Nonlinear regression

Menu:	QCExpert	Nonlinear regression

Nonlinear regression module allows you to fit and analyze regression models of the general form

$$y = F(\mathbf{x},\mathbf{p}) \qquad (1\text{-}1)$$

Where y is a response variable, $\mathbf{x} = (x_1, x_2, \ldots x_q)$ are values of the explanatory variables (written as a vector). q is the number of explanatory variables in the regression model. There are m parameters, $\mathbf{p} = (p_1, p_2, \ldots, p_m)$ in the model. $F(\mathbf{x},\mathbf{p})$ is a function of explanatory variables and parameters. Maximum number of parameters is 32, maximum number of variables is 254. Ideally, **x** is assumed to be a deterministic, i.e. non-random vector, which is either purportedly set to pre-specified values or its values are found out via an essentially error-free procedure. y depends on **x,** but the dependence is blurred by the presence of a random error ε. Vector of model parameters **p** are estimated from data by the nonlinear least squares method. The user can specify a desired nonlinear model either in the Nonlinear regression dialog panel (Fig. 1) or in the *Model specification window*.

Note: If the desired model is linear with respect to the parameters, that is in the form of (1), use linear regression module instead (see the previous chapter), where the computations are of non-iterative nature (no initial parameter estimates are needed). A typical example of models which are linear in parameters (albeit nonlinear in explanatory variables) are: $y = p_1x + p_2\ln(x)$ or polynomial models like $y = p_1x + p_2x^2 + p_3x^3 + p_4$. Other models might be linearized easily, for instance $y = p_1\exp(p_2x)$ can be linearized $\ln y = \ln p_1 + p_2x$ (quasilinearization might be needed to suppress possible error distribution distortion).

Data and parameters

Unknown parameters $\mathbf{p} = (p_1, p_2, \ldots, p_m)$ are estimated from data contained in the current data sheet. Each column of the sheet corresponds to a variable. Names of the variables appear in the column headers. Parameters and variables are part of model declaration which can be completed either upon clicking the *Model…* button, or directly in the *Nonlinear regression* window (when the desired model was already specified previously). Detailed model specification instructions can be found later in this chapter. Once a model is specified, it appears in the *Model* window of the Nonlinear regression dialog panel.

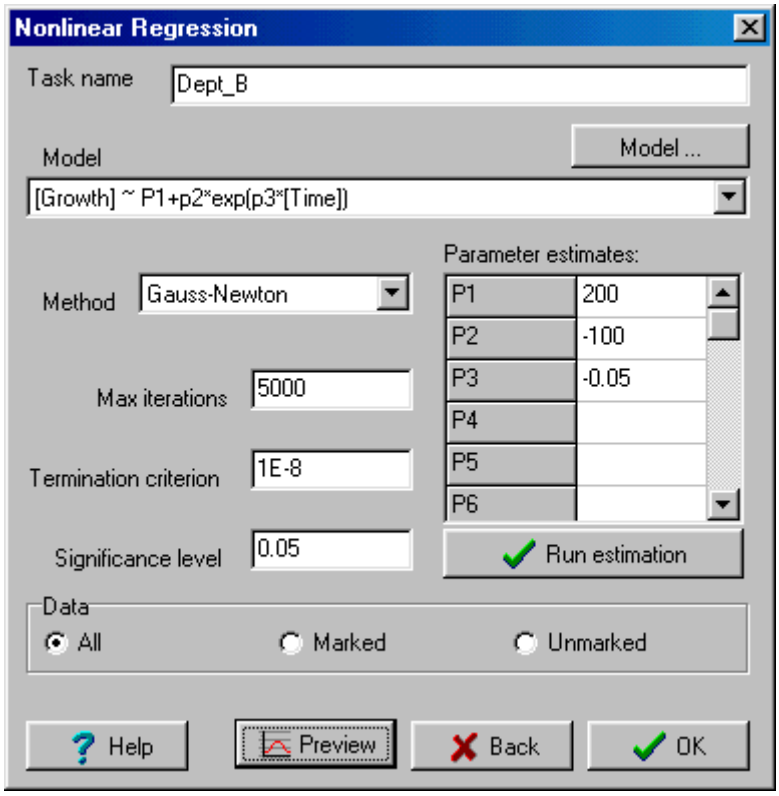


Fig. 1 Nonlinear regression dialog panel

A project identification (it will appear in the header of all protocol pages and graphical output) can be inputted in the *Project name* field. Optimization method to be used by the regression procedure can be selected from the *Method* list (*Gauss-Newton*, *Marquardt*, *gradient*, *dog-leg*, *simplex* are possible choices). Maximum number of iterations is entered in the *Max iteration* field. The algorithm stops when either maximum gradient element meets a termination requirement or when the norm of the parameter change from one iteration to the next is smaller than a specified value. Alpha is the significance/confidence level used for all tests/confidence intervals. Buttons in the *Data* section of the dialog panel can be used to determine which part of the available data will be used (possible choices are: all data, selected data only, not-selected data only). Initial guess for all parameter values $p_1, \ldots p_m$ has to be inputted in the *Parameter estimates* field. Generally, the initial estimates should be close to the final regression estimates. Take care and time to supply as good initial estimates as possible. Rough initial estimates might produce incorrect final estimates, or it can happen that final estimates cannot be produced from rough initial values at all. Improper initial values might also lead to convergence problems resulting in very large number of iterations that the procedure needs to find final estimates (which might take considerable amount of computer time). Adequacy of the final parameter estimates can be checked by pressing the *View* button from the *Nonlinear regression* dialog panel. It displays data and model fit together with the residual sum of squares. The red (model) line should be close to data points. If the fitted model contains more than one explanatory variable, *View* produces observed versus predicted response plot together with the *y=x* line, corresponding to an ideal fit (with no model-data discrepancy). The *View* window does not support any interactive features. It cannot be copied (using the *Ctrl-C* command) either. When finished with the plot inspection, press the *OK* button to get back to the *Nonlinear regression* dialog panel.

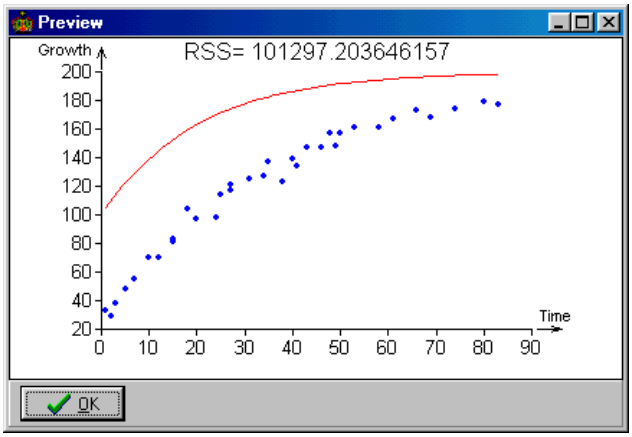


Fig. 2 *Preview* window

After the initial parameter estimates had been entered, computations are started upon pressing the *Compute* button. Progress of the iterative computational procedure can be monitored in the *Monitor* panel (Fig. 3) which is invoked automatically.

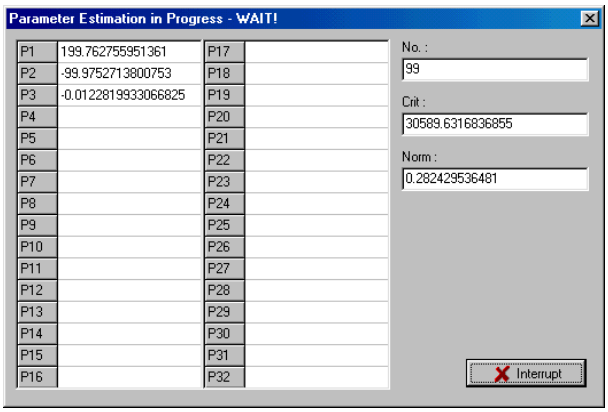


Fig. 3 *Monitor* panel

The panel contains current iteration information: parameter values, iteration number, residual sum of squares and other information, depending on the optimization method. *Norm* field contains the norm of parameter vector change from one iteration to the next. The norm is compared with a pre-specified number (termination criterion) and the procedure when the actual norm is smaller, procedure stops. Computations can be stopped manually at any time by the S*top* button (the procedure then returns parameter values from the last iteration).

When the computation procedure stops, the program returns to the *Nonlinear regression* panel. The *Parameter estimates* field contains parameter from the last completed iteration. When the procedure stops in a normal way (meeting the stopping criteria), returned parameters should be equal to the optimal values (nonlinear least squares estimates). When in doubt about convergence status of the procedure at termination, you should check carefully parameter values plausibility. You can use the *Preview* window to inspect the estimated model fit visually (Fig. 2). It is generally recommended to try different optimization methods in case of problems (e.g. slow convergence, divergence). Final estimates from one method can be used as starting values for the next method (or they can be somewhat edited before starting the computations). These steps might need to be repeated several times (checking the parameter estimates plausibility through *View* along the way). It might be also useful to check whether slightly perturbed final estimates used as starting values will yield the same values as before perturbation. After performing these various checks, the final estimates are accepted by pressing the *OK* button in the *Nonlinear regression* dialog panel. QC.Expert™ then produces protocol and graphical output. ***Warning***: if *OK* is pressed before running the computations, the starting parameter estimates are accepted as the final estimates (without any optimization)!

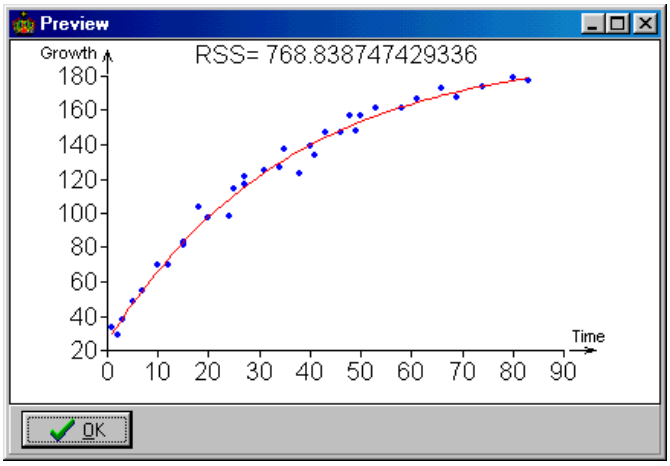


Fig. 4 *Preview* after finishing computations

Model specification: The *Model…* button opens a new panel for model specification (Fig. 5). If you specified some models previously, you can select one of them without opening the *Model specification* panel. Make sure that the variable names in the current data sheet and in the model are the same in such case. There is a list of current data sheet variables in the left part of the *Model specification* panel. These are the only variables you can use in the model building. The response variable is inputted in the upper right part of the panel. When desired, you can check the Weights option, which will enable you to specify name of a data sheet column which contains weights w_i. They correspond to coefficients by which individual residuals (not squared residuals) are multiplied. Inputted weights are automatically standardized to sum to n (number of data points). When the Weights option is not checked, unit weights, w_i=1 are used by default. There are shortcut buttons for model specification in the central part of the panel. Input line, where the model is actually specified, appears at the bottom of the panel. There you can also find a list of previously defined models. The *Save button* saves a model after it was completely specified. The newly saved model appears in the list of previously defined models in the „current“ position. The *Read* button reads a selected model in and places it in the input line, where it can be modified.

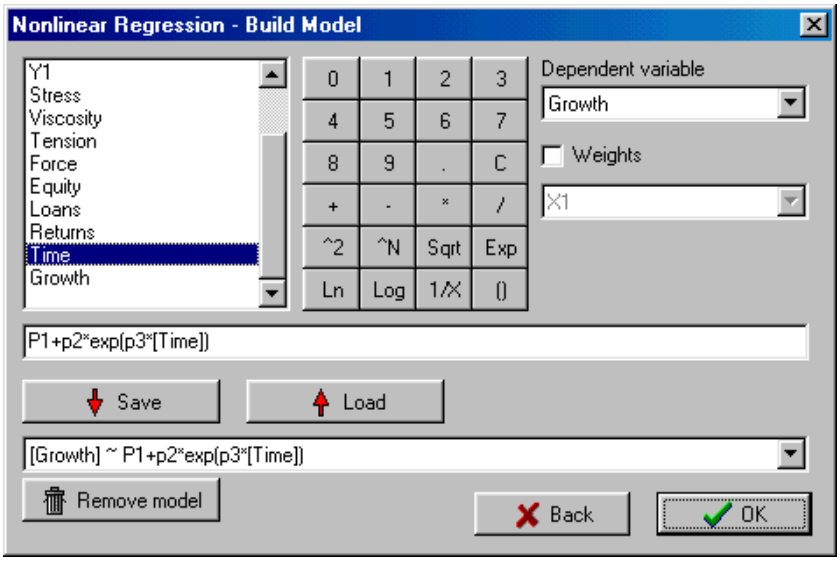


Fig. 5 Model specification dialog panel

Model specification instructions:

Double click on a variable name in the current data sheet variable list to copy the name to the input line. Variable name is always enclosed in square brackets. Parameters have to be represented by the P1, P2, … codes. These codes are then used in the *Parameter estimates* field in the *Nonlinear regression* panel.

Shortcut function buttons can be helpful when writing more complicated expressions. Highlighting a part of the model input line and clicking a function button subsequently, applies the function on the highlighted part as an argument. For instance, the expression ln([*x*]+1) can be assembled in the following way: double click on the variable *x* (there has to be a column of this name in the current data sheet): **`[x]`**; write + 1 manually; highlight the whole expression: **`[x]+1`**; click the *Ln* button, this action results into: **`ln([x]+1)`**. Application of *^2*, *^A*, *Sqrt*, *Exp*, *Log*, *1/X*, *( )* is similar. The *C* button erases model input line. Other functions have to be inputted manually (writing their name in the model input line). Available functions are listed in the Linear Regression chapter. When finished with specification, the model is saved by the *Save* button. Then, it automatically appears in the list of previously specified models located in the bottom part of the panel. The *Read* button reads in a model from the set of previously defined models. Its terms can be edited then. The *Erase model* deletes a selected model from the model list. Warning: this operation is irreversible! The *OK* button finishes model specification.

A model can be defined without using mouse and shortcut buttons as well. Usual syntactic rules apply, keep in mind that variable names have to be enclosed in square brackets. Previously defined models can be selected from list of models in the *Nonlinear regression* dialog panel without going through the *Model specification* dialog panel (variable names in the data sheet and in the model specification have to agree).

Computational methods

Nonlinear regression procedure implementation can be viewed as a procedure for finding such parameter values that minimize some kind of distance between model predicted and actual response values,

$$\min_{\mathbf{p}} S(\mathbf{p}) = \min_{\mathbf{p}} D(\mathbf{y}, \hat{\mathbf{y}}), \tag{1-2}$$

where *D* stands for a distance, $\mathbf{y}$ is a vector of observed response values and $\hat{\mathbf{y}}$ is a vector of model predicted response values. Euclidean distance is used most commonly for D,

$$S(\mathbf{p}) = \|\mathbf{y} - \hat{\mathbf{y}}\| = \|\mathbf{e}\| = \sqrt{\sum_{i=1}^{n} [y_i - \hat{y}_i]^2} = \sqrt{\sum_{i=1}^{n} e_i^2} \tag{1-3}$$

As far as minimization is concerned, we can look at squared distance in place of the distance itself, so that we get a simple expression, without the square root sign. This is to say that the original minimization is equivalent to minimizing the sum of squared differences between data and model predictions. The minimization typically has not closed form solution so that it is performed numerically, through an iterative procedure. The procedure is based on some kind of nonlinear optimization algorithm. Different algorithms have different properties and no universally best algorithm exists. Therefore, QC.Expert™ implements six different algorithms. Each of them requires a vector of initial estimates (initial guess) $\mathbf{p}_0$ to start a search for the parameter estimates $\mathbf{p}^*$ (i.e. the final or „optimal“ values). First five implemented algorithms belong to derivative based methods, which use first and possibly also second derivatives of the optimized function. The derivatives are taken with respect to parameters. The sixth implemented method is the simplex method, which does not require derivatives, all it needs to evaluate is the minimized function (i.e. $S(\mathbf{p})$ or its square). The derivative based algorithm tend to be more efficient when the initial estimates $\mathbf{p}_0$ are sufficiently close to $\mathbf{p}^*$. How close they need to be, it depends mainly on how nonlinear a particular model is. When there is a strong nonlinearity and/or it is difficult to produce initial estimates $\mathbf{p}_0$ reasonably close to $\mathbf{p}^*$, the derivative based methods may fail badly. The simplex algorithm might be an alternative to try then. The simplex method might take a very long time to converge. Hence, it is sometimes useful to start with the simplex method, stop it prematurely after the estimates stabilize to some extent, and to use the returned values as the initial values for a derivative based method in the second step. The following algorithms are implemented in the *Nonlinear regression* module:

Gauss-Newton: A classical derivative based algorithm. It is built on the idea of model linearization. When the model is not very nonlinear and/or the initial estimate $\mathbf{p}_0$ is close to $\mathbf{p}^*$, it tends to converge very fast. It can diverge in less ideal situations. To reduce step length problems, the length can be reduced by a damping parameter, $Damp \leq 1$, which is displayed during computations. Its starting value is 1.

Marquardt: A mixed type of derivative based algorithm, which combines Gauss-Newton and gradient approach. It tends to be more reliable than any of the two methods.

Gradient-Cauchy: A derivative based method which uses direction of steepest descent direction together with Cauchy step length found by minimization in the gradient direction. The Cauchy point is determined by a heuristic approach in order to prevent the algorithm from „being locked in" a banana shaped valley. To reduce step length problems, the length can be reduced by a damping parameter, *Damp*, which is displayed during computations. Its starting value is 1. The algorithm can be slow in a banana shaped valley.

Dog Leg: A derivative method which is, like the Marquardt method, based on a combination of gradient and linearization. It uses previous iteration history to improve Hessian (the matrix of second derivatives) approximation, see Denis Mei paper in the Literature. To reduce step length problems, the length can be reduced by a damping parameter, *Damp*, which is displayed during computations. Its starting value is 1. Two additional values *Theta* and *T* are displayed during computations.

Gradient, fixed step length: A derivative method based on the gradient of $S(\mathbf{p})$ only. The method is useful in the earlier stages of optimization. It can be very slow for strongly nonlinear models in later optimization stages (closer to the minimum). To reduce step length problems, the length can be reduced by a damping parameter, *Damp*, which is displayed during computations. Its starting value is 1.

Simplex: This method does not require $S(\mathbf{p})$ derivatives. Geometrically, it corresponds to flipping a simplex (with $m+1$ points) in the parametric space. The QC.Expert™ implementation uses a heuristic approach and so called „mutations" when constructing the simplex. Because the method does not need derivatives at all, it is useful for strongly nonlinear models. It can be very slow, compared to derivative based methods (when the later can be applied). In the course of computations, the simplex expansion coefficient *Norm* is displayed.

Protocol

Task name	Project name as entered in the dialog panel.
Significance level	Alpha, significance/confidence level which is used for all tests/confidence intervals.
Degrees of freedom	Degrees of freedom, $n–m$ (number of data points minus the number of model parameters).
Quantile t(1-alpha/2,n-m)	t-distribution quantile.
Quantile F(1-alpha,m,n-m)	F-distribution quantile.
Method	Method used (least squares method)
Number of data points	Number of complete data rows, having information on all model variables.
Number of parameters	Number of the regression model parameters.
Method	User -selected numerical optimization method.
Explanatory variables	List of explanatory variables which appear in the regression model.
Response	Response variable.
Model	The regression model; response variable appears before the „~" sign.
Initial values	Initial parameter values.

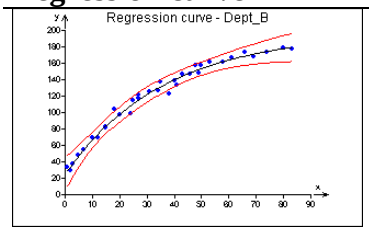
Computations	
Iterations	Number of iterations.
Termination	Optimization algorithm termination; the word *Convergence* is displayed when the algorithm ended in a normal way, reaching convergence; when computations were manually interrupted by pressing the *Stop* button, the word *Interrupted* is displayed; when a pre-specified maximum number of iterations is exceeded without meeting termination criterion, the word *Divergence* is displayed; when no computations were performed, the words *Was not optimized* appear. **Warning**: the word *Convergence* might not necessarily mean that the returned parameter estimates are correct! You should always check adequacy of the fitted model visually and inspect all parts of the output carefully (e.g. correlation matrix of estimates).
Computation time	CPU time (in seconds) spent by the procedure.
Max. iteration number	Pre-specified maximum iteration number. When the number is exceeded without meeting stopping criteria, divergence is claimed.
Termination criterion	Norm of the parameter vector change has to be smaller than this number in order to claim convergence.
Parameter estimates	Parameter estimates found by an optimization algorithm, accompanied by the asymptotic standard errors of the estimates and asymptotic confidence intervals (using a pre-specified α).
Parameter correlation matrix	Asymptotic pairwise correlations for all parameter pairs. Ones appear on the diagonal necessarily. Correlation between parameters should be expected, but when some correlations are close to +1 or –1, results are suspect. It might be useful to reparametrize the model.
Residual analysis	
Characteristic	
Y observed	Observed response value, as it appears in the current data sheet.
Y predicted	Predicted response value.
Std. error of Y	Estimated standard error of the prediction.
Raw residual	Difference between observed and predicted response value.
Residual [%Y]	Relative residual, raw residual divided by the response value.
Weights	Weights for individual observations as inputted by the user.
Residual sum of squares	Residual sum of squares cannot decrease when a new variable is included in the model (usually, it increases).
Mean of absolute residuals	Mean of absolute residuals.
Residual standard deviation	Standard deviation estimated from residuals.
Residual variance	Variance estimated from residuals.
Residual skewness	Skewness estimated from residuals.
Residual kurtosis	Kurtosis estimated from residuals.

Characteristics of the model fit	
Multiple correlation coefficient, R	Multiple correlation coefficient characterizes how closely the model fits the data. It does not necessarily express how good the model is. R cannot decrease when a new variable is included in the model (usually increases whenever a new variable is added)!
Coefficient of determination R^2	Square of the multiple correlation coefficient.

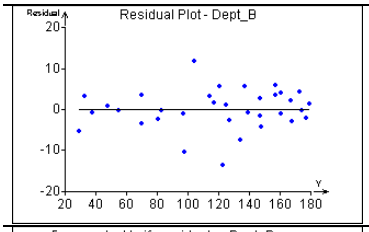
Mean square error of prediction, MEP	The i^{th} error is the difference between actual value of the i^{th} observation and its prediction. The prediction comes from the model based on data with the i^{th} row omitted. MEP is sensitive indicator of problems like multicollinearity and outliers. It is an important characteristics of the regression model quality.
Akaike information criterion	AIC in the regression context is related to the residual sum of squares, penalized by the model size (number of explanatory variables).

Graphs

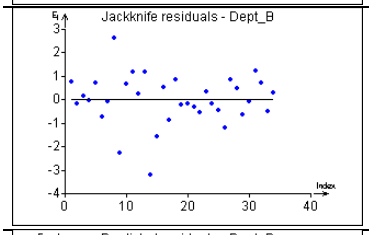
Regression curve



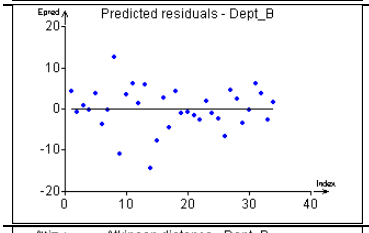
This plot is not produced when the model contains more than one explanatory variable. When only one explanatory variable appears in the model, the plot displays the regression curve. Red curves show the confidence band around the regression curve, computed for a pre-specified confidence coefficient. It should be noted that the confidence band is realistic only when the fitted model is (approximately) correct. This is even more important when predictions further from bulk of available data points are considered. Details of the plot can be inspected upon zooming part of it. The regression curve can be inspected even outside of the interval containing explanatory variable values actually used in model fitting by the inverse zooming.



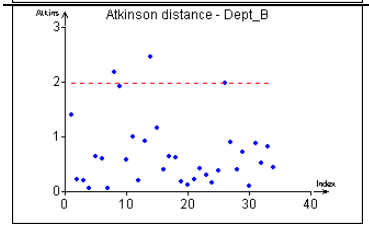
Standardized residuals plot. Predicted response is plotted on the *X* axis, while the standardized residuals are plotted on the *Y* axis. Horizontal line corresponds to the mean of residuals. Any systematic plot pattern suggests an incorrect or incomplete model, or incorrect estimates.



Jackknife residuals plot. Data index is plotted on the *X* axis, while the jackknife residuals are plotted on the *Y* axis. Horizontal line corresponds to the zero residual. Any systematic plot pattern may suggests an incorrect or incomplete model, or incorrect estimates. Outliers are detected much more precisely than on the Standardized residuals plot. Compare Linear regression.

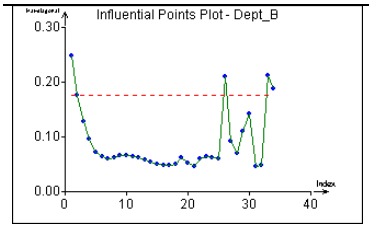


Predicted residuals plot. Data index is plotted on the *X* axis, while the predicted residuals are plotted on the *Y* axis. Horizontal line corresponds to the zero residual. Outliers are detected much more precisely than on the Standardized residuals plot. Compare Linear regression.



Atkinson distance. Data index is plotted on the *X* axis, Atkinson distances are plotted on the *Y* axis. Horizontal red dashed line corresponds to the 95% quantile of the distribution of this statistics, which is used to detect influential data. Points above the red line are assumed highly influential.

Influence



Plot of the projection matrix $\mathbf{H}=\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ diagonal elements. ($\mathbf{X}$ is the matrix of the first partial derivatives w.r.t. the model parameters). The points plotted above the red horizontal line are considered to be potentially influential.