

Package ‘quantchem’

April 19, 2009

Type Package

Title Quantitative chemical analysis: calibration and evaluation of results

Version 0.12-1

Date 2007-04-11

Depends R (>= 2.0), MASS, outliers

Author Lukasz Komsta

Maintainer Lukasz Komsta <luke@novum.am.lublin.pl>

Description Statistical evaluation of calibration curves by different regression techniques: ordinary, weighted, robust (up to 4th order polynomial). Log-log and Box-Cox transform, estimation of optimal power and weighting scheme. Tests for heteroscedascity and normality of residuals. Different kinds of plots commonly used in illustrating calibrations. Easy “inverse prediction” of concentration by given responses and statistical evaluation of results (comparison of precision and accuracy by common tests).

License GPL (>= 2)

URL <http://www.r-project.org>, <http://www.komsta.net/>

Repository CRAN

Date/Publication 2007-04-11 09:10:01

R topics documented:

AIC.cal	2
anova.lmcal, anova.nlscal	3
confint.cal	4
derivative	5
dstat	6
ibuprofen, genisten, biochanin, pseudoephedrine, nitrate	7
lmcal, nlscal	8
lof	10

plot.lmcal, plot.nlscal	11
predict.lmcal, predict.nlscal	13
residuals.cal	14
summary.lmcal, summary.nlscal	15
tablets	16
vstat	17
Index	19

AIC.cal

Akaike's An Information Criterion for calibration models

Description

This function computes a table of AIC values for given object inheriting from class 'cal' ('lmcal' and 'nlscal').

Usage

```
AIC.cal(object, ..., k = 2)
```

Arguments

object	a fitted calibration model of class 'lmcal' or 'nlscal'
...	additional arguments (ignored)
k	the k parameter, for more information see AIC

Value

A matrix with AIC values for each fitted model.

Note

The AIC values should not be directly compared, when models differ in response (for example log-log transformed and fitted with no transformation).

Author(s)

Lukasz Komsta

See Also

[lmcal](#), [nlscal](#)

Examples

```
data(ibuprofen)
attach(ibuprofen)
fit = lmcal(conc, area)
AIC(fit)
fit = nlscal(conc, area)
AIC(fit)
```

```
anova.lmcal, anova.nlscal
      ANOVA tests for calibration models
```

Description

This function performs ANOVA tests between fitted calibration models.

Usage

```
anova.lmcal(object, ...)
anova.nlscal(object, ...)
```

Arguments

object	an object of class 'lmcal' or 'nlscal'
...	additional arguments (ignored)

Details

For 'lmcal' models this function performs four tests: two ANOVAs between linear and quadratic model, without and with log-log transform (first is mentioned in literature as Mandel's fitting test), and two ANOVAs comparing four polynomial models, weighted and unweighted.

For 'nlscal' models two ANOVAs are performed - between asymptotic models without and with intercept term, and between three and four parameter logistic models.

Value

A list of 'anova' objects:

mandel	The test between linear and quadratic fit
logmandel	The same test, between two log-log models
table	The test between linear, quadratic, cubical and 4th order models without weighting
wtable	The test between linear, quadratic, cubical and 4th order models with weighting

Note

The result is printed in readable form, and returned via `invisible()`.

Author(s)

Lukasz Komsta

See Also[lmcal](#), [nlscal](#)**Examples**

```
data(nitrate)
attach(nitrate)
fit = lmcal(conc, area)
anova(fit)
fit2 = nlscal(conc, area)
anova(fit2)
```

`confint.cal`*Confidence intervals for calibration curve parameters*

Description

Computes confidence intervals for parameters in a calibration fitted model.

Usage

```
confint.cal(object, parm, level = 0.95, sort.models = FALSE, ...)
```

Arguments

<code>object</code>	an object of class 'lmcal' or 'nlscal' (inheriting from 'cal')
<code>parm</code>	ignored in this case
<code>level</code>	confidence level of intervals
<code>sort.models</code>	should the intervals be sorted by models, or variable names
<code>...</code>	additional arguments (ignored)

Value

A matrix containing upper and lower bounds for confidence interval is returned.

Author(s)

Lukasz Komsta

See Also[lmcal](#), [nlscal](#)

Examples

```
data(nitrate)
attach(nitrate)
fit = lmcal(conc, area)
confint(fit)
```

derivative	<i>Derivative of fitted polynomial</i>
------------	--

Description

Calculate derivative of polynomial for given x.

Usage

```
derivative(obj, x)
```

Arguments

obj	an object of class 'lm', fitted in $y \sim x + I(x^2) + I(x^3) + \dots$ way.
x	a vector of x values

Details

This function is called via the other high level functions, but it can be also called directly.

Value

A vector of calculated derivatives.

Author(s)

Lukasz Komsta

See Also

[lm](#)

Examples

```
x = 1:10
y = jitter(x+x^2)
fit = lm(y~x+I(x^2))
derivative(fit,1:10)
```

dstat

*Descriptive statistics of quantitative analysis results***Description**

Performs comprehensive statistical evaluation of quantitative analysis results.

Usage

```
dstat(x, expected = median(unlist(x)), sort = TRUE, inverse.f = TRUE,
      na.rm = FALSE, conf.level = 0.95, alternative = c("two.sided", "less", "greater"))
```

Arguments

<code>x</code>	a vector of results, of a dataframe with results to compare
<code>expected</code>	expected reference value
<code>sort</code>	if TRUE, the matrices are sorted by means, variances or p-values.
<code>inverse.f</code>	if F value in variance comparison is below 1, the inverse is taken (has no effect on p-value, but there are sometimes need to have such F)
<code>na.rm</code>	logical: should NA values be removed?
<code>conf.level</code>	level for calculate confidence intervals
<code>alternative</code>	alternative for all tests performed.
<code>ansari</code>	due to reports of errors on some datasets, the <code>ansari.test()</code> on data is turned off by default since 0.12. you can turn it on by setting this variable to TRUE

Details

If argument is vector, several one-row matrices are produced (see below). If argument is a `data.frame`, there are also additional matrices with pairwise comparisons. The comparison of all groups by appropriate test are also calculated. This function prints its results with significance stars and returns a list invisibly.

Value

A list containing following matrices (if data is a vector, only 5 first are returned):

<code>mean</code>	mean, its confidence interval and t-test for expected value
<code>median</code>	median, its confidence interval and Wilcoxon test for expected value
<code>var</code>	variance, standard deviation, standard error and Dixon test for outlier
<code>rsd</code>	relative standard deviation, its confidence interval and Grubbs test for outlier
<code>range</code>	minimum and maximum value, range, IQR, MAD and Shapiro-Wilk test for normality
<code>vartest</code>	ratios of variances, their confidence intervals and F test with p-value

<code>ttest</code>	differences between means, their confidence intervals and t test with p-value
<code>atest</code>	nonparametric differences in scale, their confidence intervals and Ansari-Bradley test with p-value
<code>atest</code>	nonparametric differences in location, their confidence intervals and Wilcoxon test with p-value
<code>anova</code>	ANOVA between all data
<code>kruskal</code>	Kruskal-Wallis test (nonparametric equivalent for ANOVA)
<code>bartlett</code>	Bartlett test for homogeneity of all variances
<code>fligner</code>	Fligner-Killeen test for equal variances (nonparametric alternative to Bartlett)

Note

This function calculates always **both** parametric and nonparametric tests, and choosing a test to take into account should be also decision of analyst, based on the other tests results.

Author(s)

Lukasz Komsta

See Also

[vstat](#)

Examples

```
set.seed(1234)
a = data.frame(x=rnorm(10), y=runif(10), z=rt(10, 1))
dstat(a, 0)
```

ibuprofen, genisten, biochanin, pseudoephedrine, nitrate
Calibration data for several compounds

Description

A sample calibration data with two replicates for each standard.

Usage

```
data(ibuprofen)
data(genisten)
data(biochanin)
data(pseudoephedrine)
data(nitrate)
```

Format

A data frames with 14 observations containing calibration curve of 5 compounds.

conc concentration of a standard

area peak area (response)

Source

The data was originally published by Kirkup and Mulholland (2004) to discuss various calibration models.

References

Kirkup, L., Mulholland, M. (2004). Comparison of linear and non-linear equations in univariate calibration. *J. Chromatogr. A*, 1029, 1-11.

Examples

```
data(ibuprofen)
attach(ibuprofen)
fit = lmcal(conc, area)
summary(fit)
anova(fit)
par(mfrow=c(2,2))
plot(fit)
```

lmcal, nlscal

Perform linear and nonlinear calibration of analytical method

Description

Fit given calibration data to several regression models.

Usage

```
lmcal(x, y, confint = 0.95, gridratio = 0.05)
nlscal(x, y, confint = 0.95, gridratio = 0.05)
```

Arguments

x	a vector of standard concentrations
y	a vector of corresponding responses (peak areas etc.)
confint	confidence interval for graphing prediction intervals
gridratio	a part of x variable range, to extend plot range (default 5 percent)

Details

For linear 'lmcal' fitting, procedure is performed as follows. First, the calibration data are fitted to OLS linear, quadratic, cubical, and 4th order polynomial. These models are called p1 - p4. Next, linear model is reweighted using x and y raised to power gamma from range (-4,4) with 0.1 accuracy. The optimal weights are detected by minimal mean relative error (MRE) according to Almeida et al. (2002). The best weighting scheme is then chosen, and data are fit to the same equations (called P1-P4, with uppercase).

Next, the optimal value of lambda for Box-Cox transform is estimated with accuracy up to 0.001, for transformation of x and y. The transformed models are then fitted (called bx and by).

Then, two next log-log transformed models, are fitted - linear called l1, and quadratic (mentioned sometimes as Wagner transform), called l2.

Last, the same models as p1 - p4 and P1 - P4, are fitted using `r1m` robust method, and called r1 - r4 and R1 - R4.

This function performs also computation of grid and corresponding predicted values for easy graphing of fitted models.

For nonlinear 'nlscal' fitting, procedure is performed as above, but there are following models fitted: asymptotic (a1), asymptotic through origin (a2), logistic (g1), four parameter logistic (g2), Michaelis-Menten (m1) and nonparametric (loess) spline (s1). There are no weighting nor transform when fitting by 'nlscal'.

Value

Returns object of class `c("lmcal", "cal")` or `c("nlscal", "cal")`, which is the list of following components:

<code>models</code>	A list of fitted models (p1-p4,P1-P4,l1,l2,bx,by,r1-r4,R1-R4)
<code>graph</code>	A list used by <code>plot()</code> method to produce graphs. Stored permanently to make custom graphing easier. Containing following elements: <code>grid</code> - a grid of x values, <code>fitted</code> - corresponding fitted values, <code>upperc,lowerc</code> - upper and lower bound for <code>interval="confidence"</code> prediction, <code>upperp,lowerp</code> - upper and lower bound for <code>interval="prediction"</code> prediction
<code>x</code>	Concentration vector
<code>y</code>	Response vector
<code>weigh</code>	A dataframe containing sequence of gamma values and corresponding mean relative errors, estimated during weighting process
<code>wx</code>	Value of gamma for oprimal weighting on x
<code>wy</code>	Value of gamma for oprimal weighting on y
<code>yw</code>	Logical, if weighting on y gives better result than on x
<code>px</code>	Optimal Box-Cox power for transform of x variable
<code>py</code>	Optimal Box-Cox power for transform of y variable

Note

This function performs **no** decision which model should be chosen! Such decision should be always made by analyst.

Author(s)

Lukasz Komsta

References

Almeida, A.M., Castel-Branco, M.M., Falcao, A.C. (2002) Linear regression for calibration lines revisited: weighting schemes for bioanalytical methods. *J. Chromatogr. B Biomed. Sci. Appl.* 774, 215-222.

Nagaraja, N.V., Paliwal, J.K., Gupta, R.C. (1999) Choosing the calibration model in assay validation. *J. Pharm. Biomed. Anal.* 20, 433-438.

Kimanani, E.K., Lavigne, J. (1998) Bioanalytical calibration curves: variability of optimal powers between and within analytical methods. *J. Pharm. Biomed. Anal.* 16, 1107-1115.

Kirkup, L., Mulholland, M. (2004). Comparison of linear and non-linear equations in univariate calibration. *J. Chromatogr. A*, 1029, 1-11.

Kimanani, E.K. (1998) Bioanalytical calibration curves: proposal for statistical criteria. *J. Pharm. Biomed. Anal.* 16, 1117-1124.

Baumann, K., Waetzig, H. (1997) Regression and calibration for analytical separation techniques. Part I. Design considerations. *Process Control and Quality*, 10, 59-73.

Baumann, K. (1997) Regression and calibration for analytical separation techniques. Part II. Validation, weighted and robust regression. *Process Control and Quality*, 10, 75-112.

Coleman, D.E., Vanatta, L.E. (1999) Lack-of-fit testing of ion chromatographic calibration curves with inexact replicates. *J. Chromatogr. A* 850, 43-51.

See Also

[lm](#), [rlm](#), [boxcox](#)

Examples

```
data(ibuprofen)
attach(ibuprofen)
fit = lmcal(conc, area)
fit
summary(fit)
```

lof

Lack-of-Fit testing of calibration models

Description

Performs 'a priori' ANOVA "Lack-of-Fit" tests on fitted calibration models.

Usage

```
lof(obj)
```

Arguments

`obj` An object inheriting from 'cal' (fitted by 'lmcal' or 'nlscal')

Details

This function performs lack-of-fit test on regression residuals. This test assumes, that overall residual error should not be significantly larger than error within groups with the same x (replicates). It is called by 'summary' methods, but also can be called directly by user.

Value

A matrix containing sum of squared residuals, sum of pure error, F-statistic and corresponding p-value.

Note

This test is possible to perform only with minimum 2 replicates of each x value.

Author(s)

Lukasz Komsta

References

see [lmcal](#)

See Also

[lmcal](#), [nlscal](#)

Examples

```
x = rep(1:10,10)
y = jitter(x)
fit = lmcal(x,y)
lof(fit)
```

`plot.lmcal, plot.nlscal`

Calibration plots

Description

Plot various plots commonly used to illustrate callibration.

Usage

```
plot.lmcal(x, type = c("curve", "residuals", "chronologic", "qqplot", "cook",
  "optimization"), trend = TRUE, confidence = TRUE, prediction = TRUE,
  lines = c(1, 2, 3, 4), colors = c(1, 1, 1, 1, 1), xlab = NULL, ylab = NULL, ...)
plot.nlscal(x, type = c("curve", "residuals", "chronologic", "qqplot"),
  trend = TRUE, lines = c(1, 2, 3, 4), colors = c(1, 1, 1, 1, 1), xlab = NULL,
  ylab = NULL, ...)
```

Arguments

<code>x</code>	An object of 'lmcal' or 'nlscal' class, respectively
<code>type</code>	Type of plots: <code>curve</code> - calibration plots, <code>residuals</code> - residuals vs. <code>x</code> values (common residual plot), <code>chronologic</code> - residuals in chronologic order (for autocorrelation detection), <code>qqplot</code> - quantile-quantile plots of residuals, <code>cook</code> - cook's distances, <code>optimization</code> - plots of weighting and Box-Cox optimization
<code>trend</code>	Logical: should be the trend of residuals marked (obtained by 'loess')
<code>confidence</code>	Logical: if the confidence interval should be plotted
<code>prediction</code>	Logical: if the prediction interval should be plotted
<code>lines</code>	Line types of the plot
<code>colors</code>	Colors of the plot
<code>xlab</code>	Label of <code>x</code>
<code>ylab</code>	Label of <code>y</code>
<code>...</code>	Additional arguments, currently ignored

Author(s)

Lukasz Komsta

Examples

```
x = rep(1:10, 5)
y = jitter(sqrt(x))
fit = lmcal(x, y)
par(mfrow=c(2, 2))
plot(fit, type="curve")
plot(fit, type="residuals")
plot(fit, type="optimization")
par(mfrow=c(1, 1))
# Low level plotting, useful for customizing plot!
plot(x, y)
lines(fit$graph$grid, fit$graph$fitted$p1, lty=2)
lines(fit$graph$grid, fit$graph$fitted$p2, lty=2)
```

```
predict.lmcal, predict.nlscal
```

Inverse predict concentration from given responses

Description

Inverse predict concentration from responses, using all fitted calibration models.

Usage

```
predict.lmcal(object, dataset, conf.int = 0.95, ...)  
predict.nlscal(object, dataset, ...)
```

Arguments

object	an object of class 'lmcal' or 'nlscal', respectively
dataset	a vector of responses
conf.int	confidence intercal (only for lmcal)
...	additional arguments, currently ignored

Details

For linear models, the concentrations are calculated by `inverse.predict()`, which calls `polyroot()` on modified polynomial coefficients. For nonlinear models, concentrations are calculated with appropriate 'inverse' formulas.

Value

A list containing following elements. Each element is a list of concentration vectors, calculated from a model, with name referring to the model.

fitted	Concentrations calculated by fitted model
upper	Upper limit of confidence interval of inverse prediction
lower	Lower limit of confidence interval of inverse prediction

Note

The confidence interval for prediction is calculated by taking standard error of prediction and dividing it by slope of calibration curve (estimated by `derivative`) Then, proper confidence interval is constructed using t statistic.

Author(s)

Lukasz Komsta

See Also[lmcal](#), [nlscal](#)**Examples**

```
set.seed(1234)
x=rep(1:10,10)
y=jitter(sqrt(x))
fit=lmcal(x,y)
predict(fit,rnorm(10,mean=2,sd=0.1))
```

`residuals.cal`*Residuals of calibration curves*

Description

Extract residuals of all fitted calibration models.

Usage

```
residuals.cal(object, ...)
```

Arguments

<code>object</code>	an object inheriting from 'cal' ('lmcal' or 'nlscal')
<code>...</code>	additional arguments, currently ignored

Value

A data frame, containing residuals of all models with their respective names

Author(s)

Lukasz Komsta

See Also[lmcal](#), [nlscal](#)**Examples**

```
set.seed(1234)
x=rep(1:8,5)
y=jitter(sqrt(x))
fit=lmcal(x,y)
residuals(fit)
boxplot(residuals(fit))
```

```
summary.lmcal, summary.nlscal
```

Summarizing fitted calibration curves

Description

A 'summary' class for 'lmcal' and 'nlscal' objects.

Usage

```
summary.lmcal(object, sort.models = FALSE, ...)
summary.nlscal(object, sort.models = FALSE, ...)
```

Arguments

object	an object of class 'lmcal' or 'nlscal'
sort.models	should the tables be sorted by models (TRUE) or variables (FALSE).
...	additional arguments, currently ignored.

Details

The function performs summarizing of fitted calibration models and produces several tables (see below). The are printed in appropriate form, and their list is returned invisibly.

Value

A list, consisting of following items:

coefficients	Estimated coefficients, their standard error, significance (t) and p-value
residuals	Quantiles of residuals and Shapiro-Wilk test of their normality
variances	Quantiles of variances (without transform, with log-log, and with Box-Cox on y) and Bartlett test for their heteroscedascity. Calculated only, if there are at least 2 replicates for each x
fit	R-squared, adjusted R-squared, AIC, residual standard error, sum of squared residuals, sum of pure error and Lack-of-Fit ANOVA test
sensitivity	sensitivity, limit of detection and quantitation, autocorrelation of residuals, Durbin-Watson test for autocorrelation

Note

The p-value of Durbin-Watson statistic is **only** approximated using normal transform algorithm! This is not critical criterion and **always** residual plot should be visually examined.

Some of values given above are not computed for 'nlscal' models.

Author(s)

Lukasz Komsta, with portion by Achim Zeileis (from `dwtest()`)

See Also

[lmcal](#), [nlscal](#)

Examples

```
set.seed(1234)
x=rep(1:8,5)
y=jitter(sqrt(x))
fit=lmcal(x,y)
fit
summary(fit)
```

tablets

Tablet mass data

Description

The mass of 30 headache tablets

Usage

```
data(tablets)
```

Source

(Data taken from own measurements)

Examples

```
data(tablets)
dstat(tablets,0.5)
```

Description

Estimate error propagation of results (inter- and intra-day, inter- and intra-batch etc.)

Usage

```
vstat(x, ...)
```

Arguments

`x` a data frame of results. Each column contain one serie of results
`...` optionally, more data frames, if additional factor is taking into account

Details

This function performs one-way or two-way ANOVA on given results and prints summary in the usable way. If we are checking variability only between and within one kind of series (days, batches), we construct a data frame with a result sets in its columns and call this function on such data.frame. In this case, we get variability analysis of between-series and within-series. If we want to check variability additionally for example between weeks, we group results in data frames for each week and call the function on all data frames. This will summarize error propagation concerning series and groups.

Value

The ANOVA object is returned, containing additional columns:

Percent	Percentage indication of error propagation (total sum of squares is 100 percent)
SD	Standard deviation (square root of Mean Sq)
RSD	Relative standard deviation (coefficient of variation)

There are also additional rows - lowest row gives overall (total) error, and if function is called on multiple data-frames, a row containing variability between all series (sum of within-groups and between-groups) is given.

Author(s)

Lukasz Komsta

See Also

[dstat](#)

Examples

```
set.seed(1234)
week1 = data.frame(mon=rnorm(6), tue=rnorm(6), wed=rnorm(6), thu=rnorm(6), fri=rnorm(6))
week2 = data.frame(mon=rnorm(6), tue=rnorm(6), wed=rnorm(6), thu=rnorm(6), fri=rnorm(6))
week3 = data.frame(mon=rnorm(6), tue=rnorm(6), wed=rnorm(6), thu=rnorm(6), fri=rnorm(6))
week4 = data.frame(mon=rnorm(6), tue=rnorm(6), wed=rnorm(6), thu=rnorm(6), fri=rnorm(6))
week1=week1+15;week2=week2+15;week3=week3+15;week4=week4+15
vstat(week1)
vstat(week1, week2, week3, week4)
```

Index

*Topic **datasets**

ibuprofen, genisten,
 biochanin,
 pseudoephedrine, nitrate,
 7
tablets, 16

*Topic **hplot**

plot.lmcal, plot.nlscal, 11

*Topic **htest**

dstat, 6

*Topic **models**

AIC.cal, 2
anova.lmcal, anova.nlscal, 3
confint.cal, 4
derivative, 5
lmcal, nlscal, 8
lof, 10
predict.lmcal,
 predict.nlscal, 13
residuals.cal, 14
summary.lmcal,
 summary.nlscal, 15
vstat, 17

AIC, 2

AIC.cal, 2

anova.lmcal (anova.lmcal,
 anova.nlscal), 3

anova.lmcal, anova.nlscal, 3

anova.nlscal (anova.lmcal,
 anova.nlscal), 3

biochanin (ibuprofen, genisten,
 biochanin,
 pseudoephedrine,
 nitrate), 7

boxcox, 10

confint.cal, 4

derivative, 5

dstat, 6, 17

genisten (ibuprofen, genisten,
 biochanin,
 pseudoephedrine,
 nitrate), 7

ibuprofen (ibuprofen, genisten,
 biochanin,
 pseudoephedrine,
 nitrate), 7

ibuprofen, genisten, biochanin,
 pseudoephedrine, nitrate,
 7

lm, 5, 10

lmcal, 2–4, 11, 14, 16

lmcal (lmcal, nlscal), 8

lmcal, nlscal, 8

lof, 10

nitrate (ibuprofen, genisten,
 biochanin,
 pseudoephedrine,
 nitrate), 7

nlscal, 2–4, 11, 14, 16

nlscal (lmcal, nlscal), 8

plot.lmcal (plot.lmcal,
 plot.nlscal), 11

plot.lmcal, plot.nlscal, 11

plot.nlscal (plot.lmcal,
 plot.nlscal), 11

predict.lmcal (predict.lmcal,
 predict.nlscal), 13

predict.lmcal, predict.nlscal, 13

predict.nlscal (predict.lmcal,
 predict.nlscal), 13

print.lmcal (lmcal, nlscal), 8

print.nlscal (lmcal, nlscal), 8

`pseudoephedrine` (`ibuprofen`,
`genisten`, `biochanin`,
`pseudoephedrine`,
`nitrate`), 7

`residuals.cal`, 14

`rlm`, 10

`summary.lmcal` (`summary.lmcal`,
`summary.nlscal`), 15

`summary.lmcal`, `summary.nlscal`, 15

`summary.nlscal` (`summary.lmcal`,
`summary.nlscal`), 15

`tablets`, 16

`vstat`, 7, 17