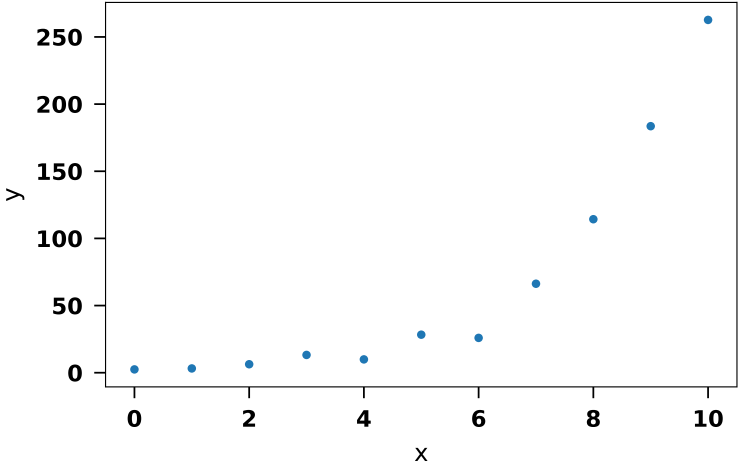
Statistical Methods with Python

Curve fitting

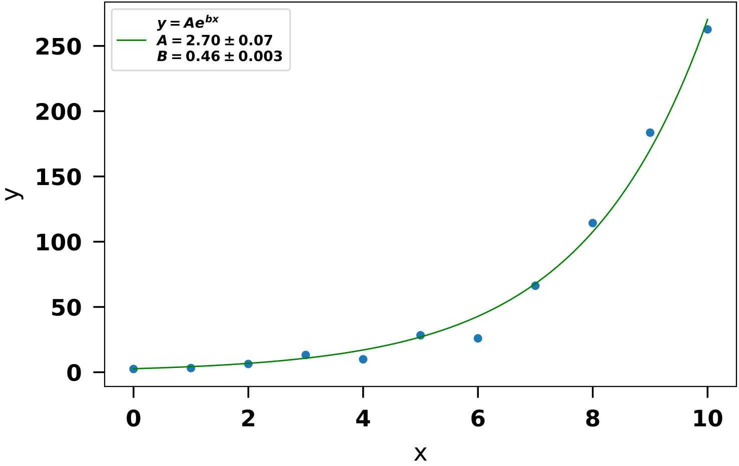
Problem Statement:

- Given some set of data points (x,y), we want to find the function $y(x)$ which most closely matches the data

We want to go from this:



To this:



Overall Procedure

Given a set of data, we must do a number of things to determine the so-called "best-fit line"

- First, we must decide what *type* of function to use
  - Does my data look like a line ($y = ax + b$)? Does it look exponential ($y = ae^{bx}$)? Parabolic ($y = ax^2 + bx + c$)?

Overall Procedure

Given a set of data, we must do a number of things to determine the so-called "best-fit line"

- First, we must decide what *type* of function to use
  - Does my data look like a line ($y = ax + b$)? Does it look exponential ($y = ae^{bx}$)? Parabolic ($y = ax^2 + bx + c$)?
- Having chosen a function, we need to find the values of the constants a, b, c ... that cause the function to most closely match the data points

Overall Procedure

Both of these items are complicated statistical problems that generally require a great deal of analysis

- Just because some function $y(x)$ “looks like a good fit” does not necessarily mean it is better than an alternative
  - The more parameters in your function, the easier it will be to get a good looking fit.
  - Comparing the “goodness of fit” of multiple functions requires an advanced understanding of statistics and probability theory; we will not get into that here

Overall Procedure

Both of these items are complicated statistical problems that generally require a great deal of analysis

- Just because some function $y(x)$ “looks like a good fit” does not necessarily mean it is better than an alternative
  - The more parameters in your function, the easier it will be to get a good looking fit.
  - Comparing the “goodness of fit” of multiple functions requires an advanced understanding of statistics and probability theory; we will not get into that here
- Bottom line: just because one function provides a better fit to the data does not mean that it is the statistically preferred function. A 5 degree polynomial will usually fit better than a simple exponential, but only because it has more “degrees of freedom”

Overall Procedure

Let's assume that you know which function you want to try and fit, and focus on how to actually find the best-fit values.

Overall Procedure

Let's assume that you know which function you want to try and fit, and focus on how to actually find the best-fit values.

- This, too is a complicated question with many possible solutions depending on your needs

Overall Procedure

Let's assume that you know which function you want to try and fit, and focus on how to actually find the best-fit values.

- This, too is a complicated question with many possible solutions depending on your needs
- I will introduce you to a popular algorithm and show you how to implement it in Python, just keep in mind there are many complementary ways of approaching this problem

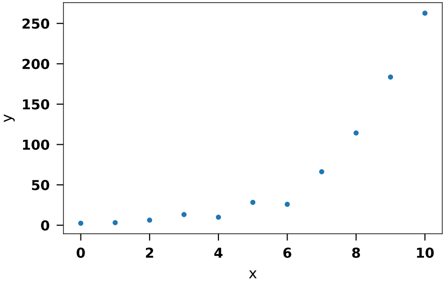
Revised Problem Statement

We have a set of data points X and Y:

$$X = x_1, x_2, x_3, \cdots, x_i, \cdots, x_n$$
$$Y = y_1, y_2, y_3, \cdots, y_i, \cdots, y_n$$

We want to fit a function $f(x|a, b, c, \cdots)$ to these data points (By this I mean a function $f(x)$ which depends on the parameters $a, b, c...$)

Example



$$X = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$$
$$Y = 2.47, 3.18, 6.31, 13.24, 9.92, 28.32, 25.93, 66.25, 114.29, 183.55, 262.68$$

$$f(x|a, b, c, \cdots) = f(x|a, b) = ae^{bx}$$

Example

$$X = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$$
$$Y = 2.47, 3.18, 6.31, 13.24, 9.92, 28.32, 25.93, 66.25, 114.29, 183.55, 262.68$$

$$f(x|a, b, c, \cdots) = f(x|a, b) = ae^{bx}$$

We want to choose the values of a and b that most closely match the data X and Y

Finding the best fit curve

How do we define whether or not a line “closely matches the data”?

- A *perfect* fit would be one where every single data point lies exactly on top of the curve (the distance between each point and the curve is 0)

Finding the best fit curve

How do we define whether or not a line “closely matches the data”?

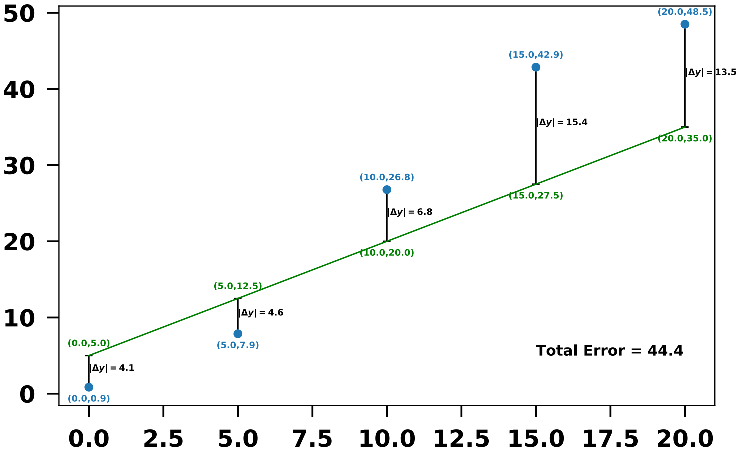
- A *perfect* fit would be one where every single data point lies exactly on top of the curve (the distance between each point and the curve is 0)
- A curve which is farther away from the data points is a worse fit

Finding the best fit curve

How do we define whether or not a line “closely matches the data”?

- A *perfect* fit would be one where every single data point lies exactly on top of the curve (the distance between each point and the curve is 0)
- A curve which is farther away from the data points is a worse fit
- One easy way to determine how “good” the fit is: just sum the distance between points and line for every point. The smaller this number is, the better the overall fit.

Fit Residuals



Fit Residuals

Each point has some error (Δy) which we usually refer to as the *residual*.

The *total* residual S is just the sum of the absolute value of each individual residual:

$$S = |\Delta y_1| + |\Delta y_2| + \cdots + |\Delta y_n| = \sum_{i=1}^{n} |\Delta y_i|$$

Fit Residuals

Each point has some error (Δy) which we usually refer to as the *residual*.
The *total* residual S is just the sum of the absolute value of each individual residual:

$$S = |\Delta y_1| + |\Delta y_2| + \cdots + |\Delta y_n| = \sum_{i=1}^{n} |\Delta y_i|$$

- In practice, it's easier to sum the *square* of the residuals, rather than their absolute values:

$$\mathbf{S \equiv \Delta y_1^2 + \Delta y_2^2 + \cdots + \Delta y_n^2 = \sum_{i=1}^{n} \Delta y_i^2}$$

Fit Residuals

$$S \equiv \Delta y_1^2 + \Delta y_2^2 + \cdots + \Delta y_n^2 = \sum_{i=1}^{n} \Delta y_i^2$$

Interpretation of S: the smaller the sum S is, the closer the data points are to the curve

The **optimal** function parameters a,b,c,... are those which *minimize S*

- click here for visualization

We can write a program to minimize S, or (in some cases) use calculus to minimize it by hand

The Method of Least Squares

This method for finding the optimal parameters for the fitted function is known as the method of least squares, since it minimizes the sum of the square of the residuals

Procedure

We have a set of data points X and Y:

$$X = x_1, x_2, x_3, \cdots, x_i, \cdots, x_n$$
$$Y = y_1, y_2, y_3, \cdots, y_i, \cdots, y_n$$

Function is $f(x|a, b, c, \cdots)$

Procedure

We have a set of data points X and Y:

$$X = x_1, x_2, x_3, \cdots, x_i, \cdots, x_n$$
$$Y = y_1, y_2, y_3, \cdots, y_i, \cdots, y_n$$

Function is $f(x|a, b, c, \cdots)$

$$S = \sum_{i=1}^{n} (Y_i - f(X_i))^2$$

Procedure

We have a set of data points X and Y:

$$X = x_1, x_2, x_3, \cdots, x_i, \cdots, x_n$$
$$Y = y_1, y_2, y_3, \cdots, y_i, \cdots, y_n$$

Function is $f(x|a, b, c, \cdots)$

$$S = \sum_{i=1}^{n} (Y_i - f(X_i))^2$$

The optimal set of parameters $a, b, c, \cdots$ are given by:

$$\frac{\partial S}{\partial a} = 0$$
$$\frac{\partial S}{\partial b} = 0$$
$$\frac{\partial S}{\partial c} = 0$$
$$\cdots$$

Special Example: Linear best fit

We have a set of data points X and Y:

$$X = x_1, x_2, x_3, \cdots, x_i, \cdots, x_n$$
$$Y = y_1, y_2, y_3, \cdots, y_i, \cdots, y_n$$

$$f(x|a, b) = ax + b$$

$$S = ((aX_1 + b) - Y_1)^2 + ((aX_2 + b) - Y_2)^2 + \cdots$$
$$S = \sum_{i=1}^{n} ((aX_i + b) - Y_i)^2$$

Special Example: Linear best fit

$$S = \sum_{i=1}^{n} ((aX_i + b) - Y_i)^2$$

$$\frac{\partial S}{\partial a} = 0 \implies \sum_{i=1}^{n} 2X_i \left((aX_i + b) - Y_i\right) = \sum_{i=1}^{n} 2 \left((aX_i^2 + bX_i) - X_i Y_i\right) = 0$$

$$\frac{\partial S}{\partial b} = 0 \implies \sum_{i=1}^{n} ((aX_i + b) - Y_i) = nb + a \sum_{i=1}^{n} X_i - \sum_{i=1}^{n} Y_i = 0$$

$$\left(nb + a \sum_{i=1}^{n} X_i - \sum_{i=1}^{n} Y_i = 0\right) \div n \to b + a\bar{X} - \bar{Y} = 0$$

$\bar{X}$ and $\bar{Y}$ are the *average values* of X and Y:

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

$$\bar{Y} = \frac{1}{n} \sum_{i=1}^{n} Y_i$$

Special Example: Linear best fit

Now we have two equations, and two unknowns (a and b)

$$\sum_{i=1}^{n}(aX_i^2 + bX_i) - X_i Y_i = 0 \tag{1}$$

$$b + a\bar{X} - \bar{Y} = 0 \tag{2}$$

Special Example: Linear best fit

I'll skip over the rest of the algebra to the final result:

$$a = \frac{\sum_{i=1}^{n} \left(X_i - \bar{X}\right) \left(Y_i - \bar{Y}\right)}{\sum_{i=1}^{n} \left(X_i - \bar{X}\right)^2}$$

$$b = \bar{Y} - a\bar{X}$$

The General Case

In general, it is difficult/impossible to minimize $S = \sum_{i=1}^{n} (Y_i - f(X_i))^2$ analytically

We instead need to find a numerical approximation for the minimum

The General Case

If the function $f = f(x|a, b, c, \cdots)$, the the sum S

$$S = S(a, b, c, \cdots) = \sum_{i=1}^{n} (Y_i - f(X_i|a, b, c, \cdots))^2$$

Is a function of the parameters $a, b, c, \cdots$

The General Case

If the function $f = f(x|a, b, c, \cdots)$, the the sum S

$$S = S(a, b, c, \cdots) = \sum_{i=1}^{n} (Y_i - f(X_i|a, b, c, \cdots))^2$$

Is a function of the parameters $a, b, c, \cdots$
All that remains is to find the parameters $a, b, c, \cdots$ for which $S(a, b, c, \cdots)$ is minimal

Minimizing S

In principle, this is just a matter of looping over all possible values of $a, b, c, \cdots$:

```
for $a$ in range($a_i$,$a_f$,$\Delta a$):
    for $b$ in range($b_i$,$b_f$,$\Delta b$):
        for $c$ in range($c_i$,$c_f$,$\Delta c$):
            $\vdots$
                $\cdots$  $s = S(a, b, c, \ldots)$
```

Minimizing S

In principle, this is just a matter of looping over all possible values of $a, b, c, \cdots$:
for a in range(a_i,a_f,Δa):
This is rarely a practical solution: the total number of iterations is:

$$N = \left(\frac{a_f - a_i}{\Delta a}\right)\left(\frac{b_f - b_i}{\Delta b}\right)\left(\frac{c_f - c_i}{\Delta c}\right)\ldots$$

So, even for a conservative precision $\Delta a = \Delta b = \Delta c = \ldots \approx 0.01$, you need:

$$N \sim \left(\frac{1}{0.01}\right)^{(\text{number of parameters})}$$

i.e.: if your function has 3 parameters, you need
$N \sim (1/0.01)^3 = 1,000,000$ iterations to find a, b, c

Minimizing S

We have now traded complicated problem for another.

Minimizing S

We have now traded complicated problem for another.

- We know the parameters of our best fit function will be those that minimize the squared residual function $S = S(a, b, c, \cdots) = \sum_{i=1}^{n} (Y_i - f(X_i|a, b, c, \cdots))^2$

Minimizing S

We have now traded complicated problem for another.

- We know the parameters of our best fit function will be those that minimize the squared residual function $S = S(a, b, c, \cdots) = \sum_{i=1}^{n} (Y_i - f(X_i | a, b, c, \cdots))^2$
- Now, how do we minimize this function?

Minimizing S

We have now traded complicated problem for another.

Minimizing S

We have now traded complicated problem for another.

- We know the parameters of our best fit function will be those that minimize the squared residual function $S = S(a, b, c, \cdots) = \sum_{i=1}^{n} \left(Y_i - f(X_i | a, b, c, \cdots) \right)^2$

Minimizing S

We have now traded complicated problem for another.

- We know the parameters of our best fit function will be those that minimize the squared residual function $S = S(a, b, c, \cdots) = \sum_{i=1}^{n} (Y_i - f(X_i | a, b, c, \cdots))^2$
- Now, how do we minimize this function (in a reasonable amount of time)?

Minimizing S

We have now traded complicated problem for another.

- We know the parameters of our best fit function will be those that minimize the squared residual function $S = S(a, b, c, \cdots) = \sum_{i=1}^{n} (Y_i - f(X_i | a, b, c, \cdots))^2$
- Now, how do we minimize this function (in a reasonable amount of time)?
- **This is a question for another class**

Minimizing S

I will not go into any depth on minimization algorithms. Many popular methods use combinations of on-the-fly derivative calculation to move toward the minimum in as few steps as possible.
The goal is to get as close to the true minimum as possible with the smallest number of iterations (ideally much less than $(1/\Delta a)^n$)

Numerical Minimization with Python

Many of these algorithms are implemented in the popular scipy (Scientific Python) package