TaylorFit

STEPWISE RESPONSE SURFACE ANALYSIS (S-RSA)

and MULTIVARIATE POLYNOMIAL REGRESSION (MPR)

Simetrica, LLC
David A. Vaccari

www.simetrica-llc.com
info@simetrica-llc.com
www.TaylorFit-RSA.com

Contents

Acknowledgements

The following people helped with the development of TaylorFit:

- Xiaoguang (Steve) Wang
- David Klappholz
- Christopher Kelley
- Patrick Grasso
- Eric Fitzpatrick
- Julia Kim
- Sarath Jagupilla

What is TaylorFit?

TaylorFit is a regression program that makes it easy to generate relatively simple equations to describe complex data. By "complex," we mean data that have nonlinear relationships, including interactions among multiple variables. This even extends to describing processes that have the behavior known as "chaotic."

Examples of situations that can be modeled with TaylorFit can be found everywhere – in business, engineering, and the sciences. One simple way to think of its applicability is to envision any situation with numerical information that can be collected into a Table or spreadsheet with two or more columns of data, and with one column expected to depend upon the values in the other columns. Each row can be considered a distinct set of measurements repeated to form the set of rows. If the data in the rows were collected at fixed time intervals (e.g. daily or annual data), then the dataset is called a *time-series*. Some specific examples of datasets are:

- A company has a set of retail outlets. For each, they collected a dozen statistics that they think can predict the profitability. The statistics include each store's local market share, how long it has been there, how many years of experience the sales people have, etc. The spreadsheet has data for each individual store in one row; the profitability of the store is in the first column, and the other statistics in other columns. They wish to predict the profitability in column one using the statistics in the other columns.

- Engineers would like a formula to predict the strength of concrete based on the recipe used to make it. They collect data on strength together with the pounds of water added per pound of cement, the amount of sand added, the amount of coarse aggregate, as well as the amount of several other additives.

- Agricultural scientists need to understand what affects the yield of a specific crop in their area. For example, they may have data from many farmers on the number of bushels of corn they produced per acre. Each farmer used different amounts of irrigation and fertilizer. The spreadsheet will have yield in one column, and irrigation and fertilizer amounts in others.

- Social scientists seek to develop a model to predict how an individual's income in a specific age bracket depends on a variety of socio-demographic factors. They collect income data, together with information on education level, the education level of individual's parents, the income of the parents, etc.

- *Time-series example*: A company has data on annual sales and advertising for a period of 54 years. How do sales in a particular year depend upon the amount spent on advertising in that year? Also, do sales in one year depend upon sales and advertising for previous years?

TaylorFit Modeling approach

The type of models TaylorFit generates are called Multivariate Polynomial Regression Models. These are extensions of the basic polynomial equations that are familiar from basic algebra, extended to multivariable data.

Consider the example above of agricultural crop yield (Y), which depends upon the amount of irrigation water (W) and fertilizer application (F). One of the best known approaches for this problem is Multilinear Regression (MLR). TaylorFit is an extension of MLR to include interactions and other nonlinearity. The MLR model for this case would be:

$$Y = a + b \cdot W + c \cdot F$$

The model has coefficients *a, b,* and *c*. The value of *b* indicates how much Y increases as W is increased. However, what if the effect of irrigation on yield depends upon how much fertilizer has been applied? This is an example of an *interaction*, and a MLR model cannot describe it. However, if we add a term $d \cdot W \cdot F$, the resulting model is capable of describing interactions:

$$Y = a + b \cdot W + c \cdot F + d \cdot W \cdot F$$

A further problem may be that yield increases with W and with F, but only up to a point. Then it saturates, or levels out. Plus, how fast yield saturates with irrigation may interact with fertilizer. The following model describes such a situation.

$$Y = 0.5 + 1.0 \cdot W + 1.0 \cdot W^2 + 0.2 \cdot F^2 - 3.5 \cdot W^2 \cdot F$$

If this looks complicated, consider that one should let the data determine the form and complexity of the model, not the other way around. If your data have such behaviors and you do not include them, then your model is biased. TaylorFit gets its name from the Taylor Theorem, which proves that any functional relationship can be described by a polynomial series equation. The models developed by TaylorFit are essentially truncated Taylor polynomials, which are capable of fitting data to an arbitrary degree of accuracy.

The previous equation is an example of a Multivariate Polynomial Regression (MPR) model. Things could be even more complicated. What if the behavior depended upon the *ratio* of two variables? This is a common situation. MLR cannot describe any such behaviors. Including negative exponents enables TaylorFit to test for the fit of ratios. This can make models more compact. For example, we could fit models that look like this:

$$Y = 0.5 + 1.0 \cdot W + 1.0 \cdot W^2 - 3.5 \cdot W^2 \cdot F + 2.5 \cdot W \cdot F^{-1} - 0.5 \cdot W^{-2} \cdot F$$

The problem with MPR models is that the number of possible terms may become very large, and it can be difficult to select which are important and which are not. TaylorFit solves this problem by using stepwise regression. For more details explaining what MPR is, see the MPR Primer at http://www.simetrica-llc.com/Products/MPR/modeling.html.

How TaylorFit Works

TaylorFit uses a stepwise algorithm to select the best terms to include in the model, while keeping the model small by excluding terms that do not improve the fit. Fitting is done by least squares using the Singular Value Decomposition (SVD) method, which prevents roundoff problems (caused by multicollinearity) often found in multiple regression.

The user selects exponents that the program can use in generating candidate polynomial terms, as well as the possible number of multiplicands in each term. If the model to be fitted is a time series, the user can input the lags to be used in generating terms. These parameters can all be changed during the fitting process. Candidate terms are picked one-at-a-time to be added to the current model if the added term improves the model. This allows the user to start with simpler models, and gradually increase model complexity until the fit can no longer be improved.

Goodness-of-Fit statistics

TaylorFit computes a number of statistics used for building and evaluating the model. These include the following statistics associated with individual terms, either candidate terms or terms in the current model: Coefficient, *t*-statistic, and *p*(*t*). There are also a number of *Data* statistics: N_d, N_p, N_f, *TSS*. Finally, there are the *Goodness-of-Fit* (*GoF*) statistics based on the overall fit of the model to the data. Any of the *GoF* statistics may be used as fitting criteria. These include:

SSE:	Sum of squares of the errors (or residuals)
MSE:	Mean square error; this is an estimate of the error standard deviation
RSQ:	R^2, the square of the multiple correlation coefficient
Adj-*RSQ*:	Adjusted R^2 – similar to R^2 , but takes into account the degrees of freedom
F:	The *F*-statistic; similar to the *t*-statistic for a term, this applies to the overall fit.
p(*F*):	The probability of *F* being this large or larger by chance.
AIC:	Akaike Information Criterion
BIC:	Bayesian Information Criterion
Max|*Err*|:	The maximum absolute value of the error

Fitting parameters

The user begins by defining the parameters of the fit. These parameters include the set of exponents {*exp*}, the maximum number of multiplicands in a single term (n_m), and (if the dataset is a time-series, the set of lags to consider {*lag*}. An intercept term is always considered, although the modeler does not have to select it into the current model.

For example , in the simplest case, if {*exp*}= {*1*} and $n_m = 1$, then TaylorFit is will only create a MLR model. If {*exp*}= {*1,2,3*} and $n_m = 1$, then univariate polynomial terms up to degree 3 are considered. If {*exp*}= {*1,2,3*} and $n_m = 2$, then multivariate polynomial terms such as (from the example) $W*F^2$ or W^3*F^2 are also considered. Once the set of exponents and lags and the maximum number of multiplicands are selected, TaylorFit generates all the possible terms and tests all the models consisting of the current model with each of the candidate terms not in the current model, one-at-a-time. It also tests the current model with each current term removed, one-at-a-time.

Stepwise regression

Stepwise regression consists of alternated addition and removal steps. In the **addition step**, candidate terms are tested to see if they improve the model, and sorted based on the selected goodness-of-fit criterion. The user may then choose to add one of the terms into the current model; usually one that improves the fitting criteria or that has a $p(t)$ that meets the selection criteria (usually being less than 5%). In the **removal step**, the terms in the current model are tested one-at-a-time to see if the criterion can be improved by removing any single term. (Sometimes, a term that was significant early in the modeling process becomes insignificant at a later point and should be removed.) The addition and removal steps are alternated repeatedly until the model criterion cannot be improved by the addition or removal of any single term. At this point, the modeler may choose to extend the search by adding to the list of exponents, lags, or to increase n_m. Or, if the model is satisfactory, the user can stop at this point.

Data are expensive, calculations are cheap.

Data Files

TaylorFit allows all or part of an input data file to be used for fitting. It also allows data files to be split into three portions, termed the "Fit", the "Test", and the “Validation” datasets.

The *FIT* dataset is used to compute statistics associated with individual terms, such as the term’s coefficient, its *t*-statistic, and its $p(t)$.

The *TEST* dataset is used for a technique called ***cross-validation***. In cross-validation, the model coefficients are still computed using the *FIT* dataset, but terms are chosen to add to the current model only if they improve the selected global model criterion for the *TEST* dataset, and the $p(t)$ for the chosen term meets the chosen (usually 5%) significance level requirement. Cross validation is optional, but is recommended whenever enough data are available. It helps eliminate problems such as overfitting or spurious correlation (discussed below).

The *VALIDATE* dataset should be used only after the best model has been produced using the *FIT* and *TEST* datasets and the model is intended for use in making predictions. Then, the model is used to compute the global statistics using the *VALIDATE* dataset. If the resulting statistics show the model is good, then the modeler’s work is done. If not, the temptation to add or remove further terms should be avoided, unless additional data will be collected for further validation. Final validation should never be done using the same dataset as was used to generate the model.

In some cases the model is not intended for use in making predictions outside the dataset used to generate the model. For example, it may only be intended to discover correlations; i.e. to identify trends or functional relationships among variables. (In other words, to answer the question, “how does *Y* depend on *X*, *Z*, etc. in these data?) In such a case, validation is not necessary. It would still be a good idea to use cross-validation, if possible.

Output files generated by TaylorFit include a model description file and a file containing model predictions based on data file inputs.

Comparison with other modeling methods

There are many other ways to model data such as described above. Here are several that are most similar in their applicability, with a comparison.

Multilinear Regression (MLR)

As described above, MLR is a widely understood methodology. Since TaylorFit stepwise response surface methodology (S-RSA) is a natural extension of MLR, it is a good fit for researchers who have used MLR. By changing to S-RSA, they will gain the following advantages:

More accurate models
Reduced bias
Ability to describe complex behaviors such as interactions and other kinds of nonlinearity

Data are expensive, calculations are cheap. Since there is a large body of research that has been analyzed using MLR, we suggest that anyone who has used MLR in the past should consider re-analyzing their data using TaylorFit, to try to glean new knowledge from those data.

Artificial Neural Networks (ANNs)

This is a powerful and increasingly popular modeling technique that is often described as a form of "artificial intelligence." It can be thought of as having layers of functions as complex as MPR models between the multiple inputs (independent variables) and the output (the dependent variable).

ANNs typically have complex structures with a very large number of adjustable coefficients. This makes them susceptible to overfitting (also known as "memorizing the noise"). It also makes ANN models difficult to communicate to others. You can't write it down as a closed-form equation, like you can with a multivariate polynomial. It is common for researchers who produce ANN models to describe how well it worked. But they can't print the form of the model for others to use. Because the ANN model cannot be easily expressed in mathematical form, it is difficult to perform standard kinds of analysis, such as computing derivatives for sensitivity analysis. You cannot provide the model to someone so they could compute it in the cell of a spreadsheet. Essentially, ANN models can provide predictions, but little other information about the behavior of the system being modeled. For these reasons, ANNs may be called "the blackest of the black box models."

> Anyone who has used Multilinear Regression in the past should consider re-analyzing their data using TaylorFit

Another characteristic of ANNs is that they may require a large number of iterative computations to obtain reasonable estimates of the model coefficients. ANNs iterate and converge more and more slowly, and have to be stopped by an arbitrary criterion. Since, as mentioned above, calculations are cheap, this is not a very serious problem. In fact, the computational intensity of ANNs is part of what inspired the development of TaylorFit. Once users were liberated from the constraint to keep the number of computations small, the large number of computations that may occur in TaylorFit were not perceived as a problem. Despite

this, TaylorFit tends to be more computationally fast than ANNs, and it converges without an arbitrary stopping criterion.

To summarize the comparison between TaylorFit and ANNs:

- TaylorFit converges absolutely. That is, iterations ultimately reach a conclusion without an arbitrary stopping criterion.
- The TaylorFit S-RSA models are easy to manipulate. That is, they are relatively simple functional relationships (like MLR models), and so are easy to communicate and use in other programs such as spreadsheets, or to analyze, such as by differentiation or by graphical means.
- TaylorFit models are much more resistant to over-fitting (especially if cross-validation is used), since every term in the model is forced to be statistically significant. It's common in ANNs for many of its coefficients to not be significant.
- They are parsimonious. This means they are compact and capture the behavior using a relatively small number of coefficients.

Response Surface Methodology (RSM)

Response Surface Methodology (RSM) is similar to, but different from, the stepwise response surface analysis (*S-RSA*) that TaylorFit implements. *RSM* and TaylorFit both can produce multivariate polynomial models of data. The major difference is that *RSM* is a Design of Experiment (*DoE*) methodology, in which the independent variables are fixed by the experimenter before conducting the experiment. On the other hand, TaylorFit is most appropriate for "natural experiments," in which data are collected as they occur, without the control of the experimenter.

In short, Response Surface *Methodology* is a Design-of-Experiment approach for developing data, while Response Surface *Analysis* is an approach for turning data into models.

Because of the cost of controlled experimentation, *RSM* is usually limited to relatively small datasets, and the resulting polynomials are low-order polynomials. In fact, the references describing *RSM* never give examples of high-order polynomial models. *RSM* literature also does not give examples of models using exponents other than positive integers. Thus, the *RSM* approach does not evaluate the possibility of ratios or other negative exponent terms. TaylorFit can fit *RSM* data; however it does not guide the user in choosing experimental conditions.

Steps in Developing a TaylorFit Model

The following is a brief summary of the 7-steps for developing a TaylorFit Model:

1 – Data Wrangling

Before using TaylorFit, you should do some initial data conditioning or cleanup, also known as data wrangling. This involves checking for problems such as:

- Missing data
- Outliers
- Influential points

When complete, the data should be saved in a comma-separated-value (CSV) Table.

2 – Starting TaylorFit

TaylorFit is launched from your web browser at www.TaylorFit-RSA.com. The Start Window shows three buttons: Import FIT Data, Import Project, and Download Manual (this document).

3 – Entering the data into TaylorFit

Data can be entered as *FIT*, *TEST (Cross-validation)*, or (*Final*) *VALIDATE* datasets. These may be from the same CSV file, or from separate files. In any case, the user can select which rows from the file are entered into each dataset.

4 – Setting Model Fitting Parameters

Once you select to import data or a project, you will note that the window displays four panes, each of which expands when you mouse over it: The *Data Pane* at the bottom left, the *Settings Pane* on the right, and the *Results Pane* at the top left, and the *Candidate Terms Table* at center left (between the results pane and the settings pane. The *Results Pane* shows four items. From left to right it includes: The *Current Model Table*; the *Global Statistics Table*; and a *Fit Plot* showing model residual errors versus predicted values of the dependent variable.

Before building the model, the user should select the model fitting parameters, which include the following pieces of information:

- The set of possible exponents, e.g. {e1, e2, ...}
- The maximum number of multiplicands, n_m
- The set of possible lag values (if the dataset is a time-series), e.g. {l1, l2, …}
- The goodness-of-fit criteria to be displayed
- The sorting criteria for the Candidate Terms Table

Once these are chosen (or the default values used), the user can start building the model.

5 – Adding and Removing Terms

The model is built by alternately clicking on terms in the Candidate Term Table to add them to the Current Model Table, then clicking on any terms in the Current Model Table that may need to be removed. Terms are picked from the Candidate Term Table that have strong fitting criteria (such as the probability of the t-statistic), and possibly that has significance to the user. After each term is added, the user should examine the terms in the Current Model Table to see if any should be removed based on their significance.

The Current Model Table shows each term of the model as a separate row, along with its term statistics. For example, consider the following Current Model Table for a model developed to correlate automobile fuel economy (MPG) with factors such as model year (YR), engine power (HP), vehicle acceleration (A), foreign versus domestic origin (OR), and vehicle weight (WT):

Term	t	p(t)
-14669	-3.2965	0.00107
+189.78(YR)	3.2328	0.00133
+0.12702(HP^{-1})(YR^2)	8.2729	2.2204e-15
-0.00841(A^2)(OR^{-1})	-4.6039	5.6403e-6
-0.01119(WT)	-7.2173	2.8465e-12
+8.4562e-5(WT^2)(YR^{-1})	5.3777	1.3114e-7
+3.7800e+5(YR^{-1})	3.3657	8.4017e-4
-0.81175(YR^2)	-3.1490	0.00177

Representing this model in the usual algebraic notation gives:

$$MPG = -14669 + 1.89.78 \cdot YR + 0.12702\frac{YR^2}{HP} - 0.00841\frac{A^2}{OR} - 0.01119 \cdot WT$$
$$+8.4562 \cdot 10^{-5} \cdot \frac{WT^2}{YR} + 3.7800 \cdot 10^5 \frac{1}{YR} - 0.81175 \cdot YR^2$$

The last term in this model, for example, is a quadratic term in model year. The associated *t*-statistic is -3.15, which has an associated two-tailed probability of 0.177%.

This addition/removal cycle ends when adding or removing any single term cannot improve the model with respect to the selected goodness-of-fit criterion.

6 – Changing Fitting Parameters

At this point, the user may change the modeling parameters by one of these changes:

- Adding new exponents to the exponent set
- Increasing the maximum number of multiplicands
- Adding additional labs to the lag set (if the data are a time-series)

7 – Ending the Modeling Session

If changing the modeling parameters as described above does not result in new terms that could improve the model, then the user should stop adding or removing terms, and perform the following two steps:

(1) Change the *FIT* dataset to include all the available data, including those data formerly used for the *TEST* and *VALIDATE* datasets. TaylorFit should now update the current model coefficients using all these data. *Caution: Do not add or remove any terms once this is done, unless you plan to obtain new data for another iteration of final validation.*

(2) Save the model for use or promulgation to other users.

The following chapters expand on this summary of the steps for using TaylorFit to model data.

Step 1 – Data Wrangling

Before using TaylorFit, you should do some initial data conditioning or cleanup, also known as data wrangling. This must be done before entering the data into TaylorFit, using spreadsheets other programs.

Data cleanup involves checking for problems such as:

- Missing data
- Outliers
- Influential points

When complete, you should save the data in a comma-separated-value (CSV) Table.

Here are the steps for preparing a data file in more detail. This assumes you are using a spreadsheet program or something similar.

a) Form a flat file structure; that is, the data should be in the form of a rectangular table, each column representing a different variable, each row a different set of measurements of those variables. If the data is a time-series, each row is a successive measurement taken at equally-spaced time intervals. Time itself should not be one of the columns, or, if it is, that column will have to be disabled within TaylorFit.

Always LOOK AT YOUR DATA!

b) The first row in the file may have text to indicate names or acronyms corresponding the the variables in the respective columns. If these are not provided, TaylorFit will use default names X_0, X_1, … for non-time-series models. For time-series models, the default names will be $X_{0,1}$, $X_{1,1}$, …, $X_{0,2}$, $X_{1,2}$, …(indicating variable 0 lag 1, variable 1, lag 1, variable 0 lag 2, variable 1 lag 2, etc.)

c) By default, the first column will be the dependent variable, although this designation can be changed within TaylorFit. The other columns will be independent (causative) variables.

d) Graphically examine the data, looking for any anomalies. Watch for outliers and influential points; these may appear as points that stand apart from the others. Graphical examination can include:
   i) Plot the dependent variable versus each independent variable
   ii) Plot each independent variable versus each other, looking for strong correlations
   iii) Examine the histogram for each variable
   iv) If the dataset is a time series, plot each column versus time

e) Look for any missing data. Missing data can be treated by one of the following:
   i) Remove the row or column containing the missing data. This may be your choice if there are many missing data points in the corresponding row or column. (If the data is a time-series, the row cannot be removed.)
   ii) Fill in missing data; one way to fill in missing data is to use the mean of the column, but avoid this if there are too many missing values

f) If possible, provide two or three similar sub-datasets, or one dataset that can be partitioned into two or three parts, for cross-validation. Make sure each dataset is "comparable" in range, distribution, etc. In other words, they should "look" similar using the graphical examination. If the user cannot ensure this kind of similarity, he/she should consider not using cross-validation. The three sub-datasets are designated:

   i) FIT: Used to compute the coefficients of the terms of the model;

   ii) TEST: Used to compute global statistics in the cross-validation process;

   iii) (FINAL) VALIDATE: Used only after the best model has been built using the FIT and TEST datasets to see if the model is capable of making good predictions when used with an independent dataset. Often data are scarce. It may be necessary to do without a separate VALIDATE dataset, or to use a single dataset for both Fit and Test purposes. However, in this case the resulting model should be used with extreme caution for making predictions. Without validation, the model may be used to express *correlation* among the variables. That is, the model can show how the independent variables affect the dependent variable. But predicting the outcome of the dependent variable using new measurements of the independent variables is risky and not scientifically valid.

How many data are enough? A rule of thumb is that each dataset should have at least six to ten times as many rows as columns. It may be possible to have fewer rows than this, but in such a case avoid having a final model that uses all the columns. In other words, the number of independent variables actually used in the final model should be less than one-sixth as many rows as in the FIT dataset. (Note that with step-wise fitting, it is possible that some columns may not contribute significantly to the model.)

g) If you have a single dataset is that is large enough to form two or three sub-datasets as just described, you can either partition it into separate files, or keep it as a single file, but note which consecutive row numbers you plan to use for each sub-dataset.

   For example, you could simply select the first one-third of the rows to use as the FIT dataset, the second one-third as the TEST dataset, and the last one-third as the VALIDATE dataset.

   Now you should graphically and statistically examine each sub-dataset to establish that they are similar to each other. If not, and the dataset is not a time-series, you can shuffle the data in one or two ways:

   i) Random shuffle: For example, in Excel®, one could create an additional column and use the RAND() function to enter random numbers. Then sort the rows by that column. Finally, delete the random number column before saving the data.

   ii) Stratified sampling: If you found that the different sub-datasets differed from each other substantially in one of the variable. Again, if you are using Excel®, sort the dataset by that variable. Now, create a new column and enter the repeating sequence 1, 2, 3, 1, 2, 3, … Next, sort on the column with the repeating sequence. Finally, delete the column of repeating numbers and save the file. You will now have three sub-datasets with similar distributions of the variable.

h) Save the datasets as comma-separated-variables (CSV) file(s).

Step 2 – Start TaylorFit

TaylorFit is run within a web browser from the website www.TaylorFit-RSA.com. The *Start Window* shows three buttons:

- *Import FIT Data* allows the user to import rows from CSV data files for *FIT*, *TEST*, and *VALIDATE*
- *Import Project* allows the user to import the data, modeling parameters, and current model that was previously saved
- *Download Manual* is this manual

The following steps assume that the user is beginning a new project. The use of the import project button allows the user to pick up where she/he left off on a previous modeling effort.

Step 3 – Entering the data into TaylorFit

The *FIT* dataset must be imported first, before the *TEST* or *VALIDATE* datasets. Simply click on *Import FIT Data* button. A dialogue box opens that allows the user to select the CSV datafile.

{Procedure for selecting rows from the datafile…}

After the *FIT* data have been entered, the window will be divided into four panes, each of which expands when moused over. The data Table is shown in the pane at the lower left. Only ten rows of the data are shown at a time. The scroll bar at right allows viewing all the rows. If the first row of the datafile contains alphabetic names or acronyms to identify the columns, they will be displayed in the heading of the Table. If the datafile does not have names, then Taylorfit will default to naming the columns as follows: X_0, X_1, X_2, …

Selecting the Dependent Variable

By default, the first column in the data Table is selected as the dependent variable; all other columns are then independent variables. The user can select any column to be the dependent variable by clicking on the heading of that column and selecting “Mark as Dependent.”

Entering Test (Cross-Validation) and (Final) Validation Data

This is an optional but recommended step. Below the FIT data Table are buttons to import TEST and VALIDATE datasets. As described in Step 1, these should be used if sufficient data are available. If there are not enough data for a separate final validation, then extreme caution is needed if the model is to be used for predicting outcomes of new cases. Predictions with an un-

validated model may not be considered scientifically valid. However, correlations and analytical dependencies revealed by the model can be considered scientifically valid.

The *TEST* and *VALIDATE* datasets are entered in the same way as the *FIT Data*, after clicking the appropriate buttons. All three datasets (FIT, TEST, VALIDATE) should have the same number of columns. Or they may be different rows selected from a single CSV file.

Step 4 – Initial Settings

Once you load the data or a project, you will note that the window displays four panes, each of which expands when you mouse over each of them:

- At the bottom left is the *Data Pane*. This displays *Data Tables* showing the data that the user enters. There may be one to three *Data Tables*; one for *FIT* data, one for *TEST* (cross-validation) data, and a third for *VALIDATE* data. The rows of the *Data Tables* correspond to dependent or independent variables. Once the user begins building a model, two additional columns are added, showing the model predicted values $\hat{y}$ and the prediction errors, e_i, or residuals.
- At the right is the *Settings Pane*. This shows buttons to select statistics for display or for use as fitting criteria, and the fitting parameters described below. These will be described in detail in Step 4.
- Above the *Data Pane* are two panes. Towards the left is the *Results Pane*. This pane includes three items:
    - The *Current Model Table* – This Table shows the term of the model as currently defined. Each row in this Table corresponds to one term of the model. The current model thus consists of the sum of each of these terms. The first column shows the coefficient of the term and the corresponding variables, their exponent and their lag (if a time-series). The second and third columns show the *t*-statistic and the probability of a larger *t*, labelled *p(t)*.
    - The *Global Statistics Table* – This Table shows summary statistics computed separately for the *FIT*, *TEST*, and *VALIDATE* datasets in columns 2, 3, and 4 respectively. These statistics are described below.
    - A *Residuals Plot* showing the residual errors versus the predicted value for the *FIT* dataset.
- On the right side above the data pane, between the results pane and the settings pane, is the *Candidate Terms Pane*.

Choosing Settings

In the *Settings Pane*, the user can choose

- The Goodness-of-Fit statistics to be used in examining the fit
- The fitting parameters (exponents, multiplicands, and lags)
- Options for clearing or saving the model or data (described in Step 7)

Statistics

The user can choose which statistics to use to evaluate the terms and the overall model.

Two statistics are shown by default in the *Candidate Terms Table* and the *Current Model Table*. These *term statistics* are associated with individual terms, not the overall model. The *term statistics* are:

t: *This is a measure of how strongly the term contributes to the fit. For example, if there is a large amount of data (>100) then a value of t = 1.96 indicates there is a 5% probability or less that this term is significant because of random variation*

p(t): *The two-tailed probability of a larger value of t. Typically, we would add a term to the model if p(t) ≥ 0.05 (5%) for that term.*

Other statistics are shown in the second Table in the Results Pane. This is the *Global Statistics Table*. The Global Statistics Table shows *Data Statistics* and *Goodness-of-Fit Statistics*.

The *Data Statistics* are computed separately for each dataset – FIT, TEST and VALIDATE, and do not depend upon the model:

Nd: The number of data (dependent variables); usually the number of data columns minus 1.

Np: The number of parameters (coefficients) in the model, including the intercept if there is one. Essentially, it is the number of terms in the model.

Nf: The number of degrees of freedom*; Nf = Nd - Np*

TSS: *Total Sum of Squares:* $TSS = \sum(y_i - \bar{y})^2$*,* where y_i are the dependent data points from the FIT dataset, $\bar{y}$ is the mean of those points

The other statistics are termed *Goodness-of-Fit (GoF) Statistics*. The *GoF* statistics are measurements that describe how well the overall (current) model fits the data. Their use is described in Step 5. The *GoF* statistics are:

SSE: Sum of squares of the errors (or residuals); $SSE = \sum(y_i - \hat{y}_i)^2$
where $\hat{y}_i$ are the model predicted values corresponding to each y_i.

MSE: *Mean square error = SSE/df;* this is an estimate of the error standard deviation

MSR: *Mean square of the regression = (TSS-SSE)/np*

RSQ: R^2*,* the square of the multiple correlation coefficient*;* $R^2 = 1 - SSE/TSS$

cRSQ: *Complementary R2 = 1 – R2 = SSE/TSS*

adjRSQ: *Adjusted* R^2 – similar to R^2 , but takes into account the degrees of freedom

F: The *F*-statistic = *MSR*/*MSE*; similar to the t-statistic for a term, but applies to the overall fit. A larger value indicates a better fit.

p(F): The probability of F being this large or larger by chance.
The smaller the value, the better the fit.

AIC: *Akaike Information Criterion = log10(MSE) + 2·np/df*

BIC: *Bayesian Information Criterion = log10(MSE) + log10(nd)·np/df*

Max|Err|: The maximum absolute value of the error

The *GoF* statistics can be seen as buttons in the Settings Pane. Selecting one or more of these *GoF* statistics by clicking on its button adds a column to the Candidate Terms Table showing what the value of that statistic would be if the corresponding term were added to the Current Model. All the *GoF* statistics are displayed in the Global Statistics Table in the Results Pane. Thus, the user can compare, say, the *F*-statistic of the current model with what it would

be if one of the terms in the Candidate Terms Table were added in. You will also be able to sort the candidate terms on the selected statistic, making it a criterion for adding and removing terms from the model.

Note that you might occasionally find the value of R^2 to be outside the range of 0.0 to 1.0, which seems illogical. This occurs in two cases:

(1) It may occur with the *FIT* dataset when there is no intercept term in the model;

(2) It may occur with the *TEST* or *VALIDATE* datasets if the current model predictions are worse than just using the average of the dependent values from the FIT data as a predictor.

This occurs because TaylorFit computes $R^2 = 1 - SSE/TSS$. Essentially, this makes the value of R^2 into a comparison between the current model and the "worst case" model: $y = \bar{y}$ where $\bar{y}$ is the mean of the dependent variable from the FIT dataset. The sum of squares of the "errors" in this worste case model is the *TSS*. Thus, if the prediction is doing worse than just using the mean as a predictor, then *SSE* > *TSS*, and R^2 will be negative (and possibly even less than -1). This is an indication of a poor model (which may occur in early stages of modeling).

Some programs (e.g. *R* and Excel®) avoid this problem by using $TSS = \sum(y_i)^2$ for the special case where there is no intercept. Essentially, this changes R^2 to a comparison with a different "worst case" model: $y = 0$. A problem with this approach is that it can cause the value of R^2 to improve when the intercept is removed from the model, even if the *SSE* increases! Some modelers recommend that the intercept should always be included in the model, which eliminates this problem. In TaylorFit this choice is up to the user.

Fitting Parameters

The real power of TaylorFit comes from the user's ability to choose three sets of fitting parameters that control the generation of candidate terms. They allow formation of polynomial interaction terms, which can create complex response surfaces. The three types of ***fitting parameters*** are:

- The maximum number of multiplicands in a candidate term, n_m
- The set of possible exponents, {*exp*}
- The set of possible lags, {*lag*}

Note: The coefficients that multiply each term in the model are also called parameters. To avoid confusion, we will always refer to "fitting parameters" for the sets of multiplicands, exponents and lags discussed here.

MAXIMUM NUMBER OF MULTIPLICANDS: This an integer that represents the maximum number of multiplicands a single term of the model may have. For example, if *Y* is to be regressed onto independent variables *Q*, *R*, and *S*, terms with one multiplicand (n_m = 1) include *Q*, or Q^2, or S^{-1}, etc. Examples of terms with n_m=2 include $Q \cdot R$, or $R \cdot S^2$, etc., but not Q·R·S. The intercept is always a potential candidate term, although the user can choose to add it to the model or not, according to needs and preference. A maximum number of multiplicands of 1 or 2 is usually sufficient, and rarely are you likely to need more than 3 or 4, even if there are many independent variables.

EXPONENTS: This section shows the exponent list which the program will use to select possible exponents in the model terms. The exponents may be integer or non-integer, positive or negative. Zero is included by default (resulting in formation of the intercept as a candidate term), and should not appear in the list. Including negative values, as in the example, enables ratios of independent variables to be tested for model inclusion.

For example consider the case where *Y* is the dependent variable, and *Q*, *R* and *S* are the independent variables. If the user selects n_m = 2 and {exp} = {1, 2, -1}, then possible candidate terms may include: Q, Q^{-1}, Q^2R^{-1}, or QR^{-1}. The last of these is the ratio *Q/R*. In this example the following terms could not be generated in the candidate term Table: QRS, Q^3R, RS^{-2}.

LAGS: A lag is an index that tells TaylorFit to point to previous time periods (in previous rows of the dataset) to find independent variables. Thus, for example, a company's sales in a particular year might depend on its sales in previous years. By default in TaylorFit the set of lags starts out as zero ({lag} = {0}). This signifies that the dataset is not a time-series. If the dataset is a time-series, then the user must signify it by replacing the zero in the set of lags with positive integer values. Lags and time-series are discussed in more detail in Step 6.

If the data are not a time-series, then lags should be omitted or include only zero. This will force TaylorFit to predict dependent data using only independent data on the same row.

Note: Selection of fitting parameters only affects the generation of candidate terms. It does not affect terms already in the current model. For example, suppose the user selects n_m = 3 and adds terms to the model with three multiplicands. Subsequently the user may then change to n_m = 2. The three-multiplicand terms in the current model stay there, but the candidate term table will only show terms with two multiplicands.

Selecting Fitting Criteria and Initial Fitting Parameters

Besides selecting the Fitting Parameters, the user has a choice of statistics to use as fitting criteria. The user should select one or more statistics to display in the candidate term table by clicking on the corresponding button at the top of the Settings Pane. Then, by clicking on the corresponding heading in the candidate table, that table will be sorted according to that statistic. This makes that statistic the effective criterion for choosing terms to add into the model.

Current recommendations are: If the FIT and TEST datasets are identical, or no separate TEST data file has been added, use a conservative criterion. The F-statistic will be a fairly conservative choice, although the AIC or BIC are also conservative. When using cross-validation (see below) and you therefore have independent FITFILE and TESTFILE (cross-validation), it is not as necessary to be so conservative in adding terms. The user should consider using the *MSE* or $p(t)$ as criterion.

Start with a linear model, then go from there.

Next, the user should select the Fitting Parameters (n_m, {exp}, {lags}). A strategy for selecting fitting parameters is to initially set n_m=1, {exp}={1}. Then, if the data are not time-series, set{lags} = {0} for regression. If the data are a time-series, set {lags} = {1}.

Then proceed to Step 5. These initial settings will result in linear models. If a non-time-series, the result will be a multilinear regression. If a time-series, the resulting model will be what is known as a first-order autoregressive (AR(1)) model. Subsequently, as described below in Step 6, each fitting parameter is increased incrementally, say by changing the multiplicand list to {1, 2}, fitting again, then sequentially adding exponents (and additional lags if a time-series).

Step 5 – Adding and Removing Terms

Once initial settings have been made, the user can begin building the model by adding and removing terms iteratively until the model cannot be improved. Only then the fitting parameters should be changed to enable more complex models (as discussed in Step 6), and the iterative procedure repeated. Steps 5 and 6 are repeated until adding multiplicands, exponents, and/or lags does not produce a significantly improved model.

Building the Model

Once the fitting parameters (n_m, {exp} and {lags}) are selected, the desired fitting criterion has been selected and set as a sort criterion in the Candidate Terms Table, the user should proceed to add and remove terms to the model. The following description assumes that cross-correlation is not being done.

Addition Step: Begin by examining the terms in the Candidate Terms Table. The first term should be the one that produces the best performance according to the selected criterion. The user decides at this point which term should be added to the current model. To be selected, a term should either have a *GoF* that is better than that of the Current Model, and/or has a p(t) below an *addition threshold value* (usually 0.05, or 5%).

In most cases the top term will be selected. But sometimes the user may select another term with almost as strong statistics, but which is preferable for some reason. For example, if a linear term is almost as good as a 3-multiplicand term, the linear term might be chosen.

Then, the user just clicks on the chosen term. This term will move from the Candidate Term Table to the bottom of the Current Model Table. TaylorFit then goes into a computation in which the *GoF* statistics Table, the residual plot, and the Candidate Term Table are all updated.

If no terms in the Candidate Terms Table meet the criterion for addition, then the user should skip to the removal step.

Removal Step: After each Addition Step, the user should examine the Current Model Table to see if any of the terms in the model have decreased in significance below a *removal threshold value*. The removal threshold value should be higher than the addition threshold value. E.g., terms may be added only if $p(t) \leq 0.05$, but removed only if $p(t) > 0.05$ or even $p(t) > 0.08$.

Iterate: After the removal step, the user should return to perform another addition step. If at any time no term meets the criterion for either addition or removal, the user then either stops the modeling exercise and going to Step 7, or continues on to Step 6 – Changing Fitting Parameters.

Cross-validation

If there are enough data to form two or three sub-datasets, it is recommended to enter them separately as FIT, TEST, and VALIDATE datasets. Refer back to Step 1 for the discussion on ensuring similarity between the FIT and TEST sub-datasets.

Cross-validation is a powerful way to prevent problems such as *overfitting* (also referred to as "memorizing the noise in the data") or *spurious correlation* (also called *chance correlation*). See the section below on Modeling Tips and Potential Pitfalls.

The TEST dataset is also known as a Cross-Validation dataset. With cross-validation, model generation proceeds almost the same as described in the previous section, with the only difference being in how the term is selected in the addition step.

For cross-validation, two criteria must be satisfied simultaneously. Specifically, p(t) must meet the addition threshold (e.g. 0.05), and a *GoF* statistic for the TEST dataset must be better than the same *GoF* statistic for the Current Model.

Keep in mind the following: For purposes of sorting the candidate terms to determine which is best for inclusion in the model, *t*-statistics are always computed using the FIT file. *However, global statistics (e.g. SSE, MSE, R^2, F-statistic, AIC and BIC) in the Candidate Terms Table are always computed using the TEST file if one has been designated, and the FIT file if not.*

As an example, suppose the *F*-statistic has been selected in the Settings Pane. Then, the user clicks on the heading for *F* in the Candidate Terms Table. The user should examine the terms in the Candidate Terms Table. Before adding one of these terms to the Current Model, she/he should ensure that both of the following are satisfied:

- The value of p(t) for the term is below the addition threshold
- The value of *F* for the term is greater than the value of *F* for the TEST file in the Global Statistics Table

If both of these criteria are satisfied, then the user should click on the term to make it part of the Current Model. If no term satisfies both of these criteria, the user should proceed to the removal step as described in the previous section.

Step 6 – Changing Fitting Parameters

After each time an addition/removal cycle is completed, the user should try expanding the domain of candidate terms by adding exponents, and/or by increasing the maximum number of multiplicands, and/or (if the data are a time-series) adding to the list of lags to check. The following will discuss strategies for increasing multiplicands and exponents first, then have a separate discussion of lags for time-series models.

Incrementing the Multiplicands and Adding Exponents

As described above, the user should start with a multilinear regression model. That is, add and remove terms with $n_m = 1$ and {exp} = {1}, until the model cannot be improved (according to the selected criterion) by the addition or removal of any single term.

Then, the maximum number of mulitplicands and the list of exponents can be increased. There is no single way to do this. This is part of the art of modeling. The Table below shows a possible sequence to consider.

Cycle	n_m	Exponents
1	1	1
2	2	1
3	2	1, 2
4	2	-1, 1, 2
5	2	-1, 1, 2, 3
6	3	-1, 1, 2, 3

As the user gains experience, she/he may make other choices. Note that as the maximum number of multiplicands and the number of exponents grows, the number of possible candidate terms may become very large. For example, if there are eight independent variables, the multilinear regression model (as in cycle 1 in the Table above) results in eight possible candidate terms. Increasing n_m to 2 results in 29 possible candidate terms. With $n_m = 2$ and two exponents (as in cycle 3 above), there will be 99 candidate terms. Finally, with up to four multiplicands and four exponents (as in cycle 6 above) there would be 2,605 possible candidate terms.

The speed of the calculations will depend not only how many candidate terms there are, but how many rows there are in the data. Although these factors may slow the computations, it's worth repeating as stated before, "data are expensive; computations are cheap." If a model produces useful results, it may be worth waiting for.

Adding to the Lags (Time-Series Only)

If the data are a time-series, then the user should select the lags to be considered by TaylorFit for forming candidate terms. Similar to choosing the other settings, the user should start simple by selecting only lag {1}. Essentially, the independent variables from the previous row is used to predict any particular row.

In subsequent iterations, the user may include additional lags. These may be added one-at-a-time sequentially. Or, the user may choose to skip lags to incorporate "seasonality" effects.

For example, if the user has data that have been collected on a daily basis, so each row represents a successive day of data, then starting with lags = {1} indicates that each day's dependent variable is modeled based only on the previous day's dependent variable measurements. This is called a *first-order autoregressive (AR(1)) model*. Lag zero should only be included if it makes physical sense for the current value of the dependent variable to depend on current values of the independent variables. Next, the user may add the dependence from two days ago, adding lag 2 so that lags = {1,2}. This would produce an AR(2) model. Subsequently, lags should be increase to {1,2,3}. Et cetera.

Now, it may be that the user does not find dependence upon 3 or more days ago, but suspects that there is a day-of-the-week effect. The user can add lag 7, to test for correlation between the dependent variable from a particular day and the value from a week ago.

Note that for all lags other than zero, TaylorFit will test for lagged values of the dependent variable, as well as lagged independent variables.

For example, consider the well-known Lydia Pinkham time-series dataset with Sales (S) and advertising budget (A) for 54 successive years from 1907 to 1960. These data are readily available online for the user to download for testing purposes. We would like to use these data to develop a model to predict sales using previous years' sales and advertising. We might expect this year's sales to depend upon this year's advertising, so lag zero would be appropriate. Lags 1 and 2 seem to be relevant. That is, sales in one year shows some dependence upon both sales and advertising in the two previous years. However, there is no expectation of a seasonal effect in this case (unless a dependence on the 11-year sunspot cycle is expected!).

In time-series modeling we have the same problem as described above in determining the order to change the Fitting Parameters. Again, is up to the user, and is part of the art of modeling. A suggestion is increase the number of lags repeatedly until no new terms are significant before increasing n_m or {exp}, and again after each time those other parameters are changed.

Step 7 – Ending the Modeling Session

When to Stop

Ideally, the modeler should only stop when the model cannot be improved (with respect to the chosen criterion) by addition or removal of any single term, after the fitting parameters have just been augmented.

Sometimes, a modeler may wish to keep models simpler than would result from this strategy. This can be done by using more strict or conservative criteria (e.g. change the p(t) threshold to 1%).

Final Validation

In order for a model to be scientifically valid for prediction purposes, it must be validated using data independent of those data used to create the model. TaylorFit is structured so that the FIT and TEST datasets are used for identifying (determining which terms should be in a model) and fitting (determining model coefficients) a model. When this task is complete, the final step is validation – determining how well the model works with independent data.

Often data are scarce. It may be necessary to do without a separate VALIDATE dataset, or to use a single dataset for both FIT and TEST purposes. However, in this case the resulting model should be used with extreme caution for making predictions. Without validation, the model may be used to express *correlation* among the variables. That is, the model can show how the independent variables affect the dependent variable. But predicting the outcome of the dependent variable using new measurements of the independent variables is risky and not scientifically valid.

Final Model Fitting and Exporting Information

Once the decision is made that the current model is satisfactory, then the following steps should be performed:

(1) Change the FIT dataset to include all the available data, including those data formerly used for the TEST and VALIDATE datasets. TaylorFit should now update the current model coefficients using all these data. Caution: *Do not add or remove any terms once this is done, unless you plan to obtain new data for another iteration of final validation*.

(2) Export the model for use or promulgation to other users.

Export Data saves the Data Tables in a CSV file. This can be useful for additional analysis, for generating graphs, etc.Although the user already has the data in a CSV file to start with, the exported files will also have *the model predictions and the residuals* in additional columns.

Export Project saves a file with extension "tf" that contains the data, the fitting parameters, and the current model. This can be used at any stage of the modeling process so the user can return later to where she/he was and continue the project from the point at which it was interrupted.

The user can also copy and paste any Table in TaylorFit into other programs. This can facilitate, for example, inputting the current model into an Excel® function.

Modeling Tips and Potential Pitfalls

The first step should always be to graphically examine the data. Plot each variable (whether independent variable or dependent variable) versus each of the others. Plot a histogram of each variable. Compare fit, test and validation data for lack of similarity, unique situations, differences ranges or other statistical properties. This step is ignored at great risk to the modeler!

The final step should always be graphical examination of residuals. The residuals should be plotted versus the predictions and versus each independent variable in turn. The purpose of the modeling is to remove any apparent trends in these plots.

The stepwise algorithm does not always converge on a global optimum. This is often not a problem; sub-optimal is often good enough. However, you can try adding and removing terms manually to find better models closer to the optimum.

Sometimes it helps to start by fitting to one independent variable (i.v.), and then add terms that include other i.v.'s. For example, in fitting wastewater treatment plant data for influent flow vs. time of day and lagged flow, first fit best model using time of day only, then allow terms with lagged flow.

Consider adding all linear terms to a model to start with, regardless of their significance. Then increase the exponent list and n_m to include nonlinear terms. Then use backward elimination to pare the model.

You can also fit a fundamental model to the data, taking into account as much as possible. For example, one investigator obtained the tidal components of waves near an ocean outlet of a bay. Then, he used TaylorFit to discover correlations between empirically related components like wind and off-shore wave parameters, and found interactions between them and the fundamental components.

OVERFITTING: This is a problem also known as "memorizing the noise in the data." The values of R^2 or SSE should never be used as a fitting criterion unless cross-correlation is being applied. Otherwise overfitting is almost guaranteed. Adding a term to a model will usually improve these statistics, even if the term is not a significant contributor to the fit.

To be scientifically valid, a model must be validated using data independent of those data used to create the model

If not using cross-validation, the number of candidate terms should not be allowed to exceed one-tenth to one-sixth of the number of data, and absolutely should not exceed the number of data.

MULTICOLINEARITY: If two or more of the independent variables are correlated with each other, a model that includes both will have misleading *t*-statistics. However, step-wise regression avoids this by adding terms to the model one-at-a-time. Once a variable or term is in the model, another variable or term that is highly correlated will not have significant additional predictive power, and is not likely to be selected into the model.

CHANCE CORRELATION: If you fit a random variable to 20 random variables, accepting 95% confidence level, you will find correlations. I tried it with two multipliers and three exponents, and got better than 50% R^2. But if you use cross-validation, no spurious correlations are found. Cross-validation (separate FIT and TEST datasets) mitigates against overfitting and chance correlation.

POLYNOMIAL WIGGLE: This is a problem in which the model works well with the FIT data, but produces results that are far "out of line" when the model is used to make predictions at points between data points used for fitting. This may be caused by too few degrees of freedom. The remedy is to use a conservative fitting criterion. Cross-correlation also mitigates this.

INFLUENTIAL POINTS: This is a problem in which a small portion of the data are distant (in n-space) from the other data. The resulting model does a good job predicting the difference between the two portions of the data, but a poor job within each group. It can also result in a misleadingly high value of R^2. This can be avoided by partitioning the data and modeling the parts separately. For example, if you have marketing data on men and women, but 95% of the data are for men, then it would be better to separate them and form two different models - one for the men and one for the women.

When partitioning multivariate data, do a multiple sort (stratified sampling). It might be necessary to bin the variables (sort into groups, number the groups, and sort on the number) so that one variable doesn't dominate the sort.

The final step in modeling should always be graphical examination of residuals

A related problem is that sometimes you might be interested in modeling one part of the data more than other parts. For example, in modeling the effect of rainfall on river flow, you might only be interested in wet periods, which might comprise a small portion of the total dataset. Ultimately the software may include weighting coefficients, but for now the only approach is to cut out some of the data to reduce its weight, or replicate other areas to increase their weight. Or the data could be partitioned to model dry season separately from wet season.

Nonlinearities may sometimes be hard to find unless noise level in the data is very low. In our experience, when R^2 is 0.75 or lower, the nonlinearity is often masked, and only linear terms are significant. Unless the nonlinearity is strong, it may not show up unless R^2 is 0.8 or 0.9, or if there are a very large number of data points.

POTENTIAL TIME-SERIES ISSUES:

If a time-series varies slowly, it is easy to get high R^2 values, even with naïvely simple models. In this case it would be more meaningful to difference the data, and model the difference. Sometimes this is not sufficient, so it might be necessary to censor the data to weight regions with larger amounts of change.

For time-series with long lags, you can cut the number of variables by using a *variable integrated lag* approach. That is, use as lagged variables linear combinations of lagged variables, such as averages of lagged variables, where you use longer averaging periods the further back in time you go. For example, create independent variables from lag 1, avg(lag 2-3), avg(lag 4-7), avg(lag 8-15), etc.

If the data show sharp changes, discontinuities, or high-frequency transitions, it may be necessary to use a high degree polynomial (say, six or eight, or even higher). And, it would be necessary to include terms with lower degrees at the same time.