NonLinearModel class

Nonlinear regression model class

Description

An object comprising training data, model description, diagnostic information, and fitted coefficients for a nonlinear regression. Predict model responses with the `predict` or `feval` methods.

Construction

Create a `NonLinearModel` object using `fitnlm`.

Properties

collapse all

`CoefficientCovariance` — Covariance matrix of coefficient estimates

numeric matrix

This property is read-only.

Covariance matrix of coefficient estimates, specified as a p-by-p matrix of numeric values. p is the number of coefficients in the fitted model.

For details, see Coefficient Standard Errors and Confidence Intervals.

Data Types: `single` | `double`

`CoefficientNames` — Coefficient names

cell array of character vectors

This property is read-only.

Coefficient names, specified as a cell array of character vectors, each containing the name of the corresponding term.

Data Types: `cell`

`Coefficients` — Coefficient values

table

This property is read-only.

Coefficient values, specified as a table. `Coefficients` contains one row for each coefficient and these columns:

- `Estimate` — Estimated coefficient value
- `SE` — Standard error of the estimate
- `tStat` — t-statistic for a test that the coefficient is zero
- `pValue` — p-value for the t-statistic

Use `anova` (only for a linear regression model) or `coefTest` to perform other tests on the coefficients. Use `coefCI` to find the confidence intervals of the coefficient estimates.

To obtain any of these columns as a vector, index into the property using dot notation. For example, obtain the estimated coefficient vector in the model `mdl`:

```
beta = mdl.Coefficients.Estimate
```

Data Types: `table`

`Diagnostics` — Diagnostic information

table

This property is read-only.

Diagnostic information for the model, specified as a table. Diagnostics can help identify outliers and influential observations. `Diagnostics` contains the following fields.

Field	Meaning	Utility
`Leverage`	Diagonal elements of `HatMatrix`	Leverage indicates to what extent the predicted value for an observation is determined by the observed value for that observation. A value close to `1` indicates that the prediction is largely determined by that observation, with little contribution from the other observations. A value close to `0` indicates the fit is largely determined by the other observations. For a model with `P` coefficients and `N` observations, the average value of `Leverage` is `P/N`. An observation with `Leverage` larger than `2*P/N` can be regarded as having high leverage.
`CooksDistance`	Cook's measure of scaled change in fitted values	`CooksDistance` is a measure of scaled change in fitted values. An observation with `CooksDistance` larger than three times the mean Cook's distance can be an outlier.
`HatMatrix`	Projection matrix to compute fitted from observed responses	`HatMatrix` is an `N`-by-`N` matrix such that `Fitted = HatMatrix*Y`, where `Y` is the response vector and `Fitted` is the vector of fitted response values.

Data Types: `table`

`DFE` — Degrees of freedom for error

positive integer

This property is read-only.

Degrees of freedom for the error (residuals), equal to the number of observations minus the number of estimated coefficients, specified as a positive integer.

Data Types: `double`

`Fitted` — Fitted response values based on input data

numeric vector

This property is read-only.

Fitted (predicted) values based on the input data, specified as a numeric vector. `fitnlm` attempts to make `Fitted` as close as possible to the response data.

Data Types: `single` | `double`

`Formula` — Model information

`NonLinearFormula` object

This property is read-only.

Model information, specified as a `NonLinearFormula` object.

Display the formula of the fitted model `mdl` by using dot notation.

```
mdl.Formula
```

`Iterative` — Information about fitting process

structure

This property is read-only.

Information about the fitting process, specified as a structure with the following fields:

- `InitialCoefs` — Initial coefficient values (the `beta0` vector)
- `IterOpts` — Options included in the `Options` name-value pair argument for `fitnlm`.

Data Types: `struct`

`LogLikelihood` — Loglikelihood

numeric value

This property is read-only.

Loglikelihood of the model distribution at the response values, specified as a numeric value. The mean is fitted from the model, and other parameters are estimated as part of the model fit.

Data Types: `single` | `double`

`ModelCriterion` — Criterion for model comparison

structure

This property is read-only.

Criterion for model comparison, specified as a structure with these fields:

- `AIC` — Akaike information criterion. `AIC = -2*logL + 2*m`, where `logL` is the loglikelihood and `m` is the number of estimated parameters.
- `AICc` — Akaike information criterion corrected for the sample size. `AICc = AIC + (2*m*(m + 1))/(n - m - 1)`, where `n` is the number of observations.
- `BIC` — Bayesian information criterion. `BIC = -2*logL + m*log(n)`.
- `CAIC` — Consistent Akaike information criterion. `CAIC = -2*logL + m*(log(n) + 1)`.

Information criteria are model selection tools that you can use to compare multiple models fit to the same data. These criteria are likelihood-based measures of model fit that include a penalty for complexity (specifically, the number of parameters). Different information criteria are distinguished by the form of the penalty.

When you compare multiple models, the model with the lowest information criterion value is the best-fitting model. The best-fitting model can vary depending on the criterion used for model comparison.

To obtain any of the criterion values as a scalar, index into the property using dot notation. For example, obtain the AIC value `aic` in the model `mdl`:

```
aic = mdl.ModelCriterion.AIC
```

Data Types: `struct`

`MSE` — Mean squared error

numeric value

This property is read-only.

Mean squared error, specified as a numeric value. The mean squared error is an estimate of the variance of the error term in the model.

Data Types: `single` | `double`

`NumCoefficients` — Number of model coefficients

positive integer

This property is read-only.

Number of coefficients in the fitted model, specified as a positive integer. `NumCoefficients` is the same as `NumEstimatedCoefficients` for `NonLinearModel` objects. `NumEstimatedCoefficients` is equal to the degrees of freedom for regression.

Data Types: `double`

`NumEstimatedCoefficients` — Number of estimated coefficients

positive integer

This property is read-only.

Number of estimated coefficients in the fitted model, specified as a positive integer. `NumEstimatedCoefficients` is the same as `NumCoefficients` for `NonLinearModel` objects. `NumEstimatedCoefficients` is equal to the degrees of freedom for regression.

Data Types: `double`

`NumPredictors` — Number of predictor variables

positive integer

This property is read-only.

Number of predictor variables used to fit the model, specified as a positive integer.

Data Types: `double`

`NumVariables` — Number of variables

positive integer

This property is read-only.

Number of variables in the input data, specified as a positive integer. `NumVariables` is the number of variables in the original table or dataset, or the total number of columns in the predictor matrix and response vector.

`NumVariables` also includes any variables that are not used to fit the model as predictors or as the response.

Data Types: `double`

`ObservationInfo` — Observation information

table

This property is read-only.

Observation information, specified as an n-by-4 table, where n is equal to the number of rows of input data. `ObservationInfo` contains the columns described in this table.

Column	Description
`Weights`	Observation weights, specified as a numeric value. The default value is `1`.

Column	Description
`Excluded`	Indicator of excluded observations, specified as a logical value. The value is `true` if you exclude the observation from the fit by using the `'Exclude'` name-value pair argument.
`Missing`	Indicator of missing observations, specified as a logical value. The value is `true` if the observation is missing.
`Subset`	Indicator of whether or not the fitting function uses the observation, specified as a logical value. The value is `true` if the observation is not excluded or missing, meaning the fitting function uses the observation.

To obtain any of these columns as a vector, index into the property using dot notation. For example, obtain the weight vector `w` of the model `mdl`:

```
w = mdl.ObservationInfo.Weights
```

Data Types: `table`

`ObservationNames` — Observation names

cell array of character vectors

This property is read-only.

Observation names, specified as a cell array of character vectors containing the names of the observations used in the fit.

- If the fit is based on a table or dataset containing observation names, `ObservationNames` uses those names.
- Otherwise, `ObservationNames` is an empty cell array.

Data Types: `cell`

`PredictorNames` — Names of predictors used to fit model

cell array of character vectors

This property is read-only.

Names of predictors used to fit the model, specified as a cell array of character vectors.

Data Types: `cell`

`Residuals` — Residuals for fitted model

table

This property is read-only.

Residuals for the fitted model, specified as a table that contains one row for each observation and the columns described in this table.

Column	Description
`Raw`	Observed minus fitted values
`Pearson`	Raw residuals divided by the root mean squared error (RMSE)
`Standardized`	Raw residuals divided by their estimated standard deviation
`Studentized`	Raw residual divided by an independent estimate of the residual standard deviation. The residual for observation i is divided by an estimate of the error standard deviation based on all observations except observation i.

Use `plotResiduals` to create a plot of the residuals. For details, see Residuals.

Rows not used in the fit because of missing values (in `ObservationInfo.Missing`) or excluded values (in `ObservationInfo.Excluded`) contain `NaN` values.

To obtain any of these columns as a vector, index into the property using dot notation. For example, obtain the raw residual vector `r` in the model `mdl`:

```
r = mdl.Residuals.Raw
```

Data Types: `table`

`ResponseName` — Response variable name

character vector

This property is read-only.

Response variable name, specified as a character vector.

Data Types: `char`

`RMSE` — Root mean squared error

numeric value

This property is read-only.

Root mean squared error, specified as a numeric value. The root mean squared error is an estimate of the standard deviation of the error term in the model.

Data Types: `single` | `double`

`Robust` — Robust fit information

structure

This property is read-only.

Robust fit information, specified as a structure with the following fields:

Field	Description

Field	Description
`WgtFun`	Robust weighting function, such as `'bisquare'` (see `robustfit`)
`Tune`	Value specified for tuning parameter (can be `[]`)
`Weights`	Vector of weights used in final iteration of robust fit

This structure is empty unless `fitnlm` constructed the model using robust regression.

Data Types: `struct`

`Rsquared` — R-squared value for model

structure

This property is read-only.

R-squared value for the model, specified as a structure with two fields:

- `Ordinary` — Ordinary (unadjusted) R-squared
- `Adjusted` — R-squared adjusted for the number of coefficients

The R-squared value is the proportion of the total sum of squares explained by the model. The ordinary R-squared value relates to the `SSR` and `SST` properties:

```
Rsquared = SSR/SST,
```

where `SST` is the total sum of squares, and `SSR` is the regression sum of squares.

For details, see Coefficient of Determination (R-Squared).

To obtain either of these values as a scalar, index into the property using dot notation. For example, obtain the adjusted R-squared value in the model `mdl`:

```
r2 = mdl.Rsquared.Adjusted
```

Data Types: `struct`

`SSE` — Sum of squared errors

numeric value

This property is read-only.

Sum of squared errors (residuals), specified as a numeric value.

Data Types: `single` | `double`

`SSR` — Regression sum of squares

numeric value

This property is read-only.

Regression sum of squares, specified as a numeric value. The regression sum of squares is equal to the sum of squared deviations of the fitted values from their mean.

Data Types: `single` | `double`

`SST` — Total sum of squares

numeric value

This property is read-only.

Total sum of squares, specified as a numeric value. The total sum of squares is equal to the sum of squared deviations of the response vector `y` from the `mean(y)`.

Data Types: `single` | `double`

`VariableInfo` — Information about variables

table

This property is read-only.

Information about variables contained in `Variables`, specified as a table with one row for each variable and the columns described in this table.

Column	Description
`Class`	Variable class, specified as a cell array of character vectors, such as `'double'` and `'categorical'`
`Range`	Variable range, specified as a cell array of vectors • Continuous variable — Two-element vector `[min,max]`, the minimum and maximum values • Categorical variable — Vector of distinct variable values
`InModel`	Indicator of which variables are in the fitted model, specified as a logical vector. The value is `true` if the model includes the variable.
`IsCategorical`	Indicator of categorical variables, specified as a logical vector. The value is `true` if the variable is categorical.

`VariableInfo` also includes any variables that are not used to fit the model as predictors or as the response.

Data Types: `table`

`VariableNames` — Names of variables

cell array of character vectors

This property is read-only.

Names of variables, specified as a cell array of character vectors.

- If the fit is based on a table or dataset, this property provides the names of the variables in the table or dataset.

- If the fit is based on a predictor matrix and response vector, `VariableNames` contains the values specified by the `'VarNames'` name-value pair argument of the fitting method. The default value of `'VarNames'` is `{'x1','x2',...,'xn','y'}`.

`VariableNames` also includes any variables that are not used to fit the model as predictors or as the response.

Data Types: `cell`

`Variables` — Input data

table

This property is read-only.

Input data, specified as a table. `Variables` contains both predictor and response values. If the fit is based on a table or dataset array, `Variables` contains all the data from the table or dataset array. Otherwise, `Variables` is a table created from the input data matrix `X` and the response vector `y`.

`Variables` also includes any variables that are not used to fit the model as predictors or as the response.

Data Types: `table`

Object Functions

`coefCI`	Confidence intervals of coefficient estimates of nonlinear regression model
`coefTest`	Linear hypothesis test on nonlinear regression model coefficients
`feval`	Evaluate nonlinear regression model prediction
`partialDependence`	Compute partial dependence
`plotPartialDependence`	Create partial dependence plot (PDP) and individual conditional expectation (ICE) plots
`plotDiagnostics`	Plot diagnostics of nonlinear regression model
`plotResiduals`	Plot residuals of nonlinear regression model
`plotSlice`	Plot of slices through fitted nonlinear regression surface
`predict`	Predict response of nonlinear regression model
`random`	Simulate responses for nonlinear regression model

Copy Semantics

Value. To learn how value classes affect copy operations, see Copying Objects.

Examples

collapse all

Fit a Nonlinear Regression Model

Fit a nonlinear regression model for auto mileage based on the `carbig` data. Predict the mileage of an average car.

View MATLAB Command

Load the sample data. Create a matrix `X` containing the measurements for the horsepower (`Horsepower`) and weight (`Weight`) of each car. Create a vector `y` containing the response values in miles per gallon (`MPG`).

```
load carbig
X = [Horsepower,Weight];
y = MPG;
```

Fit a nonlinear regression model.

```
modelfun = @(b,x)b(1) + b(2)*x(:,1).^b(3) + ...
    b(4)*x(:,2).^b(5);
beta0 = [-50 500 -1 500 -1];
mdl = fitnlm(X,y,modelfun,beta0)
```

```
mdl =
Nonlinear regression model:
    y ~ b1 + b2*x1^b3 + b4*x2^b5

Estimated Coefficients:
          Estimate      SE        tStat       pValue
          ________    _______    ________    ________

    b1     -49.383     119.97    -0.41164     0.68083
    b2      376.43     567.05     0.66384     0.50719
    b3    -0.78193    0.47168     -1.6578    0.098177
    b4      422.37     776.02     0.54428     0.58656
    b5    -0.24127    0.48325    -0.49926     0.61788


Number of observations: 392, Error degrees of freedom: 387
Root Mean Squared Error: 3.96
R-Squared: 0.745,  Adjusted R-Squared 0.743
F-statistic vs. constant model: 283, p-value = 1.79e-113
```

Find the predicted mileage of an average car. Because the sample data contains some missing (`NaN`) observations, compute the mean using `mean` with the `'omitnan'` option.

```
Xnew = mean(X,'omitnan')
```

```
Xnew = 1×2
10^3 ×

    0.1051    2.9794
```

```
MPGnew = predict(mdl,Xnew)
```

```
MPGnew = 21.8073
```

More About

collapse all

Hat Matrix

The hat matrix H is defined in terms of the data matrix X and the Jacobian matrix J:

Here f is the nonlinear model function, and β is the vector of model coefficients.

The Hat Matrix H is

$$H = J(J^TJ)^{-1}J^T.$$

The diagonal elements H_{ii} satisfy

where n is the number of observations (rows of X), and p is the number of coefficients in the regression model.

Leverage

Leverage is a measure of the effect of a particular observation on the regression predictions due to the position of that observation in the space of the inputs.

The leverage of observation i is the value of the ith diagonal term h_{ii} of the hat matrix H. Because the sum of the leverage values is p (the number of coefficients in the regression model), an observation i can be considered an outlier if its leverage substantially exceeds p/n, where n is the number of observations.

Cook’s Distance

The Cook’s distance D_i of observation i is

where

- is the jth fitted response value.
- is the jth fitted response value, where the fit does not include observation i.
- MSE is the mean squared error.
- p is the number of coefficients in the regression model.

Cook’s distance is algebraically equivalent to the following expression:

where e_i is the ith residual.

See Also

`fitnlm` | `GeneralizedLinearModel` | `LinearModel` | `nlinfit` | `predict`

Topics

Nonlinear Regression Workflow

Nonlinear Regression