Introduction to Programming/ Week 5: Plotting and fitting curves
/ Curve fitting

Curve fitting

First let's import some modules we are going to use.

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit
```

We'll use matplotlib to plot some charts. We'll use numpy and curve_fit to fit a curve to some data. It is conventional to load numpy as `np` and pyplot as `plt`.

Loading data

Please download some data from data.csv and save this in the same folder as your Python file.

We can import this CSV data file using numpy.

```
data = np.genfromtxt("data.csv", delimiter=",")
```

The data file contained two columns. We'll read these into two lists `xdata` and `ydata`:

```
xdata = data[:,0]
ydata = data[:,1]
```

Let's plot these data and see what they look like.

```
plt.scatter(xdata,ydata)
plt.show()
```

Fitting an exponential curve

Hopefully you get a plot that looks a little like an $a\mathrm{e}^{bx}$ curve.

If we guess the curve is of this form, we can write a Python function to evaluate this. We will cover functions in more depth later, just for now know that this returns values of the function $f(x) = a\mathrm{e}^{bx}$. Note that we are using `exp` from numpy rather than from sympy as we have previously - this is because we are working with numeric data rather than abstract symbols.

```
def f(x, a, b):
    return a * np.exp(b * x)
```

Now we try to fit a curve using this function. What this does is to try to adjust the parameters `a` and `b` to make the curve match the data as well as possible.

```
fit = curve_fit(f, xdata, ydata)
```

We can see the values for `a` and `b` that this settled on by running

```
print(fit[0])
```

It will be useful to save these as separate values, which we do using

```
a = fit[0][0]
b = fit[0][1]
```

We can see the function that `curve_fit` has come up with

```
print(f"{a} exp({b} x)")
```

Finally, let's plot the function alongside the data.

```
plt.scatter(xdata,ydata) # plotting xdata against ydata
plt.plot(xdata, f(xdata, a, b)) # plotting f
plt.show()
```

Fitting a sin curve

Let's try a second data file. Download data2.csv.

Again we can read the data into `xdata` and `ydata` and plot these.

```
data = np.genfromtxt("data2.csv", delimiter=",")

xdata = data[:,0]
ydata = data[:,1]

plt.scatter(xdata,ydata)
plt.show()
```

This time the data looks like it might suit a sine wave. Let's try $f(x) = \sin(ax)$.

```
def f(x, a):
    return np.sin(a * x)
fit = curve_fit(f, xdata, ydata)
print(fit[0])
```

We store the value obtained as `a`:

```
a = fit[0][0]
```

Now we can print the function:

```
print(f"sin({a} x)")
```

Finally, we can plot the original data with the curve we have fitted to it.

```
plt.scatter(xdata,ydata)
plt.plot(xdata, f(xdata, a))
plt.show()
```

Exercises

1. Fit a curve to the data in data3.csv. Hint: the curve looks like $y = ae^{bx+c}$.

2. Fit a curve to the data in data4.csv. Hint: the curve looks like $y = a\sin(bx + c) + d$.