
TaylorFit

User's Manual

RESPONSE SURFACE ANALYSIS (RSA)



using
STEPWISE MULTIVARIATE POLYNOMIAL REGRESSION
(MPR)

Simetrica, LLC

David A. Vaccari, Ph.D., P.E., BCEE

www.simetrica-llc.com

info@simetrica-llc.com

www.TaylorFit-RSA.com

2023 Version

Contents

Acknowledgements.....	5
New Features	6
What is <i>TaylorFit</i> ?	7
<i>TaylorFit</i> Modeling approach.....	8
How <i>TaylorFit</i> Works.....	9
Goodness-of-Fit statistics.....	9
Fitting parameters	10
Stepwise regression	10
Data Files.....	10
Comparison with other modeling methods.....	12
Multilinear Regression (MLR)	12
Artificial Neural Networks (ANNs)	12
Response Surface Methodology (RSM)	13
Quick Start for Developing a <i>TaylorFit</i> Model	14
1 – Data Wrangling	14
2 – Starting <i>TaylorFit</i>	14
3 – Entering the data into <i>TaylorFit</i>	14
4 – Selecting Model Fitting Parameters.....	15
5 – Adding and Removing Terms	15
6 – Changing Fitting Parameters.....	16
7 – Ending the Modeling Development/Validation	17
8 – Model Analysis and Evaluation	17
Detailed Steps for Modeling with <i>TaylorFit</i>	17
Step 1 – Data Wrangling	18
Step 2 – Start <i>TaylorFit</i>	21
Step 3 – Manipulating Data in <i>TaylorFit</i>	22
Importing Data	22
Arrangement of the <i>TaylorFit</i> Window.....	23
Selecting the Dependent and Independent Variable	24
Data Transformation.....	24

Logarithmic transform	25
k-order Differencing.....	25
Standardize	25
Scale by RMS	25
Examining the Data – Plotting and Column Statistics.....	26
Step 4 – Initial Settings.....	27
The Settings Button.....	28
Term, Dataset and GoF Statistics	28
Fitting Parameters.....	31
Setting <i>TaylorFit</i> up for Time-Series Analysis	32
Selecting Fitting Criteria and Initial Fitting Parameters	33
Step 5 – Stepwise Addition and Removal of Terms	34
Building the Model.....	34
Cross-validation	35
AutoFit	37
Step 6 – Changing Fitting Parameters.....	37
Incrementing the Multiplicands and Adding Exponents	37
Changing α	38
Adding to the Lags (Time-Series Only).....	39
Step 7 – Ending the Model Development/Validation.....	40
When to Stop	40
Examine the Statistical Goodness-of-Fit	41
Examine the Graphical Goodness-of-Fit	42
Final Validation	43
Final Model Re-Fitting.....	44
Exporting Information.....	45
Step 8 – Model Analysis and Evaluation	46
Graphical analysis	46
Confidence Intervals and Prediction Intervals.....	46
Sensitivity analysis	47
Importance ratios for the Independent Variables	47
Sensitivity and Importance Ratios of the Terms.....	48

Comparing different models	49
Example Datasets.....	50
Auto MPG Dataset	50
Lydia Pinkham Dataset.....	51
Modeling Tips and Potential Pitfalls	52
Potential Modeling Pitfalls.....	52
Potential Time-Series Issues	54
Design-of-Experiments with Multivariate Polynomial Regression	54
Known Bugs.....	55
Formal Definitions.....	56
Non-time-series Multivariate Polynomial Regression Model.....	56
Number of Candidate Terms.....	57
Computing Confidence Intervals and Prediction Intervals	58
Goodness-of-Fit Statistics	59
Dataset statistics	59
Global Goodness-of-Fit Statistics	59
Model statistics	61
Transformations.....	61
Sensitivity and Importance Ratio.....	62
Importance ratio of the TERMS in the Current Model Table.....	62
Bibliography	63
Classic references on regression and time-series analysis	63
MPR References.....	63
Other references.....	63

Acknowledgements

The following people have helped with the development of *TaylorFit* RSA software:

Xiaoguang (Steve) Wang
Sarath Jagupilla

David Klappholz

2016 Design Team:
Patrick Grasso
Julia Kim

Christopher Kelley
Eric Fitzpatrick

2017 Design Team:
Matthew R Crepea
Jaemin Shim

William Mosca
Daniel Heyman
Casey C Johnston

2018 Design Team:
Justin Tsang
Weronica Zamlynny

Kexian Wu
Tianwei Liu
Shuangchang (Jack) Li

2019 Design Team:
Jared Davis
Zhilin Chang

Yi Jing
David Cortinas
Eric Zhen

2020 Design Team:
Dominick DiMaggio
Kurt Louie

Michael Chunko
Zachary Lee
Akihira Maher

Thanks also go to:

2015 Design Team:
David Adam Perez
Chris Barry

Tim Barbara
Brandon Schur

2018 Business Technology Team:
Elena Y Oh
Dillon T Jacoby

Victoria Piskarev
Emma J Murphy
Benjamin Gilman

Research Assistants:
Zoey Meng

Michael Botros

New Features

Here is a list of changes from the previous version of *TaylorFit* and the *TaylorFit User's Manual*:

- **Pop-up help** boxes have been added that show when the user rolls over certain features on the page.
- An improved Data Partitioning dialog box was created for importing data.

CHANGES IN THE SETTINGS BOX:

- A **Max|Err|** statistic has been added to the GoF statistics to enable “MiniMax” optimization.
- A **Recalculate** button was added in Settings to force *TaylorFit* to update the *Candidate Terms* Table.
- 2021: A button for “**AutoFit**” adds the feature of having *TaylorFit* selected terms to add and remove from the current model, although limited to current model parameters. This is a beta-test function, and should be used with care.

CHANGES IN THE RESULTS PANE:

- Importance Ratios for each individual term has been added to the Current Model Table.

CHANGES IN THE DATA PANE:

- 2021: Confidence Intervals and Prediction Intervals can now be computed by *TaylorFit* from a drop-down menu item under the column for “Predicted.”
- 2021: A **Replace Data** button was added.
- **Column statistics** have been added to the data pane to show summary statistics for each column, including of data and of generated columns.
- **Replace Data** button added to allow user to import a new data file or to use different rows from the same data file.
- The drop-down box in the column headers of the data pane have the following new features:
 - **Ignore Column** and **Show Column** commands have been added to instruct *TaylorFit* to not use (or to use) a particular column for fitting.
 - **Transform Column** command to generate a new column with data transformations, such as log-transform, has been added.
 - A new plot, the **QQ Plot**, was added for graphical testing of normal distribution for a column.
- **Show Sensitivity** and **Show Importance Ratio** commands were added to create new columns with the corresponding calculations.

CHANGES TO THE USER'S MANUAL (other than documenting the new features above):

- A new chapter has been added, “Step 8 – Model Analysis and Evaluation.”
- New details about selecting a value for α are provided.
- Formal mathematical descriptions have been added for MPR models, the statistics used in *TaylorFit*.
- A bibliography and reference section has been added.
- 2021: A section on “**Known bugs**” was added.

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.”

The *TaylorFit* application works entirely client-side in your browser, so there's no need to download or install anything and no need to create an account. You can run the application simply by going to www.TaylorFit-RSA.com.

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 values that they think can predict the profitability. The values include each store's local market share, how long it has been there, how many years of experience the salespersons 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 values in other columns. They wish to predict the profitability in column one using the values 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 application rate 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 one year depend upon sales and advertising in previous years?

You can model any numerical data that can be collected into a Table or spreadsheet with two or more columns of data, where one column depends upon the others

TaylorFit Modeling approach

The type of models *TaylorFit* generates are called Multivariate Polynomial Regression (MPR) Models. These are extensions of the basic polynomial equations that are familiar from basic algebra, extended to multivariable data. A formal mathematical description of *MPR* is provided in a later chapter of this manual.

Consider the example above of agricultural crop yield (Y), which depends upon the amount of irrigation water (W) and fertilizer application rate (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 nonlinearities. The *MLR* model for this case would be:

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

This 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 an *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 complex behaviors and you do not include them, then your model will be biased.

If your data have complex behaviors and you do not include them in the model, then it will be biased

TaylorFit gets its name from the Taylor Theorem, which proves that any functional relationship can be described by a polynomial series equation. The Taylor polynomial may have an infinite number of terms. However, it can also be proven that an arbitrary degree of accuracy can be achieved by truncating the Taylor polynomial to a finite number of terms. The models developed by *TaylorFit* can be viewed as truncated Taylor polynomials.

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 and cross-correlation. For more details explaining what *MPR* is watch the 23-minute video at <https://www.youtube.com/watch?v=bLOXW5LR8-Y>.

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 “fitting 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 best model is obtained.

Goodness-of-Fit statistics

TaylorFit computes the coefficient for each term and 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: the *t*-statistic, and the probability of erroneously concluding that the coefficient is not zero, $p(t)$. There are also a number of *Dataset* statistics: n_d , n_p , df , and *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:

<i>SSE</i> :	Sum of squares of the model errors (aka residuals)
<i>MSE</i> :	Mean square error; this is an estimate of the residuals sample variance
<i>Skew</i> :	A measure of skewness, or asymmetrical distribution of the residuals
<i>Excess Kurtosis</i> :	Kurtosis minus 3.0. This is a shape factor that describes a deviation of the residuals distribution relative to the “normal” distribution.
<i>RSQ</i> :	R^2 , the square of the multiple correlation coefficient
<i>Adj-RSQ</i> :	Adjusted R^2 – similar to R^2 , but with a penalty for model complexity (the number of terms)
<i>AIC</i> :	Akaike Information Criterion
<i>BIC</i> :	Bayesian Information Criterion
<i>Max/Err/</i> :	Maximum of the absolute values of the errors; used for minimax optimization
<i>F</i> :	The <i>F</i> -statistic; similar to the <i>t</i> -statistic for a term, but indicates the fit of the overall model, rather than of a particular term
$p(F)$:	The probability of <i>F</i> being this large or larger by chance

The formal mathematical definitions of these values are given in a later chapter below.

Fitting parameters

The user begins by defining the *fitting parameters*. These 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. (In fact, the model will always initialize with only an intercept. If the modeler wishes, it may be removed.)

For example, in the simplest case, if $EXP = \{1\}$ and $N_m = 1$, then *TaylorFit* will only create an 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) $W \cdot F^2$ or $W^3 \cdot F^2$ 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 one-at-a-time 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 probability $p(t)$ less than the selection criteria called α (α is typically 5% but can be changed by the user). In the **removal step**, the terms in the current model that have a $p(t)$ less than α are removed one-at-a-time until all such terms have been removed. (Sometimes, a term that was statistically significant early in the modeling process is displaced by a term added later; it may then become insignificant and should be removed.) The addition and removal steps are alternated repeatedly until the model 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

The data for *TaylorFit* will generally be in a tabular format, such as in an *Excel* spreadsheet, but saved as a "comma separated variable (CSV)" format. Each column in the table represents a variable, and each row is a single measurement or instance of all the variables. As a rule of thumb, there should be at least six to ten times as many rows as columns. But the more rows there are, the more complexity it may be possible to discover in the data.

TaylorFit can divide the datafile into up to three partitions. These are termed the "*FIT*", the "*CROSS*", and the "*VALIDATION*" sub-datasets, and each has a different purpose.

The ***FIT* sub-dataset** is used to compute statistics associated with individual terms, such as the term's coefficient, its t -statistic, and its probability $p(t)$. A model can be built using the *FIT* sub-dataset alone, although incorporating the others has important advantages.

The **CROSS sub-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 prediction for the *CROSS* dataset (as measured with the selected global GoF criterion), and the $p(t)$ for the chosen term meets the chosen (default 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 (see section on modeling pitfalls).

The **VALIDATE sub-dataset** should be used only after the best model has been produced using the *FIT* and *CROSS* data and the model has been finalized for use in making predictions. Then, the model is used to compute the global statistics using the *VALIDATE* data. 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.

The ALL dataset: This would be the combined dataset that the *FIT*, *CROSS*, and *VALIDATE* data sets were drawn from. The *ALL* dataset will be used to update the model coefficients after the model is built and validated. This is the version of the model that should be put into use.

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. However, it is always a good idea to use cross-validation and final validation if possible.

If there are not enough data for final validation or for cross-validation, one option is for the modeler to consider using more conservative fitting criteria. For example, the user might add terms only to the model only if the $p(t)$ or $p(F)$ is less than 1%, instead of 5% (i.e. $\alpha = 0.01$ instead of 0.05). Or use minimization of the F -statistic as a fitting criterion instead of minimizing MSE or maximizing adjusted R^2 .

TaylorFit can generate a number of **output files** including a model description file and a data file. The data file contains the original data plus the model predictions and residual errors. The current model can be exported as code in *Javascript*, *Excel*, *C++*, *Matlab*, *Python* or *R* formats. The graphs can be downloaded as PNG files.

Comparison with other modeling methods

There are many other ways to model multivariate 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* response surface analysis (RSA) with stepwise multivariate polynomial regression (MPR) is a natural extension of MLR, it is a good fit for researchers who have used MLR. By changing to RSA with MPR, 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 data that have 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)

ANNs are 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 an ANN model 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 don’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 must 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 computationally faster than *ANNs*, and it converges without an arbitrary stopping criterion.

To summarize the advantages of *TaylorFit* over *ANNs*:

- *TaylorFit* converges absolutely. That is, iterations ultimately reach a conclusion without an arbitrary stopping criterion.
- *TaylorFit* MPR models are much easier to manipulate than *ANNs*. 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 than *ANNs* (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.
- *TaylorFit* models are parsimonious. This means they are compact and capture the behavior using a relatively small number of coefficients.

TaylorFit RSA models are comparatively tractable, transparent, transferable, accurate and simple – better!

In short, *TaylorFit* RSA models are comparatively tractable, transparent, transferable, and simple – better!

Response Surface Methodology (RSM)

Response Surface Methodology (RSM) is like, but different from, the response surface analysis with stepwise multivariate polynomial regression (*RSA-MPR*) 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* 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 being selected by the experimenter. For example, data collected from the field are often of this type. But laboratory data in which variables are randomized may be of this sort.

Because of the cost of controlled experimentation, *RSM* is usually limited to small datasets, and the resulting polynomials are of low-order. In fact, the references describing *RSM* rarely give examples of polynomials above degree 2. *RSM* literature also does not give examples of models using exponents other than positive integers. It does not evaluate the possibility of ratios or other negative exponent terms. If the underlying response includes ratios or some other polynomial relationship, *RSM* will produce mis-specified models. *TaylorFit* can fit data produced by *RSM*; however, it does not guide the user in choosing experimental conditions.

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. If the data have not been generated yet, and the experimenter can control the independent variables, and needs to know the best combination of independent variables to use, *RSM* is the way to go. If the data are “field” or “found” data, then *TaylorFit* RSA is the best way to explore and model the data. Having said all this, a section of this manual below will describe some design-of-experiments considerations for *MPR*.

Quick Start for Developing a *TaylorFit* Model

The following is a brief summary of the eight steps for developing a *TaylorFit* Model. These will be described with more detail in subsequent chapters.

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

If you will be dividing the dataset into separate subsets for fitting, cross-validation, and/or final validation, you should randomly sort the rows (if not a time-series).

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

It may be a good idea to make various transformations of the data, such as by differencing or taking logarithms. Although this can be done externally before producing the CSV file, *TaylorFit* has built-in functions to do this.

It is also a good idea to examine the data before beginning the modeling exercise, in order to become generally familiar with its behavior. This can be done by examining the numerical data itself, summary statistics of the data, or by performing some preliminary graphical analysis. *TaylorFit* can help with graphical analysis by making it easy to generate X-Y plots or histograms of the data. *TaylorFit* also generates summary statistics of each column in the datasets.

2 – Starting *TaylorFit*

TaylorFit is launched from your web browser at www.TaylorFit-RSA.com. The Start Window shows several buttons, including: Import Data and Open Users' Manual (this document).

Note: You can stop *TaylorFit* at any time and close the tab on your browser. But on most browsers, the next time you start *TaylorFit*, it will begin where you left off. If you want to start a new project, you must click on the *Settings* button at the upper right, then click on the *Clear Project* button towards the bottom of the Settings Pane.

3 – Entering the data into *TaylorFit*

Start by selecting "Import Data." A file directory box will open where you can select a CSV file with your data. Choose the row numbers or percentages to be partitioned into the *FIT*, *CROSS* (for cross-validation), or (Final) *VALIDATE* datasets. These will all be from the same CSV file. For example, you may select 50% FIT, 25% CROSS, and 25% Validate. If you have very few data, you may select 100% FIT and 0% for the others. A fourth file containing *ALL* data should be maintained as well.

4 – Selecting Model Fitting Parameters

Once you import data or a project, you will note that the window displays three panes, each of which expands when you move the cursor over it: The *Data Pane* at the bottom, the *Results Pane* at the top left, and the *Candidate Terms Table* at upper right.

The *Results Pane* shows three items. From left to right it includes: The *Current Model Table*; the *Global Statistics Table*; and a *Residuals Plot* showing model residual errors versus predicted values of the dependent variable. At the upper right you will see a button for *Settings*.

During model building, the user can select the model fitting parameters, which are found by clicking the *Settings* button at the top right. A box will open showing the following parameters:

- The set of possible exponents, $EXP = \{e_1, e_2, \dots\}$
- The maximum number of multiplicands, N_m
- The set of possible lag values (if the dataset is a time-series), $LAGS = \{lag_1, lag_2, \dots\}$
- The goodness-of-fit criteria to be displayed in the Candidate Terms Table and used for cross-validation

Start with $EXP = \{1\}$ and $N_m = 1$, the default values.

If the dataset is a time-series, you must start by turning on the *Time Series* switch under the *Settings Button*. Then, set $LAG = \{1\}$ (this means you must click on the zero to turn it off). Later you may add to these fitting parameters. Once these are chosen (or the defaults used), the user can start building the model.

After you are done inside the *Settings* box, click the button **Recalculate**, then click *Settings* to exit the box and return to the other display panes.

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, $p(t)$), and possibly that has meaning to the user. When a term in the *Candidate Term Table* is significant (i.e. $p(t) > \alpha$), then $p(t)$ will be highlighted in green.

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. For those terms, $p(t) < \alpha$, and $p(t)$ is highlighted in light red. Before adding new terms from the *Candidate Terms Table*, all non-significant terms should be removed from the *Current Model Table* by clicking on the X at the beginning of those terms.

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:

$$MPG = -14669 + 189.78 \cdot YR + 0.12702 \frac{YR^2}{HP} - 0.00841 \frac{A^2}{OR} - 0.01119 \cdot WT + 8.4562 \cdot 10^{-5} \cdot \frac{WT^2}{YR} + 3.7800 \cdot 10^5 \frac{1}{YR} - 0.81175 \cdot YR^2$$

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 $p(t) = 0.00177$ (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. In other words, until there are no terms highlighted in **red** in the *Current Model Table* or highlighted in **green** in the *Candidate Term Table*.

6 – Changing Fitting Parameters

At the end of each addition/removal cycle, the user may change the fitting parameters by one of these changes:

- Increasing the maximum number of multiplicands (N_m). E.g. try $N_m = 2$
- Adding new exponents to the exponent set. E.g. try $EXP = \{1, 2, -1\}$
- Adding additional lags to the lag set (if the data are a time-series). E.g. try $LAGS = \{1, 2, 3\}$

Start by clicking the up arrow next to “*Max Multiplicands*” once so that 2 shows in the associated field. This enables *TaylorFit* to test for more terms consisting of one-at-a-time or two-at-a-time independent variables multiplied together.

Repeat Step 5 (addition/removal cycle) until the model cannot be improved. Then return to settings and click the (+) next to “*Exponents*.” Use this to add exponents 2 (and the 3, -1, 4, etc. later). Repeat Step 5 again. Increment *Max Multiplicands* and add to *Exponents* repeatedly until the model new terms are not found after such an increment, then stop.

Don’t forget to click *Recalculate* after changing any parameters in the *Settings* box.

7 – Ending the Modeling Development/Validation

If the fitting parameters are changed as described above, and an addition/removal cycle does not result in new terms that could improve the model, then the user should stop model building, and perform the following steps:

- (1) Check the Goodness-of-Fit statistics for the VALIDATE sub-dataset and verify that the model performance is good. The checks should also be done by a graphical examination of residuals.
- (2) (Optional) Re-fit the model with the same terms but using ALL the data. Repeat the *GoF* and residuals checks. This step will optimize the model using all the data, which should improve its performance when used with new 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.

- (3) Save the model for future use or dissemination and for analysis and evaluation.

8 – Model Analysis and Evaluation

Once you are satisfied with your model, you can use it for prediction or forecasting, or to reveal system behavior using graphical analysis and sensitivity analysis.

In the **Results** pane, you can click on the “Export to Code” button to generate a code version of the current model in *Javascript*, *Excel*, *C++*, *Matlab*, *Python* or *R* formats. You can copy and paste from the export popup into your programs.

Try clicking the dropdown arrow in the data area by the column labeled “Residual.” You can select several options, including the display of graphical plots such as histograms, cumulative frequency distribution and, using “XY Plot,” a graph of the residuals versus any other column in the data file. *TaylorFit* also has options for computing model sensitivity for each data column. This will be described in chapter Step 8 below.

Detailed Steps for Modeling with *TaylorFit*

The following chapters expand on the previous summary of the steps for using *TaylorFit*. Despite the broad functionality, the program maintains an easy-to-use interface. It is common to be able to generate well-developed models from data in just ten or twenty minutes.

Pop-up Help: Note that when you hover over most of the headings and buttons in *TaylorFit*, a brief description of its function will pop up in a small box.

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 or other programs.

Data cleanup involves checking for problems such as:

- Missing data
- Outliers
- Influential points

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 should be disabled using the *Ignore Column* function within *TaylorFit*.

Always LOOK AT
YOUR DATA!

- b) The first row in the file may have text to indicate names or acronyms corresponding to the variables in the respective columns. The acronyms must start with an alphabetic character, or else *TaylorFit* will assume they are numerical data. If acronyms are not provided, *TaylorFit* will use default names X_0, X_1, \dots for non-time-series models. For time-series models, the default names will be $X_{0 \text{ lag}1}, X_{1 \text{ lag}1}, \dots, X_{0 \text{ lag}2}, X_{1 \text{ lag}2}, \dots$ (indicating variable₀ lag₁, variable₁ lag₁, variable₀ lag₂, variable₁ lag₂, etc.)
- c) The first column will be the dependent variable by default, although this is not required. A different column can be chosen within *TaylorFit* by selecting “Mark as dependent” in the drop-down box in the header for the column. The other columns will be independent (causative) variables.
- d) Graphically examine the data, looking for any anomalies, such as outliers and influential points; these may appear as points that stand apart from the others. Both can distort the fit. **Outliers** have unusually large or small values of the dependent variable, compared to other cases for similar values of the independent variables. The user should see if the reason for the outlier can be explained, such as due to some error in the data collections. If such a reason can be found, then the outlying point may be justifiably removed. **Influential points**, on the other hand, are far from the other data in terms of the independent variables. They may be removed from the dataset if it is ok to restrict the range of the model to the range of the remaining data. If the data occur in one or more groupings far from each other, it may be better to model each grouping separately.

Graphical examination can include:

- i) Plot the dependent variable versus each independent variable (X-Y plots).
- ii) Plot each independent variable versus each other, looking for strong correlations

- iii) Examine each variable using histogram, cumulative frequency distribution, quantile-quantile (QQ) plot, and/or autocorrelation plot.
 - iv) If the dataset is a *time series*, plot each column versus time or sequence.

This step can also be done within *TaylorFit* as described in Step 3.
- 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 dataset is a time-series, the row cannot be removed as you will then no longer have a continuous sequence of data.)
 - ii) Fill in missing data; one way to fill in missing data is to use the mean of the column. If it is a time-series, you could linearly interpolate between the previous and next values. However, you should avoid this approach if there are too many missing values. In such a case it may be better to remove that column.
- f) If there are enough data, you should plan to divide the data into two or three similar sub-datasets. This will be done in Step 3, but you should plan for it at the start. You should ensure that each sub-dataset is “comparable” in range, distribution, etc. In other words, they should “look” similar using graphical examination.

The three sub-datasets are designated:

- i) FIT: Used to compute the coefficients and *t*-statistics of the terms of the model;
- ii) CROSS: Used to compute global statistics for cross-validation;
- iii) (FINAL) VALIDATE: Used only after the best model has been built using the FIT and CROSS sub-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 all three 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 relationships between the variables. But predicting the outcome of the dependent variable using new measurements of the independent variables is risky and not scientifically valid.

In addition, there is the dataset we will refer to as “ALL”: This refers to the complete dataset combining all of the data in the FIT, CROSS and VALIDATE datasets combined.

You should determine the percentages or row numbers from the data file that you plan to use for the FIT, CROSS and VALIDATE sub-datasets. You will provide this information to *TaylorFit* in Step 3 when you upload the data. For example, you might choose to use the first 40% of the rows for the FIT data, the next 40% for the CROSS data, and the last 20% for the VALIDATE data. The proportions are a matter of judgement, subject to having enough data in each sub-dataset for fitting and validation.

How many data are enough? A rule of thumb is that each sub-dataset, and especially the FIT and CROSS sub-datasets, 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 stepwise fitting, it is possible that some columns may turn out to not contribute significantly to the prediction capability of the model, and thus may not appear in the final result.)

Is it ever OK to use the same data for fitting and validation? Sometimes: If the cause-and-effect relations are well-established, and if the model will not be used for making predictions and you are looking only to identify the strength and geometry of the relationships. However, this must be done with caution.

- g) If you have a single dataset is that is large enough to form two or three sub-datasets as just described, you should graphically and statistically examine each sub-dataset to ensure that they are similar to each other.

If not, and the dataset is not a time-series, you can shuffle the data in one of two ways:

- i) Random shuffle: For example, in *Excel*, follow these steps:
 - create an additional column and use the RAND() function to enter random numbers.
 - sort the rows by the random number column
 - delete the random number column and then save the data into a CSV file.
- ii) Stratified sampling: Suppose it is found that one of the sub-datasets differed from the others substantially in one of the variables. Again, if you are using *Excel*, perform these steps:
 - sort the dataset by the variable of concern
 - create a new column and enter the repeating sequence 1, 2, 3, 1, 2, 3, ...
 - sort on the column with the repeating sequence
 - delete the column with the 1-2-3 sequence and save the CSV file.

When you load the stratified dataset into *TaylorFit* you can use the first 33% of the data for the *FIT* data; the second third for the *CROSS* data, and the remainder for the *VALIDATE* data.

If the dataset is large and well-randomized (all the data columns have similar statistics and distributions), then cross-validation may not improve matters and could be skipped.

When complete, save the entire dataset in a single comma-separated-value (CSV) file.

Step 2 – Start *TaylorFit*

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

- *Import Data* allows the user to import data from a single CSV file containing the *FIT*, *CROSS*, and *VALIDATE* sub-datasets
- *Import Project* allows the user to import the data, fitting parameters, and current model that was previously saved
- *Open Users' Manual* opens this manual as a PDF in a separate browser tab
- *Simetrica LLC* opens a new tab at a site with additional information
- *AutoMPG* downloads an example CSV file with data describing automobile fuel economy and variables that could explain and predict the fuel economy {Provide reference}
- *Lydia Pinkham* downloads an example *time-series* data file describing a business application
- *Give Us Feedback* opens an online form for contacting us with questions and providing suggestions. You can also email us at info@simetrica-llc.com.

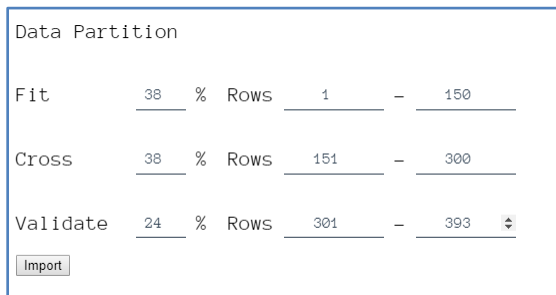
The subsequent steps below assume that the user is beginning a new project. The use of the ***import project*** button allows the user to pick up where she/he left off on a previous modeling effort.

Step 3 – Manipulating Data in *TaylorFit*

To get started, simply click on the *Import Data* button. A file directory box opens that allows the user to select the CSV data file.

Importing Data

Once you have selected and open a datafile, a *Data Partition* dialogue box opens as shown below that allows you to partition the data between the *FIT*, *CROSS* and *VALIDATE* sub-datasets. You can select the partitions by percentage of the datafile or by selecting non-overlapping row numbers.



Data Partition				
Fit	38	%	Rows	1 - 150
Cross	38	%	Rows	151 - 300
Validate	24	%	Rows	301 - 393

Import

If there are not enough data available for separate sub-datasets, the user can proceed with just a *FIT* sub-dataset. However, 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 unvalidated model may not be considered scientifically valid. On the other hand, correlations and analytical dependencies revealed by the model can be considered scientifically valid.

It is also possible to enter the *FIT*, *CROSS* and *VALIDATE* sub-datasets from separate CSV files. This is done using the *Replace Data* button in the data pane. But all three datafiles must have the same number of columns and corresponding columns must contain the same variables.

If the *VALIDATION* sub-dataset is entered from the beginning of the session, the user must avoid the temptation of looking at the model performance with those data.

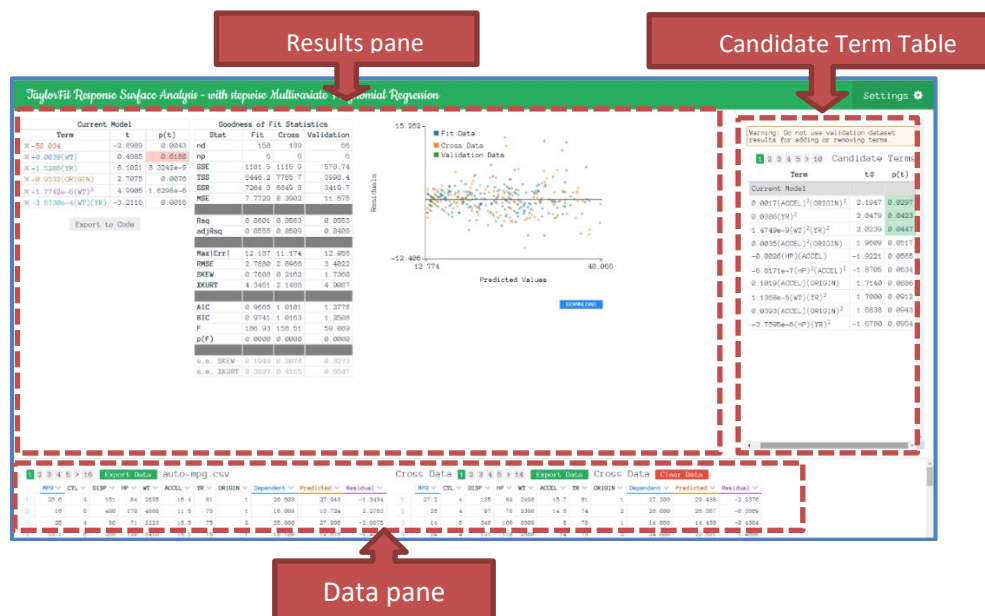
Validation data should not be used for making decisions about adding or removing terms from the model.

In fact, to avoid this temptation, it is recommended that the *VALIDATION* dataset not be imported into *TaylorFit* until model development is completed (Step 7). There is a separate button in the *Data Pane* labeled “*Import Validation Data*” for doing this.

Arrangement of the *TaylorFit* Window

After the data have been imported, you will see that the window is divided into three “panes,” each of which expands when the cursor is moved over it (see Figure 1). The **Data Pane** is at the bottom. Each sub-dataset is shown as a separate **Data Table**. Only ten rows of each dataset are shown at a time. Additional pages of ten at a time can be viewed by clicking on the numbers or arrows at the top of each table. 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, \dots

Figure 1. Arrangement of the *TaylorFit* window.



Note the other two other panes in the *TaylorFit* window: The upper right pane will be referred to as the **Candidate Term Table**. Above that is a button called **Settings**.

The larger pane to the upper left is referred to as the **Results Pane**. The *Results Pane* contains three objects: The **Current Model Table**, the **Goodness-of-Fit Table**, and the **Residuals Plot**. These will be discussed below with Step 4.

Selecting the Dependent and Independent 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 other column to be the dependent variable by clicking on the down-arrow in the heading of that column and selecting “Mark as Dependent.” Changing the dependent variable in the any dataset makes the same change in all the other datasets that have been imported.

Columns containing alphanumeric data (other than the headings in the first row, if present) are ignored. Also, the user can click on the down-arrow in the column heading in the Data Table and select “**Ignore column.**” Ignored columns will not be used in generating candidate terms for the model. To reverse this action, click on “**Show column**” in the heading drop-down menu.

Note that *TaylorFit* automatically generates three columns in each *Data Table* to the right of the data you entered as follows:

- **Dependent** – This column repeats the column selected as the dependent variable.
- **Predicted** – This is the value predicted by the current model using the independent variables from the corresponding row.
- **Residual** – Also known as “error” values, this is the measured dependent variable minus the predicted variable.

Data Transformation

Often a modeler will want to make mathematically transform a variable in the *Data Table* before using it for modeling. *TaylorFit* has the following transformations built-in: logarithmic (base 10), *k*-order difference, standardized, or scaled. To transform a column, click the down arrow in the heading of the column and select **transform column**. You can then select the desired transform. When this is done, a new column will appear to the right of the data containing the transformed data, and the original column will be marked to be ignored. For example, you cannot perform the regression on both X_3 and $\log(X_3)$.

You can use the graphical tools to examine the transformed column. If you decide not to use that particular transform, you can use the drop-down menu on the transformed column to **Delete Transform Column**. The *Transform Column* and *Delete Transform Column* menu items only appear in the *FIT* dataset but affects all the datasets. You can also transform a transformed column, if desired!

There are a number of reasons for using transformations. The meaning and possible motivations for using these transformations are as follows.

Logarithmic transform

If a variable's range covers many orders of magnitude, or its distribution is skewed to the left, it might be better to fit the logarithm of the variable instead of the variable itself. This transformation is also used with data that are always positive and non-zero, but for which the standard deviation is almost as large, or even larger, than the mean. This often occurs, for example, with measurements of pollution concentrations in the environment. Most levels may be low, but some will be much higher than the average. Logarithmic transformation may produce a more symmetric distribution that is easier to model.

k-order Differencing

This may be appropriate for time-series data. With such data, if the dependent variable changes slowly, time-series models may have very high values of R^2 . But only reflects the strong predictive effect of the previous value. For example, to predict today's temperature, you would do very well to just use yesterday's temperature, and you would get a very high R^2 . A more meaningful prediction would be to predict the change in temperature.

If you choose *k-order Difference*, you can select the value of k in the field next to the button. If $k = 1$ the transformation is a 1st-order difference. If $y_{i,k}$ is the value of y_i at time step k , then the 1st-order difference of $X_{i,k}$ is $D_{i,1} = y_{i,k} - y_{i,k-1}$. Higher-order differences are formed by taking the difference of the next-lower order difference. I.e. $D_{2,k} = D_{1,k} - D_{1,k-1}$. In general, $D_{n,k} = D_{n-1,k} - D_{n-1,k-1}$.

Standardize

If different columns have very large or very small numerical values (i.e. are in scientific notation with large positive or negative exponents), this transformation will "normalize" them. This transformation is also used to avoid having model coefficients that are very large or very small. The original values will have the mean of the column subtracted, and then be divided by the sample standard deviation. This gives the transformed column a mean of zero and standard deviation of 1. Values above the mean will be positive, and those below the mean will be transformed to negative numbers, which will prevent the column from being used with negative exponents.

Scale by RMS

With this transformation, each value in the selected column is divided by the root-mean-square of that column. This has similar motivation to *standardize* but preserves the sign of the variable so it can be used with negative exponents.

Examining the Data – Plotting and Column Statistics

A professor once said, “Always look at your data.” In other words, don’t rely only on computed statistical calculations. By their nature statistics hide some of the information in the data. In fact, the *degrees of freedom* is the number of additional pieces of information that would be needed in addition to the model to fully describe the data. You should study the raw data, plus examine them graphically and using summary statistics before you start the modeling process.

Start by scanning through the numerical values. Of course, with very large datasets, it may not be possible to look at every individual value. But one should at least examine the distribution of each independent and dependent variable. This can be done both graphically and by using numerical statistics. We will call these the column statistics because *TaylorFit* computes them for each column in the *Data Pane*.

Column statistics include the mean, standard deviation, root-mean-square (rms), and various percentile values such as the maximum (the 100th percentile), the median (the 50th percentile), and the minimum (the 0th percentile). *TaylorFit* computes all these statistics for each column in the dataset and displays them just below the column headers in the data pane. Check that these values are similar for the *FIT*, *CROSS*, and *VALIDATE* datasets.

Graphical examination of the distribution can start with the histogram and/or the cumulative distribution (which is the integral of the histogram). Next, look at how the variables compare with each other. Graphically, this means creating X-Y plots of each variable versus each other variable. (For N variables, there will be $(N^2+N)/2$ combinations to check.) Numerically, it involves creating a cross-correlation table, which can be done in a spreadsheet or similar programs. Other plots that may be of interest are the quantile-quantile plot (Q-Q Plot) and the autocorrelation plot. Although these may be of interest for examining the data, they are of most use in examining model residuals. Accordingly, these two plots will be discussed in Step 7.

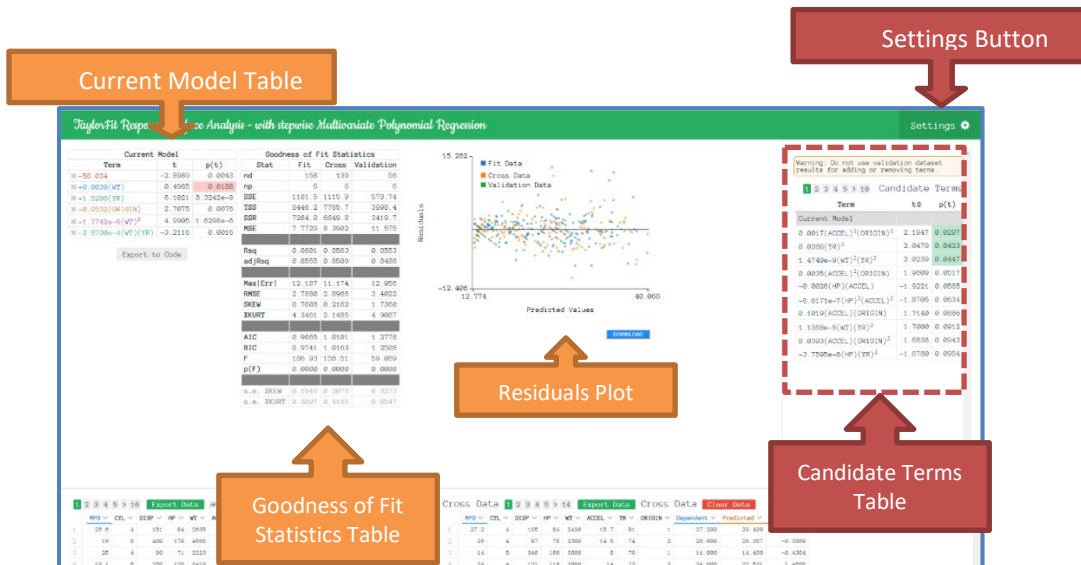
If the statistical or graphical analysis shows differences between the *FIT*, *CROSS*, and *VALIDATE* datasets, go back to the data wrangling step where you either randomize or do stratified sampling (described above).

Step 4 – Initial Settings

As previously mentioned, the *TaylorFit* window displays three “panes,” each of which expands when you move the cursor over them (see Figure 1):

- Across the bottom of the window is the *Data Pane*, discussed in Step 3.
- Above the *Data Pane* are two panes. Towards the left is the *Results Pane*. This pane includes three items (Figure 2):

Figure 2. Parts of the Results pane.



- 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 occurring by chance, labelled $p(t)$. The button below this Table “Export to Code” opens a box with the current model in a form that can be used in *Javascript*, *Excel*, *C++*, *Matlab*, *Python* or *R* formats. You can copy and paste from the popup into your programs.
- The *Goodness of Fit Statistics Table* – This Table shows summary statistics computed separately for the *FIT*, *CROSS*, 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. Residuals for the *FIT* dataset are shown as blue points; residuals for the *CROSS* dataset are shown as orange points. If a *VALIDATE* file has been imported, the residuals associated with those data are shown as green points. Clicking on the legend inside the plot turns the corresponding set of data points on or off. For example, if you only want to see the residuals for the *FIT* data, click on the legend for *Cross Data* and *Validation Data*.

- On the right side above the data pane is the *Candidate Terms Pane/Table*.
- At the upper right is the *Settings Button*. This shows buttons opens a box that allows the user to select statistics for display or for use as fitting criteria, and to select the fitting parameters described below. These will be described in detail below.

The Settings Button

The *Settings Button* opens a box in which the user can choose:

- Buttons for *Recalculation* of the *Candidate Terms Table* based on the current fitting parameters, and for clearing the fitting parameters (*Clear Settings*)
- The global *Goodness-of-Fit (GoF) statistics* for comparing candidate terms
- $N_{Candidates}$: The number of possible candidate terms (based on the maximum number of multiplicands, the number of exponents considered, and the number of lags considered if a time-series). A section at the end of this manual shows how this number is computed.
- The *fitting parameters* (multiplicands, exponents, and lags), which include:
 - *Max Multiplicands* = N_{max} : This is the maximum number of independent variables that can be multiplied together within a single candidate term. E.g., if there are eight independent variables (X_1, X_2, \dots, X_8) and $N_{max} = 2$, the candidate terms will only show terms having two of these at a time, such as $X_1 \cdot X_3$ or $X_5 \cdot X_8$, but not $X_1 \cdot X_3 \cdot X_4$
 - *Exponents*: a set of possible exponents for use in forming candidate terms
 - *Lags* (time-series only): a set of possible lagged values for forming candidate terms
- Buttons for turning *time-series* on and off, and fields for changing the display precision, and a field for changing the value of α
- Options for clearing or saving the project (described in Step 7)

When done inside this pane, click on the *Settings* button again to close the pane.

(Although if you have changed the fitting parameters, you should click on *Recalculate* first.)

Term, Dataset and GoF Statistics

The user can choose which *Goodness-of-Fit (GoF) statistics* to display in the *Candidate Terms Table* to help with selecting the terms and the overall model. The use of these statistics is described here. Their formal descriptions and computational formulae are provided in a later chapter.

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 one of the most important statistics during model building. It is a measure of how significantly a term contributes to the fit. For example, if there is a large amount of data (at least 100 “degrees of freedom” – see below) then a value of $t = 1.98$ indicates there is a 5% probability ($\alpha = 0.05$) or less that this term occurs

because of chance; $t = 2.28$ corresponds to a 2.5% probability ($\alpha = 0.025$). Some software packages compute the “*F*-to-add” in multiple regression. This gives the same information about the statistical significance of a term.

$p(t)$: The two-tailed probability *for* the hypothesis that the coefficient is equal to zero. Typically, we would add a term to the model only if $p(t) \leq \alpha$ (typically $\alpha = 0.05$) for that term and remove a term from the current model if its $p(t) > \alpha$.

Other statistics are shown in the second Table in the Results Pane. This is the *Global Goodness of Fit Statistics* Table. The values shown in this Table are global statistics; that is, they relate to the entire model, rather than individual terms in the model.

The *Dataset Statistics* are computed separately for each sub-dataset – *FIT*, *CROSS* and *VALIDATE*. All the other statistics are computed from these plus the *SSE*:

- nd : The number of data points; usually the number of rows in the data file.
- np : The number of coefficients in the model, including the intercept if there is one. Essentially, it is the number of terms in the model.
- df : Although not shown in the Table, df is easily computed as the number of degrees of freedom of the residuals; $df = n_d - n_p$
- 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
- TMS : *Total mean square* = $TSS/(n_d - 1)$ (Used below; not shown in the Table)

The other statistics are termed *Global 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. Most of these are based on the *residuals*, also known as the *errors* (e_i):

$$e_i = y_i - \hat{y}_i$$

where \hat{y}_i are the model predicted values for each measured dependent variable, y_i .

The *GoF* statistics are:

- SSE : Sum of squares of the errors (or residuals); $SSE = \sum (e_i)^2$
- MSE : *Mean square error* = SSE/df ; this is an estimate of the sample variance of the residuals
- $RMSE$: *Root-mean-square error*: The square root of MSE ; this is an estimate of the sample standard deviation of the residuals
- $Max|Err|$: The largest of the absolute value of the residuals
- MSR : *Mean square of the regression* = $(TSS - SSE)/n_p$ (Used below; not shown in the Table)
- RSQ : R^2 , the square of the multiple correlation coefficient; $R^2 = 1 - SSE/TSS$
- $adjRSQ$: *Adjusted R^2* – similar to R^2 , but takes into account the degrees of freedom; $adjusted R^2 = 1 - MSE/TMS$

<i>F</i> :	The <i>F</i> -statistic = MSR/MSE ; similar to the <i>t</i> -statistic for a term, but applies to the overall fit. A larger value indicates a better fit.
$p(F)$:	The probability of <i>F</i> being this large or larger by chance. Put another way, it is the probability that <i>MSE</i> is smaller than <i>MSR</i> by chance. The smaller the value, the better the fit.
<i>AIC</i> :	<i>Akaike Information Criterion</i> = $\log_{10}(MSE) + 2 \cdot n_p/df$
<i>BIC</i> :	<i>Bayesian Information Criterion</i> = $\log_{10}(MSE) + \log_{10}(n_d) \cdot n_p/df$
<i>SKEW</i>	Skewness
<i>s.e. SKEW</i>	Estimated standard error of the skewness
<i>XKURT</i>	Excess kurtosis = kurtosis – 3.0
<i>s.e. XKURT</i>	Estimated standard error of the excess kurtosis

All the *GoF* statistics are displayed in the *Global Statistics Table* in the Results Pane. But they will only show in the *Candidate Terms Table* if you select them. 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. The top row of the selected *GoF* column in the *Candidate Terms Table* shows the current value of that *GoF* for the *CROSS* sub-dataset, if one has been imported. This is used in cross-validation, as described below. The *Candidate Terms Table* can be sorted by any column by clicking on the corresponding heading.

Note that you might occasionally find the value of R^2 or $adjR^2$ to be outside the range of 0.0 to 1.0, which seems illogical for a squared value. 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 *CROSS* or *VALIDATE* datasets if the current model predictions are worse than just using the average of the dependent values from the *FIT* data as predictor.

This occurs because *TaylorFit*'s computational formula is $R^2 = 1 - SSE/TSS$. If $SSE > TSS$ then $R^2 < 0$. 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 corresponding dataset. The sum of squares of the "errors" (*SSE*) in this worst-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: $\hat{y} = 0$. 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 decision is up to the user. *TaylorFit* always starts by

including the intercept as the first term in the model. However, the modeler can choose to remove it at any time by clicking on the *X* at the left of the term in the *Current Model* Table.

If you find that the *VALIDATE* and/or the *CROSS* sub-datasets have much poorer *GoF* statistics than the *FIT* sub-dataset, it may be that these sub-datasets do not equally represent the distributional properties of each other. For example, this could occur if the one of the sub-datasets contains outliers or influential points not contained in the other sub-datasets.

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. This allows 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, $EXP = \{exp\}$
- The set of possible lags, $LAGS = \{lag\}$

The default values for the *fitting parameters* when *TaylorFit* is first started are:

$$N_m = 1; EXP = \{1\}; LAGS = \{0\}$$

If these settings for the *fitting parameters* are used with *TaylorFit*, the result will be a *Multilinear Regression (MLR)* model. *TaylorFit*'s power comes from extending the *fitting parameters* to produce *Multivariate Polynomial Regression (MPR)* models.

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. We will only use the term "coefficient" for the multiplier at the front of each term in the model.

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*, *Q·R*, or $R·S^2$, 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 to remove it, according to needs and preference. A maximum number of multiplicands of 2 or 3 is often sufficient.

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. This ability to include exponents that are not positive integers is also known as *fractional polynomial regression*. In most cases you will just want to use positive and negative integers.

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 $EXP = \{-1, 1, 2\}$, 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} . If one wishes to include such terms the set of exponents should be expanded to, for example, $EXP = \{-2, -1, 1, 2, 3\}$.

LAGS: A lag is an index that tells *TaylorFit* to try independent variables from previous time periods. Use of lags is described in the next section.

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 one or two multiplicands.*

There is also a field under the *Settings* button to change the “*Display Precision*.” This is the number of digits that *TaylorFit* displays for the model coefficients and all statistics. Internally, *TaylorFit* stores these values with about 16 digits of precision. Changing the *Display Precision* does not alter the internal storage or the precision used in the model produced by the *Export to Code* button. Also, you can see the full precision for the coefficients of the current model by clicking on the *Export to Code* model just below the *Current Model* Table.

Setting TaylorFit up for Time-Series Analysis

If the dataset is a time-series, you must start by turning on the *Time Series* switch under the *Settings Button*. Then, you should set the lags.

As was previously mentioned, 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 ($LAGS = \{0\}$). This means that only independent variables on the same row as the dependent variable are used in the prediction.

To start with, use $LAG = \{1\}$. To do this, click the + sign in the *LAGS* field and enter 1. Then click on the zero lag to disable it. There may be cases where the zero should not be disabled. This will occur only if the independent variables *on the same row as the dependent variable* would be expected to be available as predictors for the dependent variable on that row. For example, if you want to predict a company’s sales for a year, you might include the advertising expenses for that year, since they may be budgeted, and therefore known, before the sales figures are known. But more often, independent variable data contemporaneous with the dependent variable is not known ahead of time, and the lag zero should be disabled.

Making subsequent changes to *LAGS* during the modeling process is discussed further in Step 6.

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 effectively makes that statistic the criterion for choosing terms to add into the model.

An extensive comparison of fitting strategies using three different datasets found that using *MSE* (or, equivalently, *adjRSQ*) as a cross-validation criterion produced the best results, as measured by performance with a separate final validation dataset.

Next, the user should select the *fitting parameters* (N_m , *EXP*, *LAGS*). A strategy for selecting initial fitting parameters is to set $N_m = 1$, $EXP = \{1\}$. If the data are a time-series, use an initial setting $LAGS = \{1\}$.

Start with a multilinear regression model, then go from there.

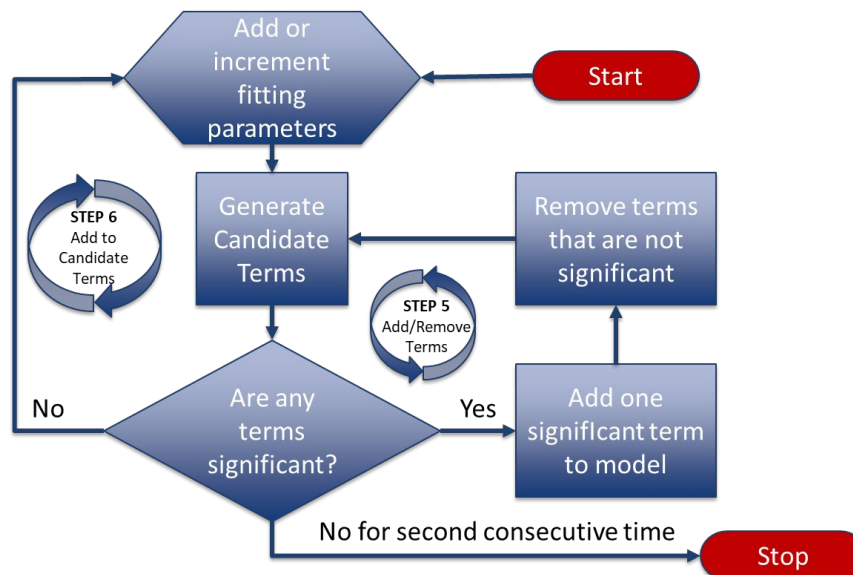
Next, proceed to Step 5. The initial settings described above will result in linear models. If a non-time-series, the result will be a multilinear regression (*MLR*). If a time-series, the resulting model will be what is known as a first-order autoregressive (*AR(1)*) model. Subsequently, as described below in Step 6, the *fitting parameters* are increased in increments, say by changing the *Maximum Multiplicands* to 2, fitting again, then sequentially adding exponents (and additional lags if a time-series).

Step 5 – Stepwise Addition and Removal of Terms

Once the data are loaded and initial *fitting parameters* have been selected, the user can begin building the model by adding and removing terms iteratively until the model cannot be improved by adding or removing any single term. Only then should the *fitting parameters* 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 any new candidate terms that significantly improve the model. The flowchart in Fig. 1 illustrates the steps involved. Step 5 is an “inner” loop of the procedure; Step 6 is the “outer” loop. These two loops are the core of the *TaylorFit* modeling procedure.

Figure 1. Flowchart for stepwise regression with multivariate polynomials.



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-validation is not being done. Recall that when a modeling project is started for the first time, the constant (i.e. intercept) term is added automatically. The user can choose to remove it at any time.

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 known as α (usually 0.05, or 5%). Notice that the program highlights $p(t)$ values that are less than α in **green background** to make it easier to select. If there

is no separate cross-validation and/or final validation dataset, then a more stringent addition threshold value for α , such as 2.5%, and/or one of the relatively conservative *GoF* criteria, could be considered.

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 to give preference to a simpler model.

To select a term 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.

After each new term is added to the *Current Model*, the user should go 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 the value of α . Notice that the program highlights $p(t)$ values in the *Current Model Table* that are greater than α in light red background to make it easier to select. Start with the term with the largest value of $p(t)$ and continue until there are no more terms with $p(t) > \alpha$.

Note: Sometimes, especially if cross-validation is not being used, the user may find the same terms being sequentially added to the model, and then removed, and then added again, in an infinite loop. A way to break this loop is to use a less strict value of α for removing terms than for adding them. For example, set α to 5%, but only remove a term if its $p(t) > 10\%$ (0.10).

Iterate: After the removal step is complete, 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 continues to Step 6 – Changing Fitting Parameters.

If after the *Setting Parameters* have been incremented in Step 6 and no *Candidate Terms* meet the criteria for addition, then the model development is complete (for the given choice of criteria). The user should then **stop** the modeling exercise and go to Step 7 for model validation.

Cross-validation

If there are enough data to form two or three sub-datasets, it is recommended to enter them separately as *FIT*, *CROSS*, and *VALIDATE* datasets. Refer back to Step 1 for the discussion on ensuring similarity between the three sub-datasets and the quantity of data needed. 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 *CROSS* 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 a term is selected in the addition step.

Note: Experiments with various datasets have found that if the dataset is large and well-randomized, then the *CROSS* dataset may contain the same information as the *FIT* dataset, and cross-validation might not improve the model performance in final validation. Ideally, the *CROSS* dataset should contain data that were collected independently of the *FIT* dataset.

For cross-validation, two criteria must be satisfied simultaneously. Specifically, $p(t)$ for a candidate term must be less than the *significance level* α , and the selected *GoF* statistic for the *CROSS* dataset with the candidate term included must be better than the same *GoF* statistic for the current model.

Note that the *GoF* statistics in the *Candidate Term Table* for each term indicate what those statistics would be if that term were added to the current model. The *GoF* for the current model without that term is displayed in a row at the top of the *Candidate Terms Table*.

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* sub-dataset. *However, global statistics (e.g. SSE, MSE, R^2 , $Max|err|$, F -statistic, AIC and BIC) in the Candidate Terms Table are always computed using the *CROSS* sub-dataset if one has been designated, and the *FIT* file if not.*

As an example, look at the *Candidate Term Table* below. Note that all the values of $p(t)$ are very small and are highlighted in green. But only the first term in the list has its value highlighted in the column for the F -statistic. Its value of F is 111.19. This is greater than the value of F for the current model applied to the *CROSS* sub-dataset, which has $F = 97.046$. Thus, the term for (WT)(ACCEL) meets the dual criteria for being added to the current model. On the other hand, look at the second term, (WT)(ORIGIN). Its t -statistic is significant, but the value of F (96.547) is lower than for the current model (97.046). Although still statistically significant, it is less so, and thus this term is judged to not improve the prediction capability with the *CROSS* sub-dataset. It doesn't satisfy the dual criteria, so this term should not be added into the *Current Model*.

Candidate Terms			
Term	F	t	p(t)
Current Model	97.046		
-2.6516e-4(WT)(ACCEL)	111.19	-7.3247	1.4614e-11
-0.0027(WT)(ORIGIN)	96.547	-6.3532	2.4909e-9
0.0472(ACCEL)(YR)	91.110	6.8359	2.0251e-10
0.3968(YR)(ORIGIN)	82.922	5.8908	2.5145e-8
-0.0020(DISP)(ACCEL)	79.212	-6.1606	6.6020e-9
-0.0039(HP)(ACCEL)	77.488	-4.3134	2.9344e-5
-0.1120(CYL)(ACCEL)	76.802	-5.8317	3.3544e-8
-0.0348(HP)(ORIGIN)	73.967	-3.7373	2.6569e-4
-0.0199(DISP)(ORIGIN)	68.770	-5.0863	1.0968e-6
-1.7797e-4(CYL)(WT)	67.788	-4.6137	8.5434e-6

The same cross-correlation procedure can be done with some of the other global statistics. Use of MSE or $adj-R^2$ will generally be less conservative (produce models with more terms). Use of F , $p(F)$, AIC or BIC tend to produce simpler models. Using $Max|Err|$ results in what is known as *mini-max optimization*.

The *Settings* pane allows the user to add other global statistics to the *Candidate Term* Table such as *SSE*, *SSR*, R^2 , *SKEW*, and *XKURT*. However, these are just for monitoring purposes. They should not be used for selecting terms to add to or remove from the model.

AutoFit

Note: This feature is a preliminary β -test version. Clicking this button will initiate a sequence in which *TaylorFit* will automatically and iteratively perform step 5 until there are no significant terms in the *Candidate Term Table* and no insignificant terms in the *Current Model Table*.

CAUTION: This feature has not been fully debugged. It sometimes leaves insignificant terms in the *Current Model Table*, which the user must remove. Also, if there are a large number of candidate terms and the dataset is large, the program may take a very long time before it halts, and may sometimes hang the program. If this occurs, see the section *Known Bugs* for the procedure to halt the program.

Step 6 – Changing Fitting Parameters

As mentioned above, the real power of *TaylorFit* comes from the user's ability add to the fitting parameters that result in the generation of multivariate polynomial candidate terms. This enables the creation of models with complex response surfaces to more accurately capture the behavior contained in the data, and with less bias than linear models.

After each time an addition/removal cycle is completed and the model cannot be improved either by adding a term from the *Candidate Term* Table, or by removing a term from the *Current Model* Table, the user should try expanding the domain of candidate terms. This is done by increasing the maximum number of multiplicands, and/or by adding to the list of exponents, and/or (if the data are a time-series) adding to the list of lags. This step is illustrated by the outer loop in Figure 1. 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, it is suggested that the user should start with a linear model. In the case of non-time series, this is the 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 multiplicands and the list of exponents can be increased. There is no single way to do this. This decision 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 (which is displayed on the *Settings* page as $N_{\text{Candidates}}$) 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, its worth repeating as stated before, “data are expensive; computations are cheap.” If a model produces useful results, it may be worth waiting for.

Changing α

The α is the threshold probability level used for tests of significance for the t -statistics. These statistics are deemed statistically significant if their calculated values (in the *Candidate Term Table* and the *Current Model Table* are less than α . The user can change the value of α on the *Settings* screen. This does not change any calculations within *TaylorFit*. The only thing that changes is the color highlighting in the *Candidate Term Table* and the *Current Model Table*:

- Values of $p(t) \leq \alpha$ in the *Candidate Term Table* are highlighted in green.
- Values of $p(t) > \alpha$ in the *Current Model Table* are highlighted in red.

The default value of α is 0.05. The user may want to use a lower value, say, 0.025 or even 0.010, which will result in simpler models with fewer terms. In a comparison of results with fitting MPR models to three different datasets, we obtained best results (measured using independent final validation *MSE*) with $\alpha = 0.05$ for two of them, and $\alpha = 0.025$ with one of the three. The user may wish to try developing models with $\alpha = 0.05$ and then again with $\alpha = 0.025$. The two models could be compared using the *MSEs* for the final validation dataset, and the better model selected for use.

Adding to the Lags (Time-Series Only)

If the dataset is a time-series, then the user should select the lags to be considered by *TaylorFit* for forming candidate terms. As with the selections of the other settings, the user should start simple by selecting only $LAGS = \{1\}$. 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.

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 $LAGS = \{1,2,3\}$. Et cetera.

The user may choose to skip lags to incorporate "seasonality" effects. For example, if the data represent daily measurements, and $LAGS = \{1, 2, 7\}$, then *TaylorFit* will test independent variables from yesterday, the day before yesterday, and one week ago for their ability to predict today's dependent variable. This could capture day-of-the-week effects.

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 two columns – Sales (S) and advertising budget (A), and 54 rows for successive years from 1907 to 1960. These data can be loaded automatically from the *TaylorFit* start page by clicking the button labeled "Lydia Pinkham – Load Example Data." 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 number of LAGs in the *Fitting Parameters*. Again, this is up to the user, and is part of the art of modeling. A suggestion is to increase the number of lags repeatedly until no new terms are significant before increasing N_m or adding to *EXP*, and again after each time those other fitting parameters are changed.

Step 7 – Ending the Model Development/Validation

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* (N_m , $\{Exp\}$ and $\{Lags\}$) have just been augmented. Before moving on to final validation, the current model should be examined statistically and graphically as described in the next two sections. If the models are found inadequate in some way, the modeling effort can be continued.

Suppose no terms are found to be significant at all. In this case it may be concluded that the proposed independent variables do not in fact have predictive power for these data!

Throughout the model-building process, it is a good idea to keep an eye on the plot of residuals versus predicted values that is displayed in the *Results Pane*. Besides model accuracy, a goal of modeling is for the residuals to be random. If trends are visible in this plot, then the model-building process is not complete. Note that the residuals for the *FIT*, *CROSS*, and *VALIDATE* sub-datasets are shown on the same plot but in different colors. Clicking on the corresponding legend can turn individual sub-dataset results on and off.

Another problem that can be revealed by this plot is the presence of outliers or influential points. **Outliers** show up as points on the *residuals plot* that are vertically far away from the other residual points, and themselves have a large residual value, compared to the other points on the plot. **Influential points** also tend to be far from the other points on the plot, but along the x-axis instead of the y-axis. However, they tend to have very small residuals.

A modeler may wish to keep models relatively simple. This can be done by using a more conservative cross-validation criteria, and/or a lower significance level (e.g. use $\alpha = 0.025$ or 0.010).

The models developed by *TaylorFit* may be described as “sub-optimal.” *TaylorFit* does not explore all the possible MPR models. But it will find ones that are “pretty good.” If the modeler wants to explore a wider family of models to seek improvement, here is a strategy to try: Add a number of terms that are not statistically significant, but are either close to significant, or just because the modeler judges those terms to have potential predictive power. Then return to the add-and-remove cycle until again the model cannot be improved with a single change. (Of course, by this time any added non-significant terms will have been removed.) It is likely that you will have found a path to a different model, possibly with better performance.

The following two sections suggest ways to examine the model fit using statistical and graphical approaches. These approaches can be used at all stages of the model development process, including during model identification, when deciding whether to stop model development, when performing final validation, and for checking the final re-fitted model.

Examine the Statistical Goodness-of-Fit

The global *Goodness-of-Fit* statistics shown in the *Results Pane* provide a variety of useful measures. Most can be used for comparing competing models, although the F -statistic, R^2 , and adjusted- R^2 provide “absolute” measures.

The MSE is one of the fundamental values at the root of the calculation of adjusted R^2 , the F -statistic, the AIC , and the BIC . The MSE is an unbiased estimate of the variance of the residuals. The square root of the MSE is the root-mean-squared-error ($RMSE$), which is an estimate of the residual standard deviation. The $RMSE$ will have the same scale and units as the dependent variable, which makes it easy to compare with the data.

The $RMSE$ also provides a rough way to estimate the 95% confidence interval of the data as $\pm 2 \cdot RMSE$. To be precise, the factor 2 should be replaced by the critical value of t , which depends upon the degrees of freedom. If $df < 100$, the critical value of t is 1.96.

The “confidence interval of the data” is properly called the “prediction interval.” This is distinguished from the confidence interval around the model predictions. This somewhat confusing nomenclature is discussed more fully in the section below on Step 8. Also, it should be noted that both the confidence interval and the prediction interval will vary depending on the values of the independent variables. *TaylorFit* can take this into account to compute the exact confidence interval and prediction interval for each row in the dataset as described in Step 8.

The maximum absolute value of the error ($Max|Err|$) is sometimes of interest. One such situation is when *TaylorFit* is being used to fit an MPR model to results of a different numerical model. The data in this case have no true “error.” The value of R^2 may be expected to be very close to 1.0 if enough terms are included, and the maximum absolute error may be a desired objective. Using $Max|Err|$ as the cross-validation criterion may be appropriate in this situation.

The R^2 is the best-known and most commonly used *GoF* criterion for regression, although it (like SSE) is likely to result in over-fitting if used as a criterion for adding terms to the model. The $adj-R^2$ is an improvement since it includes a penalty for model complexity. This makes it a better criterion. Incidentally, $adj-R^2$, MSE , and $RMSE$ are equivalent to each other for purposes of optimization, and should produce the same model if used as a fitting criterion.

The F -statistic also penalizes model complexity. It has the advantage that it can be used in a statistical test of the hypothesis that the modeled variance is greater than the residual variance. That is, that the model as a whole is a significantly better predictor of the dependent variable than just using the mean of the dependent variable. This test is done by computing the probability that a larger value of F could occur by chance. This probability is denoted $p(F)$. As long as $p(F) < \alpha$, the model may be accepted. Generally, the higher the value of F , the lower will be $p(F)$, and the more certain one may be of the statistical significance of the model. Besides serving for this test, F can be used as a criterion for cross-validation. The higher the value of F , the better the model. This is a more conservative criterion than using, say, MSE , and will result in a simpler model (with fewer terms). Note that the value computed for $p(F)$ depends on the residuals being normally distributed and independent of each other (as tested below). But, even if they do not satisfy this condition $p(F)$ can be used as a relative probability for comparing fits.

AIC and *BIC* are criteria based on information-theoretic considerations. They may be used to compare models with different number of terms. If different models have the same number of terms, then these statistics correspond to comparison based on *MSE*. As with the *F*-statistic, they are fairly conservative criteria that may result in simpler models.

Skewness and *Excess Kurtosis* are statistical parameters that can help determine how well the distribution of the residuals correspond to the normal, or Gaussian, distribution. Skewness measures asymmetry of the distribution; a large excess kurtosis indicates a greater probability of extreme values (“fat tails”). Ideally both values should be close to zero. The *Goodness of Fit Statistics* Table also includes the standard error of the skewness and kurtosis – s.e. SKEW and s.e. XKURT, respectively. As an approximation, both statistics should be within 2 times their standard error of zero or, to put it more simply, the absolute value of each statistic should be less than twice their standard error. For example, if s.e. SKEW = 0.1205, then SKEW = -0.2534 is significant skew, but SKEW = -0.1727 is not.

Summary statistics should be examined for predictions, residuals and terms. These include: mean, standard deviation, skewness, excess kurtosis, and various percentile values such as the maximum (the 100th percentile), the median (the 50th percentile), and the minimum (the 0th percentile). These quantities (except for skewness and kurtosis) are displayed by *TaylorFit* in the *column statistics* area of the *Data Pane*.

Examine the Graphical Goodness-of-Fit

One of the most popular ways to graphically display goodness of fit in the literature is by a plot of predicted values on the Y-axis versus observed dependent data on the X-axis. When is this appropriate? **Never!** Instead, one should plot observed versus predicted. The reason is that observed values equal predictions plus residuals. This will distribute the data vertically around the line, rather than horizontally. Distributing the data horizontally will skew the plot and make it appear to have outliers at the upper and lower extremes. (See Draper and Smith reference for more.)

An even better graphical approach is to plot the residuals on the y-axis and predicted values on the x-axis. This is what is done in the plot that shows continuously in the *Results Pane* of *TaylorFit*. One goal of modeling is for this plot to appear as a random scatter of points, with no apparent trends.

Furthermore, the residuals should be plotted versus each of the dependent variables as well as the sequence in which they appear in the data. This can be done using the XY Plot function in the column headers in the *Data Pane*. Click on the heading of the *Residual* column and select XY

Never plot predicted versus observed!

Plot. What appears is a plot of the residuals on the y-axis, and “index” on the x-axis.

“Index” means that the values are plotted in sequence in the order they appear in the

corresponding sub-dataset. Click on the down-arrow next to “Index” within the plot to select any of the other columns for the x-axis to examine the residuals versus each of the independent variables. None of these plots should show obvious correlations. Do not use the dependent variable as the x-axis value for the reason described above.

Several specialized plots are included to assist in verifying that the residuals are independent and normally distributed. These include the histogram, the CD plot, the Q-Q Plot, and the autocorrelation plot.

The *histogram* is displayed as a bar graph. The number of bars (or “buckets”) in the bar graph can be increased or decreased by clicking on the “+bucket” or “-bucket” buttons in the graph. This plot should approximate a symmetrical “bell-shape” or Gaussian shape.

The *cumulative distribution* (CD) plot is the integral of the histogram but shows all the data. If the data are Gaussian the CD plot will have a sigmoidal shape (i.e. be “S”-shaped).

The final check on Gaussian distribution is the *quantile-quantile* or “QQ” plot. This is a direct comparison to the Gaussian distribution. If the residuals are perfectly normally distributed, they will fall exactly on the straight line shown. Some deviation is acceptable though, especially at the extreme values.

The last graphical examination technique built into *TaylorFit* is the *autocorrelation* plot. This plot is essential to checking residuals in time-series models but may also be useful with non-time-series. This is essentially a bar-graph of correlation coefficient of a residual with its lagged value. The correlation coefficient is on the y-axis and the lag on the x-axis. Thus, the bar at lag 2 represents the correlation of residuals at any time step with the residual from two time steps before. The orange and green lines are the upper and lower (respectively) confidence intervals for the correlation coefficient. When the model is complete, all bars should fall between the upper and lower confidence limits. Autocorrelation is not expected with non-time-series data, especially if the data were randomized before being exported into *TaylorFit*. But it might occur if the data were collected with the dependent and/or one or more of the independent variables exhibiting a strong trend.

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 *CROSS* sub-datasets are used for *identification* (determining which terms should be in a model) and *fitting* (determining model coefficients). (Note that with *TaylorFit*, only the *FIT* sub-dataset is used to compute model coefficients. Both *FIT* and *CROSS* sub-datasets can be used for model identification.) When model identification and fitting are complete, the final step is validation – determining how well the model works with independent data.

Final validation is done as follows: First of all, import the validation data. If this was not done at the start of the modeling project, it can be done when the model is completed. In fact, it is

Independent final validation is the gold standard for modeling

better to wait until the modeling is completed, to avoid the temptation of looking at the performance of the model with the validation data during the model identification process. If the validation data were not imported at the beginning, it can be done now by clicking on the

button in the *Data Pane* called “Import Validation Data.” Notice that a new column appears in

the *Goodness of Fit Statistics* Table in the *Results Pane* with the heading “*Validation*.” This shows the statistics used to judge the validity of the current model. Only $p(F)$ is meaningful on an absolute scale. As long as it is less than α , then the model is statistically significant.

Often data are scarce and there are not enough to partition the dataset into two, let alone three, parts. In such cases it may be necessary to do without a separate *VALIDATE* dataset, or to use a single dataset for both *FIT* and *CROSS* purposes. However, in this case the resulting model should be used with extreme caution for making predictions, and the modeler should emphasize these limitations to whomever may use the model or its results. 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. In statistics-speak, another way to describe this is to say that without final validation the model is representative of the *sample* and should be used with caution to make inferences about the *population*.

There is a situation in which cross-validation and final validation are not required. This is when *TaylorFit* is being used to generate an MPR model of data that were themselves generated by a model. Here, all relevant independent variables are known, and there are, in a fundamental sense, no errors. (There may be a small amount of error due to rounding off the original computed results. This is called *discretization error*.) Why would one want to replace one model with another? There are several reasons. One would be to replace a computationally complex model with a much simpler one. For example, one could replace an implicit iterative computation with an explicit one, or a numerical simulation with a time-series model. Another reason might be to simplify model analysis, such as sensitivity analysis.

Final Model Re-Fitting

Once the decision is made that the current model is valid, the user can optionally make best use of all the data by re-fitting the model coefficients to all the data combined – without adding or removing any terms. This is done by changing the *FIT* dataset to include all the available data, including those data formerly used for the *CROSS* and *VALIDATE* datasets. This is done by performing the following steps:

- a. In the *Data Pane*, in the top row above the *FIT* data, click on the **Replace Data** button. A box will open to view files on the local computer. Navigate to the directory with the data files, and click on the CSV file containing all of the data.
- b. A dialog box will open similar to the one used when first entering the *FIT*, *CROSS* and *VALIDATE* data. But this time there is only one line, for entering new *FIT* data. The first field shows that 100% of the data will be imported. Leave this value and click on **Import**, and then **Close**.
- c. You will notice that the *FIT* datafile now has a new value of *nd* as shown in the *Goodness of Fit Statistics* Table, showing that all the data have been imported for model fitting. You will also notice one or two changes to the *Current Model* Table.
First, all the coefficients may have changed. *TaylorFit* will re-fit the current model to all the data in the revised *FIT* dataset, which has changed.

The second change that you may notice is that if your original model did not include an intercept term, *TaylorFit* will have added it to the current model. This is a minor bug in the current program. Simply click on the X on the left of the intercept term to remove it.

*DO NOT MAKE ANY OTHER CHANGES TO THE CURRENT MODEL,
I.E. BY EITHER ADDING OR REMOVING TERMS.*

- d. At this point the *Current Model* Table will have the re-fitted model with the coefficients updated using all the data, and the *FIT* column of the *Goodness of Fit Statistics* Table will have the updated statistics corresponding to the re-fitted model.

For clarity, we call the result the *Final Re-Fitted Model*. You can use the final re-fitted model for your ultimate purposes including making predictions and for model analysis, evaluation and distribution.

Exporting Information

The next step is to save your model and the statistical results.

The Export Data button in the *Data Pane* saves the *Data Tables* in a CSV file. Although the user already has the original data in a CSV file to start with, the exported files will also have the final re-fitted model *predictions and the residuals* in additional columns, as well as any additional columns that have been generated such as *transformed columns, sensitivities, and importance ratios*. This will be useful for any additional graphical or numerical data analysis the user may wish to perform on these derived results, such as using external programs for producing publication-quality graphics.

The Export Project button under the *Settings* button saves a file with filename "*model.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 of interruption. However, depending upon the browser being used, even if the *TaylorFit* browser tab is closed, the project may still be there if the *TaylorFit* link is reopened. The *model.tf* file is saved in the *Downloads* directory on the local computer.

The user can also copy and paste the *Current Model* Table or the *Goodness of Fit Statistics* Table from *TaylorFit* into other programs such as spreadsheets or word processors.

Finally, the Export to Code button below the *Current Model Table* generates code with the proper syntax that can be copied and pasted into *Javascript, Excel, C++, MatLab, Python* or *R* software for computing the model predictions. This version of the model contains the coefficients of the terms with full precision, as opposed to the *Current Model* Table, which displays only a user-selected number of digits (default five digits).

Note: You can stop *TaylorFit* at any time and close the tab on your browser. But on most browsers, the next time you start *TaylorFit*, it will begin where you left off. If you want to start a new project, you must click on the *Settings* button at the upper right, then click on the *Clear Project* button towards the bottom of the *Settings* Pane.

Step 8 – Model Analysis and Evaluation

Now it is time to look at what the model is telling you and what behaviors it reveals using a combination of graphical and analytical methods.

Graphical analysis

The *final re-fitted model* should be examined using the same graphical methods described for Step 7 above. In addition, you might want to produce plots of the model separately from the data. You can do this using software outside of *TaylorFit* such as *MatLab* or *Excel*. The *Export to Code* button under the *Current Model* Table facilitates this.

Since the essence of *Multivariate Polynomial Regression* is the “multivariate” part, it may be a problem to plot the response of the model to more than one or two independent variables. The model can be plotted versus a single independent variable in an ordinary X-Y plot.

The problem then becomes what value(s) to use for the other independent variables. A starting point may be to hold those other values at their mean or median values. Or, a parametric plot can show several model curves corresponding to several values of the second independent variable over its range. Only variables that appear together in the same term will interact with each other and would benefit from such a parametric plot. Constructing plot such as these involving one or two of the most sensitive independent variables may be adequate, although a thorough analysis would require a systematic examination of all the variables.

Confidence Intervals and Prediction Intervals

The *Confidence Interval* (C.I.) is the range of values that the *model prediction* is expected to fall into with a probability of $1-\alpha$, given a particular set of independent variables. The *Prediction Interval* (P.I.) is the wider range that the *data* are expected to fall into with the same probability. Specifically, the expected value of the prediction would be $\hat{y} \pm (C.I.)$, and the expected value of the individual data points would be $\hat{y} \pm (P.I.)$.

TaylorFit can calculate the C.I and the P.I. for each row in the *Data Table*. This feature is found in the dropdown menu in the heading for the Prediction Column (and only that column). By selecting either “Show Confidence Interval” or “Show Prediction Interval” a new column will be generated with those values, labeled “C.I.” or “P.I.” respectively.

For example: Suppose we developed a simple univariate model for how automobile fuel efficiency depends upon the weight of the vehicle (and nothing else). Using the included dataset, this model would be:

$$MPG = 46.927 - 0.0078556 \cdot WT$$

For $WT = 3664$ (the first case of the included Auto-MPG FIT dataset), the predicted fuel efficiency for this vehicle is 18.144 mpg. Assume the default value of $\alpha = 5\%$. *TaylorFit* computes C.I. = 1.0322 and P.I. = 9.2774.

Now, consider the following thought experiment: Suppose we had thousands of different cars, all with the exact same weight (keep in mind that this is a thought experiment!), and we grouped them randomly in groups of 500. We would expect the *average* MPG of the groups (i.e. 500 cars at a time) of this large population to fall within a range of values. This range is the *confidence interval*. If we performed this hypothetical experiment of thousands of different cars each weighing 3664 lbs and took the average of many groups of 500, the average MPG of the groups would be expected to fall in the range of 18.14 ± 1.03 (17.1 mpg to 19.2 mpg) 95% of the time.

Of course, the actual MPG values from all these cars would vary over a wider range than would the averages of the groups. This range is the prediction interval. In our example, the MPG values of the individual cars themselves would fall in the wider range of 18.14 ± 9.28 (8.87 mpg to 27.4 mpg) 95% of the time.

In the modeling literature it is the usual practice to report only the confidence interval, and not the prediction interval.

Sensitivity analysis

One important way to analyze the behavior of a model is to examine its sensitivity to particular independent variables (i.e., how the model “depends upon” a particular independent variable). The sensitivity of the model prediction with respect to the variable (x_i) may be defined as the slope of the model prediction curve with respect to x_i (see mathematical definitions in chapter *Formal Definitions*).

For *MLR* models, the sensitivity of the model to a particular independent variable x_i is a constant equal to the coefficient of the term containing that independent variable (a_i). But for *MPR* models the sensitivity may not be constant. In general, it will depend on the value of the independent variable and may also depend upon the values of other independent variables that interact with it. Thus, it would be appropriate to represent the sensitivity as a distribution.

TaylorFit can compute the sensitivity of the model to any independent variable by computing a value for the sensitivity for each row in the dataset. This is done by clicking on the down arrow in the heading of one of the independent variables and selecting “Show Sensitivity.” *TaylorFit* computes the sensitivities for each row and displays them in a new column to the right of the residuals. This column can then be examined using the same graphical tools as for the other columns. Of particular interest would be the *histogram* and *cumulative distribution* plots. If the *Export Data* button is clicked the new sensitivity column will be saved in a CSV file along with the other data.

Importance ratios for the Independent Variables

A problem with sensitivity as defined above is that values computed for different variables cannot be compared with each other. Sensitivity has units equal to the ratio of the units for the dependent variable and the selected independent variable. Thus, sensitivity itself depends upon the units used. A way to make the different sensitivities comparable is to cancel out the units using a measure of the spread of the variables. *TaylorFit* uses the standard deviation to do this. The resulting measure of sensitivity is called the *Importance Ratio (IR)*. The *IR* is proportional to

the sensitivity by the ratio of the standard errors of the respective variables. (See equations for sensitivity and *IR* in the section on *Formal Definitions* below.)

A narrative explanation of the meaning of the *IR* is: “*The Importance Ratio is the ratio of how much the dependent variable will change relative to its spread (as represented by its standard deviation) in response to a given change in one of the independent ratios relative to its range.*”

To further clarify with an example: An Importance Ratio of 0.5 means that if you change the independent variable by, say, 1% of its standard deviation, the dependent variable will change by 0.5% of its standard deviation.

Taylorfit will compute the *IR* values for any independent variable column in the dataset selected by the user. As with computing sensitivity, the user can click the down-arrow in the heading for the desired independent variable and select “Show Importance Ratio.”

Once the *IRs* have been produced for a particular column in the dataset, use the *Cumulative Distribution* Plot in the dropdown menu for the *I.R.* column to examine the range of values for the sensitivity or the *I.R.* Note that it might be possible for them to change sign over their range. I.e., the sensitivity may be positive for some sets of independent variable values, and positive for others. This is an example of a type of behavior that linear models cannot describe.

If a single value is desired to represent the overall sensitivity or importance ratio, it is recommended to use the *RMS* value shown in the column statistics in the *Data Pane*. Thus the *RMS IR* can be used to determine which of the independent variables is “most important.”

Note: At the present time, *TaylorFit* can only properly compute sensitivity and importance ratios for non-time-series models. The program will compute them for time-series, but the results will not be meaningful and should not be used.

Sensitivity and Importance Ratios of the Terms

The user may be interested in determining which terms in the *Current Model* contributes the most to the predictive power of the model. The Importance Ratio for the Term is a suitable measure of this. Each term in the model can be described as a coefficient (a) times a polynomial function (Z). Term i is written as $a_i \cdot Z_i$. We want to compare not only how individual independent variables contribute to the model, but how each term contributes. By treating Z as a variable, we can do a sensitivity and importance ratio analysis similar to that described above.

For sensitivity, the result is very simple: The sensitivity of the model prediction is simply the coefficient a_i . This tells you how much the prediction changes as you change Z_i . However, as with the independent variables, we cannot compare the different values of sensitivity because they depend upon the units used in the original independent variables. Therefore, we can define an importance ratio for the terms in a similar way by multiplying the coefficient by the ratio of the standard deviation of Z_i to the standard deviation of the dependent variable y . Unlike the *IR* of the independent variables, the *IR* of terms is a constant.

The values of the *IR* of the terms are automatically displayed in the fourth column of the *Current Model Table*. These can be compared to each other to indicate the relative contribution of each term to the prediction.

Comparing different models

As was noted above, *TaylorFit* may produce different models if different strategies are used. For example, a model using *MSE* for cross-validation may be different than one produced using the *F*-statistic. The user could also develop different models using different values of α . Or the sequence selected to increment the *Fitting Parameters* may also influence which particular terms wind up in the model. Modeling is art as well as science, and there is no single guaranteed route to a perfect model. But how can the user choose which of these models is best?

This raises the issue of how to compare any two different models. The approach described here will work not only for comparing two *MPR* models, but to compare *MPR* with *MLR*, or *MPR* with *ANN*, etc. This can be carried out using the data from *TaylorFit*, but the calculation itself will have to be done in external software such as *Excel*.

Disparate models can be compared by using the ratio of their *MSE* values. Note that $MSE = SSE/df$. The *df* is unambiguous with *MLR* or *MPR* models. In the case of *ANN* models there is disagreement in the literature about what the *df* should be. Some references indicate that *df* is much less than the number of weights in the *ANN* (which can be quite high); others indicate it should be higher. A conservative approach would be to just assume that *df* is equal to the number of weights in the *ANN*.

In any case, once the *MSE* is computed for each of the two models, the ratio of the model with the larger *MSE* (call it “model 1”) and the model with the smaller *MSE* (model 2) will be an *F*-statistic for the comparison:

$$F = \frac{MSE_{model\ 1}}{MSE_{model\ 2}}$$

Then, the probability that that value of *F* or larger can occur by chance can be computed from the *F*-statistic, and the degrees of freedom for the two models:

$$p(F) = f(F, df_1, df_2)$$

In *Excel*, for example, the value of $p(F)$ can be computed using the built-in function “=FDIST(F,df1, df2).” If $p(F) \leq \alpha$ one may accept that model 2 has a significantly smaller (better) *MSE*.

For example, a multilinear regression (*MLR*) model of the Auto MPG dataset produced $MSE = 9.635$ with $df = 129$, using a validation dataset. The *MPR* model resulted in $MSE = 6.7082$ with $df = 124$. Is the *MPR* model significantly better than the *MLR* model? The *F*-statistic is $9.635/6.7082 = 1.4363$. Using the degrees of freedom in the *Excel* formula one can compute the probability of a larger *F* as $p(F) = 0.022$. So it is unlikely (2.2% probability) that this difference occurred by chance, and we can conclude that the *MPR* model significantly out-performed *MLR*.

Example Datasets

The startup page for www.TaylorFit-RSA.com includes buttons for:

- **Additional information:** This button takes you to the website for *Simetrica, LLC*. This has a primer on *MPR* modeling and some other information.
- **Download Example Non-time-series Data - Auto MPG:** This download automatically loads an example *non-time-series* dataset into *TaylorFit*.
- **Download Example Time-series - Lydia Pinkham:** This download automatically loads an example *time-series* dataset into *TaylorFit*.

Both example datasets are described below. Many other example datasets can be found on the web. A good source of data is at the following website maintained by the University of California, Irvine (UCI):

<https://archive.ics.uci.edu/ml/datasets.html?format=&task=&att=&area=&numAtt=&numIns=&type=&sort=taskUp&view=table>

Reference:

Lichman, M. (2013). UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]. Irvine, CA: University of California, School of Information and Computer Science.

Auto MPG Dataset

This dataset was taken from a library maintained at Carnegie Mellon University and was downloaded from the UCI website:

<https://archive.ics.uci.edu/ml/datasets/Auto+MPG>

There are data for 398 vehicles from model years 1970 to 1982. The variables, corresponding to the columns of the CSV file, are:

1. MPG: City-cycle fuel consumption in miles per gallon
2. CYL: The number of cylinders in the engine.
3. DISP: The piston displacement volume in cubic inches
4. HP: Engine power in horsepower units
5. WT: Vehicle weight in pounds
6. ACCEL: Acceleration
7. YR: Model year in integer value as last two digits of the year (i.e. 79 means 1979)
8. ORIGIN: This is an integer-valued index indicated whether the origin of the vehicle is from the United States (index = 1), Europe (index = 2), or Japan (index = 3).

The first column should be designated the dependent variable. An additional column was in the original download file containing the name of the car. This was removed because it is unique for each car (row in the data file) and cannot be used as a fitting variable.

As the fitting proceeds, you will note some interesting results that show how the modeler needs to use judgement in the process. Once the modeling process has progressed to a fairly good model, examination of the residuals plot shows that there are a number of cars that are high-MPG outliers. The variables in the data file are not able to explain the high MPG values for

these cars. This indicates that there is likely to be some additional variable, not in the dataset, that would explain the outliers. The modeler should examine the outliers to see if such an explanatory variable could be found. For example, it might be found that these cars have some special technology to improve mileage, such as a hybrid power train or engine cylinder deactivation.

Lydia Pinkham Dataset

The Lydia Pinkham is a company that markets a tonic to women. As a result of a lawsuit, the company's annual sales results (S) and advertising budget (A) for the years 1907 to 1960 were made public. This dataset has been used often as a test case for time-series models. The values in the dataset are in thousands of U.S. dollars.

Modeling Tips and Potential Pitfalls

The modeler's first step should always be to statistically and 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, different ranges or other statistical properties. This step is ignored at great risk to the modeler!

The final step should always be the statistical and graphical examination of residuals and the model behavior. 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. (The same will be true for other kinds of models such as ANNs.) This is usually not a problem; close to optimal may be good enough. With empirical modeling there will always be other approaches to modeling a specific dataset that may give somewhat different behavior. However, with *MPR* models you can try adding and removing terms manually to find better models closer to the optimum. Use the approach described above in the section *Comparing Different Models* to select which model works best.

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 until only significant terms remain.

You can also fit a fundamental (aka “mechanistic”) model to the data, taking into account as much as possible, and then model the residuals left over after the mechanistic model predictions were subtracted from the data. For example, one investigator obtained the tidal components of waves near an ocean outlet of a bay using the well-known sinusoidal behavior of the tides. Then, he used *TaylorFit* to discover correlations between those residuals and components such as wind and offshore wave parameters that are not accounted for by the mechanistic models.

Nonlinearities may sometimes be hard to find unless noise level in the data is very low. From experience, when R^2 is 0.75 or lower, the nonlinearity is sometimes 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 Modeling Pitfalls

Here is a summary of several well-known problems that occur in any kind of modeling. The modeler needs to be aware of them and watchful for them.

Overfitting: This is a problem also known as “memorizing the noise in the data.” You will recognize this when you observe that the model does well predicting the *FIT* dataset but does poorly in fitting the final validation data not used in developing the model. The simplest way to

avoid this is to use conservative fitting criteria such as *F-statistic*, *AIC*, or *BIC*, which penalize model complexity. However, this may not be sufficient. A better approach is to use cross-validation. The values of R^2 or *SSE* should never be used as a fitting criterion. Adding a term to a model will usually improve these statistics, even if the term is not a significant contributor to the fit. The adjusted- R^2 or *MSE* may be used in place of these.

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.

Multicollinearity: If two or more of the independent variables are correlated with each other, a model that includes both will have misleading *t*-statistics. However, the step-wise regression strategy helps avoid this problem. Once a variable or term is in the model, another variable or term that is highly correlated with it usually 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 or more random variables, accepting 95% confidence level, you may find correlations. An experiment like this modeled with two multipliers and three exponents got better than 50% R^2 purely by chance. But when cross-validation was used with the same data no spurious correlations were found. Cross-validation helps prevent both 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 and/or a larger value of α . Cross-validation 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 may be better to separate them and form two different models - one for the men and one for the women.

The final step in modeling should always include graphical examination of the residuals

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.

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. A future version of the software may include weighting factors, 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 periods separately from the wet ones.

Non-orthogonality: Once criticism that is sometimes levied at MPR models is that it often may produce models whose terms are not orthogonal. Orthogonality is a property that is desired because it can reduce roundoff errors during fitting. This is overcome by *TaylorFit* by using *Singular Value Decomposition (SVD)* to do the fitting, as well as by using stepwise term selection to eliminate multicollinearity. The criticism may also be unfounded because in the literature when fitting is done by orthogonal polynomials, a final step may be to remove insignificant polynomial terms, resulting in a final model that is not orthogonal anyway.

Potential Time-Series Issues

If a time-series varies slowly, it is easy to get high R^2 values, even with naïvely simple models. For example, in modeling daily temperature in a river, simply using yesterday's value as a predictor would produce high R^2 . In such cases it would be more meaningful to difference the data and develop a model to predict the difference. That is, instead of predicting tomorrow's value, predict the difference between today and tomorrow.

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, average(lags 2 to 3), average(lags 4 to 7), average(lags 8 to 15), etc. Yet another approach is to use an exponentially decaying weighted moving average.

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 test candidate terms with lower degrees at the same time.

Design-of-Experiments with Multivariate Polynomial Regression

As was mentioned above, Response Surface *Methodology (RSM)* is a Design-of-Experiment (*DOE*) approach for developing data, while Response Surface *Analysis (RSA)* is an approach for turning data into models. *TaylorFit* is well-adapted for *RSA* using “found” data, i.e. data obtained from uncontrolled conditions in nature. As also mentioned, typical *RSM* approaches limit the experimenter to low-order integer polynomials. If it is desired to explore more complex response surfaces, experiments can be designed to take advantage of *TaylorFit's* capabilities.

The approach that will be described here is the most basic type, called *full factorial design*. Suppose you would like to do a non-time-series experiment with three independent variables, and you would like to consider two-way interactions ($N_m = 2$) with three possible exponents ($n_e = 3$). (Say, $Exp = \{1, 2, -1\}$, although it doesn't matter which (non-zero) exponents you use, only that there are three of them.) According to the calculation described in the section below, there would be 37 possible candidate terms. You would have to do at least 37 experiments (neglecting replicates) in order to explore every possible combination of variables and exponents.

Known Bugs

TaylorFit is available free and without restriction. It's development has been mostly supported by senior design teams in the computer science program at Stevens Institute of Technology in Hoboken, NJ. As a result of the annual development iterations by independent teams, there remain bugs in the code. Most of these bugs have workarounds or are easily avoided. They are listed here:

- Sensitivity and I.R. do not always update when the current model is changed. It is recommended not to select this feature until the model development is complete.
- *Clear Project* button does not clear the exponents list. The user should either use the *Clear Settings* button or refresh the page after hitting *Clear Project*.
- If program hangs and cannot be stopped even by closing the browser, these are some steps to try (based on Chrome browser; other browsers may have similar steps):
 - Hit F12 to get into debug mode
 - Select the tab for "Applications" (you might have to click a right-arrow to scroll the tabs)
 - Click on "Local Storage"
 - Select "tf-model"
 - Delete
 - Close the browser tab or hit "Refresh"
- If the intercept term has been removed from a model and the user hits the "refresh page" button on the browser, the intercept is brought back into the model. The user should then just delete the intercept term again.

Bugs specific to Time-Series modeling:

- TaylorFit cannot load univariate time series (i.e. time-series data with only one column). Workaround: Add a dummy column filled with 1's. Then ignore any terms containing the dummy variables. (Do not use the "Ignore Column" feature, as this will affect display of terms for interactions between lags.)
- The Sensitivity and I.R. calculations do not work for lagged variables (i.e. for Time-series models in general).

Formal Definitions

This chapter provides formal mathematical definitions for the models and statistics used in *TaylorFit* suitable for use in publications.

Non-time-series Multivariate Polynomial Regression Model

The following is a formal definition of the non-time-series Multivariate Polynomial Regression (MPR) model:

Data: $(y_i, x_{1,i}, x_{2,i}, \dots, x_{n_v,i})^T; i = 1, \dots, n_d$

Model: $\hat{y}_i = (\sum_{k=1}^{n_p} a_k Z_{k,i}); Z_{k,i} = \prod_{j=1}^{n_v} x_{j,i}^{b^{j,k}}$

$$e_i = y_i - \hat{y}_i$$

$$\sigma^2 \simeq s^2 \equiv MSE = \frac{\sum_{i=1}^{n_d} e_i^2}{n_d - n_p}$$

Definitions:

y_i	(Measured) dependent variable in row i
\bar{y}	Mean of dependent variables
\hat{y}_i	Predicted value using independent variables from row i
$x_{j,i}$	Independent variable j in row i
n_v	Number of independent variables (i.v.'s)
n_d	Number of data points (rows in dataset)
n_p	Number of coefficients (terms) in the model (including constant term)
$Z_{k,i}$	Model term k using independent variables from row i
$b^{j,k}$	Exponent of independent variable j in term k
e_i	Residual (error) for row i
σ^2	Variance of residuals
s^2	Estimated variance of residuals

Number of Candidate Terms

The total number of possible candidate terms ($N_{\text{candidates}}$) is displayed in the *Settings* box. This value is of interest both because it relates to how much data are needed for fitting (see section on *Design-of-Experiments* above) and because the more candidate terms that there are, the longer your computer will take for each recalculation.

The number of possible candidate terms, n_t , depends upon the number of independent variables, n_v , the number of exponents being considered, n_e , and (if the model is a time-series) the number of lags, n_l . Adding lagged values increases the effective number of independent variables. The total number of independent variables, n_v , will be:

$$n_v = (n_x - 1) \cdot n_l + v_0 \cdot (n_l - l_0) \quad [\text{for time-series}]$$

Where

n_x is the number of variables (columns) in the dataset (dependent variable plus independent variables),

l_0 equals one if the list of lags includes zero, and equals zero otherwise, and

v_0 equals one if lagged dependent variables are included as independent variables, and zero otherwise. By default, *TaylorFit* assumes $v_0 = 1$.

If the dataset is **not** a time-series, then $n_l = 1$ and the second term is dropped, so:

$$n_v = n_x - 1 \quad [\text{for non-time-series}]$$

The maximum value that n_m can take is n_v . If $n_m = n_v$, then the maximum total number of candidate terms is:

$$n_{t,\text{max}} = (n_e + 1)^{n_v}$$

This may result in a large number of terms to be tested for selection into the model. For example, if there are three independent variables ($n_x=4$) and no lags ($n_l=1$), and 10 exponents ($n_e=10$), then there are $n_{t,\text{max}} = 1331$ possible terms. Experience with a wide variety of datasets has shown that n_m can often be restricted to two or three. For $n_m \leq n_v$:

$$n_t = \sum_{m=0}^{n_m} C(n_v, m) \cdot n_e^m$$

where $C(n_v, m)$ is the number of combinations of n_v objects taken m at a time.

For the example above, if $n_m=2$, the number of candidate terms drops to $n_t=331$. As n_v and n_e increase, this restriction becomes more important. For example, with $n_x=10$, $n_l=2$, $n_e=2$, $l_0=1$, and $v_0=0$, $n_v=18$. If $n_m=n_v$, then there are $n_t = 387,420,489$ possible terms. Restricting to $n_m=2$ gives $n_t = 649$.

Computing Confidence Intervals and Prediction Intervals

As mentioned above, a rough estimate of the prediction interval for the residuals (and therefore for the data) can be estimated as $\pm 2 \cdot RMSE$. More precisely, the value of 2 should be replaced by the critical value of the t -statistic for α and df :

$$\hat{y}_i \pm t_{crit} \cdot \sqrt{MSE}$$

However, this does not take into account the effect of the independent variables. Using the values of the independent variables allows more precise estimation of the confidence intervals for each prediction.

To compute confidence intervals, it is first necessary to construct the $n_d \times n_p$ matrix Z in which each row corresponds to a row in the dataset, and each column is a corresponding term computed from the independent variables in those rows. If there is a constant term in the model, the corresponding column should be filled with 1's. Z_i is then the vector corresponding to row i of matrix Z . Then, the confidence interval ($C.I.$) for the prediction for row i is:

$$C.I. = \hat{y}_i \pm t_{crit} \cdot \sqrt{MSE \cdot (Z_i^T (Z^T Z)^{-1} Z_i)}$$

It is interesting to note that if the dependent values are plotted along with the predictions and their confidence intervals, most of the data would be outside the $C.I.$ This seems paradoxical. One might expect 95% of the data to be within the 95% confidence interval. The explanation is that the $C.I.$ is the range for the *model prediction*, not for the *data*. Since the data are the predictions plus their corresponding residuals, they range more widely. The interval for the data is called (ironically) the *prediction interval (P.I.)*, and can be computed as follows:

$$P.I. = \hat{y}_i \pm t_{crit} \cdot \sqrt{MSE \cdot (1 + Z_i^T (Z^T Z)^{-1} Z_i)}$$

The prediction intervals and confidence intervals are computed by *TaylorFit* when the user selects them from the drop-down menu for the column "predictions" in the Data Table. Note that only the *predictions column* drop-down menu shows this option.

Goodness-of-Fit Statistics

The following are formal definitions of the values found in the *Goodness of Fit Statistics* Table. (Note that some of the following are intermediate values not displayed in *TaylorFit*, including *df*, *TMS*, *SSR*, and *MSR*.)

Dataset statistics

n_d : The number of data used for fitting. For non-time-series models this is the same as the number of rows in the corresponding sub-dataset. For time-series models it is the number of rows minus the maximum value of *LAGS*.
(Same as n_d in the formal model definition.)

n_p : The number of coefficients (= number of terms) in the current model, including the constant term, if included in the model.
(Same as n_p in the formal model definition.)

df : The residual (or error) degrees of freedom;
$$df = n_d - n_p$$

TSS : Total Sum of Squares:

$$TSS = \sum_{i=1}^{n_d} (y_i - \bar{y})^2$$

TMS : Total Mean Square, the estimated variance of the dependent variable

$$TMS = \frac{TSS}{n_d - 1}$$

Global Goodness-of-Fit Statistics

SSE : Sum of squares of the model errors (aka sum of squares of the residuals)

$$SSE = \sum_{i=1}^{n_d} e_i^2$$

MSE : Mean square error; the sample variance of the residuals

$$MSE = \frac{SSE}{df}$$

$RMSE$: Root mean square error; the estimated standard error of the residuals

$$RMSE = \sqrt{MSE}$$

$Max|Err|$: The maximum of the absolute values of the errors; used for minimax optimization

$$Max|Err| = Max|e_i|$$

SSR : Sum of squares of the regression:

$$SSR = TSS - SSE$$

MSR: Mean square of the regression (intermediate value not displayed in *TaylorFit*)

$$MSR = \frac{SSR}{n_p}$$

RSQ: R^2 , the square of the multiple correlation coefficient

$$R^2 = 1 - \frac{SSE}{TSS}$$

Adj-*RSQ*: Adjusted R^2 – similar to R^2 , but with a penalty for model complexity (the number of terms)

$$R_{adj}^2 = 1 - \frac{MSE}{TMS}$$

F: The *F*-statistic; similar to the *t*-statistic for a term, but indicates the fit of the overall model, rather than of a particular term

$$F = \frac{MSR}{MSE}$$

$p(F)$: The probability of *F* being this large or larger by chance. The numerator degrees of freedom is n_p , the denominator degrees of freedom is *df*.

AIC: Akaike Information Criterion

$$AIC = \log(MSE) + 2 \cdot \frac{n_p}{n_d}$$

BIC: Bayesian Information Criterion

$$BIC = \log(MSE) + \log(n_d) \cdot \frac{n_p}{n_d}$$

SKEW: Skewness, a measure of asymmetry of the residuals distribution

s.e. *SKEW* Standard error of the skewness

XKURT: Excess Kurtosis = Kurtosis - 3.0; This is a shape factor that describes a deviation of the residuals distribution relative to the “normal” distribution; a large excess kurtosis indicates a greater probability of extreme values

s.e. *XKURT* Standard error of the kurtosis

Model statistics

t : The computed t statistic:

$$t = \frac{a_1}{s.e._{a1}}$$

$s.e._{ai}$ Standard error of model coefficient a_i

$p(t)$: The two-tailed probability of a larger value of t . Typically, we would add a term to the model if $p(t) \leq \alpha$ for that term and remove a term if its $p(t) > \alpha$. The degrees of freedom for computing this is df .

α The significance level. Default value set to $\alpha = 0.05$, but it can be changed in *SETTINGS*.

Transformations

Logarithmic

$$\hat{y} = \log_{10}(y)$$

Standardized

$$\hat{y} = \frac{y - \bar{y}}{s.e._y}$$

Scaled by *RMS*

$$\hat{y} = \frac{y}{RMS_y}$$

k -order Difference

$$k = 1: D_{1,i} = y_i - y_{i-1}$$

$$k = 2: D_{2,i} = D_{1,i} - D_{1,i-1}$$

$$k = n: D_{n,i} = D_{n-1,i} - D_{n-1,i-1}$$

Sensitivity and Importance Ratio

Sensitivity of \hat{y} with respect to x_i :

$$\psi_{x_i} = \frac{\partial \hat{y}}{\partial x_i}$$

Importance ratio of \hat{y} with respect to x_i :

$$IR_{x_i} = \frac{\partial \hat{y} / s.e._y}{\partial x_i / s.e._{x_i}} = \psi_{x_i} \cdot \frac{s.e._{x_i}}{s.e._y}$$

Importance ratio of the TERMS in the Current Model Table

$$\psi_{Z_i} = \frac{\partial \hat{y}}{\partial Z_i} = a_i$$

$$IR_{Z_i} = \frac{\partial \hat{y} / s.e._y}{\partial Z_i / s.e._{Z_i}} = a_i \cdot \frac{s.e._{Z_i}}{s.e._y}$$

Bibliography

Classic references on regression and time-series analysis

Draper N.R., Smith H. (1981) Applied Regression Analysis. 4th ed. New York: John Wiley.

Box G.E.P., Draper N.R. (2007) Response surfaces, mixtures and ridge analyses. John Wiley.

Box, G.E.P., G.M. Jenkins, G.C. Reinsel, G.M. Ljung, (2015) Time Series Analysis, Forecasting and Control. John Wiley.

MPR References

Vaccari, D.A., Nonlinear Analysis of Retail Performance, *Proceedings of the IEEE/IAFE Conference on Computational Intelligence for Financial Engineering (CIFER)*, pp252-258, New York, NY, March 24-26, 1996.

Vaccari, D.A. and J. Levri, "Multivariable Empirical Modeling of ALS Systems Using Polynomials," *Life Support and Biosphere Science*, vol. 6 pp. 265-271, (1999).

Vaccari, D. A. and H.K. Wang, "Multivariate polynomial regression for identification of chaotic time series," *Mathematical and Computer Modelling of Dynamical Systems*, v13, n4, pp395–412 (2007).

Jagupilla, S.J.K., D.A. Vaccari, and R.I. Hires, "Multivariate Polynomial Time-Series Models and Importance Ratios to Identify Fecal Coliform Sources," *ASCE Journal of Environmental Engineering*, v136, n7, pp657-665 (2010).

Simetrica (2015) TaylorFit Multivariate Regression Software, www.Simetrica-LLC.com.

Rutner, M.P. and D.A. Vaccari (2016) Preliminary and time-efficient vulnerability assessment of structural columns subjected to blast loading, *Engineering Structures* 128 (2016) 55–66.

Watson, R., Nieradka, B. and Vaccari, D. A. (2017), Empirical Dynamic Material Flow Model for Tungsten in the USA. *Journal of Industrial Ecology*. <http://doi.org/10.1111/jiec.12555>.

Vishwa Shah, Sarath Chandra K. Jagupilla *, David A. Vaccari, Daniel Gebler (2021). Non-Linear Visualization and Importance Ratio Analysis of Multivariate Polynomial Regression Ecological Models based on River Hydromorphology and Water Quality. Section: Ecohydrology, Water 2021, 13, 2708. <https://doi.org/10.3390/w13192708>.

Other references

Chen, S. and Billings, S.A., 1989, Representations of nonlinear systems: The NARMAX model. *Int. J. Control*, 49(3), 1013 – 1032.

Billings, S.A. and Aguirre, L.A., 1995, Effects of the sampling time on the dynamics and identification of nonlinear model. *Int. J. Bifurc. & Chaos*, 5(6), 1541 – 1556.

Guo, L.Z. and Billings, S.A., 2005, A comparison of polynomial and wavelet expansions for the identification of chaotic coupled map lattices. *Int. J. Bifurc. & Chaos*, 15(9), 2927 – 2938.