# NUMERICAL METHODS WEEK 1

# CURVE FITTING 0

## OR SCIENTIFIC COMPUTING REFRESHER

This introduces the ideas of curve fitting - this week the simplest case of fitting a line to data we expect to be linearly related.

Learning outcomes:

- Revise some material from Scientific Computing last year.
- Code a working version of linear regression using C++.
- Check your code works correctly, via an external reference.


## MATT WATKINS MWATKINS@LINCOLN.AC.UK

## WHAT IS NUMERICAL METHODS?

Using computers to solve numerical problems in applied mathematics and physics.

## WHAT IS NUMERICAL METHODS NOT?

More programming training.

## WHY AM I LEARNING THIS?

Numerical competency will be one of the major skills you can bring to the market place alongside soft and professional skills.

# PHILOSOPHY

Break down problems into small chunks.

Use pen and paper and plan your work before attacking the keyboard.

Test, test and test again.

## NO REALLY - TRY AND TEST AFTER EVERY SINGLE LINE YOU ADD.

## SAVE - ON ONEDRIVE IT WILL KEEP BACKUPS TOO.

# LEAST SQUARES REGRESSION

suppose that you think a set of paired observations $(x_0, y_0), (x_1, y_1), \ldots, (x_{n-1}, y_{n-1})$ are related as

$$y_i = a_0 + a_1 x_i + e$$

where $e$ is the error, or residual, between the model and the observations.

We think there is a linear relationship between $x$ and $y$, but there is some error in the measurements.

# BEST FIT

given our assumption of a straightline

$$y_i = a_0 + a_1 x_i + e_i$$

the error at each point is given by

$$e_i = y_i - a_0 - a_1 x_i$$

So in some sense the some of the total errors would be given by the sum of the errors. We will take the sum of the squares of the errors

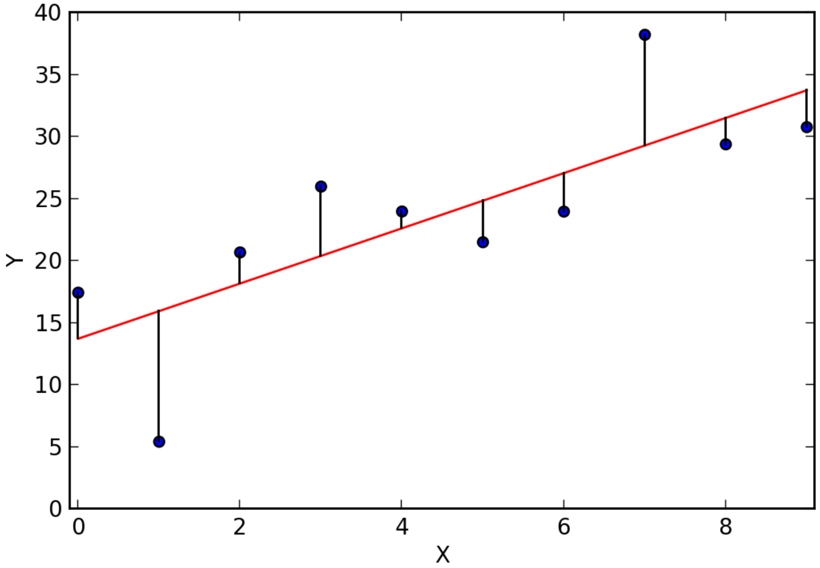
$$S_r = \sum_{i=0}^{n-1} e_i^2 = \sum_{i=0}^{n-1} (y_i - a_0 - a_1 x_i)^2$$

as our error criterion.

So in some sense the some of the total errors would be given by the sum of the errors. We will take the sum of the squares of the errors

$$S_r = \sum_{i=0}^{n-1} e_i^2 = \sum_{i=0}^{n-1} (y_i - a_0 - a_1 x_i)^2$$

as our error criterion.



# OPTIMAL PARAMETERS

If we look at our model

$$S_r = \sum_{i=0}^{n-1} e_i^2 = \sum_{i=0}^{n-1} (y_i - a_0 - a_1 x_i)^2$$

we see that there are 2 parameters, $a_0$ and $a_1$ that control the slope and intercept of our model.

It is a model, we are assuming that there is a linear relationship between $x$ and $y$.

We want to minimize the value of $S_r$, so we differentiate with respect to our parameters

$$\frac{\partial S_r}{\partial a_0} = -2 \sum (y_i - a_0 - a_1 x_i)$$

$$\frac{\partial S_r}{\partial a_1} = -2 \sum [(y_i - a_0 - a_1 x_i) x_i]$$

where the summations go from $0$ to $n - 1$ (this is to agree with C style arrays).

Note that the points $(x_i, y_i)$ are not variables, they are things we have measured. What we can vary is the parameters of our model. So $S_r$ is a function of the two parameters $a_0$ and $a_1$.

For more general models we will have more parameters and a more complex relations ship than the straight line assumed here.

# OPTIMAL PARAMETERS

We want to minimize the value of $S_r$, so we differentiate with respect to our parameters

$$\frac{\partial S_r}{\partial a_0} = -2\sum(y_i - a_0 - a_1 x_i) = 0$$

$$\frac{\partial S_r}{\partial a_1} = -2\sum[(y_i - a_0 - a_1 x_i)x_i] = 0$$

This gives us a pair of simultaneous linear equations, sometimes called the normal equations.

We can solve these for $a_1$ and $a_0$.

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2} \tag{1}$$

and plug this into the first equation to get

$$a_0 = \frac{\sum y_i}{n} - a_1 \frac{\sum x_i}{n} = \bar{y} - a_1 \bar{x} \tag{2}$$

where $\bar{y}$ and $\bar{x}$ are the means of the $x$ and $y$ values.

# CODING UP LINEAR REGRESSION

You will want to use arrays to store data. Remember arrays are like a list, or ordered set, of numbers. The type of number is defined in the normal way.

Here is some code to allocate an array of size 100, place the numbers 0 to 99, in that order, into the array.

Then we add up the elements of the array, and print them out.

```cpp
/* C++ code*/
#include <iostream>
using namespace std;

int main()
{
  double x[100] ;
  for (int i = 0; i < 100; i++) {
    x[i] = i;
  }

  double sumx = 0.0;
  for (int i = 0; i < 100; i++) {
    sumx += x[i];
  }
  cout << "The sum of the numbers 0 to 99 is " << sumx <<"\n";
}
```

highlight: c++ hljs cpp

```python
# python code
# create an empty array
x = []

for i in range(100):
  x.append(i)

sumx = 0
for i in range(len(x)): # range(n) command creates a list of values from 0 to n-1
  sumx += i
  # print(i)

print("the sum of the numbers 0 to 99 is " + str(sumx)) # str(sumx) converts sumx into a strin
g
```

highlight: python hljs

# EXERCISES

Alter the previous code to answer the following questions:

- What is $\sum_{n=0}^{99} 2n^2$
- What is $\sum_{n=1}^{100} n$
- What is $\sum_{n=2}^{200} 2n$
- What is $\sum_{n=0}^{99} 2n^2$

# EXERCISES

Find the intercept ($a_0$)and slope ($a_1$) of the least squares best fit to the following data using the formulae given a few slides previously:

```
x = [
0.526993994,
0.691126852,
0.745407955,
0.669344512,
0.518168748,
0.291558862,
0.010870453,
0.71818573,
0.897190954,
0.476789102,
]

y = [
3.477982975,
4.197925374,
4.127080815,
3.365719179,
3.387060084,
1.829099436,
0.658137249,
4.023164612,
5.074088869,
2.752890033,
]
```

# TEST YOURSELF

That is it for the lecture! The really important thing this week is that you get a computer setup so that you can try the problems as we go forward.

If you have problems getting the software to work on your laptop or desktop let us know.