

Appendix: *Mathematica*, Excel, Regression, and Matrix Algebra

Introduction

The purpose of this appendix is to look inside the regression process so that the reader may see the computations required to produce regression output. There are a variety of different ways to present many particular results. This appendix derives results several ways to illustrate the different presentations. Readers should not view regression output as computer magic. Enhanced understanding comes with knowing the equations and the calculations that take place when one clicks on an icon or executes a regression command. (Even greater understanding comes from understanding the formal proof of these equations, something that exceeds the scope of this work).

This material draws heavily upon two sources. One is *Econometric Analysis*, 5th Ed., by William H. Greene. The other is *The Classical Regression Model*, lecture notes of Prof. Herman J. Bierens, <http://econ.la.psu.edu/~hbierens/index.htm>, of the Economics Department of the College of Liberal Arts at The Pennsylvania State University.

The example we use here elaborates the univariate case of Chapter 6 involving house prices and sizes.

The model is of the following form

$$y_i = \alpha + \beta x_i + \epsilon_i, i = 1, \dots, n$$

Where:

- y = the regressand or dependent variable to be explained
- x = the regressor or independent variable offering explanation
- α = intercept term where the function crosses the y axis (when $\beta x_i = 0$)
- β = coefficient to be estimated indicating the effect of x on y in the form of a slope
- ϵ = error or disturbance term

The model is estimated under the following assumptions

1. The model is in the correct functional form, in this case linear;
2. $E[\epsilon_i] = 0$ for all i , that is the mean of the disturbance terms is zero;
3. $\text{Var}[\epsilon_i] = \sigma^2$, a constant, for all i , the property of homoscedasticity;
4. $\text{Cov}[\epsilon_i, \epsilon_j] = 0$ if $i \neq j$, meaning the error terms are independent of each other, not correlated with each other;
5. $\text{Cov}[x_i, \epsilon_i] = 0$ for all i and j , meaning that the regressor and the disturbances are uncorrelated.

Although not strictly required, it is very common to further assume

6. $\epsilon_i \sim N[0, \sigma^2]$, meaning that the error terms are normally distributed with a zero mean and a variance of σ^2 .

Assumptions #3 and #4, if they hold, mean that the data is independently and identically distributed, referred to as "iid". The idea is that the error term is the sum of many small effects that are individually unimportant, random, both positive and negative. Assumption #2 assumes that the sum of these have a zero effect on the system.

Assumptions #4 and #5 are hard to maintain for real estate. For instance, #4, the non-autocorrelation assumption, assumes that there is no relationship between the error terms. Much financial data has evidence of autocorrelation.

Also, as valuation in real estate sales usually depend on recent sales of nearby properties correlation may be based on

proximity. There are tests to adjust results to consider autocorrelation and heteroscedasticity but these are beyond the scope of this primer.

For reasons found deep in probability theory, if data is distributed normally it is by definition independent. Thus, Assumption #6 is common. The point is that while normality is not required, independence is and when you assume normality independence comes with it. In several places in the text we question the validity of this assumption for real world data. It is mathematically convenient nonetheless.

It is our purpose here to consider the regression process in a critical way. Regression is very good for a host of purposes. The general notion of conditional probability underlying regression is a powerful concept. But like all tools it must be viewed in terms of its limitations and used correctly. To the extent assumptions cannot be maintained results will be distorted. Conclusions based on distorted results obtained under incorrect assumptions can be wrong.

A note about notation: Convention in most texts is to represent algebraic equations in standard font and matrix algebraic symbols in **BOLD** and sometimes upper case. This conflicts with the *Mathematica* convention of user defined variables being lower case and the capitalization of *Mathematica* built-in variable names and routines. To replicate the convention in most texts the *Mathematica* convention may be violated at times.

The organization of this appendix is as follows: Working in reverse (from bottom to top) we present the four components of the Excel regression output in Table 6-7. We begin with the Residual Output, then the parameter table, then the Analysis of Variance, concluding with the Summary. The reason for working in reverse is that the creation of variable names in *Mathematica* format is simplified. Where appropriate, for many results we show the algebraic equation, the matrix algebra notation equivalent and the matching output performed by *Mathematica*'s routines. This appendix is provided to only to show the calculations underlying regression output, not interpret or explain the output. For interpretation the reader is referred to Greene or any of the many fine econometric or statistics text books that cover regression.

```
In[1]:= $Post := If[MatrixQ[#1], MatrixForm[#1], #1] &
<< "MultivariateStatistics`"; << "ComputationalGeometry`"
<< "HypothesisTesting`"
```

The Data

Using our House Price example as familiar input, here is the same information we saw in Chapter 6:

```
In[4]:= ydata = {195 000, 210 000, 225 000, 240 000, 275 000, 285 000, 190 000, 239 000, 249 000, 185 000};
xdata = {1500, 1750, 1600, 1700, 1900, 1875, 1650, 1810, 1975, 1550};
houses = Transpose[{xdata, ydata}]
```

Out[6]//MatrixForm=

$$\begin{pmatrix} 1500 & 195\,000 \\ 1750 & 210\,000 \\ 1600 & 225\,000 \\ 1700 & 240\,000 \\ 1900 & 275\,000 \\ 1875 & 285\,000 \\ 1650 & 190\,000 \\ 1810 & 239\,000 \\ 1975 & 249\,000 \\ 1550 & 185\,000 \end{pmatrix}$$

We need the y values arranged as a vector named **Y**

```
In[7]:= Y = Transpose[{Transpose[houses][[2]]}]
```

```
Out[7]//MatrixForm=
```

$$\begin{pmatrix} 195\,000 \\ 210\,000 \\ 225\,000 \\ 240\,000 \\ 275\,000 \\ 285\,000 \\ 190\,000 \\ 239\,000 \\ 249\,000 \\ 185\,000 \end{pmatrix}$$

We create a matrix (**X**) composed of a vector of 1's for the intercept (alpha) and vectors for each of the independent values (x_i).

Vector length of **X** must match that of **Y**.

```
In[8]:= X = Transpose[{Table[1, {10}], Transpose[houses][[1]]}]
```

```
Out[8]//MatrixForm=
```

$$\begin{pmatrix} 1 & 1500 \\ 1 & 1750 \\ 1 & 1600 \\ 1 & 1700 \\ 1 & 1900 \\ 1 & 1875 \\ 1 & 1650 \\ 1 & 1810 \\ 1 & 1975 \\ 1 & 1550 \end{pmatrix}$$

We sometimes need the independent variables in the form of a vector that we will name "size".

```
In[9]:= size = Transpose[{Transpose[X][[2]]}]
```

```
Out[9]//MatrixForm=
```

$$\begin{pmatrix} 1500 \\ 1750 \\ 1600 \\ 1700 \\ 1900 \\ 1875 \\ 1650 \\ 1810 \\ 1975 \\ 1550 \end{pmatrix}$$

For equations developed below we need to set values for n (number of observations) and k (the width of **X**, one more than the number of independent variables). This makes **X** an $n \times k$ matrix (10 x 2 in our example).

```
In[10]:= n = Length[Y]; k = Length[Transpose[X]];
```

```
In[11]:= Print["The X matrix is ", n, " x ", k]
```

```
The X matrix is 10 x 2
```

Descriptive Statistics

We will compute two descriptive statistics, the mean and the variance for each variable.

The mean may be computed using *Mathematica*'s function or directly

Here is the mean of Y produced by Mathematic's built-in function

```
In[12]:= Mean[Y][[1]]
```

```
Out[12]= 229 300
```

Using $\frac{\sum_{i=1}^n y_i}{n}$ we get the same answer. This time we name the result for future use

```
In[13]:=  $\mu_Y = \frac{\text{Plus @@ Y}}{n} [ [1]]$ 
```

```
Out[13]= 229 300
```

We follow the same procedure for the independent variables

```
Mean[X][[All, 2]]
```

```
1731
```

```
In[14]:=  $\mu_X = \frac{\text{Plus @@ X}}{n} [ [2]]$ 
```

```
Out[14]= 1731
```

Here are two presentations arriving at the same sample variance of the size vector, the sum of the squared differences between individual values of the independent variables and the mean, all divided by n-1.

This is *Mathematica*'s automated output

```
In[15]:= N[Variance[size]][[1]]
```

```
Out[15]= 25 137.8
```

To produce it using an equation we must define \bar{X} , a vector of the mean of **X** having a length equal to **X**

```
In[16]:=  $\bar{X} = \mu_X * \text{Table}[i, \{i, n\}, \{i, 1\}];$ 
```

This is the algebraic version, $\frac{\sum (x - \bar{x})^2}{(n-1)}$

```
In[17]:=  $\frac{(\text{Plus @@ (size - \bar{X})^2) [ [1]]}{n - 1} // N$ 
```

```
Out[17]= 25 137.8
```

The mean and the variance are known as, respectively, the first and second moment of the distribution. Other useful measures are skewness and kurtosis, the third and fourth moments of the distribution. These, if significant, represent a deviation from normality because a normal distribution has a skewness of zero (it is symmetric) and an excess kurtosis of zero (it has light tails). Our Y data has a slight skewness and lighter than normal tails.

```
In[23]:= Skewness[Y] // N
```

```
Kurtosis[Y] - 3 // N
```

```
Out[23]= {0.231227}
```

```
Out[24]= {-1.14215}
```

Regression

■ Preliminary calculations, matrices and the normal equations

Crucial to the regression process is the creation of some special matrices.

For instance, the dot product of $X^T X$ must be a non-singular matrix. This means the $X^T X$ matrix must have an inverse. This property leads to a unique solution of the minimization problem required to solve for the least squares estimators.

```
In[25]:= Transpose[X].X
```

```
Out[25]//MatrixForm=
```

$$\begin{pmatrix} 10 & 17310 \\ 17310 & 30189850 \end{pmatrix}$$

The above creates a matrix with the sum of the squares of the individual vectors in X on the diagonal and the sum of the values of the independent variable on the off diagonal. Note that this also produces the coefficients of the "normal equations". These are simultaneous equations which, when solved, produce the estimated a and b of the regression. The two equations in two unknowns, a and b, are shown as equations (4.1) and (4.2).

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

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

Let's unravel this notation one term at a time.

On the left hand side of equation (4.1), the sum of the dependent variables (observed house prices), $\sum_{i=1}^n y_i$ is

```
In[26]:= (Plus @@ Y) [[1]]
```

```
Out[26]= 2293000
```

n is simple enough. Note that it is the upper left term in the $X^T X$ matrix.

```
In[27]:= n
```

```
Out[27]= 10
```

The upper right and lower left (off diagonal) terms in the $X^T X$ matrix is the sum of the independent variables (size of houses), $\sum_{i=1}^n x_i$. This is the coefficient of b on the right hand side of equation (4.1).

```
In[28]:= (Plus @@ size) [[1]]
```

```
Out[28]= 17310
```

Moving to equation (4.2), the term on the left hand side, $\sum_{i=1}^n x_i y_i$, is the sum of the product of each house price and its size

```
In[29]:= (Transpose[Y].size) [[1]][[1]]
```

```
Out[29]= 4009490000
```

On the right side of equation (4.2) we again encounter the term in off-diagonal of the $X^T X$ matrix, the sum of the independent variables (size of houses), $\sum_{i=1}^n x_i$. This time it is the coefficient of a.

```
In[30]:= (Plus @@ size) [[1]]
```

```
Out[30]= 17310
```

Finally, the lower right element in the $X^T X$ matrix, the sum of the square of the size of each house, $\sum_{i=1}^n x_i^2$ is the coefficient of b in equation (4.2).

```
In[31]:= (Transpose[size].size)[[1]][[1]]
```

```
Out[31]= 30 189 850
```

Inserting the values we have found for the terms in equations (4.1) and (4.2) produce two equations in two unknowns.

$$\begin{aligned} 2293000 &= 10 a + 17310 b \\ 4009490000 &= 17310 a + 30189850 b \end{aligned}$$

These can be combined into a pair of equations using *Mathematica*'s Solve function to find the values for the a and b unknowns.

```
In[32]:= N[Solve[{(Plus@@Y)[[1]] == n a + b (Plus@@size)[[1]], (Transpose[Y].size)[[1]][[1]] ==  
  (a (Plus@@size) + b (Transpose[size].size)[[1]][[1]])[[1]]}, {a, b}]] // TableForm
```

```
Out[32]/TableForm=
```

```
a → -79 095.6    b → 178.16
```

Students are often confused by the different notation styles used to present equations and mathematical identities. One usually learns algebra and its symbols and then "graduates" to matrix algebra and its notation form. In time one recognizes the similarities between the two styles and moves easily between them. Until that time one may endure much frustration.

For instance, equation (1.1) may be expressed in matrix notation as

$$Y = X \beta_i + \epsilon$$

Where the symbols stand for vectors and matrices.

In general, above we have used both methods to present the normal equations and solve for their two unknown coefficients. This is easy enough when there are only two equations and two unknowns. But algebra becomes unwieldy in the multivariate case when the number of independent variables exceeds one. The multivariate case involves multiple coefficients (b_i , $i = 1 \dots n$) to allow for many independent variables (such as not only the size of the house but the size of the lot the house is on, the number of bedrooms, etc.). This results in a system of many unknowns with an equal number of equations that is too cumbersome for the algebraic format. Matrix algebra streamlines the notation in the multivariate case.

Above we said that the $X^T X$ must have an inverse. $(X^T X)^{-1}$ is the inverse of $X^T X$.

```
In[33]:= N[Inverse[Transpose[X].X]]
```

```
Out[33]/MatrixForm=
```

$$\begin{pmatrix} 13.3442 & -0.00765117 \\ -0.00765117 & 4.42008 \times 10^{-6} \end{pmatrix}$$

Here are two ways *Mathematica* "picks" the lower right corner term. We name the second one "LwrRt" for later use in forming the t-statistic for the estimated beta coefficient

```
In[34]:= N[Inverse[Transpose[X].X]][[2, 2]]
```

```
Out[34]= 4.42008 × 10-6
```

```
In[35]:= LwrRt = N[{0, 1}.Inverse[Transpose[X].X].{{0}, {1}}]
```

```
Out[35]= {4.42008 × 10-6}
```

The product of the inverse matrix and X^T also produces a matrix. Using that matrix and the Y values produces a vector of parameter estimates (a and b) for the regression:

```
In[36]:= N[Inverse[Transpose[X].X].Transpose[X]]
```

```
Out[36]//MatrixForm=
```

$$\begin{pmatrix} 1.86742 & -0.0453722 & 1.1023 & 0.337186 & -1.19305 & -1.00177 & 0.719745 \\ -0.00102104 & 0.0000839816 & -0.000579031 & -0.000137023 & 0.000746994 & 0.000636492 & -0.000358027 \end{pmatrix}$$

The parameter estimates are the solutions to a minimization problem. We present them as a vector, $\hat{\theta} = \begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \end{pmatrix}$. The elements in this vector should be compared with values for a and b reached with the normal equations above and values for the intercept and "SF" shown in the parameter table of Table 6-7 of Chapter 6.

The matrix algebra equation for the parameter estimate vector is $\hat{\theta} = (X^T X)^{-1} X^T Y$

```
In[37]:=  $\hat{\theta} = N[Inverse[Transpose[X].X].Transpose[X].Y]$ 
```

```
Out[37]//MatrixForm=
```

$$\begin{pmatrix} -79095.6 \\ 178.16 \end{pmatrix}$$

Mathematica puts these in the form of a regression equation using the function "Fit"

```
In[38]:= Fit[houses, {1, x}, x] // TraditionalForm
```

```
Out[38]//TraditionalForm=
```

$$178.16x - 79095.6$$

Mathematica also uses LinearModelFit to produce an object that can be used for a variety of diagnostics, listed below

```
In[40]:= lmf = LinearModelFit[houses, {1, x}, x]
```

```
Out[40]= FittedModel[ $-79095.6 + \ll 19 \gg x$ ]
```

```
In[91]:= lmf["Properties"]
```

```
Out[91]= {AdjustedRSquared, AIC, ANOVATable, ANOVATableDegreesOfFreedom, ANOVATableEntries, ANOVATableFStatistics, ANOVATableMeanSquares, ANOVATablePValues, ANOVATableSumsOfSquares, BasisFunctions, BetaDifferences, BestFit, BestFitParameters, BIC, CatcherMatrix, CoefficientOfVariation, CookDistances, CorrelationMatrix, CovarianceMatrix, CovarianceRatios, Data, DesignMatrix, DurbinWatsonD, EigenstructureTable, EigenstructureTableEigenvalues, EigenstructureTableEntries, EigenstructureTableIndexes, EigenstructureTablePartitions, EstimatedVariance, FitDifferences, FitResiduals, Function, FVarianceRatios, HatDiagonal, MeanPredictionBands, MeanPredictionConfidenceIntervals, MeanPredictionConfidenceIntervalTable, MeanPredictionConfidenceIntervalTableEntries, MeanPredictionErrors, ParameterConfidenceIntervals, ParameterConfidenceIntervalTable, ParameterConfidenceIntervalTableEntries, ParameterConfidenceRegion, ParameterErrors, ParameterPValues, ParameterTable, ParameterTableEntries, ParameterTStatistics, PartialSumOfSquares, PredictedResponse, Properties, Response, RSquared, SequentialSumOfSquares, SingleDeletionVariances, SinglePredictionBands, SinglePredictionConfidenceIntervals, SinglePredictionConfidenceIntervalTable, SinglePredictionConfidenceIntervalTableEntries, SinglePredictionErrors, StandardizedResiduals, StudentizedResiduals, VarianceInflationFactors}
```

Each of which can be separately researched. Example (follow link): AIC

The LinearModelFit object can be represented in standard form by wrapping it with "Normal"

```
In[46]:= houseFit = Normal[lmf]
          TraditionalForm[%]
```

```
Out[46]= -79 095.6 + 178.16 x
```

```
Out[47]/TraditionalForm=
178.16 x - 79 095.6
```

We pick the intercept and the parameter estimate, each, out of the vector of estimated parameters, $\hat{\theta}$, and name them for later use.

```
In[44]:= a = {1, 0}. $\hat{\theta}$ 
```

```
Out[44]= {-79 095.6}
```

```
In[45]:= b1 = {0, 1}. $\hat{\theta}$ 
```

```
Out[45]= {178.16}
```

Before continuing, it is important to relate all of this to the SP distribution issues of Chapter 6. A central purpose of regression is to predict the conditional mean. For our example that is the average house value *conditioned upon knowing some other fact*, in our case the size of the house). This prediction is dependent on the accurate estimate of β , shown as the value of b_1 above. After performing the regression, our claim is that the value of a house goes up by \$178.16 for each added square foot. This is only correct if we have correctly estimated the mean itself. Outliers, as we have said, have a significant affect on the mean. Thus, if the assumption of normality is violated by significant outliers in the distribution the estimated regression coefficient(s) are distorted. For an excellent and quite technical explanation of this situation using a real estate example see McCulloch (1998).

■ Residuals

The regression report in Table 6-7 of Chapter 6 is reproduced here for comparison to the results below.

RESIDUAL OUTPUT

Observation	Predicted Price	Residuals
1	188145.0	6855.0
2	232685.0	-22685.0
3	205961.0	19039.0
4	223777.0	16223.0
5	259409.1	15590.9
6	254955.1	30044.9
7	214869.0	-24869.0
8	243374.7	-4374.7
9	272771.1	-23771.1
10	197053.0	-12053.0

Mathematica will also perform the above as part of diagnostics and analysis for lmf

```
In[51]:= resids = LinearModelFit[houses, {1, x}, x] ["FitResiduals"]
```

```
Out[51]= {6855.04, -22685., 19039., 16223., 15590.9, 30044.9, -24869., -4374.67, -23771.1, -12053.}
```

The values predicted by the model, our estimated house values \hat{Y} , are the product of the vector of independent variables and the vector of parameters, $\hat{Y} = \hat{X} \hat{\theta}$

```
In[49]:=  $\hat{Y} = \mathbf{x} \cdot \hat{\theta}$ 
```


Out[49]//MatrixForm=

$$\begin{pmatrix} 188145. \\ 232685. \\ 205961. \\ 223777. \\ 259409. \\ 254955. \\ 214869. \\ 243375. \\ 272771. \\ 197053. \end{pmatrix}$$

Our model is expected to differ from reality. The regression residuals, the error terms, are the result of subtracting the predicted values (\hat{Y}) from the actual (Y) values.

The equation for the error terms is $\epsilon = Y - \hat{Y} = Y - X\hat{\theta}$

In[54]:= $\epsilon = Y - X.\hat{\theta}$

Out[54]//MatrixForm=

$$\begin{pmatrix} 6855.04 \\ -22685. \\ 19039. \\ 16223. \\ 15590.9 \\ 30044.9 \\ -24869. \\ -4374.67 \\ -23771.1 \\ -12053. \end{pmatrix}$$

Notice these are the same as the residuals

In[55]:= **Chop[resids - %]**

Out[55]//MatrixForm=

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

■ Analysis of Variance

From Table 6-7 of Chapter 6 we reproduce for reference the analysis of variance table.

ANOVA					
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>	<i>Significance F</i>
Regression	1	7181109658	7181109658	15.6069079	0.004232704
Residual	8	3680990342	460123792.8		
Total	9	10862100000			

Mathematica will produce the same ANOVA table, parts of which will will derive individually below.

```
In[56]:= LinearModelFit[houses, {1, x}, x] ["ANOVATable"]
```

	DF	SS	MS	F-Statistic	P-Value
Out[56]:= x	1	7.18111×10^9	7.18111×10^9	15.6069	0.0042327
Error	8	3.68099×10^9	4.60124×10^8		
Total	9	1.08621×10^{10}			

We need a vector, \bar{Y} in which each element is the mean of Y

```
In[57]:=  $\bar{Y} = \mu_Y * \text{Table}[i, \{i, n\}, \{i, 1\}];$ 
```

The sum of the squares of the regression (known both as SSR and $b^2 S_{xx}$) is $(\hat{Y} - \bar{Y})^T (\hat{Y} - \bar{Y})$.

```
In[58]:= SSR = Transpose[ $\hat{Y} - \bar{Y}$ ] . ( $\hat{Y} - \bar{Y}$ )
```

```
Out[58]//MatrixForm=
```

(7.18111×10^9)

The Sum of the Squared Errors (SSE aka $\Sigma[\epsilon_i^2]$)

```
In[59]:= SSE = Transpose[ $\epsilon$ ] .  $\epsilon$ 
```

```
Out[59]//MatrixForm=
```

(3.68099×10^9)

The total sum of squares (known either as SST or S_{yy} or $\Sigma(y_i - \hat{y})^2$) is the dot product $(Y - \bar{Y})^T (Y - \bar{Y})$.

```
In[60]:= SST = Transpose[Y -  $\bar{Y}$ ] . (Y -  $\bar{Y}$ )
```

```
Out[60]//MatrixForm=
```

(10862100000)

The Mean Square of the Regression (MSR) is the Sum of Squares of the Regression (SSR) divided by the degrees of freedom for the regression (k-1).

```
In[61]:= MSR = SSR / (k - 1)
```

```
Out[61]//MatrixForm=
```

(7.18111×10^9)

The Mean Square of the Error terms (MSE) is the Sum of the Squared Errors divided by the degrees of freedom for the errors.

```
In[62]:= MSE = SSE / (n - k)
```

```
Out[62]//MatrixForm=
```

(4.60124×10^8)

The F-test for model utility (F-statistic) is the result of dividing the MSR by the MSE

```
In[63]:= F = MSR / MSE
```

```
Out[63]//MatrixForm=
```

(15.6069)

■ The Parameter Table

From Table 6-7 of Chapter 6 we reproduce for reference the parameter table of the regression output. This provides, in addition to parameter estimates, the standard errors, t-statistics and p values for each estimate.

	Coefficients	Standard Error	t Stat	P-value
Intercept	-79095.58434	78357.96111	-1.009413507	0.342328491
Size	178.1603607	45.09751892	3.950557923	0.004232704

Mathematica's parameter table produces the same information. We will compute these individually for the independent variable only.

```
In[64]:= LinearModelFit[houses, {1, x}, x] ["ParameterTable"]
```

```
Out[64]=
```

	Estimate	Standard Error	t-Statistic	P-Value
1	-79095.6	78358.	-1.00941	0.342328
x	178.16	45.0975	3.95056	0.0042327

The standard error of the b coefficient uses SSE and the square root of the lower right element in $(X^T X)^{-1}$. We will shortly see that the term on the left, $\text{Sqrt}[SSE/(n-k)]$, is also known as the Standard Error of the regression, S.

```
In[65]:= Sb1 = Sqrt[SSE / (n - k)] * Sqrt[LwrRt]
```

```
Out[65]//MatrixForm=
( 45.0975 )
```

The t-stat for the b coefficient test of the null hypothesis that $\beta = 0$ is the ratio of the coefficient to its standard error. This is sometimes referred to as a "signal to noise ratio". Depending on the asymptotic properties of the distribution we view the size of this measure as a level of confidence that the value of our estimator did not occur by chance. A large value indicates that our coefficient provides more "signal than noise"

```
In[66]:= tB1 = (b1 / (Sb1)) [[1]] [[1]]
```

```
Out[66]= 3.95056
```

■ Summary Output

Finally, the Summary Output below is provided from the Table 6-7

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.813090489
R Square	0.661116143
Adjusted R Square	0.618755661
Standard Error	21450.49633
Observations	10

The sum of the squared differences between X and \bar{X} (known both as S_{xx} and as $\sum(x_i - \bar{x})^2$) is the product $(x - \bar{x})^T \cdot (x - \bar{x})$.

```
In[67]:= Sxx = Transpose[size - X] . (size - X)
```

```
Out[67]//MatrixForm=
( 226 240 )
```

There are several ways to compute the R^2 -value. the first is to have *Mathematica* compute it as an option in the Regression Report.

```
In[68]:= LinearModelFit[houses, {1, x}, x] ["RSquared"]
```

```
Out[68]= 0.661116
```

R^2 is the ratio of the sum of the squared regression to the total sum of squares.

```
In[69]:= Rsqd = SSR / SST
```

```
Out[69]//MatrixForm=
( 0.661116 )
```

Or it is 1 minus the ratio of the sum of the squared errors to the total sum of squares.

```
In[70]:= 1 - (SSE / SST)
```

```
Out[70]//MatrixForm=
( 0.661116 )
```

Or it is the product of the estimated coefficient, b_1 and the ratio of the sum of the squared differences between \mathbf{X} and \overline{X} and the total sum of squares, $b^2 S_{xx} / S_{yy}$ or $b^2 S_{xx} / SST$

```
In[71]:= (b1^2) * Sxx / SST
```

```
Out[71]//MatrixForm=
( 0.661116 )
```

R^2 is also the square of the correlation coefficient

```
In[72]:= N[Correlation[xdata, ydata] ^ 2]
```

```
Out[72]= 0.661116
```

The standard error of the regression (S) is the square root of the ratio of the SSE to the degrees of freedom (n-k):

```
In[73]:= S = Sqrt[SSE / (n - k)]
```

```
Out[73]//MatrixForm=
( 21450.5 )
```

Notice that the square of S is also the Mean Squared Error

```
In[74]:= TrueQ[S^2 == MSE]
```

```
Out[74]= True
```

The standard deviation of the observed Y values, s_y , may be computed by *Mathematica*'s StandardDeviation function or as the square root of the ratio of the total sum of squares to (n-1).

```
In[75]:= StandardDeviation[ydata] // N
```

```
Out[75]= 34740.5
```

```
In[76]:= Sy = N[Sqrt[SST / (n - 1)] [[1]] [[1]]]
```

```
Out[76]= 34740.5
```

It is, of course, also the square root of the variance.

```
In[77]:= Sqrt[Variance[ydata]] // N
```

```
Out[77]= 34740.5
```

Dividing the square of s_y into the square of the standard error of the regression and subtracting the result from 1 gives the adjusted R^2 :

```
In[78]:= LinearModelFit[houses, {1, x}, x] ["AdjustedRSquared"]
```

```
Out[78]= 0.618756
```

```
In[79]:= adjRsqr = N[1 - (S^2 / Sy^2)]
```

```
Out[79]//MatrixForm=
( 0.618756 )
```

■ Alternate presentations

Here are some other ways to use *Mathematica* to display regression output.

We can produce all the output options at once.

```
In[80]:= LinearModelFit[houses, {1, x}, x][{"ANOVATable", "ParameterTable", "FitResiduals",
      "SinglePredictionConfidenceIntervalTable", "ParameterConfidenceRegion"}]
```

```
Out[80]= { x      | DF  SS           MS           F-Statistic  P-Value
      Error | 8   3.68099×109  4.60124×108
      Total | 9   1.08621×1010

      | Estimate  Standard Error  t-Statistic  P-Value
1 | -79095.6  78358.      -1.00941  0.342328 , {6855.04, -22685., 19039.,
x | 178.16   45.0975      3.95056  0.0042327
   16223., 15590.9, 30044.9, -24869., -4374.67, -23771.1, -12053.},

Observed  Predicted  Standard Error  Confidence Interval
195000    188145.    24792.4        {130974., 245316.}
210000    232685.    22513.8        {180768., 284602.}
225000    205961.    23260.2        {152323., 259599.}
240000    223777.    22540.9        {171798., 275756.}
275000    259409.    23753.4        {204634., 314184.} , FittedModels`ParameterEllipsoid[
285000    254955.    23416.         {200958., 308952.}
190000    214869.    22792.1        {162310., 267428.}
239000    243375.    22777.8        {190849., 295900.}
249000    272771.    25044.4        {215019., 330524.}
185000    197053.    23932.5        {141865., 252241.}

{-79095.6, 178.16}, {234000., 11.6584}, {{-1., 0.000573371}, {-0.000573371, -1.}}}]
```

We can name the errors and predicted values in separate lists and plot them together to get a visual look at whether there is a relationship such as bigger errors for larger predictions.

```
In[83]:= errors = LinearModelFit[houses, {1, x}, x]["FitResiduals"];
% // TableForm
```

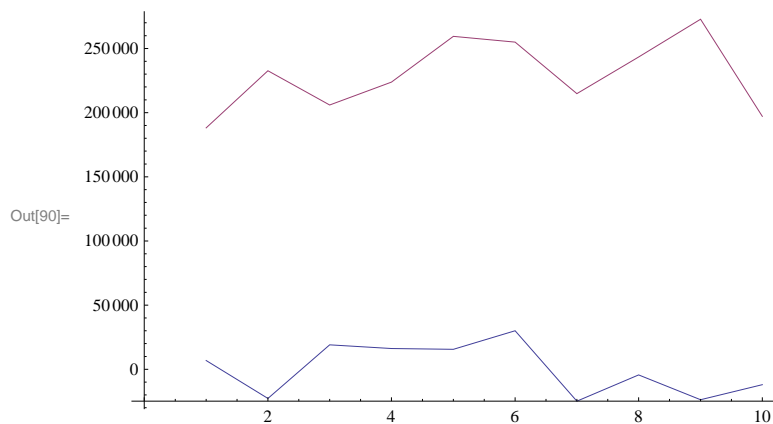
```
Out[84]//TableForm=
6855.04
-22685.
19039.
16223.
15590.9
30044.9
-24869.
-4374.67
-23771.1
-12053.
```

```
In[85]:= predValues = LinearModelFit[houses, {1, x}, x]["PredictedResponse"];
% // TableForm
```

```
Out[86]//TableForm=
```

```
188 145.
232 685.
205 961.
223 777.
259 409.
254 955.
214 869.
243 375.
272 771.
197 053.
```

```
In[90]:= ListPlot[{errors, predValues}, Joined → True, AxesOrigin → {0, Min[errors]}]
```



■ References

- Greene, William H. (2002). *Econometric Analysis*, 5th Ed., Upper Saddle River, NJ: Prentice Hall
- McCulloch, J. H. (1998). Linear Regression with Stable Disturbances. R. J. F. R. E. T. M. S. Adler (Editors), *A Practical Guide to Heavy Tails* (First ed., pp. 359-376). Boston, MA: Birkhauser.
- Nolan, J. P. (1997). Numerical Calculation of Stable Densities and Distribution Functions. *Communications in Statistics - Stochastic Models*, 13(4), 759-774.
- Young, M. S. , & Graff, R. A. (1995). Real Estate is Not Normal: A Fresh Look at Real Estate Return Distributions. *Journal of Real Estate Finance and Economics*, 10(3), 225-259 (1995).