Notes on Linear Regression

Introduction

In these notes we will present a brief discussion of the methods and formulas for fitting a straight line to approximately linear data using a least squares fit. Much of the material is based on Taylor's *An Introduction to Error Analysis*[1]. Sample code in *Python* is presented.

Simple Averages

We begin by looking at a familiar concept, the ordinary average. We will look at it in a slightly complicated manner that will illustrate the method we use below for fitting a straight line to linear data, a *linear regression*. Intuitively, we know that an average is "in the middle" of a set of numbers. If a group of values are all supposed to represent the same thing then, in some sense, the average represents the "best" value for the group. Here we will give an explicit meaning to "in the middle" or "best".

If we have N points x_i or pairs of points (x_i, y_i), where $i = 1..N$, and functions $f(x)$ or $g(x, y)$ we can define the *average* of the functions over the points as

$$\langle f \rangle \equiv \frac{1}{N} \sum_{i=1}^{N} f(x_i) \tag{1}$$

$$\langle g \rangle \equiv \frac{1}{N} \sum_{i=1}^{N} g(x_i, y_i) \tag{2}$$

The average, $\langle f \rangle$, is a linear function of its argument:

$$\langle a f + b g \rangle = a \langle f \rangle + b \langle g \rangle \tag{3}$$

where a and b are constants. Note that the ordinary average of N points x_i is given by $\langle x \rangle$ in this notation.

An important property of the average $\langle x \rangle$ can be derived by defining the sum of the squares of the distances from the individual points to a particular point. This is called the variance, $\sigma_x^2(\bar{x})$, of a set of points, x_i, from a value $\bar{x}$ and is defined as

1 An excellent reference for this material is *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements, Second Edition* by John R. Taylor, University Science Books, Sausalito, CA. 1997. ISBN 0-935702-75-X.

$$\sigma_x^2(\bar{x}) \equiv \langle (x-\bar{x})^2 \rangle \tag{4}$$

If we expand the square and use linearity we get

$$\sigma_x^2(\bar{x}) = \langle x^2 \rangle - 2\bar{x}\langle x \rangle + \bar{x}^2 \tag{5}$$

The value of $\bar{x}$ that minimizes $\sigma_x^2(\bar{x})$ can be found by differentiating Equation 5 with respect to $\bar{x}$ and setting the result equal to zero.

$$\frac{\partial \sigma_x^2(\bar{x})}{\partial \bar{x}} = -2\langle x \rangle + 2\bar{x} = 0 \tag{6}$$

which gives the familiar result that $\bar{x} = \langle x \rangle$ and allows the simplification of Equation 5, often used in calculations, that

$$\sigma_x^2 = \langle x^2 \rangle - \langle x \rangle^2 \tag{7}$$

Note that none of these results would change if we used the "sample mean" definition, with N replaced by $N-1$, instead of the "population mean".

This calculation illustrates two important points. First, it gives a very specific meaning to the concept that the average is the middle or best value. *The average is the value that minimizes the variance, or the sum of the squared differences between the individual values and average.* In other words, the average is a *least squares* fit to the data set.

Second, it illustrates the technique of using simple differentiation to find the value that minimizes the variance. We will use this same method below to find the "middle" or "best" straight line to fit a group of roughly linear points.

Linear Regression

The problem of a linear fit to data is to find the equation of the straight line, $y = mx + b$, that is the "best" fit to a set of N pairs of data points (x_i, y_i). Following the method we used above for averages, we will look at the distance between our data points and a straight line and pick a slope and intercept that minimizes the sum of the squares of those distances. This will be a "least squares fit" or a *linear regression.*

We begin by defining the two simple averages

$$\bar{x} \equiv \langle x \rangle \quad \& \quad \bar{y} \equiv \langle y \rangle \tag{8}$$

the "variance" of the *x* and the *y* data

$$\sigma_x^2 \equiv \langle (x - \bar{x})^2 \rangle = \langle x^2 \rangle - \langle x \rangle^2 \tag{9}$$

$$\sigma_y^2 \equiv \langle (y - \bar{y})^2 \rangle = \langle y^2 \rangle - \langle y \rangle^2 \tag{10}$$

and the "covariance" (discussed in ***Quality of the Fit*** below)

$$\sigma_{xy} \equiv \langle (x - \bar{x})(y - \bar{y}) \rangle = \langle xy \rangle - \langle x \rangle \langle y \rangle \tag{11}$$

where the second forms of Equations 9-11 are derived using Equation 8. Finally, we define the standard error of the estimate (squared) as,

$$\sigma_{\ln}^2(m, b) = \langle (y - mx - b)^2 \rangle \tag{12}$$

This is the sum of the squares of the difference between the actual points y_i and those predicted by the equation $m x_i + b$. We define the "best fit" as the line, defined by *m* and *b*, that minimizes $\sigma_{\ln}^2(m, b)$. To find the values of *m* and *b* that minimize we differentiate with respect to *m* and *b* and set the results equal to zero. Taking the partial derivatives

$$\frac{\partial \sigma_{\ln}^2(m, b)}{\partial b} = -2\langle y - mx - b \rangle = 0 \tag{13}$$

$$\frac{\partial \sigma_{\ln}^2(m, b)}{\partial m} = -2\langle x(y - mx - b) \rangle = 0 \tag{14}$$

we find that

$$\langle y \rangle = m\langle x \rangle + b \tag{15}$$

$$\langle xy \rangle = m\langle x^2 \rangle + b\langle x \rangle \tag{16}$$

Using these two equations, we can solve for *m* and b as

$$m = \frac{\langle xy \rangle - \langle x \rangle \langle y \rangle}{\langle x^2 \rangle - \langle x \rangle^2} = \frac{\sigma_{xy}}{\sigma_x^2} \tag{17}$$

$$b = \bar{y} - m\,\bar{x} \tag{18}$$

With these *m* and *b,* some simple algebra shows that

$$\sigma_{\ln}^2 = \langle y^2 \rangle - m\langle xy \rangle - b\langle y \rangle \tag{19}$$

Finally, it can be shown[2] that the standard errors in the slope and intercept are given by

$$\sigma_m^2 = \frac{1}{\sigma_x^2}\frac{\sigma_{\ln}^2}{N} \tag{20}$$

$$\sigma_b^2 = \frac{\langle x^2 \rangle}{\sigma_x^2}\frac{\sigma_{\ln}^2}{N} \tag{21}$$

The application of these equations in an actual fragment of code to perform a linear repression and return the regression coefficients will be presented below in the section ***Linear Regression using Python***. However, before we present the actual code we will display another use for the covariance σ_{xy} and discuss the way to measure the quality of our fit.

Quality of the Fit

In the physical sciences it is usually self-evident when a data set of *(x,y)* points is related by a linear relation *y=mx+b*, ether from simple inspection of the data or from theoretical considerations. In this case, finding the slope and intercept of is clearly justified. In more complex situations one might try to fit a quadratic or an exponential, but the *existence* of a relation between the *x* and *y* values is not in question.

In the biological or social sciences this is often not the case. Frequently, when presented with a data set, the first question that must be addressed is whether or not there is a relation between the two variables. Does a change in one predict a change in the other? Can you compute a predicted value of *y* from an *x* value with any confidence? In statistics, one asks if the two variables are *correlated*. In order to discuss this question we will define two important statistical measures, the *covariance*[3], σ_{xy} of a data set and the closely related *correlation coefficient*, *r*.

The covariance was defined in Equation 11 as:

$$\sigma_{xy} \equiv \langle (x - \bar{x})(y - \bar{y}) \rangle = \langle x\,y \rangle - \langle x \rangle \langle y \rangle \tag{22}$$

2 See: Taylor, 1997, *op cit*, p. 188.

3 Note that some authors use σ_{xy}^2 for covariance. Following Taylor, I will not use the superscript "2" since the quantity is *not* positive definite as any true square must be.

where $\bar{x}=\langle x\rangle$ and $\bar{y}=\langle y\rangle$. The correlation coefficient, *r*, is defined as:

$$r=\frac{\sigma_{xy}}{\sqrt{(\sigma_x^2\sigma_y^2)}} \tag{23}$$

where σ_x^2 and σ_y^2 were defined in Equations 9 and 10 above. It can be shown[4] that $|\sigma_{xy}|\le\sigma_x\sigma_y$ from which it follows that $r^2\le 1$. The correlation coefficient is a normalized covariance.

The intuitive meaning of covariance and the correlation coefficient is rather easy to see. If, when *x* increases *y* tends to also increase then a value of *x* that is greater than average is likely to accompany a value of *y* that is also greater than average or *vice-versa* and $x\ge\bar{x}$ will tend to occur when $y\ge\bar{y}$ and $x\le\bar{x}$ when $y\le\bar{y}$. In ether case, the contribution to σ_{xy} will be positive and the overall average will also be positive. If an increase in *x* goes with a decrease in *y* then the contributions will tend to be negative. If there is no relation between *x* and *y* then an *x* value above the *x* average $\bar{x}=\langle x\rangle$ will be just as likely to go with a *y* value that is above $\bar{y}$ as one that is below and the average will tend towards *zero*. The extreme case of perfect correlation is data pairs which are all related by the *same* linear relation, $y_i=mx_i+b$, in which case it is easy to show that $r^2=1$. In summary, uncorrelated data will produce a correlation coefficient that approaches zero while data with a good underlying linear relation with have an r^2 near unity.

Linear Regression using Python

Here is a code fragment that illustrates the formulas above with real code using the open-source programming language Python and the SciPy library of software for mathematics, science, and engineering.

`linear_regression``(x,y)` is based on **`linregress`** from the SciPy stats module with the addition of code to return errors on the slope and intercept and removal of the code to find the two-tailed probability. The input form is also much less flexible. SciPy and its companion NumPy are Python modules for fast scientific and numerical computing using n-dimensional arrays. SciPy uses NumPy for speed on large arrays. For further information see http://www.scipy.org/ .

4 See: Taylor, 1997, *op cit*, p. 224. This is a form of the *Schwartz Inequality*.

```
def linear_regression(x,y):
    """Calculates a regression line on two arrays, x and y.
    Input:
        (x,y) a pair of NumPy arrays
    Returns:
        slope, intercept, r, stderr-of-the-estimate,
        stderr-of-the-slope, stderr-of-the-intercept
    Warning:
        This is not production code.  You should test for a zero in the
        denominator of r and set r = 0 if it occurs.  You should also set
        r=1 if r>1 due to round-off error and check that len(x)=len(y).
    """
    from scipy import mean, add, math, stats
    n = len(x)                          # the length of the array x
    xmean = mean(x)                     # the average of the array x
    ymean = mean(y)
    x_xm,y_ym = x-xmean, y-ymean        # term-by-term array subtraction
    sig2x = stats.ss(x_xm)/n            # ss(x) returns the sum of the squares
    sig2y = stats.ss(y_ym)/n
    sumx2 = stats.ss(x)
    sigxy = add.reduce(x_xm*y_ym)/n     # sum of the term-by-term array product
    r = sigxy/math.sqrt(sig2x*sig2y)
    slope = sigxy/sig2x
    intercept = ymean - slope*xmean
    err_estimate = math.sqrt((1-r*r)*sig2y)
    err_slope = err_estimate/math.sqrt(n*sig2x)
    err_intercept = err_estimate*math.sqrt(sumx2/sig2x)/n
    return (slope, intercept, r, err_estimate,
            err_slope, err_intercept)
```

Text Box 24: Code to program a linear regression

One important feature of this code is that it never subtracts two large numbers that might be nearly equal. However, it does require that you have all the data in several arrays in memory when you perform the regression.

Another Computational Method

In certain computational or programming environments it may be very useful to keep a running sum of numbers rather than storing them all and doing all the computations at the end. You also may not know in advance how many data points you will be handling. For example, certain hand-held calculators can easily be programmed to store six cumulative sums but do not handle large arrays easily. With this issue in mind, we present an alternate method of computing our regression coefficients.

First, make the following six definitions:

$$N=\sum 1 \quad , \quad \Sigma_x=\sum x_i \quad , \quad \Sigma_y=\sum y_i \tag{25}$$

$$\Sigma_{x^2}=\sum x_i^2 \quad , \quad \Sigma_{y^2}=\sum y^2 \quad , \quad \Sigma_{xy}=\sum x_i y_i \tag{26}$$

Note that each of these sums can be computed as data is collected. We also define the auxiliary quantity

$$\Delta_x=N\Sigma_{x^2}-(\Sigma_x)^2 \tag{27}$$

With these definitions, we can rewrite the results for the average and the standard deviation as:

$$\bar{x}=\frac{\Sigma_x}{N} \tag{28}$$

$$\sigma_x^2=\frac{\Delta_x}{N^2} \tag{29}$$

There are two other similar formulas for $\bar{y}$ and σ_y^2. The slope and intercept of a linear fit to the data of the form $y=mx+b$ become

$$m=(N\Sigma_{xy}-\Sigma_x\Sigma_y)/\Delta_x \tag{30}$$

$$b=(\Sigma_{x^2}\Sigma_y-\Sigma_x\Sigma_{xy})/\Delta_x \tag{31}$$

The standard deviation from the fitted line, $\sigma_{\ln}^2$, is given by

$$\sigma_{\ln}^2=\frac{1}{N}\sum_{i=1}^{N}(y_i-mx_i-b)^2=\frac{1}{N}(\Sigma_{y^2}-m\Sigma_{xy}-b\Sigma_y) \tag{32}$$

and the errors in the slope and intercept become

$$\sigma_m=\sigma_{\ln}\sqrt{N/\Delta_x} \tag{33}$$

$$\sigma_b=\sigma_{\ln}\sqrt{\Sigma_{x^2}/\Delta_x} \tag{34}$$

One final warning on implementation. In a numerical or computer environment these computations will have round-off errors. In particular, for large *N,* Equations 30 and 31 for *m* and *b* will involve taking the difference between very large numbers in both the numerator and

the denominator. This is a very hazardous activity. If the precision of the numerical calculation is not good enough, large machine errors can creep in. In this case the alternative algorithms presented in Equations 17 and 18 and used in the code fragment on page 6 should be considered.