

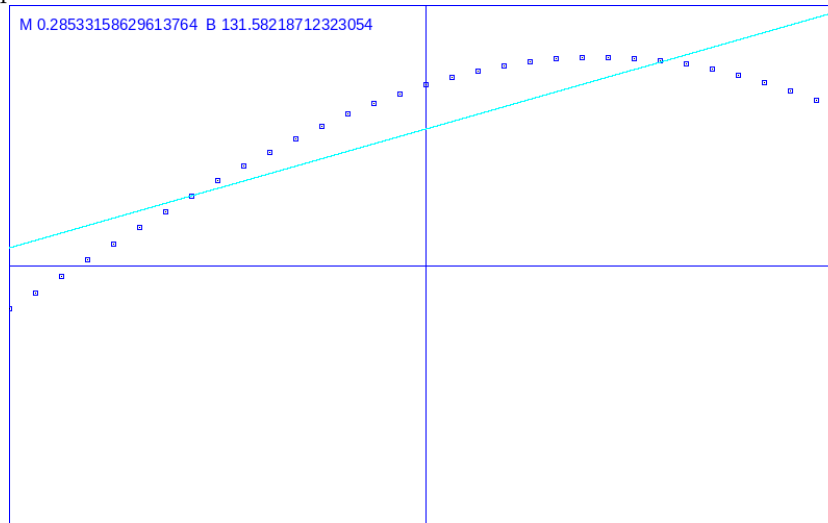
Linear Least Squares Line Fitting Calculus

Sean Brennan
www.zettix.com
github.com/zettix/LeastSquares

2025 March

1 Introduction

This is a description of fitting a line to a collection of two dimensional points (x, y) by minimizing the error defined as the square of the differences between the line $y = mx + b$ and each y_i for every x_i in the point set, when added together. Below is an example of a solution of the best fitting line given the set of points below.



To find the line, we first define our error function. Described below, it is sometimes called the sum of residuals. The important idea is that it is a function, and thus has parameters and a derivative.

To find the minimum, we take the derivative, or gradient, and set it to zero. This is possible because the error function is similar to the function $z = (x - y)^2$, which is convex and increases as x and y differ, thus the minimum value has gradient zero.

2 The error function

The error function is defined as the differences between the line's y values and the y values of the data set squared for each pair.

$$\sum_{i=1}^n (mx_i + b - y_i)^2 \quad (1)$$

We can see this as a function f taking two variables, m and b :

$$f(m, b) = \sum_{i=1}^n (mx_i + b - y_i)^2 \quad (2)$$

3 Finding the minimum: Simplify

The current definition of $f(m, b)$ would require the chain rule for a derivative, in a sum, things we can avoid by trying to isolate our terms by carrying out the square, isolating the sums from our parameters \mathbf{m} and \mathbf{b} and separating constants from our variables.

Dropping the sum for the moment, we expand:

$$\begin{aligned} f(m, b) &= (mx_i + b - y_i)^2 \\ &= (mx_i + b - y_i)(mx_i + b - y_i) \\ &= mx_i(mx_i + b - y_i) + b(mx_i + b - y_i) - y_i(mx_i + b - y_i) \\ &= mx_i mx_i + mx_i b - mx_i y_i + bmx_i + b^2 - by_i - y_i mx_i - y_i b + y_i^2 \\ &= m^2 x_i^2 + mbx_i - mx_i y_i + mbx_i + b^2 - by_i - my_i x_i - by_i + y_i^2 \\ &= m^2 x_i^2 + 2mbx_i - 2mx_i y_i - 2by_i + b^2 + y_i^2 \end{aligned} \quad (3)$$

If we add the sum back we get:

$$f(m, b) = \sum_{i=1}^n (m^2 x_i^2 + 2mbx_i - 2mx_i y_i - 2by_i + b^2 + y_i^2) \quad (4)$$

Expanding the sums, carefully:

$$f(m, b) = \sum_{i=1}^n m^2 x_i^2 + \sum_{i=1}^n 2mbx_i - \sum_{i=1}^n 2mx_i y_i - \sum_{i=1}^n 2by_i + \sum_{i=1}^n b^2 + \sum_{i=1}^n y_i^2 \quad (5)$$

Then factoring and simplifying:

$$f(m, b) = m^2 \sum_{i=1}^n x_i^2 + 2mb \sum_{i=1}^n x_i - 2m \sum_{i=1}^n x_i y_i - 2b \sum_{i=1}^n y_i + nb^2 + \sum_{i=1}^n y_i^2 \quad (6)$$

4 Summed Constants

We know the sums using x_i and y_i are constant, that is, we cannot change them, we are simply using them to define our gradient. So let's rename them for our calculations, to make them much easier to work with.

$$\chi = \sum_{i=1}^n x_i \quad (7)$$

And

$$\Omega = \sum_{i=1}^n x_i^2 \quad (8)$$

With

$$\nu = \sum_{i=1}^n y_i \quad (9)$$

And finally

$$\phi = \sum_{i=1}^n x_i y_i \quad (10)$$

Equation (6) becomes much shorter:

$$f(m, b) = m^2 \Omega + 2mb\chi - 2m\phi - 2b\nu + nb^2 + \sum_{i=1}^n y_i^2 \quad (11)$$

The final sum $\sum y_i^2$ does not involve our parameters m or b so we don't simplify it, we will ignore it.

5 Finding the minimum: derivatives of $f(m, b)$

Now let's get the gradient by taking partial derivatives of (11) using our two knobs: m and b : Let us first solve for m .

$$\frac{\partial f(m, b)}{\partial m} = 2m\Omega + 2b\chi - 2\phi \quad (12)$$

Then, by b

$$\frac{\partial f(m, b)}{\partial b} = 2m\chi - 2\nu + 2nb \quad (13)$$

Setting both to zero, and dividing by two, we get:

$$\frac{\partial f(m, b)}{\partial m} = m\Omega + b\chi - \phi = 0 \quad (14)$$

And

$$\frac{\partial f(m, b)}{\partial b} = m\chi + nb - \nu = 0 \quad (15)$$

Find a common factor for \mathbf{b} by multiplying (14) by \mathbf{n} and (15) by $-\chi$

$$m\mathbf{n}\Omega + b\chi\mathbf{n} - \phi\mathbf{n} = 0 \quad (16)$$

And

$$-m\chi^2 - b\chi n + v\chi = 0 \quad (17)$$

Adding them and removing the $-b\chi n$ term.

$$\mathbf{m}(n\Omega - \chi^2) + v\chi - \phi n = 0 \quad (18)$$

Move $v\chi$ and ϕn to the other side, and divide by $n\Omega - \chi^2$ and we get the first half of our solution, m

$$\mathbf{m} = \frac{\phi n - v\chi}{n\Omega - \chi^2} \quad (19)$$

Should we divide by zero, the slope will become ∞ ! We deal with that in the next section. Should $n\Omega - \chi^2$ not be zero, we can find b with:

$$\mathbf{b}\chi = \phi - m\Omega \quad (20)$$

Finally:

$$\mathbf{b} = \frac{\phi - m\Omega}{\chi} \quad (21)$$

6 Corner cases

Notice the two pitfalls: $\chi = 0$ and $n\Omega - \chi^2 = 0$! What do these mean? $\chi = 0$ implies:

$$\sum_{i=1}^n x_i = 0 \quad (22)$$

This is not difficult. Consider the points: $(-1, 1), (0, 2), (1, 3)$. Starting at $(-1, 1)$, the line has slope 1 and y intercept of 2. Nevertheless, the sum of \mathbf{x}_i is zero i.e. χ is zero and the equation for \mathbf{b} fails.

Hence we try an alternate strategy: solve for \mathbf{b} instead of \mathbf{m} . Find a common factor for \mathbf{m} by multiplying (14) by χ and (15) by $-\Omega$

$$m\Omega\chi + b\chi^2 - \phi\chi = 0 \quad (23)$$

$$-m\chi\Omega - nb\Omega + \nu\Omega = 0 \quad (24)$$

Add them up, and we get:

$$b\chi^2 - nb\Omega + \nu\Omega - \phi\chi = 0 \quad (25)$$

Collecting \mathbf{b} and moving constants to the other side:

$$b(\chi^2 - n\Omega) = \phi\chi - \nu\Omega \quad (26)$$

We are in a contingency, where $\chi = 0$, so let us exploit this to simplify.

$$n\Omega b = \nu\Omega \quad (27)$$

Immediately:

$$b = \frac{\nu\Omega}{n\Omega} = \frac{\nu}{n} \quad (28)$$

Things get even more weird using (15) with $\chi = 0$ to find \mathbf{m} :

$$m\Omega - \phi = 0 \quad (29)$$

And hence:

$$m = \frac{\phi}{\Omega} \quad (30)$$

Now to deal with pitfall two: $n\Omega - \chi^2 = 0$, or:

$$n \left(\sum_{i=1}^n x_i^2 \right) - \left(\sum_{i=1}^n x_i \right)^2 = 0 \quad (31)$$

For example, a single point will create that problem, and a single point has no solution, nor does a set where all the x values are identical. Here is what we want to avoid:

$$n \left(\sum_{i=1}^n x_i^2 \right) = \left(\sum_{i=1}^n x_i \right)^2 \quad (32)$$

If $|\tilde{\mathbf{x}}| > 0$ then we cannot have the two sums equal due to the Cauchy-Schwarz Inequality:

$$|\langle \mathbf{u}, \mathbf{v} \rangle|^2 \leq \langle \mathbf{u}, \mathbf{u} \rangle \cdot \langle \mathbf{v}, \mathbf{v} \rangle \quad (33)$$

where \mathbf{u} is \vec{x} and \mathbf{v} is $[1, 1, 1, \dots, 1]$, and \langle, \rangle is the inner (dot) product, while the actual dot(\cdot) is a scalar multiply. Hence we see that $\langle \mathbf{v}, \mathbf{v} \rangle = n$, and the others match the above equation as well. A situation can arise where every \mathbf{x}_i is identical with different (or equal) \mathbf{y}_i , the line that fits would be vertical (or undefined if all \mathbf{y}_i are equal).

So if we get a divide by zero possibility, we simply report an error.

7 Code

A function to obtain **m** and **b** is given below, as you can see, it is quite short.

```
def FindLine(points=[]):
    """points is a 2-D array
       [[x, y], [x, y], ...]"""
    chi = 0.0
    omega = 0.0
    nu = 0.0
    phi = 0.0

    n = len(points)
    for x, y in points:
        chi += x
        omega += x * x
        nu += y
        phi += x * y

    if chi == 0.0: # special case.
        m = phi / omega
        b = nu / n
        return (m, b)

    d = n * omega - chi * chi
    if d == 0.0:
        raise Exception("Bad data!")
    m = (phi * n - nu * chi) / d
    b = (phi - m * omega) / chi
    return (m, b)
```

8 Conclusion

By using partial derivatives and some algebra, it is fairly easy to understand how to fit a line to a collection of points. This is called Gradient Descent and is used in Inverse Kinematics and Artificial Intelligence.