Package ‘nlstools’

October 13, 2022

Version 2.0-0
Title Tools for Nonlinear Regression Analysis
Imports graphics, grDevices, stats
Suggests knitr, rmarkdown, rticles
Description Several tools for assessing the quality of fit of a gaussian nonlinear model are provided.
URL https://github.com/aursiber/nlstools
BugReports https://github.com/aursiber/nlstools/issues
License GPL-3
Encoding UTF-8
LazyData true
VignetteBuilder knitr
NeedsCompilation no
Author Florent Baty [aut],
       Marie-Laure Delignette-Muller [aut],
       Sandrine Charles [ctb],
       Jean-Pierre Flandrois [ctb],
       Christian Ritz [ctb],
       Aurelie Siberchicot [aut, cre]
Maintainer Aurelie Siberchicot <aurelie.siberchicot@univ-lyon1.fr>
Repository CRAN
Date/Publication 2021-09-17 11:40:02 UTC

R topics documented:

confint2 ................................................................. 2
L.minor ................................................................. 3
michaelisdata .......................................................... 3
michaelismodels ...................................................... 4
nlsBoot ................................................................. 5
nlsBootPredict ......................................................... 7
Description

Produces confidence intervals for the parameters in nonlinear regression model fit. The intervals can either be based large sample results or on profiling.

Usage

confint2(object, parm, level = 0.95, method = c("asymptotic", "profile"), ...)

Arguments

object object of class nls.
parm a vector character strings with names of the parameter for which to calculate confidence intervals (by default all parameters).
level the confidence level required.
method method to be used: "asymptotic" for large sample and "profile" for profiling approach.
... additional argument(s) to pass on the method doing the profiling.

Details

The profiling used is the method confint.nls.

Value

A matrix with columns giving lower and upper confidence limits for each parameter.

Author(s)

Christian Ritz

Examples

L.minor.m1 <- nls(rate ~ Vm*conc/(K+conc), data = L.minor, start = list(K=20, Vm=120))
confint2(L.minor.m1)
confint2(L.minor.m1, "K")
L. minor

Enzyme kinetics

Description
Enzyme kinetics

Usage
data(L.minor)

Format
A data frame with 8 observations on the following 2 variables.

conc  a numeric vector
rate  a numeric vector

Source

---

michaelisdata  Michaelis Menten data sets

Description
Michaelis Menten data sets

Usage
data(vmkm)
data(vmkmki)

Format
*vmkm* is a data frame with 2 columns (S: concentration of substrat, v: reaction rate)
*vmkmki* is a data frame with 3 columns (S: concentration of substrat, I: concentration of inhibitor, v: reaction rate)

Source
These datasets were provided by the French research unit INRA UMR1233.
michaelismodels

Examples

data(vmkm)
data(vmkmki)
plot(vmkm)
plot(vmkmki)

michaelismodels Michaelis-Menten model and derived equations to model competitive and non-competitive inhibition

Description

Formula of Michaelis-Menten model commonly used to describe enzyme kinetics, and derived formulas taking into account the effect of a competitive or a non-competitive inhibitor

Usage

michaelis
compet_mich
non_compet_mich

Details

These models describe the evolution of the reaction rate ($v$) as a function of the concentration of substrate ($S$) and the concentration of inhibitor ($I$) for compet_mich and non_compet_mich.

michaelis is the classical Michaelis-Menten model (Dixon, 1979) with two parameters ($K_m$, $V_{max}$):

$$v = \frac{S}{S + K_m} V_{max}$$

compet_mich is the Michaelis-Menten derived model with three parameters ($K_m$, $V_{max}$, $K_i$), describing a competitive inhibition:

$$v = \frac{S}{S + K_m(1 + \frac{I}{K_i})} V_{max}$$

non_compet_mich is the Michaelis-Menten derived model with three parameters ($K_m$, $V_{max}$, $K_i$), describing a non-competitive inhibition:

$$v = \frac{S}{(S + K_m)(1 + \frac{I}{K_i})} V_{max}$$
Value

A formula

Author(s)

Florent Baty, Marie-Laure Delignette-Muller

References


Examples

# Example 1

data(vkmkm)
nls1 <- nls(michaelis, vkmkm, list(Km=1, Vmax=1))
plotfit(nls1, smooth = TRUE)

# Example 2

data(vkmkmki)
def.par <- par(no.readonly = TRUE)
par(mfrow = c(2,2))
nls2_c <- nls(compet_mich, vkmkmki, list(Km=1, Vmax=20, Ki=0.5))
plotfit(nls2_c, variable=1)
overview(nls2_c)
res2_c <- nlsResiduals(nls2_c)
plot(res2_c, which=1)
nls2_nc <- nls(non_compet_mich, vkmkmki, list(Km=1, Vmax=20, Ki=0.5))
plotfit(nls2_nc, variable=1)
overview(nls2_nc)
res2_nc <- nlsResiduals(nls2_nc)
plot(res2_nc, which=1)
par(def.par)

---

**nlsBoot**

*Bootstrap resampling*

**Description**

Bootstrap resampling
Usage

nlsBoot (nls, niter = 999)
## S3 method for class 'nlsBoot'
plot(x, type = c("pairs", "boxplot"),
     mfr = c(ceiling(sqrt(ncol(x$coefboot))),
            ceiling(sqrt(ncol(x$coefboot)))),
     ask = FALSE, ...)
## S3 method for class 'nlsBoot'
print(x, ...)
## S3 method for class 'nlsBoot'
summary(object, ...)

Arguments

nls an object of class 'nls'
niter number of iterations
x, object an object of class 'nlsBoot'
type type of representation (options are "pairs" or "boxplot")
mfr layout definition (number of rows and columns in the graphics device)
ask if TRUE, draw plot interactively
... further arguments passed to or from other methods

Details

Non-parametric bootstrapping is used. Mean centered residuals are bootstrapped. By default, 999 resampled data sets are created from which parameter estimates are obtained by fitting the model on each of these data sets. Whenever the fit fails to converge, a flag reports the number of non-convergences. If the fitting procedure fails to converge in more than 50% of the cases, the procedure is interrupted with a flag and no result is given. The function summary returns the bootstrap estimates (mean and std. dev. of the bootstrapped estimates) and the median and 95 percent confidence intervals (50, 2.5, and 97.5 percentiles of the bootstrapped estimates). The bootstrapped estimate distributions can be visualized using the function plot.nlsBoot either by plotting the bootstrapped sample for each pair of parameters or by displaying the boxplot representation of the bootstrapped sample for each parameter. Notice that nlsBoot does not currently handle transformed dependent variables specified in the left side of the nls formula.

Value

nlsBoot returns a list of 5 objects:

coeefboot contains the bootstrap parameter estimates
bootCI contains the bootstrap medians and the bootstrap 95% confidence intervals
estiboot contains the means and std. errors of the bootstrap parameter estimates
rse is the vector of bootstrap residual errors
nls the object of class 'nls' given in input
Author(s)
Florent Baty, Marie-Laure Delignette-Muller

References


Examples

```r
formulaExp <- as.formula(VO2 ~ (t <= 5.883) * VO2res + (t > 5.883) *
                          (VO2res + (VO2peak - VO2res) *
                           (1 - exp(-(t - 5.883) / mu))))
O2K.nls1 <- nls(formulaExp, start = list(VO2res = 400, VO2peak = 1600,
                                           mu = 1), data = O2K)
O2K.boot1 <- nlsBoot(O2K.nls1, niter = 200)
plot(O2K.boot1)
plot(O2K.boot1, type = "boxplot", ask = FALSE)
summary(O2K.boot1)
```

nlsBootPredict

**Prediction from Bootstrap resampling**

Description
Computation of confidence intervals on predictions from Bootstrap resampling

Usage

```r
nlsBootPredict(nlsBoot, newdata, interval = c("confidence", "prediction"))
```

Arguments

- `nlsBoot` An object of class 'nlsBoot'.
- `newdata` A data frame in which to look for values of independent variables for the predictions. If omitted, the data used for fitting are used.
- `interval` Type of interval to compute, "confidence", or "prediction".
Details

nlsBootPredict produces confidence intervals on predicted values that can be obtained using predict.nls for values of the independent variable(s) defined in the data frame newdata. Non-parametric bootstrapping is used (results of nlsBoot). For confidence intervals the bootstrap sample of predictions is simply computed from the bootstrap sample of estimations of the model parameters, by evaluating the mean value of the model on each new data. For prediction intervals, to take into account the residual errors, a residual error sampled in the mean centered residuals is added to each mean predicted value. In both cases, bootstrap predictions are summarized by the median and 95 percent confidence intervals (50, 2.5, and 97.5 percentiles of the bootstrapped values).

Value

nlsBoot returns a matrix of predictions with three columns respectively corresponding to the 50, 2.5 and 97.5 percentiles of bootstrap predictions.

Author(s)

Florent Baty, Marie-Laure Delignette-Muller

References


See Also

See nlsBoot and predict.nls.

Examples

formulaExp <- as.formula(VO2 ~ (t <= 5.883) * VO2res + (t > 5.883) * (VO2res + (VO2peak - VO2res) * (1 - exp(-(t - 5.883) / mu)))))
O2K.nls1 <- nls(formulaExp, start = list(VO2res = 400, VO2peak = 1600, mu = 1), data = O2K)
niter <- 200

### To reach stable prediction intervals use far greater niter (>> 1000)
O2K.boot1 <- nlsBoot(O2K.nls1, niter = niter)
newdata <- data.frame(t = seq(0, 12, length.out = 50))
(pred.clim <- nlsBootPredict(O2K.boot1, newdata = newdata, interval = "confidence"))
(pred.plim <- nlsBootPredict(O2K.boot1, newdata = newdata, interval = "prediction"))

plotfit(O2K.nls1, smooth = TRUE, ylim = c(200, 1800))
lines(newdata$t, pred.clim[, 2], col = "red")
lines(newdata$t, pred.clim[, 3], col = "red")
lines(newdata$t, pred.plim[, 2], col = "blue")
lines(newdata$t, pred.plim[, 3], col = "blue")

### An example without giving newdata

# plot of data
nlsConfRegions

plot(O2K$t, O2K$VO2)

# add of predictions computed using predict.nls()
pred <- predict(O2K.nls1)
points(O2K$t, pred, pch = 16)

# add of prediction intervals using nlsBootPredict()
(pred.plim <- nlsBootPredict(O2K.boot1, interval = "prediction"))
segments(O2K$t, pred.plim[, 2], O2K$t, pred.plim[, 3], col = "blue")

---

### nlsConfRegions

**Confidence regions**

#### Description

Draws parameter values in the Beale’s 95 percent unlinearized confidence region

#### Usage

```r
nlsConfRegions (nls, length = 1000, exp = 1.5)
```

#### Arguments

- `nls`: an object of class `nls`
- `length`: number of points to draw in the confidence region
- `exp`: expansion factor of the hypercube in which random values of parameters are drawn
- `x`: an object of class `nlsConfRegions`
- `bounds`: logical defining whether bounds of the drawing hypercube are plotted
- `ask`: if TRUE, draw plot interactively
- `...`: further arguments passed to or from other methods

#### Details

A sample of points in the 95 percent confidence region is computed according to Beale’s criterion (Beale, 1960). This region is also named the joint parameter likelihood region (Bates and Watts, 1988). The method used consists in a random sampling of parameters values in a hypercube centered on the least squares estimate and rejecting the parameters values whose residual sum of squares do not verify the Beale criterion. The confidence region is plotted by projection of the sampled points in each plane defined by a couple of parameters. Bounds of the hypercube in which random values of parameters are drawn may be plotted in order to check if the confidence region was totally included in the hypercube defined by default. If not the hypercube should be expanded in order to obtain the full confidence region.
Value

\texttt{nlsConfRegions} returns a list of four objects:

- \texttt{cr} a data frame containing the sample drawn in the Beale’s confidence region
- \texttt{rss} a vector containing the residual sums of squares corresponding to \texttt{cr}
- \texttt{rss95} the 95 percent residual sum of squares threshold according to Beale (1960)
- \texttt{bounds} lower and upper bounds of the hypercube in which random values of parameters have been drawn

Author(s)

Florent Baty, Marie-Laure Delignette-Muller

References


See Also

\texttt{ellipse.nls} in the \texttt{ellipse} library

Examples

```r
formulaExp <- as.formula(VO2 ~ (t <= 5.883) * VO2rest + (t > 5.883) *
                          (VO2rest + (VO2peak - VO2rest) *
                           (1 - exp(-(t - 5.883) / mu))))
O2K.nls1 <- nls(formulaExp, start = list(VO2rest = 400, VO2peak = 1600,
                                          mu = 1), data = O2K)
O2K.conf1 <- nlsConfRegions(O2K.nls1, exp = 2, length = 200)
plot(O2K.conf1, bounds = TRUE)
```

\texttt{nlsContourRSS} \hspace{1cm} \textit{Surface contour of RSS}

Description

Provides residual sum of squares (RSS) contours
Usage

nlsContourRSS (nls, lseq = 100, exp = 2)
## S3 method for class 'nlsContourRSS'
plot(x, nlev = 0, col = TRUE, col.pal = terrain.colors(100),
   ask = FALSE, useRaster = TRUE, ...)
## S3 method for class 'nlsContourRSS'
print(x, ...)

Arguments

nls an object of class 'nls'
lseq length of the sequences of parameters
exp expansion factor of the parameter intervals defining the grids
nlev number of contour levels to add to the likelihood contour at level 95 percent
col logical. Contours are plotted with colors if TRUE
col.pal Palette of colors. Colors to be used as background (default is terrain.colors(100); unused if col is FALSE)
x an object of class 'nlsContourRSS'
ask if TRUE, draw plot interactively (default is FALSE)
useRaster a bitmap raster is used to plot the image instead of polygons (default is TRUE)
... further arguments passed to or from other methods

Details

The aim of these functions is to plot the residual sum of squares (RSS) contours which correspond
to likelihood contours for a Gaussian model. For each pair of parameters the RSS is calculated on a
grid centered on the least squares estimates of both parameters, the other parameters being fixed to
their least square estimates. The contours of RSS values are then plotted for each pair of parameters.
For each pair of parameters, one of this contour corresponds to a section of the 95 percent Beale’s
confidence region in the plane of these parameters. This contour is plotted in a different color.

Value

nlsContourRSS returns a list of three objects:

seqPara a matrix with the sequence of grid values for each parameter
lrss a list of matrices with logarithm values of RSS in the grid for each pair of parameters
lrss95 the logarithm of the 95 percent residual sum of squares threshold according to
   Beale (1960)

Author(s)

Florent Baty, Marie-Laure Delignette-Muller
References


Examples

```r
formulaExp <- as.formula(VO2 ~ (t <= 5.883) * VO2rest + (t > 5.883) * 
                        (VO2rest + (VO2peak - VO2rest) * 
                         (1 - exp(-(t - 5.883) / mu))))
O2K.nls1 <- nls(formulaExp, start = list(VO2rest = 400, VO2peak = 1600, 
                                      mu = 1), data = O2K)
O2K.cont1 <- nlsContourRSS(O2K.nls1)
plot(O2K.cont1)
```

```
## S3 method for class 'nlsJack'
plot(x, mfr = c(nrow(x$reldif),1), ask = FALSE, ...)
## S3 method for class 'nlsJack'
print(x, ...)
## S3 method for class 'nlsJack'
summary(object, ...)
```

nlsJack

### Jackknife resampling

Description

Jackknife resampling

Usage

```
nlsJack (nls)
## S3 method for class 'nlsJack'
plot(x, mfr = c(nrow(x$reldif),1), ask = FALSE, ...)
## S3 method for class 'nlsJack'
print(x, ...)
## S3 method for class 'nlsJack'
summary(object, ...)
```

Arguments

- `nls` an object of class 'nls'
- `x, object` an object of class 'nlsJack'
- `mfr` layout definition, default is k rows (k: number of parameters) and 1 column
- `ask` if TRUE, draw plot interactively
- `...` further arguments passed to or from other methods
Details

A jackknife resampling procedure is performed. Each observation is sequentially removed from the initial data set using a leave-one-out strategy. A data set with \( n \) observations provides thus \( n \) resampled data sets of \( n-1 \) observations. The jackknife estimates with confidence intervals are calculated as described by Seber and Wild (1989) from the results of \( n \) new fits of the model on the \( n \) jackknife resampled data sets. The leave-one-out procedure is also employed to assess the influence of each observation on each parameter estimate. An observation is empirically defined as influential for one parameter if the difference between the estimate of this parameter with and without the observation exceeds twice the standard error of the estimate divided by \( \sqrt{n} \). This empirical method assumes a small curvature of the nonlinear model. For each parameter, the absolute relative difference (in percent of the estimate) of the estimates with and without each observation is plotted. An asterisk is plotted for each influential observation.

Value

`nlsJack` returns a list with 7 objects:

- `estijack`: a data frame with jackknife estimates and bias
- `coefjack`: a data frame with the parameter estimates for each jackknife sample
- `reldif`: a data frame with the absolute relative difference (in percent of the estimate) of the estimates with and without each observation
- `dfb`: a data frame with dfbetas for each parameter and each observation
- `jackCI`: a data frame with jackknife confidence intervals
- `rse`: a vector with residual standard error for each jackknife sample
- `rss`: residual a vector with residual sum of squares for each jackknife sample

Author(s)

Florent Baty, Marie-Laure Delignette-Muller

References


Examples

```r
formulaExp <- as.formula(V02 ~ (t <= 5.883) * V02rest + (t > 5.883) * (V02rest + (V02peak - V02rest) * (1 - exp(-(t - 5.883) / mu))))
O2K.nls1 <- nls(formulaExp, start = list(V02rest = 400, V02peak = 1600, mu = 1), data = O2K)
O2K.jack1 <- nlsJack(O2K.nls1)
plot(O2K.jack1)
summary(O2K.jack1)
```
**nlsResiduals**

**NLS residuals**

**Description**

Provides several plots and tests for the analysis of residuals

**Usage**

```r
nlsResiduals(nls)
## S3 method for class 'nlsResiduals'
plot(x, which = 0, ...)
test.nlsResiduals(x)
## S3 method for class 'nlsResiduals'
print(x, ...)
```

**Arguments**

- `nls`: an object of class 'nls'
- `x`: an object of class 'nlsResiduals'
- `which`: an integer:
  - 0 = 4 graphs of residuals (types 1, 2, 4 and 6)
  - 1 = non-transformed residuals against fitted values
  - 2 = standardized residuals against fitted values
  - 3 = sqrt of absolute value of standardized residuals against fitted values
  - 4 = auto-correlation residuals (i+1th residual against ith residual)
  - 5 = histogram of the residuals
  - 6 = qq-plot of the residuals
- `...`: further arguments passed to or from other methods

**Details**

Several plots and tests are proposed to check the validity of the assumptions of the error model based on the analysis of residuals.

The function `plot.nlsResiduals` proposes several plots of residuals from the nonlinear fit: plot of non-transformed residuals against fitted values, plot of standardized residuals against fitted values, plot of square root of absolute value of standardized residuals against fitted values, auto-correlation plot of residuals (i+1th residual against ith residual), histogram of the non-transformed residuals and normal Q-Q plot of standardized residuals.

`test.nlsResiduals` tests the normality of the residuals with the Shapiro-Wilk test (`shapiro.test` in package stats) and the randomness of residuals with the runs test (Siegel and Castellan, 1988). The `runs.test` function used in nlstools is the one implemented in the package tseries.
Value

nlsResiduals returns a list of five objects:

- std95: the Student value for alpha=0.05 (bilateral) and the degree of freedom of the model
- resi1: a matrix with fitted values vs. non-transformed residuals
- resi2: a matrix with fitted values vs. standardized residuals
- resi3: a matrix with fitted values vs. sqrt(abs(standardized residuals))
- resi4: a matrix with i-th residuals vs. (i+1)-th residuals

Author(s)

Florent Baty, Marie-Laure Delignette-Muller

References


Examples

# Plots of residuals
formulaExp <- as.formula(VO2 ~ (t <= 5.883) * VO2rest + (t > 5.883) * (VO2rest + (VO2peak - VO2rest) * (1 - exp(-(t - 5.883) / mu))))
O2K.nls1 <- nls(formulaExp, start = list(VO2rest = 400, VO2peak = 1600, mu = 1), data = O2K)
O2K.res1 <- nlsResiduals(O2K.nls1)
plot(O2K.res1, which = 0)

# Histogram and qq-plot
plot(O2K.res1, which = 5)
plot(O2K.res1, which = 6)

# Tests
test.nlsResiduals(O2K.res1)
Usage

preview (formula, data, start, variable = 1)
plotfit (x, smooth = FALSE, variable = 1, xlab = NULL, ylab = NULL,
        pch.obs = 1, pch.fit = "+", lty = 1, lwd = 1, col.obs = "black",
        col.fit = "red", ...)
overview (x)

Arguments

formula      formula of a non-linear model
data         a data frame with header matching the variables given in the formula
start        a list of parameter starting values which names match the parameters given in
             the formula
variable     index of the variable to be plotted against the predicted values; default is the first
             independent variable as it appears in the original dataset
x            an object of class ‘nls’
smooth       a logical value, default is FALSE. If smooth is TRUE, a plot of observed values
             is plotted as a function of 1000 values continuously taken in the range interval
             [min(variable),max(variable)]. This option can only be used if the number of
             controlled variables is 1.
xlab         X-label
ylab         Y-label
pch.obs      type of point of the observed values
pch.fit      type of point of the fitted values (not applicable if smooth=TRUE)
lty          type of line of the smoothed fitted values (if smooth=TRUE)
lwd          thickness of line of the smoothed fitted values (if smooth=TRUE)
col.obs      color of the observed points
col.fit      color of the fitted values
...          further arguments passed to or from other methods

Details

The function preview helps defining the parameter starting values prior fitting the model. It
provides a superimposed plot of observed (circles) and predicted (crosses) values of the dependent
variable versus one of the independent variables with the model evaluated at the starting values of
the parameters. The function overview returns the parameters estimates, their standard errors as
well as their asymptotic confidence intervals and the correlation matrix (alternately, the function
confint provides better confidence interval estimates whenever it converges). plotfit displays a
superimposed plot of the dependent variable versus one the independent variables together with the
fitted model.

Author(s)

Florent Baty, Marie-Laure Delignette-Muller
References


See Also

nls in the stats library and confint.nls in the package MASS

Examples

```r
formulaExp <- as.formula(VO2 ~ (t <= 5.883) * VO2rest + (t > 5.883) *
                        (VO2rest + (VO2peak - VO2rest) * 
                        (1 - exp(-(t - 5.883) / mu))))
preview(formulaExp, O2K, list(VO2rest = 400, VO2peak = 1600, mu = 1))
O2K.nls1 <- nls(formulaExp, start = list(VO2rest = 400, VO2peak = 1600, 
                              mu = 1), data = O2K)
overview(O2K.nls1)
plotfit(O2K.nls1, smooth = TRUE)
```

---

**Defunct Functions in Package nlstools**

Description

The models or data sets listed here are no longer part of package nlstools. In order to access these models and data set in the future, please load the additional package nlsMicrobio.

Details

Defunct functions are:
geeraerd
geeraerd_without_Nres
gleeraerd_without_Sl
mafart
albert
trilinear
bilinear_without_Nres
bilinear_without_Sl
baranyi
baranyi_without_Nmax
baranyi_without_lag
buchanan
buchanan_without_Nmax
buchanan_without_lag
O2K  

Oxygen kinetics during 6-minute walk test data set

Description

Oxygen uptake kinetics during a 6-minute walking test in a patient with pulmonary disease. The first 5.83 minutes correspond to the resting phase prior to exercise.

Usage

data(O2K)

Format

O2K is a data frame with 2 columns (t: time, VO2: oxygen uptake)

Source

This data set was provided by the Cantonal Hospital St. Gallen, Switzerland.

Examples

data(O2K)
plot(O2K)
Index

* datasets
  L.minor, 3
  michaelisdata, 3
  O2K, 18
* models
  confint2, 2
  michaelismodels, 4
* nonlinear
  confint2, 2
  nlsBoot, 5
  nlsBootPredict, 7
  nlsConfRegions, 9
  nlsContourRSS, 10
  nlsJack, 12
  nlsResiduals, 14
  nlstools, 15
  compet_mich (michaelismodels), 4
  confint.nls, 2
  confint2, 2
  L.minor, 3
  michaelis (michaelismodels), 4
  michaelisdata, 3
  michaelismodels, 4
  nls, 2
  nlsBoot, 5, 8
  nlsBootPredict, 7
  nlsConfRegions, 9
  nlsContourRSS, 10
  nlsJack, 12
  nlsResiduals, 14
  nlstools, 15
  nlstools-defunct, 17
  non_compet_mich (michaelismodels), 4
  O2K, 18
  overview (nlstools), 15
  plot.nlsBoot (nlsBoot), 5
  plot.nlsConfRegions (nlsConfRegions), 9
  plot.nlsContourRSS (nlsContourRSS), 10
  plot.nlsJack (nlsJack), 12
  plot.nlsResiduals (nlsResiduals), 14
  plotfit (nlstools), 15
  predict.nls, 8
  preview (nlstools), 15
  print.nlsBoot (nlsBoot), 5
  print.nlsConfRegions (nlsConfRegions), 9
  print.nlsContourRSS (nlsContourRSS), 10
  print.nlsJack (nlsJack), 12
  print.nlsResiduals (nlsResiduals), 14
  summary.nlsBoot (nlsBoot), 5
  summary.nlsJack (nlsJack), 12
  test.nlsResiduals (nlsResiduals), 14
  vmkm (michaelisdata), 3
  vmkmki (michaelisdata), 3