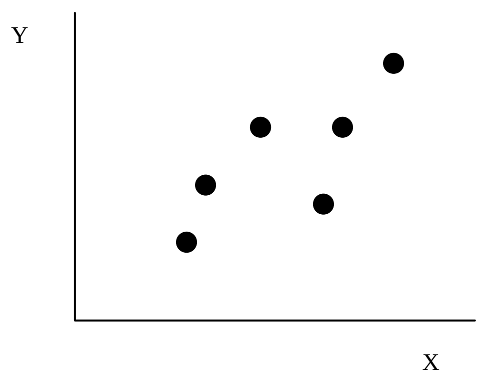
Linear Regression using Matrix Arithmetic

Shankar Manamalkav (11/2/2011)

The most common use of Regression analysis is to ascertain the causal effect of one variable upon another. In order to understand the mathematical foundations of linear regression analysis using matrices, it is helpful to first consider a simple regression: a scenario that involves one dependent variable(Y) and one independent variable(X).

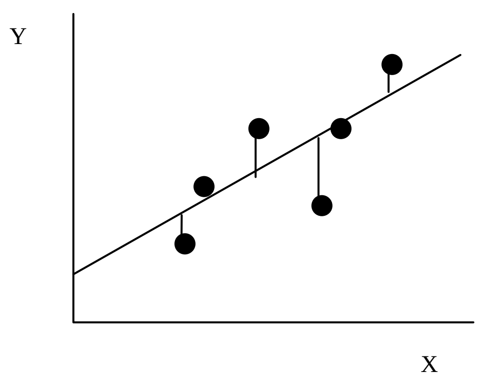
It is possible to plot the information obtained from individuals in the sample using a simple scatter diagram, where each point in the diagram represents an individual.



A generic linear regression function is expressed mathematically as:

$$y_i = ax_i + b + \varepsilon_i$$

Which is the geometric equation of a line having an Y-intercept a, and a slope of b ($\varepsilon_i$is the noise term that is usually ignored).
Evidently, there can be multiple straight lines that pass through the points on the scatter plot (of course, each having a different intercept and slope). Our problem then reduces to one of determining the “best line”. The criterion employed by regression analysis to arrive at this perfect line is to pick the one that has the “least sum of squared errors”.



i.e. line with minimum

$$\sum (data\ y - model\ y)^2$$

The least SSE (sum of squared errors) method is an attractive criterion as it is very easy to compute using matrix algebra.

Given data points $\{(x1, y1), (x2, y2), \ldots (xn, yn)\}$, and a straight line $y = ax + b$, the SSE can be written as

$$E = \sum_{i=1}^{N} (y_i - (ax_i + b))^2$$

The goal then, is to minimize this error.

Minimizing this sum with respect to ‘b’ requires that its derivative with respect to ‘b’ be zero,

$$\frac{dE}{db} = \sum_{i=1}^{n} 2\big(y_i - (ax_i + b)\big).1 = 0$$

Similarly, minimizing with respect to ‘a’ gives us

$$\frac{dE}{da} = \sum_{i=1}^{n} 2\big(y_i - (ax_i + b)\big).-x_i = 0$$

Simplifying the above equations, we get

$$\sum_{i=1}^{n} \big(y_i - (ax_i + b)\big) = 0$$

$$\sum_{i=1}^{n} \big(y_i - (ax_i + b)\big).x_i = 0$$

These may further be written as:

$$\left(\sum_{i=1}^{n} x_i\right) a + \left(\sum_{i=1}^{n} 1\right) b = \sum_{i=1}^{n} y_i$$

$$\left(\sum_{i=1}^{n} {x_i}^2\right) a + \left(\sum_{i=1}^{n} x_i\right) b = \sum_{i=1}^{n} x_i y_i$$

In matrix terms, the above equations can be re-written as:

$$\begin{bmatrix} \sum_{i=1}^{n} 1 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} {x_i}^2 \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \end{bmatrix}$$

Or

$$\begin{bmatrix} b \\ a \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} 1 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} {x_i}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \end{bmatrix} \qquad (Equation\ 1)$$

Getting back to our main regression equation, we have

$$y_i = ax_i + b + \varepsilon_i$$

So, for multiple points, i=1,2,3…n, we obtain n different equations

$$y_1 = ax_1 + b + \varepsilon_1$$
$$y_2 = ax_2 + b + \varepsilon_2$$

$$y_n = ax_n + b + \varepsilon_n$$

These can be written easily in matrix notation as follows:

$$\begin{bmatrix} y1 \\ y2 \\ . \\ yn \end{bmatrix} = \begin{bmatrix} 1 & x1 \\ 1 & x2 \\ . & . \\ 1 & xn \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix} + \begin{bmatrix} \varepsilon 1 \\ \varepsilon 2 \\ . \\ \varepsilon n \end{bmatrix}$$

OR

$$Y = X\beta + E$$

Where, Y, X, β and E are matrices.

The simplified equation for a (simple) linear regression in matrix terms is $Y = X\beta + E$

Assume that the average error (e) will equal 0. The equation becomes: $Y = X\beta$. We need to find the value of β, the coefficient matrix.

Multiply both sides of the equation by X' (transpose of X) , we get: $X'Y = X'X\beta$

In order to eliminate $(X'X)$ on the RHS, multiply both sides of the equation with $(X'X)^{-1}$, the inverse of $(X'X)$

$$(X'X)^{-1}X'Y = (X'X)^{-1}X'X\beta$$

Any matrix multiplied by its inverse is the identity matrix I, so we get

$$\boldsymbol{\beta} = (\boldsymbol{X'X})^{-1}\boldsymbol{X'Y}$$

[**Note**: the above equation contains the Moore-Penrose pseudoinverse matrix which ensures that inverses take place on SQUARE matrices. A $^{\text{pseudo-1}}$=(A'A)$^{-1}$ *A'
Recall that matrix inverses are only defined on Square matrices. We cannot guarantee that the matrix of independent variables will be a square matrix, in fact, it never is! The sample size needs to be reasonable to ensure correct results]

Now, let's look at the pieces of this new formula:

$$(X'X)^{-1} = \left(\begin{bmatrix} 1 & 1 & . & 1 \\ x1 & x2 & . & xn \end{bmatrix} \begin{bmatrix} 1 & x1 \\ 1 & x2 \\ . & . \\ 1 & xn \end{bmatrix}\right)^{-1} = \begin{bmatrix} \sum_{i=1}^{n} 1 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} {x_i}^2 \end{bmatrix}^{-1}$$

$$(X'Y) = \begin{bmatrix} 1 & 1 & . & 1 \\ x1 & x2 & . & xn \end{bmatrix} \begin{bmatrix} y1 \\ y2 \\ . \\ yn \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \end{bmatrix}$$

Plug these into the equation for

$$\boldsymbol{\beta} = (\boldsymbol{X'X})^{-1}\boldsymbol{X'Y}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \sum_{i=1}^{n} 1 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} {x_i}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i y_i \end{bmatrix} \qquad (Equation\ 2)$$

Note that Equations 1 and 2 are identical.. Meaning that the solution of our matrix equation, $(X'X)^{-1}X'Y$ does in fact give us the coefficients with the least SSE!

Now that we have the basics of simple regression established, the same discussion can be extrapolated for modeling multiple regression. Multiple regression allows additional factors to enter the analysis separately so that the effect of each can be estimated. The modified model has the equation

$$y = a + b1x1 + b2x2 + \cdots$$

Where x1, x2 ..xn are n independent variables.

Instead of a line, we are looking at planes in multidimensional space. We use the least squares criterion and locate the hyper-plane that minimizes the sum of squares of the distances from the points around the plane to the plane.

Implementation in PHP

A Matrix helper class with the following methods was programmed in PHP. Internally, the matrix is represented as an (m*n) array.

Method (Public)	**Description**
DisplayMatrix()	Formatted display of matrix for debugging
GetInnerArray()	Return the stored array
NumRows()	Number of Rows
NumColumns()	Number of Columns
GetElementAt($row, $col)	Returns the element at $row, $col
isSquareMatrix()	Returns true of square matrix
Subtract(ZF_Matrix $matrix2)	Subtract two matrices, return a new matrix with the result
Add(ZF_Matrix $matrix2)	Add two matrices, return a new matrix with the result
Multiply(ZF_Matrix $matrix2)	Multiply two matrices, return a new matrix with the result
ScalarMultiply($scalar)	Does a scalar multiplication.. Return a new matrix with the result
ScalarDivide($scalar)	Does a scalar division.. Return a new matrix with the result
GetSubMatrix($crossX, $crossY)	Get a submatrix blocking off $crossX and $crossY. This is for computing the determinant using the Expansion By Minors method (http://mathworld.wolfram.com/DeterminantExpansionbyMinors.html)
Determinant()	Return the determinant of the matrix
Transpose()	Return a new matrix with the transpose of the current matrix
Inverse()	Return the inverse using the cofactor method. http://www.mathwords.com/i/inverse_of_a_matrix.htm A new matrix is returned.

A Regression class that utilizes the matrix class above to compute the various parameters was developed. The following table summarizes the relevant parameters:

Regression Parameter	**Formula**	**Description**
b	(X’X)-1X’Y	Regression coefficients. This is an n X 1 array
SSR	b’X'Y – (1/n) (Y’UU’Y)	Sum of squares due to regression – this is a scalar. U is a unit vector of dimensions n X 1
SSE	Y’Y-b’X'Y	Sum of squares due to errors – scalar
SSTO	SSR+SSE	Total sum of squares – scalar
dfTotal	sample_size – 1	Total degrees of freedom
dfModel	num_independent_vars – 1	Model degrees of freedom
dfResidual	dfTotal – dfModel	Residual degrees of freedom
MSE	SSE/dfResidual	Mean square error – scalar
SE	(X’X)-1 *(MSE) then take Square Root of elements on the diagonal	Standard error array
t-stat	b[i][j]/SE[i][j]	t-statistic
R2	SSR/SSTO	R Square of regression
F	(SSR/dfModel)/(SSE/dfResidual)	F Value of regression