Mathematics for Sustainability (part 2)/ Python/ Plotting data

## Creating plots with MatPlotLib

MatPlotLib is a library for creating plots. There are a variety of plots available within matplotlib.pyplot, including:

- Line plot (e.g. plotting a function): plot;
- Histogram: hist;
- Scatter plot: scatter;
- Bar chart: bar;
- Pie chart: pie (though note that when representing data, a pie chart is almost never the right chart);
- 3D plot: plot surface.

## **Exercise: Plotting carbon from energy**

We'll download some data in a CSV file and plot this.

First, go to Carbon Intensity and scroll down to find 'Download The Data', click to expand it and it should look like this.

## Download The Data (hide) ^

Select a start date and end date in the boxes below in the format YYYY-MM-DD. Data is retrieved in UTC time and is returned as a CSV file. Only 30 days of data can be downloaded at a time. Data cannot be downloaded between years. Data is only available after 2017-09-26.

Start:	
End:	
`	



Enter the start date as a week ago and the end date as today, and click Download GB Data.

Save this file in the same folder as you create a new Python file.

We start by loading some modules we are going to use. We will use csv to import the data, datetime to process the dates/times and matplotlib.pyplot to plot the data.

Peter Rowlett 1

```
import matplotlib.pyplot as plt
import csv
from datetime import datetime
```

## Import CSV data

First, let's import the data from the CSV file.

We enter the filename and use file() to open the file.

```
# update to match your downloaded file
f = open("Carbon_Intensity_Data.csv", "r")
```

Now we start a csv.reader() to process the data from the file.

```
csvfile = csv.reader(f)
```

We can get the first row (headings) and store this as a list fields. If we print fields, we can see what headings there are in our file.

```
fields = next(csvfile)
print(fields)
```

You should see something like this

```
['Datetime (UTC)', 'Actual Carbon Intensity (gCO2/kWh)', 'Forecast Carbon Inter
```

This tells us that the first column (column 0) is the datetime and the second (column 1) is "Actual Carbon Intensity (gCO2/kWh)". Let's get these ready to plot.

First we start two new list in which we will store our data.

```
dates = []
actual_carbon = []
```

Now we loop through the rows of the CSV file, storing the entry in column 0 in dates and the entry in column 1 in  $actual\_carbon$ . We use a command from datetime to process the entry in column 0 into a date that matplotlib will understand.

```
for row in csvfile:
    dates.append(datetime.strptime(row[0], '%Y-%m-%dT%H:%MZ'))
    actual_carbon.append(int(row[1]))
```

Since we are finished using it, we close the file.

```
f.close()
```

Peter Rowlett 2

Now we have our data in two lists, dates and actual\_carbon, we can plot these.

```
plt.plot(dates, actual_carbon)
plt.show()
```

We label the axes using plt.xlabel for the x-axis and plt.ylabel for the y-axis. Put these commands after plt.plot() and before plt.show().

```
plt.xlabel('Date/time')
plt.ylabel('Actual Carbon')
```

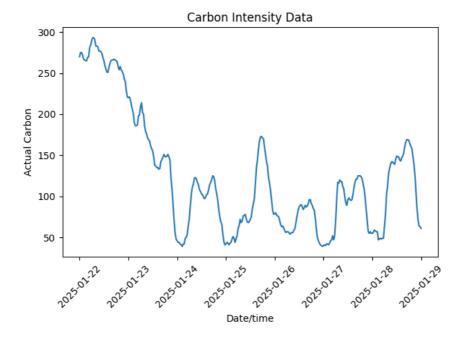
We can add a title to the plot

```
plt.title('Carbon Intensity Data')
```

Finally, we'll rotate the dates on the x axis so they don't overlap so easily, and  $plt.tight_layout()$  just helps deal with overlaps. Again, put these commands before plt.show().

```
plt.xticks(rotation=45)
plt.tight_layout()
```

When finished, you should be looking at a plot that is something like this.



Peter Rowlett 3