Best Multiple Regression Models Using Excel Linest Function

By Namir C. Shammas

In this article I present three versions of VBA subroutines that select the best (empirical) regression models among a larger set of models. In all cases, the subroutines obtain the values of observed variables and then apply a number of transformations on each variable to determine which combination of transformations yield the best set of models. While we are certainly interested in the very best regression model, it is a very wise to view the best, say, 50 models. The reason is that if the user has several sets of similar data (each having its own errors) then a possible best empirical regression model will consistently appear in the set of top models. I will nickname this set of best models the *Hit Parade*. Once you select one or more regression models, you can use subroutine **MLR**, which I present in the first article, to obtain the regression ANOVA table for these selected regression models.

This article presents three approaches for obtaining the best (empirical) regression models:

- Scheme M1: Applying a combination of powers and the natural logarithm to the observed variables.
- Scheme M2: Applying a combination of expressions to transform the observed variables.
- Scheme M3: Applying a set of regression models.

Basic Linearized Regression Approach

When you perform regression calculations you have the following ingredients:

- A set of observed values for variables. These variables include the dependent variable and at least one independent variable. Multiple independent variables are very common in multiple regression.
- One or more regression models (i.e. equations) that describes the relationship between the dependent variable and the independent variables.
- Transformed regression variables that are calculated based on the observed variables and the given regression model(s). The calculation of the regression coefficients of the model are based on the transformed variables.

In the case of simple linear models, the observed variables are also the regression variables. By contrast, in the case of linearized models, the regression variables have their values calculated based on the observed variables. Depending on the regression model, the regression variables can be straightforward transformation of corresponding observed variables (i.e. one-to-one relation). More complex regression model create regression variables based on the mathematical expressions that involve multiple observed variables.

Here are examples of simple multiple regression models:

$Y = a_0 + a_1 X_1 + a_2 X_2$

$Y = a_0 + a_1 X_1 + a_2 X_2 + a_3 X_3$

The variables Y, X_1, X_2, and X_3 are the observed variables and are also the regression variables.

Here are examples of regression models where some of the variables are linearized. The linearization uses a simple one-to-one transformation:

$1/Y = a_0 + a_1 1/X_1 + a_2 1/X_2$

$Y = a_0 + a_1 \ln(X_1) + a_2 1/X_2 + a_3 \sqrt{}X_3$

$1/Y = a_0 + a_1 1/X_1 + a_2 1/X_2\text{^}2 + a_3 X_3\text{^}3$

Notice that the models are linearized by applying transformations to the various observed variables. For example, the first model has the observed variables Y, X_1 and X_2. The model has the regression variables of 1/Y, $1/X_1$, and $1/X_2$. The above examples show simple linearized terms.

Here are examples for more elaborate linearized regression models:

$Y = a_0 + a_1 1/(2 X_1 X_2 + 1) + a_2 1/(X_1{*}X_2)$

$Y = a_0 + a_1 \ln(1 + X_1 + X_2) + a_2 \ln(X_2)/(3X_2 + 1) + a_3 \sqrt{}(X_1\text{^}2 + X_3\text{^}2)$

$1/Y = a_0 + a_1 1/(X_1\text{^}2 + 1) + a_2 X_1/X_2\text{^}2 + a_3 X_1 X_2 X_3\text{^}3$

$Y = a_0 + a_1 X + a_2 X^2 + a_3 X^3$

The above sample regression models show several terms that are expressions containing one or more variables. The number of regression variables may or may not equal the number of observed variables. This equality (or inequality) is not an issue. For example, the first model has the observed variables Y, X_1 and X_2. The model has the regression variables of Y, $1/(2 X_1 X_2 + 1)$, and $1//(X_1{*}X_2)$. These regression models can still benefit from the function **Linest**, because they are linearized multiple regression models. The last model has the observed variables X and Y, and the regression variables Y, X, X^2, and X^3. This polynomial model is typical of polynomials that have a single independent variable.

It is very important to make a distinction between the observed variables (dependent and independent) and the regression variables. The two types of variables *can* be the same, but this equivalence serves to support only simple multiple regression models. The regression coefficients are based directly on the values of the regression variables.

The Results of the Linest Function

The **Linest** function is a powerful Excel function that returns a matrix of results. The arguments for the function **Linest** are data ranges. When calling the function in VBA you can pass the ranges as such or as variant-type variables. The function's results include the regression coefficients, the standard errors for the regression coefficients, and other statistics that allow you to obtain the regression ANOVA table. You can display the results of function **Linest** as an array of ranges, or store its results in a variant-typed variable. You can then access various elements of that variable to obtain the different values returned by the function **Linest**.

The **Linest** function returns an irregular r-like shaped matrix of results. Table 1 shows an outline for these results.

Slope a_n	Slope a_{n-1}	Slope a_{n-2}	…	Intercept a_0
Standard error for a_n	Standard error for a_{n-1}	Standard error for a_{n-2}	…	Standard error for a_0
R^2	SE(y)			
F statistics	Degrees of freedom			
Regression SS	Residual SS			

Table 1. Outline for the results of function Linest.

The Scheme M1 Approach for Best Regression Models

The best way to present and explain scheme M1 is to show you the worksheet for this approach to getting the best regression models. Figures 1 and 2 show the left and right sides, respectively, of a sample scheme M1 worksheet. Notice the following individual cells and columns of cells that provide input for the regression program:

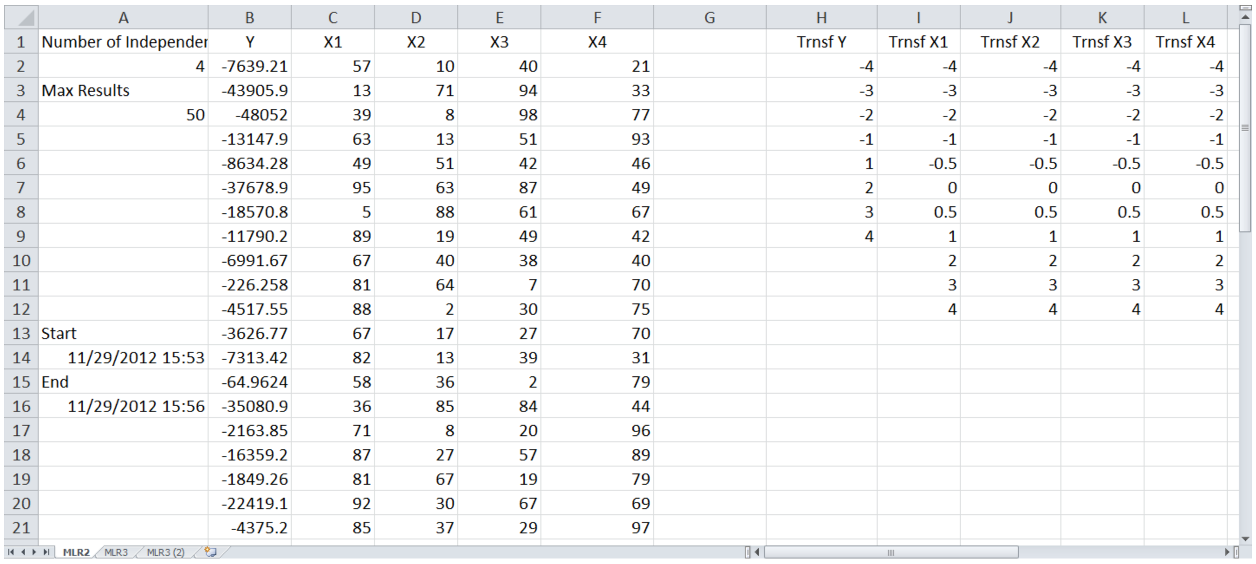
- Cell A2 contains the number of independent variables entering the regression. These regression variables can be the values of observed variables and/or their transformations.
- Cell A4 contains the values for the maximum number of best models. The current value is 50.
- Column B has the values of the observed dependent variable Y, starting at cell B2.
- Column C stores the values of the first observed independent variable, starting at cell C2.
- Columns D and beyond store the values of additional observed independent variables, starting at the columns' second rows.
- The column that comes after the last column of independent variable MUST BE empty.
- The first column of transformations for variable Y.
- The second column of transformations for variable X1.
- The third column of transformations for variable X2.

- Other columns of transformations for additional variables.
- The column that comes after the last column of transformations MUST BE empty.

The following worksheet columns provide the regression output generated by running the VBA macro **BestMLR1**. You need to prepare the headers manually:

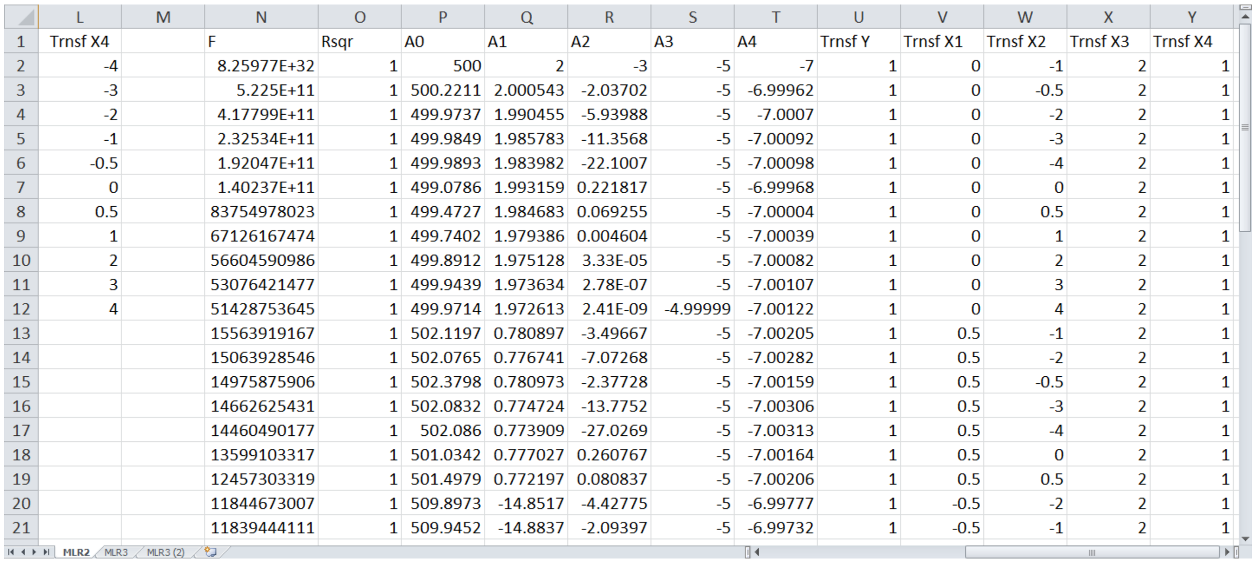
- The first column shows the values of the F statistic.
- The second column shows the values of the R^2 statistic
- The third column shows the values of the intercepts.
- The fourth column shows the values for the regression slope of variable X1.
- The fifth column shows the values for the regression slope of variable X2.
- The next columns show the values for the regression slopes of other variables.
- The next set of columns show the transformations associated with each model.

The rows that start with the column for the F statistic represent the record for each model. These rows are sorted in a descending order using the values of the F statistic.



	A	B	C	D	E	F	G	H	I	J	K	L
1	Number of Independer	Y	X1	X2	X3	X4		Trnsf Y	Trnsf X1	Trnsf X2	Trnsf X3	Trnsf X4
2	4	-7639.21	57	10	40	21		-4	-4	-4	-4	-4
3	Max Results	-43905.9	13	71	94	33		-3	-3	-3	-3	-3
4	50	-48052	39	8	98	77		-2	-2	-2	-2	-2
5		-13147.9	63	13	51	93		-1	-1	-1	-1	-1
6		-8634.28	49	51	42	46		1	-0.5	-0.5	-0.5	-0.5
7		-37678.9	95	63	87	49		2	0	0	0	0
8		-18570.8	5	88	61	67		3	0.5	0.5	0.5	0.5
9		-11790.2	89	19	49	42		4	1	1	1	1
10		-6991.67	67	40	38	40			2	2	2	2
11		-226.258	81	64	7	70			3	3	3	3
12		-4517.55	88	2	30	75			4	4	4	4
13	Start	-3626.77	67	17	27	70						
14	11/29/2012 15:53	-7313.42	82	13	39	31						
15	End	-64.9624	58	36	2	79						
16	11/29/2012 15:56	-35080.9	36	85	84	44						
17		-2163.85	71	8	20	96						
18		-16359.2	87	27	57	89						
19		-1849.26	81	67	19	79						
20		-22419.1	92	30	67	69						
21		-4375.2	85	37	29	97						

Figure 1. The left side of worksheet for scheme M1.



	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	Trnsf X4		F	Rsqr	A0	A1	A2	A3	A4	Trnsf Y	Trnsf X1	Trnsf X2	Trnsf X3	Trnsf X4
2	-4		8.25977E+32	1	500	2	-3	-5	-7	1	0	-1	2	1
3	-3		5.225E+11	1	500.2211	2.000543	-2.03702	-5	-6.99962	1	0	-0.5	2	1
4	-2		4.17799E+11	1	499.9737	1.990455	-5.93988	-5	-7.0007	1	0	-2	2	1
5	-1		2.32534E+11	1	499.9849	1.985783	-11.3568	-5	-7.00092	1	0	-3	2	1
6	-0.5		1.92047E+11	1	499.9893	1.983982	-22.1007	-5	-7.00098	1	0	-4	2	1
7	0		1.40237E+11	1	499.0786	1.993159	0.221817	-5	-6.99968	1	0	0	2	1
8	0.5		83754978023	1	499.4727	1.984683	0.069255	-5	-7.00004	1	0	0.5	2	1
9	1		67126167474	1	499.7402	1.979386	0.004604	-5	-7.00039	1	0	1	2	1
10	2		56604590986	1	499.8912	1.975128	3.33E-05	-5	-7.00082	1	0	2	2	1
11	3		53076421477	1	499.9439	1.973634	2.78E-07	-5	-7.00107	1	0	3	2	1
12	4		51428753645	1	499.9714	1.972613	2.41E-09	-4.99999	-7.00122	1	0	4	2	1
13			15563919167	1	502.1197	0.780897	-3.49667	-5	-7.00205	1	0.5	-1	2	1
14			15063928546	1	502.0765	0.776741	-7.07268	-5	-7.00282	1	0.5	-2	2	1
15			14975875906	1	502.3798	0.780973	-2.37728	-5	-7.00159	1	0.5	-0.5	2	1
16			14662625431	1	502.0832	0.774724	-13.7752	-5	-7.00306	1	0.5	-3	2	1
17			14460490177	1	502.086	0.773909	-27.0269	-5	-7.00313	1	0.5	-4	2	1
18			13599103317	1	501.0342	0.777027	0.260767	-5	-7.00164	1	0.5	0	2	1
19			12457303319	1	501.4979	0.772197	0.080837	-5	-7.00206	1	0.5	0.5	2	1
20			11844673007	1	509.8973	-14.8517	-4.42775	-5	-6.99777	1	-0.5	-2	2	1
21			11839444111	1	509.9452	-14.8837	-2.09397	-5	-6.99732	1	-0.5	-1	2	1

Figure 2. The right side of worksheet for scheme M1.

Looking at the columns H to L in Figure 1, you see the list of transformations for each variable. Column H shows the list of numerically coded transformations for variable Y. Column I through L show the list of numerically coded transformations for variables X1 through X4, respectively. The question poses itself. What are these numerically coded transformations? The numerically coded transformations represent powers used to raise the variables in order to calculate their transformations. For example, the value of –4 in cell H2 tells the program that one of the transformations for variable Y is 1/Y^4. You will no doubt notice that columns I through L contain the value of 0. The VBA code treats 0 as a special case and applies the natural logarithm to the corresponding variable.

The transformations of the observed variables require that you obey the following simple rules:

- Each variable has its own set of transformations, independent, in number and sequence, of those used with the other variables.
- The transformation can be integers and non-integers, as well as negative, zero, or positive. The VBA subroutine has an error handler and will catch runtime errors. Any set of transformation that generates a runtime error is out of the contest, so to speak!

Looking at the sample results in Figure 2, the best model is:

$$Y = 500 + 2 \ln(X1) – 3/X2 – 5 X2^2 – 7 X4$$

This model has the F statistic of 8.259E+32 and the R^2 value of 1. Notice that there is quite the difference in the values of the F statistic between the best model and the second best model. It is also interesting to see that the first few models in the Hit Parade list only differ in the transformations of variable X2.

The BestMLR1 VBA Subroutine

Listing 1 shows the source code for subroutine **BestMLR1**. This subroutine performs the following general tasks:

- Read the input from the leading columns. This includes the observations for the different variables and the lists of numerically coded transformations.
- Establish the first set of transformations.
- Transform the observed variables into the regression variables using the current set of powers.
- Invoke the **Linest** function and store its results in variable **vRegResultsMat**. The subroutine compares the value of the F statistic with the lowest value of F in the Hit Parade list. If the newly calculated F statistic is greater than that lowest F value, the subroutine overwrites the last Hit Parade row with newly calculated data and then invokes Excel's Sort method to quickly re-sort the Hit Parade list using values of the F statistics. Sorting the data maintains the order for the best regression model combination of transformations.
- Use the **For VarIdx** loop to simulate nested loops that systematically change the transformations for the variables.

The subroutine use the status bar to keep you informed of the calculations progress. The subroutine displays a message box when done. This message shows the starting and ending time for the calculations. These two values also appear in column A.

```
Option Explicit
Option Base 1

Function ExFx(ByVal sFx As String, ByRef X() As Variant, _
              ByRef Y() As Variant, _
              ByVal I As Integer, ByVal NumIVars As Integer) As Double
 Dim J As Integer
 ' replace Xnn starting with the higher indices just in case there
 ' are more than 9 variables.
 sFx = UCase(sFx)
 For J = NumIVars To 1 Step -1
   sFx = Replace(sFx, "X" & J, "(" & X(I, J) & ")")
 Next J
 sFx = Replace(sFx, "Y", "(" & Y(I, 1) & ")")
 ExFx = Evaluate(sFx)
End Function


Sub BestMLR2()

  Const MAX_ERRORS As Double = 1000000# ' initial max error value
  Dim ErrorCounter As Double, MaxErrors As Double, sMaxErr As String
  Dim NumIndpVars As Integer ' number of independent variables
  Dim TN As Integer ' total number of variables = NIV+1
  Dim N As Integer ' number of data points
  Dim MaxTrans As Integer ' max transformations
  Dim MaxResults As Integer ' max results
  Dim Col1 As Integer, Col2 As Integer, Col3 As Integer
  Dim Col4 As Integer, Col5 As Integer
  Dim I As Integer, J As Integer, K As Integer
  Dim M1 As Integer, M As Integer
  Dim VarIdx As Integer, Low As Integer, Hi As Integer
  Dim TransfMat() As String, sFx As String
  Dim CurrentTransf() As String, CountTransf() As Integer
  Dim NumTransf() As Integer ' number of transformations
  Dim Y() As Variant, X() As Variant
  Dim Yt() As Variant, Xt() As Variant
  Dim vRegResultsMat As Variant
  Dim F As Double, Rsqr As Double, xval As Double
  Dim fMaxCount As Double, fCount As Double, fMilestone As Double
  Dim dt1 As Date, dt2 As Date

  dt1 = Now
  ErrorCounter = 0
  MaxErrors = MAX_ERRORS
  NumIndpVars = [A2].Value
  MaxResults = [A4].Value
  TN = NumIndpVars + 1
  Col1 = 2              ' first column of data
  Col2 = Col1 + TN + 1  ' first column of transformations
  Col3 = Col2 + TN + 1  ' first column of results
  Col4 = Col3 + TN + 1  ' first column of tranformatioons
  Col5 = Col4 + TN - 1  ' last column of transformations

  Range(Cells(2, Col3), Cells(1 + 2 * MaxResults, Col3 + 3 *
TN)).Value = ""
  Range(Cells(2, Col3), Cells(1 + MaxResults, Col3 + 1 + TN)).Value =
0
  MaxTrans = Range(Cells(2, Col2), Cells(1,
Col2)).CurrentRegion.Rows.Count - 1
  ReDim NumTransf(TN), TransfMat(TN, MaxTrans), CurrentTransf(TN)
  ReDim CountTransf(TN)

  fMaxCount = 1
  For I = 1 To TN
```

```
    J = 2
    Do While Trim(Cells(J, Col2 + I - 1)) <> ""
      TransfMat(I, J - 1) = Cells(J, Col2 + I - 1)
      J = J + 1
    Loop
    NumTransf(I) = J - 2
    fMaxCount = fMaxCount * NumTransf(I)
  Next I

  N = Range("B1").CurrentRegion.Rows.Count - 1
  Y = Range("B2:B" & N + 1).Value
  X = Range(Cells(2, 3), Cells(N + 1, 2 + NumIndpVars)).Value
  Yt = Range("B2:B" & N + 1).Value
  Xt = Range(Cells(2, 3), Cells(N + 1, 2 + NumIndpVars)).Value

  ' set the initial transformations
  For I = 1 To TN
    CurrentTransf(I) = TransfMat(I, 1)
    CountTransf(I) = 1
  Next I

  fCount = 0
  fMilestone = 0.1
  Do

    On Error GoTo HandleErr

    For I = 1 To N
      DoEvents

      If fCount / fMaxCount > fMilestone Then
        DoEvents
        Application.StatusBar = "Processed " & CStr(fMilestone * 100)
& " %"
        If fMilestone < 1 Then fMilestone = fMilestone + 0.05
      End If


      ' sFx = CurrentTransf(1) '[A5].Value
      Yt(I, 1) = ExFx(CurrentTransf(1), X, Y, I, NumIndpVars)

      For J = 1 To NumIndpVars
        'sFx = CurrentTransf(J + 1)  ' Range("A" & M).Value
        Xt(I, J) = ExFx(CurrentTransf(J + 1), X, Y, I, NumIndpVars)
      Next J
    Next I

    ' perform the regression calculations
    vRegResultsMat = Application.WorksheetFunction.LinEst(Yt, Xt,
True, True)

    Rsqr = vRegResultsMat(3, 1)
    F = vRegResultsMat(4, 1)
    ' check if F > F of last result
    If F > Cells(MaxResults + 1, Col3) Then
      xval = fCount / fMaxCount * 100
      xval = CInt(100 * xval) / 100
      Application.StatusBar = "Processed " & CStr(xval) & " %"
      M1 = MaxResults + 1
      ' write new results to row M
      Cells(M1, Col3) = F
      Cells(M1, Col3 + 1) = Rsqr
      For I = 1 To TN
        Cells(M1, Col3 + I + 1) = vRegResultsMat(1, TN - I + 1)
      Next I
      For I = 1 To TN
        Cells(M1, Col4 + I) = CurrentTransf(I)
      Next I

      Range(Cells(2, Col3), Cells(MaxResults + 1, Col5 + 1)).Select
      Range(Cells(2, Col3), Cells(MaxResults + 1, Col5 + 1)).Sort
Key1:=Range(Cells(2, Col3), Cells(MaxResults + 1, Col3)),
Order1:=xlDescending

    End If ' If F > Cells(MaxResults + 1, Col3)

    GoTo Here

HandleErr:
  fCount = fCount - 1
  ErrorCounter = ErrorCounter + 1
  If ErrorCounter > MaxErrors Then
    If MsgBox("Reached maximum error limits of " & ErrorCounter &
vbCrLf & _
    "Want to stop the process?", vbYesNo + vbQuestion, "Confirmation
requested") = vbYes Then
      Exit Sub
    Else
      sMaxErr = InputBox("Update maximum number of errors", "Max
Errors Input", MaxErrors)
```

```
      If Trim(sMaxErr) = "" Then
        MsgBox "User canceled calculations process", vbOKOnly +
vbInformation, "End of Process"
        Exit Sub
      End If
      MaxErrors = CDbl(sMaxErr)
      ErrorCounter = 0
    End If
  End If
  Resume Here
Here:

    ' ---------------------------------------------------------
    ' ---------------------------------------------------------
    '  ------------ Simulate Nested Loops ---------------------
    ' ---------------------------------------------------------
    ' ---------------------------------------------------------

    For VarIdx = 1 To TN
      DoEvents
      If CountTransf(VarIdx) >= NumTransf(VarIdx) Then
        If VarIdx < TN Then
          CurrentTransf(VarIdx) = TransfMat(VarIdx, 1)
          CountTransf(VarIdx) = 1
        Else
          Exit Do
        End If
      Else
        CountTransf(VarIdx) = CountTransf(VarIdx) + 1
        CurrentTransf(VarIdx) = TransfMat(VarIdx, CountTransf(VarIdx))
        fCount = fCount + 1
        Exit For
      End If
    Next VarIdx

  Loop

  On Error GoTo 0
  dt2 = Now
  [A13].Value = "Start"
  [A14].Value = dt1
  [A15].Value = "End"
  [A16].Value = dt2
  Application.StatusBar = "Done"
  MsgBox "Start at " & CStr(dt1) & vbCrLf & _
```

```
         "End at " & CStr(dt2), vbOKOnly + vbInformation, "Success!"
End Sub
```

Listing2. The BestMLR2 subroutine.

Handling Runtime Error in Scheme M1, M2, and M3

The subroutines **BestMLR1**, **BestMLR2**, and **BestMLR3** (which I will introduce later in this article) have an error handler and an error counting scheme. The constant **MAX_ERRORS** represents the initial value for the maximum number of occurring runtime errors. The subroutine assigns the value of this constant to the variable **MaxErrs**. The subroutine also uses the variable **ErrorCounter** to count the number of occurring runtime errors. When the value in variable **ErrorCounter** exceeds that in variable **MaxErrs**, the subroutine displays a message box telling you that the number of occurring runtime errors has reached its maximum limit. This message should not convey a mandatory end to the calculations. The message box has a **Yes** and **No** buttons allowing you to end the program by clicking the **Yes** button. If you click the **No** button, the subroutine displays an input box prompting you to enter a new maximum error limit. The input box shows the current maximum error limit as the default value. You can alter the maximum error limit and/or simply press the **OK** button to accept it and proceed with the statistical calculations. In this case, the subroutine resets the value in **ErrorCounter** to 0. You can also stop the subroutine altogether by clicking the **Cancel** button. Thus, the subroutine offers you two points of exit.

The Scheme M2 Approach for Best Regression Models

The scheme M1 transformation basically relies on raising the variables to different powers (and taking the natural logarithms). As Listing 1 shows, coding the transformations for scheme M1 is easy and compact. The price for this advantage is that scheme M1 comes with some limitations. These limitations include:

1. The inability to dynamically shift and scale particular variables. There are cases where shifting and scaling a particular variable produces better regression models.
2. The transformations use a one-to-one mapping. Scheme 1 cannot create regression variables that are calculated using the expressions (that add, subtract, multiply, and divide) that involve multiple observed variables.

Scheme M2 overcomes the above limitations and supports a highly flexible set of transformations. Basically, scheme M2 replaces the numerically coded transformations of scheme M1, with expression-based transformations. These transformations can create regression variables based on expressions that use any observed variable, Excel-supported functions, and shift/scale values. With scheme M2, the sky is the limit!

Since a picture is worth a thousand words, Figures 3 and 4 show the left and right sides, respectively, of a sample scheme M2 worksheet. Notice the following individual cells and columns of cells that provide input for the regression program:

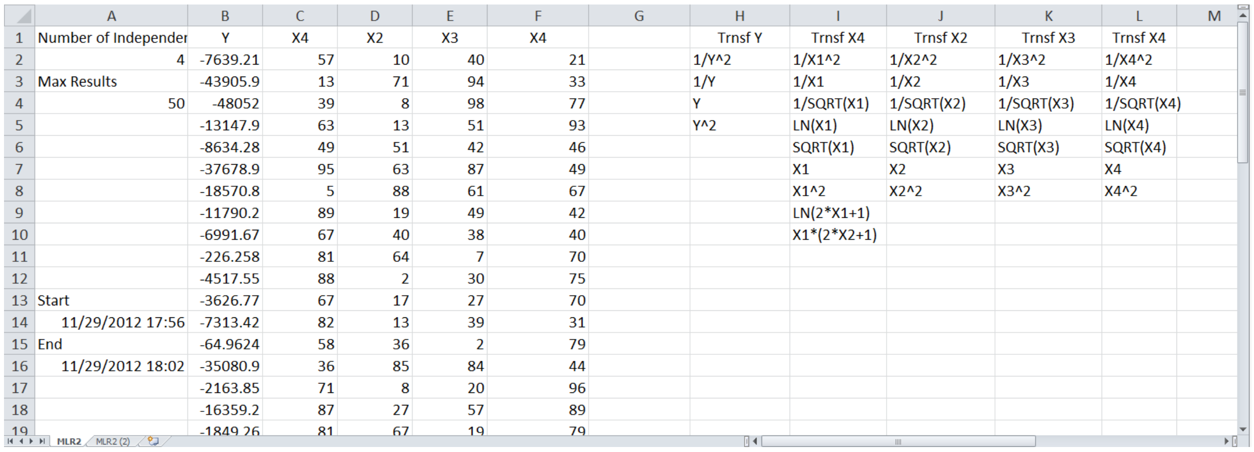
- Cell A2 contains the number of independent variables entering the regression. These regression variables can be the values of observed variables and/or their transformations.
- Cell A4 contains the values for the maximum number of best models. The current value is 50.
- Column B has the values of the observed dependent variable Y, starting in cell B2.
- Column C stores the values of the first observed independent variable, starting at cell C2.
- Columns D and beyond store the values of additional observed independent variables, starting at the columns' second rows.
- The column that comes after the last column of independent variable MUST BE empty. .
- The first column of transformations for variable Y.
- The second column of transformations for variable X1.
- The third column of transformations for variable X2.
- Other columns of transformations for additional variables.
- The column that comes after the last column of transformations MUST BE empty.

Notice that the columns of transformations display expressions for the various observed variables. For example, the transformations for variable X1 in column I are 1/X1^2, 1/X1, 1/SQRT(X1) ,LN(X1), SQRT(X1), X1, and X1^2.

The following worksheet columns provide the regression output generated by running the VBA macro **BestMLR2**. You need to prepare the headers manually:

- The first column shows the values of the F statistic.
- The second column shows the values of the R^2 statistic
- The third column shows the values of the intercepts.
- The fourth column shows the values for the regression slope of X1.
- The fifth column shows the values for the regression slope of X2.
- The next columns show the values for the regression slopes of additional variables.
- The next set of columns show the transformations associated with each model.

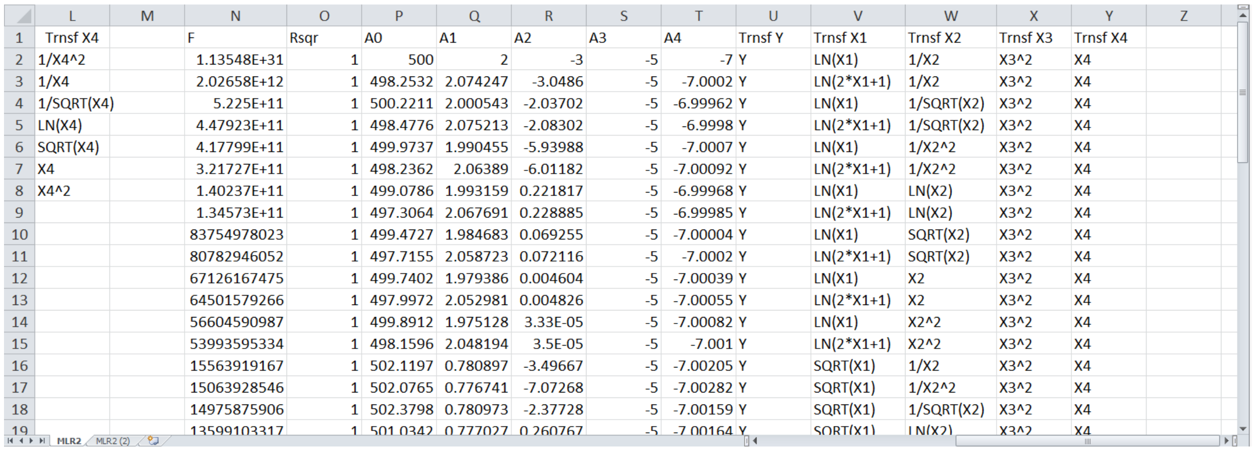
The rows that start with the column for the F statistic represent the record for each model. These rows are sorted in a descending order using the values of the F statistic.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Number of Independer	Y	X4	X2	X3	X4		Trnsf Y	Trnsf X4	Trnsf X2	Trnsf X3	Trnsf X4	
2	4	-7639.21	57	10	40	21		1/Y^2	1/X1^2	1/X2^2	1/X3^2	1/X4^2	
3	Max Results	-43905.9	13	71	94	33		1/Y	1/X1	1/X2	1/X3	1/X4	
4	50	-48052	39	8	98	77		Y	1/SQRT(X1)	1/SQRT(X2)	1/SQRT(X3)	1/SQRT(X4)	
5		-13147.9	63	13	51	93		Y^2	LN(X1)	LN(X2)	LN(X3)	LN(X4)	
6		-8634.28	49	51	42	46			SQRT(X1)	SQRT(X2)	SQRT(X3)	SQRT(X4)	
7		-37678.9	95	63	87	49			X1	X2	X3	X4	
8		-18570.8	5	88	61	67			X1^2	X2^2	X3^2	X4^2	
9		-11790.2	89	19	49	42			LN(2*X1+1)				
10		-6991.67	67	40	38	40			X1*(2*X2+1)				
11		-226.258	81	64	7	70							
12		-4517.55	88	2	30	75							
13	Start	-3626.77	67	17	27	70							
14	11/29/2012 17:56	-7313.42	82	13	39	31							
15	End	-64.9624	58	36	2	79							
16	11/29/2012 18:02	-35080.9	36	85	84	44							
17		-2163.85	71	8	20	96							
18		-16359.2	87	27	57	89							
19		-1849.26	81	67	19	79							

MLR2 MLR2 (2)

Figure 3. The left side of worksheet for scheme M2.



	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	Trnsf X4		F	Rsqr	A0	A1	A2	A3	A4	Trnsf Y	Trnsf X1	Trnsf X2	Trnsf X3	Trnsf X4	
2	1/X4^2		1.13548E+31	1	500	2	-3	-5	-7	Y	LN(X1)	1/X2	X3^2	X4	
3	1/X4		2.02658E+12	1	498.2532	2.074247	-3.0486	-5	-7.0002	Y	LN(2*X1+1)	1/X2	X3^2	X4	
4	1/SQRT(X4)		5.225E+11	1	500.2211	2.000543	-2.03702	-5	-6.99962	Y	LN(X1)	1/SQRT(X2)	X3^2	X4	
5	LN(X4)		4.47923E+11	1	498.4776	2.075213	-2.08302	-5	-6.9998	Y	LN(2*X1+1)	1/SQRT(X2)	X3^2	X4	
6	SQRT(X4)		4.17799E+11	1	499.9737	1.990455	-5.93988	-5	-7.0007	Y	LN(X1)	1/X2^2	X3^2	X4	
7	X4		3.21727E+11	1	498.2362	2.06389	-6.01182	-5	-7.00092	Y	LN(2*X1+1)	1/X2^2	X3^2	X4	
8	X4^2		1.40237E+11	1	499.0786	1.993159	0.221817	-5	-6.99968	Y	LN(X1)	LN(X2)	X3^2	X4	
9			1.34573E+11	1	497.3064	2.067691	0.228885	-5	-6.99985	Y	LN(2*X1+1)	LN(X2)	X3^2	X4	
10			83754978023	1	499.4727	1.984683	0.069255	-5	-7.00004	Y	LN(X1)	SQRT(X2)	X3^2	X4	
11			80782946052	1	497.7155	2.058723	0.072116	-5	-7.0002	Y	LN(2*X1+1)	SQRT(X2)	X3^2	X4	
12			67126167475	1	499.7402	1.979386	0.004604	-5	-7.00039	Y	LN(X1)	X2	X3^2	X4	
13			64501579266	1	497.9972	2.052981	0.004826	-5	-7.00055	Y	LN(2*X1+1)	X2	X3^2	X4	
14			56604590987	1	499.8912	1.975128	3.33E-05	-5	-7.00082	Y	LN(X1)	X2^2	X3^2	X4	
15			53993595334	1	498.1596	2.048194	3.5E-05	-5	-7.001	Y	LN(2*X1+1)	X2^2	X3^2	X4	
16			15563919167	1	502.1197	0.780897	-3.49667	-5	-7.00205	Y	SQRT(X1)	1/X2	X3^2	X4	
17			15063928546	1	502.0765	0.776741	-7.07268	-5	-7.00282	Y	SQRT(X1)	1/X2^2	X3^2	X4	
18			14975875906	1	502.3798	0.780973	-2.37728	-5	-7.00159	Y	SQRT(X1)	1/SQRT(X2)	X3^2	X4	
19			13599103317	1	501.0342	0.777027	0.260767	-5	-7.00164	Y	SQRT(X1)	LN(X2)	X3^2	X4	

MLR2 MLR2 (2)

Figure 4. The right side of worksheet for scheme M2.

Looking at the sample results in Figure 4, the best model is:

$Y = 500 + 2 \ln(X1) – 3/X2 – 5 X2^2 – 7 X4$

This model has the F statistic of 1.13548E+31 and the R^2 value of 1. Notice that there is quite the difference in the values of the F statistic between the best model and the second best model. It is also interesting to see that the first few models in the Hit Parade list have different transformations for variables X1 and X2.

The BestMLR2 VBA Subroutine

Listing 2 shows the source code for subroutine **BestMLR2**. This subroutine performs the following general tasks:

- Read the input from the leading columns. This includes the observations for the different variables and the lists of transformations.
- Establishes the first set of transformations.

- Transform the observed variables into the regression variables using the current transformation expressions.
- Invoke the **Linest** function and store its results in variable **vRegResultsMat**. The subroutine compares the value of the F statistic with the lowest value of F in the Hit Parade list. If the newly calculated F statistic is greater than that lowest F value, the subroutine overwrites the last Hit Parade row with newly calculated data and then invokes Excel's Sort method to quickly re-sort the Hit Parade list using values of the F statistics. Sorting the data maintains the order for the best regression model combination of transformations.
- Use the **For VarIdx** loop to simulate nested loops that systematically change the transformations for the variables.

The subroutine use the status bar to keep you informed of the calculations progress. The subroutine displays a message box when done. This message shows the starting and ending time for the calculations. These two values also appear in column A.

```
Option Explicit
Option Base 1

Function ExFx(ByVal sFx As String, ByRef X() As Variant, _
              ByRef Y() As Variant, _
              ByVal I As Integer, ByVal NumIVars As Integer) As Double
 Dim J As Integer
 ' replace Xnn starting with the higher indices just in case there
 ' are more than 9 variables.
 sFx = UCase(sFx)
 For J = NumIVars To 1 Step -1
   sFx = Replace(sFx, "X" & J, "(" & X(I, J) & ")")
 Next J
 sFx = Replace(sFx, "Y", "(" & Y(I, 1) & ")")
 ExFx = Evaluate(sFx)
End Function


Sub BestMLR2()

  Const MAX_ERRORS As Double = 1000000# ' initial max error value

  Dim ErrorCounter As Double, MaxErrors As Double, sMaxErr As String
  Dim NumIndpVars As Integer ' number of independent variables
  Dim TN As Integer ' total number of variables = NIV+1
  Dim N As Integer ' number of data points
  Dim MaxTrans As Integer ' max transformations
  Dim MaxResults As Integer ' max results
  Dim Col1 As Integer, Col2 As Integer, Col3 As Integer
  Dim Col4 As Integer, Col5 As Integer
  Dim I As Integer, J As Integer, K As Integer
  Dim M1 As Integer, M As Integer
  Dim VarIdx As Integer, Low As Integer, Hi As Integer
  Dim TransfMat() As String, sFx As String
  Dim CurrentTransf() As String, CountTransf() As Integer
  Dim NumTransf() As Integer ' number of transformations
  Dim Y() As Variant, X() As Variant
  Dim Yt() As Variant, Xt() As Variant
  Dim vRegResultsMat As Variant
  Dim F As Double, Rsqr As Double, xval As Double
  Dim fMaxCount As Double, fCount As Double, fMilestone As Double
  Dim dt1 As Date, dt2 As Date

  dt1 = Now
  ErrorCounter = 0
  MaxErrors = MAX_ERRORS
  NumIndpVars = [A2].Value
  MaxResults = [A4].Value
  TN = NumIndpVars + 1
  Col1 = 2              ' first column of data
  Col2 = Col1 + TN + 1  ' first column of transformations
  Col3 = Col2 + TN + 1  ' first column of results
  Col4 = Col3 + TN + 1  ' first column of tranformatioons
  Col5 = Col4 + TN - 1  ' last column of transformations

  Range(Cells(2, Col3), Cells(1 + 2 * MaxResults, Col3 + 3 *
TN)).Value = ""
  Range(Cells(2, Col3), Cells(1 + MaxResults, Col3 + 1 + TN)).Value =
0
  MaxTrans = Range(Cells(2, Col2), Cells(1,
Col2)).CurrentRegion.Rows.Count - 1
  ReDim NumTransf(TN), TransfMat(TN, MaxTrans), CurrentTransf(TN)
  ReDim CountTransf(TN)

  fMaxCount = 1
  For I = 1 To TN
    J = 2
    Do While Trim(Cells(J, Col2 + I - 1)) <> ""
      TransfMat(I, J - 1) = Cells(J, Col2 + I - 1)
      J = J + 1
    Loop
    NumTransf(I) = J - 2
    fMaxCount = fMaxCount * NumTransf(I)
  Next I

  N = Range("B1").CurrentRegion.Rows.Count - 1
  Y = Range("B2:B" & N + 1).Value
  X = Range(Cells(2, 3), Cells(N + 1, 2 + NumIndpVars)).Value
  Yt = Range("B2:B" & N + 1).Value
  Xt = Range(Cells(2, 3), Cells(N + 1, 2 + NumIndpVars)).Value

  ' set the initial transformations
  For I = 1 To TN
    CurrentTransf(I) = TransfMat(I, 1)
    CountTransf(I) = 1
  Next I

  fCount = 0
  fMilestone = 0.1
  Do

    On Error GoTo HandleErr

    For I = 1 To N
      DoEvents

      If fCount / fMaxCount > fMilestone Then
        DoEvents
        Application.StatusBar = "Processed " & CStr(fMilestone * 100)
& " %"
        If fMilestone < 1 Then fMilestone = fMilestone + 0.05
      End If


      ' sFx = CurrentTransf(1) '[A5].Value
      Yt(I, 1) = ExFx(CurrentTransf(1), X, Y, I, NumIndpVars)

      For J = 1 To NumIndpVars
        'sFx = CurrentTransf(J + 1)  ' Range("A" & M).Value
        Xt(I, J) = ExFx(CurrentTransf(J + 1), X, Y, I, NumIndpVars)
      Next J
    Next I

    ' perform the regression calculations
    vRegResultsMat = Application.WorksheetFunction.LinEst(Yt, Xt,
True, True)

    Rsqr = vRegResultsMat(3, 1)
    F = vRegResultsMat(4, 1)
    ' check if F > F of last result
    If F > Cells(MaxResults + 1, Col3) Then
      xval = fCount / fMaxCount * 100
      xval = CInt(100 * xval) / 100
      Application.StatusBar = "Processed " & CStr(xval) & " %"
      M1 = MaxResults + 1
      ' write new results to row M
      Cells(M1, Col3) = F
      Cells(M1, Col3 + 1) = Rsqr
      For I = 1 To TN
        Cells(M1, Col3 + I + 1) = vRegResultsMat(1, TN - I + 1)
      Next I
      For I = 1 To TN
        Cells(M1, Col4 + I) = CurrentTransf(I)
      Next I

      Range(Cells(2, Col3), Cells(MaxResults + 1, Col5 + 1)).Select
      Range(Cells(2, Col3), Cells(MaxResults + 1, Col5 + 1)).Sort
Key1:=Range(Cells(2, Col3), Cells(MaxResults + 1, Col3)),
Order1:=xlDescending

    End If ' If F > Cells(MaxResults + 1, Col3)

    GoTo Here

HandleErr:
  fCount = fCount - 1
  ErrorCounter = ErrorCounter + 1
  If ErrorCounter > MaxErrors Then
    If MsgBox("Reached maximum error limits of " & ErrorCounter &
vbCrLf & _
    "Want to stop the process?", vbYesNo + vbQuestion, "Confirmation
requested") = vbYes Then
      Exit Sub
    Else
      sMaxErr = InputBox("Update maximum number of errors", "Max
Errors Input", MaxErrors)
      If Trim(sMaxErr) = "" Then
        MsgBox "User canceled calculations process", vbOKOnly +
vbInformation, "End of Process"
        Exit Sub
      End If
      MaxErrors = CDbl(sMaxErr)
      ErrorCounter = 0
    End If
  End If
  Resume Here
Here:

    ' ---------------------------------------------------------
    ' ---------------------------------------------------------
    '  ------------ Simulate Nested Loops ---------------------
    ' ---------------------------------------------------------
    ' ---------------------------------------------------------

    For VarIdx = 1 To TN
      DoEvents
      If CountTransf(VarIdx) >= NumTransf(VarIdx) Then
        If VarIdx < TN Then
          CurrentTransf(VarIdx) = TransfMat(VarIdx, 1)
          CountTransf(VarIdx) = 1
        Else
          Exit Do
        End If
      Else
        CountTransf(VarIdx) = CountTransf(VarIdx) + 1
        CurrentTransf(VarIdx) = TransfMat(VarIdx, CountTransf(VarIdx))
        fCount = fCount + 1
        Exit For
      End If
    Next VarIdx

  Loop

  On Error GoTo 0
  dt2 = Now
  [A13].Value = "Start"
  [A14].Value = dt1
  [A15].Value = "End"
  [A16].Value = dt2
  Application.StatusBar = "Done"
  MsgBox "Start at " & CStr(dt1) & vbCrLf & _
         "End at " & CStr(dt2), vbOKOnly + vbInformation, "Success!"
End Sub
```

Listing 2. The BestMLR2 subroutine.

The Scheme M3 Approach for Best Regression Models

Schemes M1 and M2 process a large number of regression models created by applying a combination of transformations to the observed variables. These schemes generally search for the best *empirical* regression model. Scheme M3 differs from schemes M1 and M2 by allowing you to focus on your custom collection of models. You can specify any number of highly customized regression models by using custom transformations for each regression variable. Scheme 3 does not apply a combination of transformations as do schemes M1 and M2. Instead, each row of transformations represents the transformations that collectively define a single regression model.

Figures 5 and 6 show the left and right sides, respectively, of a sample scheme M3 worksheet. Notice the following individual cells and columns of cells that provide input for the regression program:

- Cell A2 contains the number of independent variables entering the regression. These regression variables can be the values of observed variables and/or their transformations.
- Cell A4 contains the values for the maximum number of best models. The current value is 6.
- Column B has the values of the observed dependent variable Y, starting in cell B2.
- Column C stores the values of the first observed independent variable, starting at cell C2.
- Columns D and beyond store the values of additional observed independent variables, starting at the columns' second rows.
- The column that comes after the last column of independent variable MUST BE empty.
- The first column of transformations for variable Y.
- The second column of transformations for variable X1.
- The third column of transformations for variable X2.
- Other columns of transformations for additional variables.
- The column that comes after the last column of transformations MUST BE empty.

Notice that each row of transformations specifies a single regression model. For example, the first row which contains the transformation expressions Y, X1, 1/X2, X3^2, and X4 represent the following model:

$Y = a_0 + a_1 X1 + a_2/X2 + a_3 X3\text{^}2 + a_4 X4$

The second row which contains the transformation expressions Y, LN(X1), 1/X2, X3^2, and X4 represent the following model:

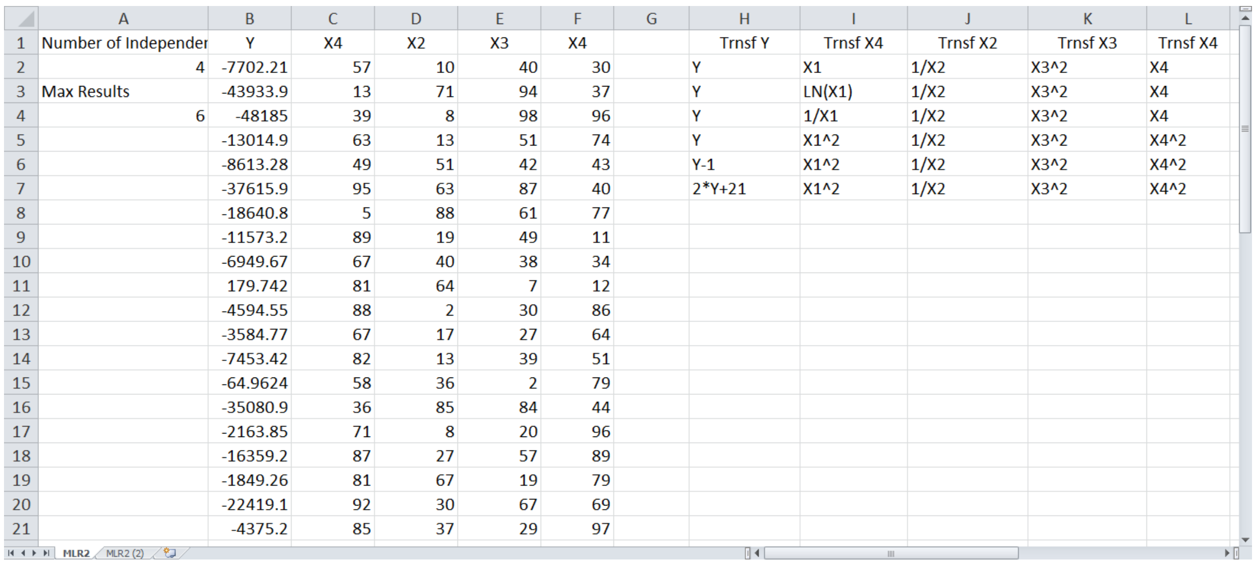
$Y = a_0 + a_1 \ln(X1) + a_2/X2 + a_3 X3\text{^}2 + a_4 X4$

And so on. Figure 5 shows a total of six regression models.

The following worksheet columns provide the regression output generated by running the VBA macro **BestMLR3**. You need to prepare the headers manually:

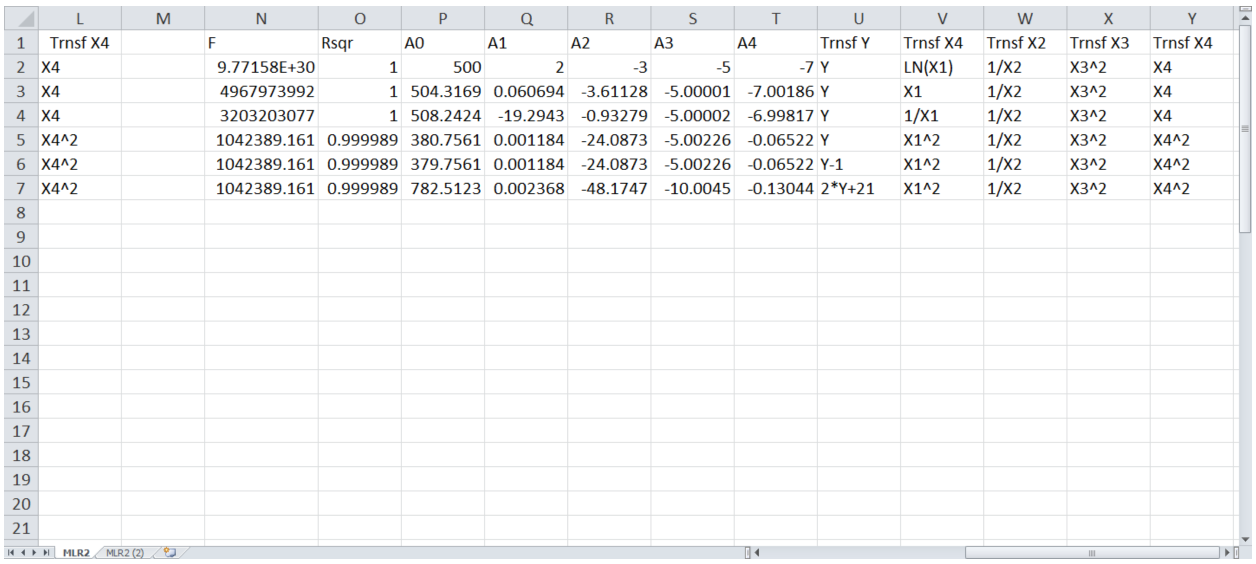
- The first column shows the values of the F statistic.
- The second column shows the values of the R^2 statistic
- The third column shows the values of the intercepts.
- The fourth column shows the values for the regression slope of X1.
- The fifth column shows the values for the regression slope of X2.
- The next columns show the values for the regression slopes of additional variables.
- The next set of columns show the transformations associated with each model.

The rows that start with the column for the F statistic represent the record for each model. These rows are sorted in a descending order using the values of the F statistic.



	A	B	C	D	E	F	G	H	I	J	K	L
1	Number of Independer	Y	X4	X2	X3	X4		Trnsf Y	Trnsf X4	Trnsf X2	Trnsf X3	Trnsf X4
2	4	-7702.21	57	10	40	30		Y	X1	1/X2	X3^2	X4
3	Max Results	-43933.9	13	71	94	37		Y	LN(X1)	1/X2	X3^2	X4
4	6	-48185	39	8	98	96		Y	1/X1	1/X2	X3^2	X4
5		-13014.9	63	13	51	74		Y	X1^2	1/X2	X3^2	X4^2
6		-8613.28	49	51	42	43		Y-1	X1^2	1/X2	X3^2	X4^2
7		-37615.9	95	63	87	40		2*Y+21	X1^2	1/X2	X3^2	X4^2
8		-18640.8	5	88	61	77						
9		-11573.2	89	19	49	11						
10		-6949.67	67	40	38	34						
11		179.742	81	64	7	12						
12		-4594.55	88	2	30	86						
13		-3584.77	67	17	27	64						
14		-7453.42	82	13	39	51						
15		-64.9624	58	36	2	79						
16		-35080.9	36	85	84	44						
17		-2163.85	71	8	20	96						
18		-16359.2	87	27	57	89						
19		-1849.26	81	67	19	79						
20		-22419.1	92	30	67	69						
21		-4375.2	85	37	29	97						

Figure 5. The left side of worksheet for scheme M3.



	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	Trnsf X4		F	Rsqr	A0	A1	A2	A3	A4	Trnsf Y	Trnsf X4	Trnsf X2	Trnsf X3	Trnsf X4
2	X4		9.77158E+30	1	500	2	-3	-5	-7	Y	LN(X1)	1/X2	X3^2	X4
3	X4		4967973992	1	504.3169	0.060694	-3.61128	-5.00001	-7.00186	Y	X1	1/X2	X3^2	X4
4	X4		3203203077	1	508.2424	-19.2943	-0.93279	-5.00002	-6.99817	Y	1/X1	1/X2	X3^2	X4
5	X4^2		1042389.161	0.999989	380.7561	0.001184	-24.0873	-5.00226	-0.06522	Y	X1^2	1/X2	X3^2	X4^2
6	X4^2		1042389.161	0.999989	379.7561	0.001184	-24.0873	-5.00226	-0.06522	Y-1	X1^2	1/X2	X3^2	X4^2
7	X4^2		1042389.161	0.999989	782.5123	0.002368	-48.1747	-10.0045	-0.13044	2*Y+21	X1^2	1/X2	X3^2	X4^2
8														
9														
10														
11														
12														
13														
14														
15														
16														
17														
18														
19														
20														
21														

Figure 6. The right side of worksheet for scheme M3.

Looking at the sample results in Figure 4, the best model is:

Y = 500 + 2 ln(X1) – 3/X2 – 5X2^2 – 7 X4

This model has the F statistic of 9.77158E+30 and the R^2 value of 1.

The BestMLR3 VBA Subroutine

Listing 3 shows the source code for subroutine **BestMLR3**. This subroutine performs the following general tasks:

- Read the input from the leading columns. This includes the observations for the different variables and the lists of numerically coded transformations.
- Repeat the next tasks for each row of transformations.
- Read the next row of transformation to collectively specify a regression model.
- Calculate the values of the transformed variables.
- Invoke the **Linest** function and store its results in variable **vRegResultsMat**. The subroutine compares the value of the F statistic with the lowest value of F in the Hit Parade list. If the newly calculated F statistic is greater than that lowest F value, the subroutine overwrites the last Hit Parade row with newly calculated data and then invokes Excel's Sort method to quickly re-sort the Hit Parade list using values of the F statistics.

```
Option Explicit
Option Base 1

Function ExFx(ByVal sFx As String, ByRef X() As Variant, _
              ByRef Y() As Variant, _
              ByVal I As Integer, ByVal NumIVars As Integer) As Double
 Dim J As Integer
 ' replace Xnn starting with the higher indices just in case there
 ' are more than 9 variables.
 sFx = UCase(sFx)
 For J = NumIVars To 1 Step -1
   sFx = Replace(sFx, "X" & J, "(" & X(I, J) & ")")
 Next J
 sFx = Replace(sFx, "Y", "(" & Y(I, 1) & ")")
 ExFx = Evaluate(sFx)
End Function


Sub BestMLR3()

  Const MAX_ERRORS As Double = 1000000# ' initial max error value
  Dim ErrorCounter As Double, MaxErrors As Double, sMaxErr As String
  Dim NumIndpVars As Integer ' number of independent variables
  Dim TN As Integer ' total number of variables = NumIndpVars +1
  Dim N As Integer ' number of data points
  Dim MaxTrans As Integer ' max transformations
  Dim MaxResults As Integer ' max results
  Dim iTransforms As Integer
  Dim Col1 As Integer, Col2 As Integer, Col3 As Integer
  Dim Col4 As Integer, Col5 As Integer
  Dim I As Integer, J As Integer, K As Integer
  Dim M1 As Integer, M As Integer
  Dim VarIdx As Integer, Low As Integer, Hi As Integer
  Dim TransfMat() As String, sFx As String
  Dim CurrentTransf() As String, CountTransf() As Integer
  Dim Y() As Variant, X() As Variant
  Dim Yt() As Variant, Xt() As Variant
  Dim vRegResultsMat As Variant
  Dim F As Double, Rsqr As Double, xval As Double

  ErrorCounter = 0
  MaxErrors = MAX_ERRORS
  NumIndpVars = [A2].Value
  MaxResults = [A4].Value
  TN = NumIndpVars + 1
  Col1 = 2              ' first column of data
  Col2 = Col1 + TN + 1  ' first column of transformations
  Col3 = Col2 + TN + 1  ' first column of results
  Col4 = Col3 + TN + 1  ' first column of tranformatioons
  Col5 = Col4 + TN - 1  ' last column of transformations

  Range(Cells(2, Col3), Cells(1 + 2 * MaxResults, Col3 + 3 *
TN)).Value = ""
  Range(Cells(2, Col3), Cells(1 + MaxResults, Col3 + 1 + TN)).Value =
0
  MaxTrans = Range(Cells(2, Col2), Cells(1,
Col2)).CurrentRegion.Rows.Count - 1
  ReDim TransfMat(TN, MaxTrans), CurrentTransf(TN)
  ReDim CountTransf(TN)

  For I = 1 To TN
    J = 2
    Do While Trim(Cells(J, Col2 + I - 1)) <> ""
      TransfMat(I, J - 1) = Cells(J, Col2 + I - 1)
      J = J + 1
    Loop
  Next I

  N = Range("B1").CurrentRegion.Rows.Count - 1
  Y = Range("B2:B" & N + 1).Value
  X = Range(Cells(2, 3), Cells(N + 1, 2 + NumIndpVars)).Value
  Yt = Range("B2:B" & N + 1).Value
  Xt = Range(Cells(2, 3), Cells(N + 1, 2 + NumIndpVars)).Value

  For iTransforms = 1 To MaxTrans

    On Error GoTo HandleErr

    For I = 1 To N
      DoEvents

      ' sFx = CurrentTransf(1) '[A5].Value
      Yt(I, 1) = ExFx(TransfMat(1, iTransforms), X, Y, I, NumIndpVars)

      For J = 1 To NumIndpVars
        'sFx = CurrentTransf(J + 1)  ' Range("A" & M).Value
        Xt(I, J) = ExFx(TransfMat(J + 1, iTransforms), X, Y, I,
NumIndpVars)
      Next J

    Next I

    ' perform the regression calculations
    vRegResultsMat = Application.WorksheetFunction.LinEst(Yt, Xt,
True, True)

    Rsqr = vRegResultsMat(3, 1)
    F = vRegResultsMat(4, 1)
    ' check if F > F of last result
    If F > Cells(MaxResults + 1, Col3) Then
      M1 = MaxResults + 1
      ' write new results to row M
      Cells(M1, Col3) = F
      Cells(M1, Col3 + 1) = Rsqr
      For I = 1 To TN
        Cells(M1, Col3 + I + 1) = vRegResultsMat(1, TN - I + 1)
      Next I
      For I = 1 To TN
        Cells(M1, Col4 + I) = TransfMat(I, iTransforms)
      Next I
      Range(Cells(2, Col3), Cells(MaxResults + 1, Col5 + 1)).Select
      Range(Cells(2, Col3), Cells(MaxResults + 1, Col5 + 1)).Sort
Key1:=Range(Cells(2, Col3), Cells(MaxResults + 1, Col3)),
Order1:=xlDescending

    End If ' If F > Cells(MaxResults + 1, Col3)

    GoTo Here

HandleErr:
  ErrorCounter = ErrorCounter + 1
  If ErrorCounter > MaxErrors Then
    If MsgBox("Reached maximum error limits of " & ErrorCounter &
vbCrLf & _
    "Want to stop the process?", vbYesNo + vbQuestion, "Confirmation
requested") = vbYes Then
      Exit Sub
    Else
      sMaxErr = InputBox("Update maximum number of errors", "Max
Errors Input", MaxErrors)
      If Trim(sMaxErr) = "" Then
        MsgBox "User canceled calculations process", vbOKOnly +
vbInformation, "End of Process"
        Exit Sub
      End If
      MaxErrors = CDbl(sMaxErr)
      ErrorCounter = 0
    End If
  End If
  Resume Here
Here:

  Next iTransforms

  On Error GoTo 0
  MsgBox "Done", vbOKOnly + vbInformation, "Success!"
End Sub
```

Listing 3. The BestMLR3 subroutine.

References

1. Shammas, Namir, *Multiple Regression Model Using Excel Linest Function, article on www.namirshammas.com web site.*
2. Shammas, Namir, *Best Polynomial Model Using Excel Linest Function, article on www.namirshammas.com web site.*

3. Wikipedia article *Coefficient of Determination*.
4. Wikipedia article *Linear Regression*.
5. Wikipedia article *Simple Linear Regression*.
6. Wikipedia article *Akaike information criterion*.
7. Draper and Smith, *Applied Regression Analysis*, Wiley-Interscience; 3rd edition (April 23, 1998).
8. Neter, Kuther, Wasserman, and Nachtsheim, *Applied Linear Statistical Models*, McGraw-Hill/Irwin; 4th edition (February 1, 1996).
9. Fox, *Applied Regression Analysis and Generalized Linear Models*, Sage Publications, Inc; 2nd edition (April 16, 2008).
10. Montgomery, Peck, and Vining, *Introduction to Linear Regression Analysis*, Wiley-Interscience; 4th edition (2006).
11. Seber and Lee, *Linear Regression Analysis*, Wiley; 2nd edition (February 5, 2003).

About this Article

Version	*Date*	*Comment*
1.0.0.0	12/7/2012	Initial release.