polyfit` to fit x,y to a quadratic.

i. Plot the points and the quadratic on the same plot. Does the quadratic fit look better than the line fit?

j. Turn off the `diary`. Save the plots of linear and quadratic fits (in pdf format) and exit the Matlab session.
`>> diary off` (Your worksheet/script is saved in the file `lab10script.txt`. The default filename is `diary` if started with `diary on`.)

k. Clean up the Matlab diary `hw10script.txt`, leave only commands and save it to `hw10script.m`, and **submit the Matlab script and the plots.**