Package 'smatr'

May 9, 2026

Version 3.4-8

Title (Standardised) Major Axis Estimation and Testing Routines

Author David Warton <David.Warton@unsw.edu.au>, Remko Duursma, Daniel Falster and Sara Taskinen.

Maintainer Remko Duursma <remkoduursma@gmail.com>

Description Methods for fitting bivariate lines in allometry using the major axis (MA) or standardised major axis (SMA), and for making inferences about such lines. The available methods of inference include confidence intervals and one-sample tests for slope and elevation, testing for a common slope or elevation amongst several allometric lines, constructing a confidence interval for a common slope or elevation, and testing for no shift along a common axis, amongst several samples. See Warton et al. 2012 <doi:10.1111/j.2041-210X.2011.00153.x> for methods description.

License GPL-2

URL http://web.maths.unsw.edu.au/~dwarton, http://www.bitbucket.org/remkoduursma/smatr

Collate 'alpha.fun.R' 'b.com.est.R' 'coef.sma.R' 'com.ci.R' 'defineAxis.R' 'elev.com.R' 'elev.test.R' 'fitted.sma.R' 'huber.M.R' 'line.cis.R' 'logLik_sma.R' 'lr.b.com.R' 'ma.R' 'makeLogMinor.R' 'meas.est.R' 'multcompmatrix.R' 'nicePlot.R' 'plot.sma.R' 'predict.sma.R' 'print.sma.R' 'residuals.sma.R' 'robust.factor.R' 'seqLog.R' 'shift.com.R' 'slope.com.R' 'slope.test.R' 'sma.R' 'summary.sma.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2018-03-18 21:15:22 UTC

Contents

`smatr-package` *(Standardised) Major Axis Estimation and Testing Routines*

Description

This package provides methods of fitting bivariate lines in allometry using the major axis (MA) or standardised major axis (SMA), and for making inferences about such lines. The available methods of inference include confidence intervals and one-sample tests for slope and elevation, testing for a common slope or elevation amongst several allometric lines, constructing a confidence interval for a common slope or elevation, and testing for no shift along a common axis, amongst several samples.

Details

The key functions in this package are `sma` and `ma`, which will fit SMA and MA respectively, and construct confidence intervals or test hypotheses about slope or elevation in one or several samples, depending on how the arguments are used.

For example:

`sma(y~x)` will fit a SMA for `y` vs `x`, and report confidence intervals for the slope and elevation.

`sma(y~x, robust=T)` will fit a robust SMA for `y` vs `x` using Huber's M estimation, and will report (approximate) confidence intervals for the slope and elevation.

`ma(y~x*groups-1)` will fit MA lines for `y` vs `x` that are forced through the origin, where a separate MA is fitted to each of several samples as specified by the argument `groups`. It will also report results from a test of the hypothesis that the true MA slope is equal across all samples.

For more details, see the help listings for `sma` and `ma`.

Note that the sma and ma functions replace the functions given in earlier package versions as line.cis, slope.test, elev.test, slope.com, elev.com and shift.com, although all of these functions and their help entries are still available.

All procedures have the option of correcting for measurement error, although only in an approximate fashion, valid in large samples.

Additional features of this package are listed below.

meas.est Estimates the average variance matrix of measurement error for a set of subjects with repeated measures

Example datasets:

leaflife leaf longevity and leaf mass per area for plant species from different sites. Used to demonstrate the functionality of the sma and ma functions.

leafmeas leaf mass per area and photosynthetic rate for plant species from different sites. Used to demonstrate the meas.est function

For more details, see the documentation for any of the individual functions listed above.

Author(s)

Warton, D. <David.Warton@unsw.edu.au>, Duursma, R., Falster, D. and Taskinen, S.

References

Warton D. I. and Weber N. C. (2002) Common slope tests for bivariate structural relationships. *Biometrical Journal* **44**, 161–174.

Warton D. I., Wright I. J., Falster D. S. and Westoby M. (2006) A review of bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

Taskinen S. and Warton D. I. (in press) Robust estimation and inference for bivariate line-fitting in allometry. *Biometrical Journal*.

See Also

sma,ma, meas.est, leaflife, leafmeas

Examples

```
# See  ?sma and ?plot.sma for a full list of examples.
```

`coef.sma` *Extract coefficients from a 'sma' or 'ma' fit*

Description

Extracts elevation and slope of a standardized major axis (sma) or major axis (ma) fit, for each of the groups if the fit was by group.

Usage

```
## S3 method for class 'sma'
coef(object, ...)
```

Arguments

`object`	Object of class 'sma'.
`...`	Further arguments ignored.

Value

A dataframe with the slope(s) and elevation(s), and their confidence intervals. If the fit was by multiple groups, fits by all groups are returned.

Author(s)

R.A. Duursma

See Also

`sma`

`elev.com` *Test for equal elevation among several lines fitted with (standardised) major axes of common slope.*

Description

Test if several major axis or standardised major axis lines share a common elevation. This can now be done via `sma(y~x+groups)`, see help on the `sma` function.

Usage

```
elev.com(y, x, groups, data = NULL, method = 'SMA', alpha = 0.05, robust=FALSE,
  V = array(0, c(2, 2, length(unique(groups)))),
  group.names = sort(unique(groups)))
```

Arguments

y	The Y-variable for all observations (as a vector).
x	The X-variable for all observations (as a vector).
groups	Coding variable identifying which group each observation belongs to (as a factor or vector).
data	Deprecated. Use with() instead (see Examples).
method	The line fitting method: **'SMA' or 1** standardised major axis (this is the default) **'MA' or 2** major axis
alpha	The desired confidence level for the 100(1-alpha)% confidence interval for the common elevation. (Default value is 0.05, which returns a 95% confidence interval.)
robust	If TRUE, uses a robust method to fit the lines and construct the test statistic.
V	The estimated variance matrices of measurement error, for each group. This is a 3-dimensional array with measurement error in Y in the first row and column, error in X in the second row and column,and groups running along the third dimension. Default is that there is no measurement error.
group.names	(optional: rarely required). A vector containing the labels for 'groups'. (Only actually useful for reducing computation time in simulation work).

Details

Calculates a Wald statistic to test for equal elevation of several MA's or SMA's with a common slope. This is done by testing for equal mean residual scores across groups.

Note that this test is only valid if it is reasonable to assume that the axes for the different groups all have the same slope.

The test assumes the following:

- each group of observations was independently sampled
- the axes fitted to all groups have a common slope
- y and x are linearly related within each group
- residual scores independently follow a normal distribution with equal variance at all points along the line, within each group

Note that we do not need to assume equal variance across groups, unlike in tests comparing several linear regression lines.

The assumptions can be visually checked by plotting residual scores against fitted axis scores, and by constructing a Q-Q plot of residuals against a normal distribution, available using the `plot.sma` function. On a residual plot, if there is a distinct increasing or decreasing trend within any of the groups, this suggests that all groups do not share a common slope.

Setting `robust=TRUE` fits lines using Huber's M estimation, and modifies the test statistic as proposed in Taskinen & Warton (in review).

The common slope ($\hat{\beta}$) is estimated from a maximum of 100 iterations, convergence is reached when the change in $\hat{\beta} < 10^{-6}$.

Value

stat	The Wald statistic testing for no shift along the common axis
p	The P-value of the test. This is calculated assuming that stat has a chi-square distribution with (g-1) df, if there are g groups
a	The estimated common elevation
ci	A 100(1-alpha)% confidence interval for the true common elevation
as	Separate elevation estimates for each group

Author(s)

Warton, D.I.<David.Warton@unsw.edu.au>, J. Ormerod, & S. Taskinen

References

Warton D. I., Wright I. J., Falster D. S. and Westoby M. (2006) A review of bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

Taskinen, S. and D.I. Warton. in review. Robust tests for one or more allometric lines.

See Also

sma, plot.sma, line.cis, slope.com, shift.com

Examples

```
# Load leaf longevity data
data(leaflife)

# Test for common SMA slope amongst species at low soil nutrient sites
# with different rainfall:
leaf.low.soilp <- subset(leaflife, soilp == 'low')
with(leaf.low.soilp, slope.com(log10(longev), log10(lma), rain))

# Now test for common elevation of the groups fitted with an axis
# of common slope, at low soil nutrient sites:
with(leaf.low.soilp, elev.com(log10(longev), log10(lma), rain))

# Or test for common elevation amongst the MA's of common slope,
# for low soil nutrient sites, and construct 99% a confidence interval
# for the common elevation:
with(leaf.low.soilp, elev.com(log10(longev), log10(lma), rain, method='MA',
   alpha=0.01))
```

`elev.test` *One-sample test of a (standardised) major axis elevation*

Description

Test if the elevation of a major axis or standardised major axis equals a specific value. This can now be done via `sma(y~x,elev.test=0)`, see help on the `sma` function.

Usage

```
elev.test(y, x, test.value = 0, data = NULL, alpha = 0.05,
    method = 'SMA', robust = FALSE, V = matrix(0,2,2) )
```

Arguments

`y`	The Y-variable
`x`	The X-variable
`test.value`	The hypothesised value of the elevation (default value is 0)
`data`	Deprecated. Use with() instead (see Examples).
`alpha`	The desired confidence level for the 100(1-alpha)% confidence interval for the common slope. (Default value is 0.05, which returns a 95% confidence interval.)
`method`	The line fitting method: **'OLS' or 0** linear regression **'SMA' or 1** standardised major axis (this is the default) **'MA' or 2** major axis
`robust`	If TRUE, uses a robust method to fit the lines and construct the test statistic.
`V`	The estimated variance matrix of measurement error. Default is that there is no measurement error.

Details

Tests if the line relating y to x has an elevation equal to test.value (which has a default value of 0). The line can be a linear regression line, major axis or standardised major axis (as selected using the input argument choice). The test is carried out usinga t-statistic, comparing the difference between estimated and hypothesised elevation to the standard error of elevation. As described in Warton et al (2006).

A confidence interval for the elevation is also returned, again using the t-distribution.

If measurement error is present, it can be corrected for through use of the input argument V, which makes adjustments to the estimated sample variances and covariances then proceeds with the same method of inference. Note, however, that this method is only approximate (see Warton et al 2006 for more details).

The test assumes the following:

1. y and x are linearly related

2. residuals independently follow a normal distribution with equal variance at all points along the line

The assumptions can be visually checked by plotting residual scores against fitted axis scores, and by constructing a Q-Q plot of residuals against a normal distribution, available using the `plot.sma` function.

Setting `robust=TRUE` fits lines using Huber's M estimation, and modifies the test statistic as proposed in Taskinen & Warton (in review).

Value

`t`	The test statistic (a t-statistic).
`p`	The P-value, taken from the t_{n-2}-distribution. This is an exact test if residuals are normally distributed.
`test.value`	The hypothesised value of the elevation.
`a`	The estimated elevation.
`ci`	A 100(1-alpha)% CI for the slope.

Author(s)

Warton, D.I.<David.Warton@unsw.edu.au>, J. Ormerod, & S. Taskinen

References

Warton D. I., Wright I. J., Falster D. S. and Westoby M. (2006) A review of bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

Taskinen, S. and D.I. Warton. in review. Robust tests for one or more allometric lines.

See Also

`sma`, `line.cis`, `slope.test`

Examples

```
#load the leaflife dataset:
data(leaflife)

#consider only the low rainfall sites:
leaf.low.rain <-  subset(leaflife, rain=="low")

#construct a plot
plot(log10(leaf.low.rain$lma), log10(leaf.low.rain$longev),
   xlab="leaf mass per area [log scale]", ylab="leaf longevity [log scale]")

#test if the SMA elevation is 0 for leaf longevity vs LMA
with(leaf.low.rain, elev.test(log10(lma), log10(longev)))

#test if the MA elevation is 2
with(leaf.low.rain,elev.test(log10(lma), log10(longev),
   test.value = 2, method = "MA"))
```

`fitted.sma` *Returns fitted values*

Description

Returns "fitted values" of a (standardized) major axis fit.

Usage

```
## S3 method for class 'sma'
fitted(object, type = "fitted", newdata=NULL, centered=TRUE, ...)
```

Arguments

`object`	Object of class `sma`.
`type`	Either 'residuals', or 'fitted'.
`newdata`	New data for which to provide fitted values.
`centered`	Logical. If TRUE (the default) returns the zero-centered fitted values.
`...`	Further arguments are currently ignored.

Details

"Fitted values" are calculated using `y+bx` for SMA or `by+x` for MA (see Warton et al. 2006, especially Table 4). Note these values are calculated differently to ordinary linear regression, and they cannot be interpreted as predicted values. The "fitted values" should be interpreted as measuring how far along the fitted axis each point lies, and are used in checking assumptions (via residual vs. fitted value plots).

Fitted values may be computed for new data, using the `newdata` arguments. Be aware that the names of the variables need to be the same as in the data used in the original model fit. Also, if the original fit used `log='xy'`, for example, this transformation will also be applied to your `newdata` (so don't do the transformation yourself).

References

Warton D. I., Wright I. J., Falster D. S. and Westoby M. (2006) A review of bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259-291.

See Also

`sma`,`residuals.sma`

leaflife — *Leaf longevity and leaf mass per area for plant species from different sites*

Description

This dataset contains the leaf longevity and leaf mass per area of 75 plant species from four sites in south-eastern Australia. Sites represent contrasts along rainfall and soil phosphorous gradients.

Usage

```
data(leaflife)
```

Format

A data frame containing 75 rows and 5 variables:

site the site at which this species was sampled (coded as 1, 2, 3, or 4)

rain The level of annual rainfall at the site (coded as "high" or "low")

soilp The level of soil phosphate concentration at the site (coded as "high" or "low")

longev Leaf longevity (years)

lma Leaf mass per area (m2/kg)

Source

Wright I. J., Westoby M. and Reich P. B. (2002) Convergence towards higher leaf mass per area in dry and nutrient-poor habitats has different consequences for leaf life span. *Journal of Ecology* **90**, 534–543.

leafmeas — *Leaf mass per area and photosynthetic rate for plant species from different sites*

Description

This dataset contains the leaf mass per area (LMA) and photosynthetic rate per meaf mass (Amass) of individual plants from 81 plant species at four sites in south-eastern Australia. Sites represent contrasts along rainfall and soil phosphorous gradients.

Usage

```
data(leafmeas)
```

Format

A data frame containing 529 rows and 4 variables:

site (factor) The site at which this species was sampled (coded as Murrua, WH, RHM, RHW)

spp (factor) Species of the observed plant species

Amass (numeric) Photosynthetic rate per leaf mass ()

lma (numeric) Leaf mass per area (m2/kg)

Source

Wright I. J., Reich P. B. and Westoby M. (2001) Strategy shifts in leaf physiology, structure and nutrient content between species of high- and low-rainfall and high- and low-nutrient habitats. *Functional Ecology* **15**, 423–434.

`line.cis` *Slope and elevation of a (standardised) major axis, with confidence intervals*

Description

Calculates the slope and elevation of a major axis or standardised major axis, and constructs confidence intervals for the true slope and elevation. This can now be fitted via calls of the form `sma(y~x, ...)`, see `sma`.

Usage

```
line.cis(y, x, alpha = 0.05, data = NULL,
method = "SMA", intercept = TRUE,
V = matrix(0, 2, 2), f.crit = 0, robust=FALSE, ...)
```

Arguments

`y`	The Y-variable
`x`	The X-variable
`alpha`	The desired confidence level for the 100(1-alpha)% confidence interval for the common slope. (Default value is 0.05, which returns a 95% confidence interval.)
`data`	Deprecated. Use with() instead (see Examples).
`method`	The line fitting method: **'OLS' or 0** linear regression **'SMA' or 1** standardised major axis (this is the default) **'MA' or 2** major axis
`V`	The estimated variance matrix of measurement error. Average measurement error for Y is in the first row and column, and average measurement error for X is in the second row and column. The default is that there is no measurement error.

intercept	(logical) Whether or not the line includes an intercept. **FALSE** no intercept, so the line is forced through the origin **TRUE** an intercept is fitted (this is the default)
f.crit	(optional - rarely required). The critical value to be used from the F distribution. (Only actually useful for reducing computation time in simulation work - otherwise, do not change.)
robust	If TRUE, uses a robust method to fit the lines and construct confidence intervals.
...	Further parameters (not passed anywhere at the moment).

Details

Fits a linear regression line, major axis, or standardised major axis, to a bivariate dataset. The slope and elevation are returned with confidence intervals, using any user-specified confidence level.

Confidence intervals are constructed by inverting the standard one-sample tests for elvation and slope (see slope.test and elev.test for more details). Only the primary confidence interval is returned - this is valid as long as it is known a priori that the (standardised) major axis is estimating the true slope rather than the (standardised) minor axis. For SMA, this means that the sign of the true slope needs to be known a priori, and the sample slope must have the same sign as the true slope.

The test assumes the following:

1. y and x are linearly related
2. residuals independently follow a normal distribution with equal variance at all points along the line

These assumptions can be visually checked by plotting residuals against fitted axis scores, and by constructing a Q-Q plot of residuals against a normal distribution. An appropriate residual variable is y-bx, and for fitted axis scores use x (for linear regression), y+bx (for SMA) or by+x (for MA), where b represents the estimated slope.

If measurement error is present, it can be corrected for through use of the input argument V, which makes adjustments to the estimated sample variances and covariances then proceeds with the same method of inference. Note, however, that this method is only approximate (see Warton et al in review for more details).

Setting robust=TRUE fits lines using Huber's M estimation, and modifies confidence interval formulae along the lines discussed in Taskinen & Warton (in review).

Value

coeff	A matrix containing the estimated elevation and slope (first column), and the lower and upper limits of confidence intervals for the true elevation and slope (second and third columns). Output for the elevation and slope are in the first and second rows, respectively.

Author(s)

Warton, D.I.<David.Warton@unsw.edu.au>, J. Ormerod, & S. Taskinen

References

Warton, D.I., I.J. Wright, D.S. Falster and M. Westoby. 2006. Bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

Taskinen, S. and D.I. Warton. in review. Robust tests for one or more allometric lines.

See Also

sma, slope.test, elev.test

Examples

```
#load the leaflife data
data(leaflife)

#consider only the low rainfall sites:
leaf.low.rain=leaflife[leaflife$rain=='low',]

#estimate the SMA line for reserve vs coat
with(leaf.low.rain, line.cis(log10(longev),log10(lma)))

#produce CI's for MA slope and elevation:
with(leaf.low.rain, line.cis(log10(longev),log10(lma),method='MA'))
```

---

makeLogMinor *Generate spacing for minor tick marks on a plot with log-scaled axes*

Description

Generates a sequence of numbers providing minor tick marks on log scaled axes.

Usage

```
makeLogMinor(major)
```

Arguments

major	a vector of values giving major tick marks.

Details

A vector is created according to the following algorithm. For each pair of adjacent values (x1, x2) in major, the function adds the values (x1, 2*x1, 3*x1, ..., x2) to the vector of return values.

This is useful for generating spacing of minor tick-values on log-transformed axes.

See Also

nicePlot, seqLog

Examples

```
#Sequence suitable for log base 10 labels
makeLogMinor(seqLog(1E-5, 1E5))
```

`meas.est` *Measurement error variance estimation from repeated measures*

Description

Estimates the average variance matrix of measurement error for a set of subjects with repeated measures.

Usage

```
meas.est(datameas, id, data=NULL )
```

Arguments

`datameas`	A data matrix containing the repeated measures of observations, with each variable in a different column and all observations on all subjects in different rows of the same column.
`id`	An id vector identifying the subject being measured for each observation in the data matrix.
`data`	Deprecated. Use with() instead (see Examples).

Details

This function allows the estimation of measurement error variance, given a set of repeated measures on different subjects. Measurement error variance is estimated separately for each subject, then averaged across subjects. This provides terms that can be used to correct for measurement error in allometric analyses.

Any number of variables can be specified in the data matrix for measurement error calculation. If more than one variable is specified, the covariance of measurement error is estimated from the repeated measures as well as the variance.

As well as the estimated measurement error variance, a data matrix is returned which contains the averages of the repeated measures for each subject.

Value

`V`	A matrix containing the average variances and average convariances of the repeated measures of subjects.
`dat.mean`	A matrix containing the values for each subject, averages across repeated measures. Subjects are in rows, variables in columns.

Author(s)

Warton, D. <David.Warton@unsw.edu.au>, translated to R by Ormerod, J. 2005-12-08

References

Warton, D.I., I.J. Wright, D.S. Falster and M. Westoby. 2006. Bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

See Also

`line.cis`, `slope.test`

Examples

```
#load the individual level leaf example dataset
data(leafmeas)

#Estimate measurement error variance matrix, store in "meas.vr"
meas.vr <- meas.est(leafmeas[,3:4], leafmeas$spp)
```

`multcompmatrix` *Multiple comparisons graphical matrix*

Description

Print a matrix of pair-wise comparisons based on an 'sma' fit.

Usage

```
multcompmatrix(smfit, sort = TRUE)
```

Arguments

`smfit`	An object of class 'sma', fit with `sma`.
`sort`	Logical. Specifies whether or not to sort groups from smallest to largest value based on the parameter of interest (slope, elevation or mean fitted value).

Details

A matrix of comparisons is drawn, based on the P values of the pair-wise tests between levels of the grouping variable. An 'X' indicates $P < 0.05$, 'o' indicates $0.05 < P < 0.1$.

Value

Invisibly returns the (character) matrix.

Author(s)

Remko Duursma and Daniel Falster.

See Also

`sma`

Examples

```
# Print the matrix of comparisons:
data(leaflife)
sm1 <- sma(lma ~ longev + site, data=leaflife, multcomp=TRUE)
multcompmatrix(sm1)

# Write the matrix to a file like this:
## Not run:
capture.output(multcompmatrix(sm1), file="sm1matrix.txt")

## End(Not run)
```

`plot.sma` *Draw an X-Y plot*

Description

Plot a (standardized) major axis fit, including the data and the fitted lines. There are many options for changing the appearance of the plot and generating high-quality publishable graphs.

Usage

```
## S3 method for class 'sma'
plot(x, which = c("default", "residual", "qq"),
use.null = FALSE, add = FALSE,  type = "o", xaxis = NULL,
yaxis = NULL, xlab = NULL, ylab = NULL, pch = NULL,
col = NULL, lty = NULL, from = NULL, to = NULL,
log = x$log, frame.plot = TRUE, tck=par("tck"),
p.lines.transparent = NA, axes = TRUE, ...)
```

Arguments

`x`	Object of class 'sma'.
`which`	If 'residual', plots a residual plot; if 'qq', plots a qq plot; otherwise an x-y plot.
`use.null`	Logical. If FALSE, plots the fitted lines (the default), otherwise those corresponding to the null hypothesis.
`add`	Logical. If TRUE, adds lines or points to an existing plot.
`type`	As in 'lm.default' : 'p' plots points only, 'l' only lines, and 'o' or 'b' plot both.
`xaxis, yaxis`	Special axis objects. See Details and examples.
`xlab, ylab`	Labels for X and Y axes.
`pch`	Plotting symbols (see `points`).
`col`	Color of points and lines.
`lty`	Line type (see `lines`).

from, to	Min and max X values for the lines (defaults to values given by sma, which are the X values corresponding the maximum and minimum fitted values in the data.).
log	One of 'x','y' or 'xy', to denote which axes to log-scale.
frame.plot	a logical indicating whether a box should be drawn around the plot, by default = TRUE.
tck	The length of tick marks as a fraction of the smaller of the width or height of the plotting region. If tck >= 0.5 it is interpreted as a fraction of the relevant side, so if tck = 1 grid lines are drawn. By default set to current system defaults (tck = par("tck")).
p.lines.transparent	Adjusts transparency level of fitted lines according to p-value of correlation between X and Y, via formula opacity = 1-p/p.lines.transparent). Setting to a value 0.1 means the line for any group with p = 0.1 would be fully transparent, while line for a group with p =0.05 would be 50 perecent transparent. by default set to NA, which means lines are fully visible.
axes	If FALSE, suppress plotting of the axes (Default TRUE)
...	Further arguments passed to plot.default.

Details

The plot.sma function produces one of three different types of plot, depending on the which argument.

The default plot, which="default", produced a plot of y vs x, with separate symbols for each group if appropriate, and MA or SMA lines fitted through each group. The formula used in the sma object that is input to the plot function determines whether there is a group structure, whether fitted lines have common slope, etc.

A residual plot can be obtained via which="residual" - this is a plot of residuals against fitted values. This can be used to check assumptions - if assumptions are satisfied there should be no pattern.

A normal quantile plot can be obtained via which="qq" - this is a normal quantile plot of residuals. This can be used to check the normality assumption - if data are close to a straight line, normality is plausible. Note that non-normality is only important to the validity of the test in small samples. In larger samples, non-normality will not effect validity, but strong non-normality will reduce the power of tests.

If use.null=TRUE then the lines added to the plot use the coefficients estimated under the null hypothesis. For example, if the sma object x was produced using a common slopes test (via y~x*groups or similar) then use.null=TRUE will plot lines that apply the common slope to each group.

The arguments pch, col, lty, from & to, are used to modify characteristics of the plotted points and lines. If a vector of values for anyone of these arguments is passed to plot.sma, then successive values are applied to each group, provided group structure is included in x and the vector length is at least as long as the number of groups.

By default, plot.sma uses the default tick spacing given by plot.default. To customise axes, users can pass special axis objects to plot.sma, obtained using the defineAxis command as in

the example below. This enables high quality publishable plots to be produced. See plotutils for more information.

Author(s)

D. Falster, R.A. Duursma, D.I. Warton

See Also

sma, plotutils, defineAxis

Examples

```
# Load leaf lifetime dataset:
data(leaflife)

# Only consider low-nutrient sites:
leaf.low.soilp <- subset(leaflife, soilp == 'low')

# Fit SMA's separately at each of high and low
# rainfall sites and test for common slope:
ft <- sma(longev~lma*rain, data=leaf.low.soilp, log="xy")

# Plot leaf longevity (longev) vs leaf mass per area (lma)
# separately for each of high and low rainfall:
plot(ft)

# As before but add lines which have a common slope:
plot(ft, use.null=TRUE)

#As above, but adding the common slope lines to an existing plot
plot(ft, type='p', col="black")
plot(ft, use.null=TRUE, add=TRUE, type='l')

# Plot with equally spaced tick marks:
plot(ft, xaxis=defineAxis(major.ticks=c(40,80,160,320,640)),
yaxis=defineAxis(major.ticks=c(0.5,1,2,4,8)) )

# Produce a residual plot to check assumptions:
plot(ft,which="res")

# Produce a normal quantile plot:
plot(ft,which="qq")
```

---

plotutils *Utility plotting functions*

Description

Functions used in conjunction with plot.sma to customize spacing of ticks on plot axes. defineAxis creates an 'axis' object, including tick spacing, limits, and labels, that can be passed into plot.sma or nicePlot. nicePlot creates an empty plot using x and y axis objects.

Usage

```
defineAxis(major.ticks, limits=NULL, minor.ticks = NULL,
major.tick.labels = major.ticks, both.sides = FALSE)
nicePlot(xaxis, yaxis, log = "", ann = par("ann"), xlab = NULL,
ylab = NULL,tck = par("tck"),frame.plot = TRUE, ...)
```

Arguments

limits	the x or y limits of the plot, (x1, x2) or (y1,y2).
major.ticks, minor.ticks	Where to draw major and minor ticks (vectors).
major.tick.labels	Labels to draw at the ticks (optional).
both.sides	a logical value indicating whether tickmarks should also be drawn on opposite sides of the plot, i.e. right or top
xaxis, yaxis	'axis' objects, the result of calling 'defineAxis'.
ann	a logical value indicating whether the default annotation (x and y axis labels) should appear on the plot.
xlab	a label for the x axis, defaults to a description of x.
ylab	a label for the y axis, defaults to a description of y.
log	One of 'x', 'y' or 'xy', specifying which axes draw in log10 scale.
frame.plot	a logical indicating whether a box should be drawn around the plot, by default = TRUE.
tck	The length of tick marks as a fraction of the smaller of the width or height of the plotting region. If tck >= 0.5 it is interpreted as a fraction of the relevant side, so if tck = 1 grid lines are drawn. By default set to current system defaults (tck = par("tck")).
...	Arguments to be passed to nicePlot, and therein to 'axis'.

Details

By default, calls to plot.sma use the values given by plot.default to determine axis limits and spacing of major and minor ticks. Users can override these values by passing two 'axis' objects created by defineAxis to plot.sma. When both x and y axis objects are passed to plot.sma, the function uses nicePlot to construct the axes according to the specified values, instead of plot.default. As a minimum, users must at least one argument (major.ticks) to defineAxis when passing these to plot.sma.

The function nicePlot can also be called to construct a new set of axes according to the specified values. For this to work, axis objects must contain both major.ticks and limits.

See Also

sma, plot.sma

Examples

```
# Load leaf lifetime dataset:
data(leaflife)

#First example of axis spacing
ft <- sma(longev~lma*rain,log="xy",data=leaflife)
xax <- defineAxis(major.ticks=c(50, 100, 200, 400))
yax <- defineAxis(major.ticks=c(0.25, 0.5, 1,2,4,8))
plot(ft, xaxis=xax, yaxis=yax)

#As above, marking axes on both sides
xax <- defineAxis(major.ticks=c(50, 100, 200, 400), both.sides=TRUE)
yax <- defineAxis(major.ticks=c(0.25, 0.5, 1,2,4,8), both.sides=TRUE)
plot(ft, xaxis=xax, yaxis=yax)

#Using labels with log10 spacing and minor tick marks
xax <- defineAxis(limits=c(10, 1E3), major.ticks=seqLog(1, 1000),
minor.ticks=makeLogMinor(seqLog(1, 1000)))
yax <- defineAxis(limits=c(1E-1, 1E1), major.ticks=seqLog(1E-2, 10),
minor.ticks=makeLogMinor(seqLog(1E-2, 10)))
plot(ft, xaxis=xax, yaxis=yax)

#As above, but using expressions as labels
xax <- defineAxis(limits=c(10, 1E3), major.ticks=seqLog(10, 1000),
minor.ticks=makeLogMinor(seqLog(10, 1000)),
major.tick.labels = parse(text=paste("10^", c( 1,2,3), sep="")),
both.sides=FALSE)
yax <- defineAxis(limits=c(1E-1, 1E1), major.ticks=seqLog(1E-1, 10),
minor.ticks=makeLogMinor(seqLog(1E-1, 10)),
major.tick.labels = parse(text=paste("10^", c( -1,0,1), sep="")),
both.sides=FALSE)
plot(ft, xaxis=xax, yaxis=yax)

#start an empty plot using nicePlot
xax <- defineAxis(limits=c(8, 1.2E3), major.ticks=seqLog(1, 1000))
yax <- defineAxis(limits=c(0.8E-1, 1.2E1), major.ticks=seqLog(1E-2, 10))
nicePlot(xax,yax,log='xy')
```

---

print.sma *Print an object of class 'sma'.*

Description

Prints the coefficients and the results of the various hypothesis tests of an sma object.

Usage

```
## S3 method for class 'sma'
print(x, ..., coefbygroup = FALSE)
```

Arguments

x	Object of class 'sma'.
coefbygroup	Whether or not to print the coefficients by group.
...	Further arguments ignored.

Author(s)

R.A. Duursma, D. Falster, D.I. Warton

See Also

sma

---

residuals.sma *Extract model residuals*

Description

Extracts the residuals of a (standardized) major axis fit.

Usage

```
## S3 method for class 'sma'
residuals(object, ...)
```

Arguments

object	Object of class 'sma'.
...	Further arguments ignored.

Details

Residuals are calculated as y-bx-a for each group. These values are useful in assumption checking, especially in constructing residual vs fitted value plots.

Value

A vector of residuals.

See Also

sma, plot.sma

seqLog *Sequence Generation*

Description

Generate multiplicative sequences, or series.

Usage

```
seqLog(from, to, base = 10)
```

Arguments

from, to	the starting and (maximal) end value of a sequence.
base	multiplication value.

Details

Starting at from, seq multiplies successively by base until the maximal value is reached. This is useful for generating tick-spacing on log-transformed axes.

See Also

nicePlot, makeLogMinor

Examples

```
#Sequence suitable for log base 10 labels
seqLog(1E-5, 1E5)

#Sequence suitable for log base 2 labels
seqLog(2, 128,base=2)
```

shift.com *Test for no mean shift along a common (standardised) major axis*

Description

Test if several groups of observations have no shift in location along major axis or standardised major axis lines with a common slope. This can now be done via sma(y~x+groups, type="shift"), see help on the sma function.

Usage

```
shift.com( y, x, groups, data = NULL, method = 'SMA',
   intercept = TRUE, robust = FALSE, V=array( 0, c( 2,2,length(unique(groups)))),
    group.names=sort(unique(groups)) )
```

Arguments

y	The Y-variable for all observations (as a vector)
x	The X-variable for all observations (as a vector)
groups	Coding variable identifying which group each observation belongs to (as a factor or vector)
data	Deprecated. Use with() instead (see Examples).
method	The line fitting method: **'SMA' or 1** standardised major axis (this is the default) **'MA' or 2** major axis
intercept	(logical) Whether or not the line includes an intercept. **FALSE** no intercept, so the line is forced through the origin **TRUE** an intercept is fitted (this is the default)
robust	If TRUE, uses a robust method to fit the lines and construct the test statistic.
V	The estimated variance matrices of measurement error, for each group. This is a 3-dimensional array with measurement error in Y in the first row and column, error in X in the second row and column, and groups running along the third dimension. Default is that there is no measurement error.
group.names	(optional: rarely required). A vector containing the labels for 'groups'. (Only actually useful for reducing computation time in simulation work).

Details

Calculates a Wald statistic to test for no shift along several MA's or SMA's of common slope. This is done by testing for equal fitted axis means across groups.

Note that this test is only valid if it is reasonable to assume that the axes for the different groups all have the same slope.

The test assumes the following:

1. each group of observations was independently sampled
2. the axes fitted to all groups have a common slope
3. y and x are linearly related within each group
4. fitted axis scores independently follow a normal distribution with equal variance at all points along the line, within each group

Note that we do not need to assume equal variance across groups, unlike in tests comparing several linear regression lines.

The assumptions can be visually checked by plotting residuals against fitted axis scores, and by constructing a Q-Q plot of residuals against a normal distribution, available using

`plot.sma(sma.object,which="residual")`.

On a residual plot, if there is a distinct increasing or decreasing trend within any of the groups, this suggests that all groups do not share a common slope.

A plot of residual scores against fitted axis scores can also be used as a visual test for no shift. If fitted axis scores systematically differ across groups then this is evidence of a shift along the common axis.

Setting `robust=TRUE` fits lines using Huber's M estimation, and modifies the test statistic as proposed in Taskinen & Warton (in review).

The common slope ($\hat{\beta}$) is estimated from a maximum of 100 iterations, convergence is reached when the change in $\hat{\beta}$ is $< 10^{-6}$.

Value

`stat`	The Wald statistic testing for no shift along the common axis
`p`	The P-value of the test. This is calculated assuming that stat has a chi-square distribution with (g-1) df, if there are g groups
`f.mean`	The fitted axis means for each group

Author(s)

Warton, D.I.<David.Warton@unsw.edu.au>, J. Ormerod, & S. Taskinen

References

Warton D. I., Wright I. J., Falster D. S. and Westoby M. (2006) A review of bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

Taskinen, S. and D.I. Warton. in review. Robust tests for one or more allometric lines.

See Also

`sma`, `plot.sma`, `line.cis`, `elev.com`, `shift.com`

Examples

```
#load leaf longevity data
data(leaflife)

#Test for common SMA slope amongst species at low rainfall sites
#with different levels of soil nutrients
leaf.low.rain=leaflife[leaflife$rain=='low',]
with(leaf.low.rain, slope.com(log10(longev), log10(lma), soilp))

#Now test for no shift along the axes of common slope, for sites
#with different soil nutrient levels but low rainfall:
with(leaf.low.rain, shift.com(log10(longev), log10(lma), soilp))

#Now test for no shift along the axes of common slope, for sites
#with different soil nutrient levels but low rainfall:
with(leaf.low.rain,shift.com(log10(longev), log10(lma), soilp, method='MA'))
```

slope.com *Common slope test amongst several allometric lines*

Description

Test if several major axis or standardised major axis lines share a common slope. This can now be done via `sma(y~x*groups)`, see help on the `sma` function.

Usage

```
slope.com(y, x, groups, method = 'SMA', alpha = 0.05,
   data = NULL, intercept = TRUE, robust=FALSE,
   V = array(0, c(2, 2, length(unique(groups)))),
   group.names = sort(unique(groups)),
   ci = TRUE, bs = TRUE, slope.test=NULL)
```

Arguments

`y`	The Y-variable for all observations (as a vector)
`x`	The X-variable for all observations (as a vector)
`groups`	Coding variable identifying which group each observation belongs to (as a factor or vector)
`method`	The line fitting method: **'SMA' or 1** standardised major axis (this is the default) **'MA' or 2** major axis **'lamest' or 3** Error variance ratio is estimated from the data
`alpha`	The desired confidence level for the 100(1-alpha)% confidence interval for the common slope. (Default value is 0.05, which returns a 95% confidence interval.)
`data`	Deprecated. Use with() instead (see Examples).
`intercept`	(logical) Whether or not the line includes an intercept. **FALSE** no intercept, so the line is forced through the origin **TRUE** an intercept is fitted (this is the default)
`robust`	If TRUE, uses a robust method to fit the lines and construct the test statistic.
`V`	The estimated variance matrices of measurement error, for each group. This is a 3-dimensional array with measurement error in Y in the first row and column, error in X in the second row and column, and groups running along the third dimension. Default is that there is no measurement error.
`group.names`	(optional: rarely required). A vector containing the labels for 'groups'. (Only actually useful for reducing computation time in simulation work).
`ci`	(logical) Whether or not to return a confidence interval for the common slope.
`bs`	(logical) Whether or not to return the slopes for the separate groups, with confidence intervals.
`slope.test`	If a value provided, tests the common slope fit against this value.

Details

For several bivariate groups of observations, this function tests if the line-of-best-fit has a common slope for all samples, when the line-of-best-fit is estimated using the major axis, standardised major axis, or a more general version of these methods in which the error variance ratio is estimated from the data.

The test assumes the following:

1. each group of observations was independently sampled
2. y and x are linearly related within each group
3. residuals independently follow a normal distribution with equal variance at all points along the line, within each group

Note that we do not need to assume equal variance across groups, unlike in the standard test for common slope for linear regression.

The assumptions can be visually checked by plotting residual scores against fitted axis scores, and by constructing a Q-Q plot of residuals against a normal distribution, available using the `plot.sma` function.

Setting `robust=TRUE` fits lines using Huber's M estimation, and modifies the test statistic as proposed in Taskinen & Warton (in review).

The common slope is estimated from a maximum of 100 iterations, convergence is reached when the change in b is $< 10^{-6}$.

Value

`lr`	The (Bartlett-corrected) likelihood ratio statistic testing for common slope
`p`	The P-value of the test. This is calculated assuming that lr has a chi-square distribution with (g-1) df, if there are g groups
`b`	The common slope estimate
`varb`	The sample variance of the common slope
`ci`	A 100(1-alpha)% confidence interval for the common slope
`lambda`	The error variance ratio - the ratio of error variance in y to error variance in x. For MA, this is assumed to be 1. for SMA, this is assumed to be b^2. For the 'lamest' method, the error variance ratio is estimated from the data under the common slope assumption.
`bs`	The slopes and confidence intervals for data from each group.

Author(s)

Warton, D.I.<David.Warton@unsw.edu.au>, J. Ormerod, & S. Taskinen

References

Warton D. I. and Weber N. C. (2002) Common slope tests for bivariate structural relationships. *Biometrical Journal* **44**, 161–174.

Warton D. I., Wright I. J., Falster D. S. and Westoby M. (2006) A review of bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

Taskinen, S. and D.I. Warton. in review. Robust tests for one or more allometric lines.

See Also

sma, line.cis, elev.com, shift.com

Examples

```
#load leaf longevity data
data(leaflife)

#plot the data, with different symbols for different groups.
plot(leaflife$lma, leaflife$longev, type='n', log='xy', xlab=
   'leaf mass per area [log scale]', ylab='leaf longevity [log scale]')
colours <- c('blue', 'red', 'green', 'yellow')
points(leaflife$lma, leaflife$longev,
   col=colours[as.numeric(leaflife$site)])
legend(55, 5, as.character(unique(leaflife$site)), col=colours,
   pch=rep(1,4))

#test for common SMA slope of log(leaf longevity) vs log(LMA),
#across species sampled at different sites:
fit <- with(leaflife, slope.com(log10(longev), log10(lma), site))
fit

#Residual vs fits plots for SMA fit of each site
y <- log10(leaflife$longev)
x <- log10(leaflife$lma)
site <- leaflife$site
par( mfrow=c(2,2) )
plot(y[site==1] + fit$bs[1,1] * x[site==1], y[site==1] - fit$bs[1,1]
   * x[site==1], xlab='fits (site 1)', ylab='residuals (site 1)')
plot(y[site==2] + fit$bs[1,2] * x[site==2], y[site==2] - fit$bs[1,2]
   * x[site==2], xlab='fits (site 2)', ylab='residuals (site 2)')
plot(y[site==3] + fit$bs[1,3] * x[site==3], y[site==3] - fit$bs[1,3]
   * x[site==3], xlab='fits (site 3)', ylab='residuals (site 3)')
plot(y[site==4] + fit$bs[1,4] * x[site==4], y[site==4] - fit$bs[1,4]
   * x[site==4], xlab='fits (site 4)', ylab='residuals (site 4)')

#Test for common SMA slope amongst species at low rainfall sites
#with different levels of soil nutrients
leaf.low.rain <- leaflife[leaflife$rain=='low',]
with(leaf.low.rain, slope.com(log10(longev), log10(lma), soilp))

#test for common MA slope:
with(leaflife, slope.com(log10(longev), log10(lma), site, method='MA'))

#test for common MA slope, and produce a 90% CI for the common slope:
with(leaflife, slope.com(log10(longev), log10(lma), site,  method='MA', alpha=0.1))
```

---

slope.test *One-sample test of a (standardised) major axis slope*

Description

Test if the slope of a major axis or standardised major axis equals a specific value. This can now be done via `sma(y~x, slope.test=1)`, see help on `sma`.

Usage

```
slope.test(y, x, test.value = 1, data=NULL, method = SMA,
    alpha = 0.05, V = matrix(0,2,2), intercept = TRUE, robust=FALSE )
```

Arguments

`y`	The Y-variable
`x`	The X-variable
`test.value`	The hypothesised value of the slope (default value is 1)
`data`	Deprecated. Use with() instead (see Examples).
`method`	The line fitting method: **'OLS' or 0** linear regression **'SMA' or 1** standardised major axis (this is the default) **'MA' or 2** major axis
`alpha`	The desired confidence level for the 100(1-alpha)% confidence interval for the common slope. (Default value is 0.05, which returns a 95% confidence interval.)
`robust`	If TRUE, uses a robust method to fit the lines and construct the test statistic.
`V`	The estimated variance matrix of measurement error. Average measurement error for Y is in the first row and column, and average measurement error for X is in the second row and column. The default is that there is no measurement error.
`intercept`	(logical) Whether or not the line includes an intercept. **FALSE** no intercept, so the line is forced through the origin **TRUE** an intercept is fitted (this is the default)

Details

Tests if the line relating y to x has a slope equal to test.value (which has a default value of 1). The line can be a linear regression line, major axis or standardised major axis (as selected using the input argument choice). The test is carried out by testing for correlation between residual and fitted values, as described in Warton et al (in review).

A confidence interval for the slope is also returned, which is the primary confidence interval found by inverting the one-sample test statistic.

If measurement error is present, it can be corrected for through use of the input argument V, which makes adjustments to the estimated sample variances and covariances then proceeds with the same method of inference. Note, however, that this method is only approximate (see Warton et al in review for more details).

The test assumes the following:

1. y and x are linearly related
2. residuals independently follow a normal distribution with equal variance at all points along the line

The assumptions can be visually checked by plotting residual scores against fitted axis scores, and by constructing a Q-Q plot of residuals against a normal distribution, available using the `plot.sma` function.

Setting `robust=TRUE` fits lines using Huber's M estimation, and modifies the test statistic as proposed in Taskinen & Warton (in review).

Value

`r`	The test statistic - the sample correlation between residuals and fitted values
`p`	The P-value, taken from the F-distribution. This is an exact test if residuals are normally distributed.
`test.value`	The hypothesised value of the slope
`b`	The estimated slope
`ci`	A 100(1-alpha)% CI for the slope.

Author(s)

Warton, D.I.<David.Warton@unsw.edu.au>, J. Ormerod, & S. Taskinen

References

Warton D. I., Wright I. J., Falster D. S. and Westoby M. (2006) A review of bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

Taskinen, S. and D.I. Warton. in review. Robust tests for one or more allometric lines.

See Also

`sma`, `line.cis`, `elev.test`

Examples

```
#load the leaflife dataset:
data(leaflife)

#consider only the low rainfall sites:
leaf.low.rain <- leaflife[leaflife$rain=='low',]

#test if the SMA slope amongst species at low rainfall sites is 1,
#for log (base 10) transformed data:
with(leaf.low.rain, slope.test(log10(longev), log10(lma)))

#test if the MA slope is 2/3
with(leaf.low.rain, slope.test(log10(longev), log10(lma), test.value = 2/3, method = 'MA'))
```

sma *(Standardized) major axis estimation and testing for one or several samples*

Description

The sma and ma functions fit SMA and MA respectively, and construct confidence intervals or test hypotheses about slope or elevation in one or several samples, depending on how the arguments are specified. Options exist to force lines through the origin, (approximately) correct for measurement error if the measurement error variance is known, and use robust estimation to ensure that inferences are valid in the presence of outliers (currently implemented for single samples only).

Usage

```
sma(formula, data, subset, na.action, log="",
 method=c("SMA","MA","OLS"), type=c("elevation","shift"),
   alpha=0.05, slope.test=NA, elev.test=NA,
   multcomp=FALSE, multcompmethod=c("default","adjusted"),
 robust=FALSE,V=matrix(0,2,2),n_min=3,quiet=FALSE,
 ...)
```

Arguments

formula	Formula of the form y ~ x etc. This determines whether a single (S)MA is fitted, or whether several lines are fitted and compared. See Details.
data	A dataframe with the x and y variables.
subset	A subset of the dataframe for fitting; optional.
na.action	What to do with missing values (na.omit, na.fail, etc).
log	One of 'x', 'y', or 'xy' to log10-transform variables.
method	If SMA, standardized major axis, if MA, major axis, and if OLS, ordinary least squares.
type	If several lines with common slope are to be compared, do you want to test for a change in 'elevation' or for a 'shift' along a common (S)MA. See Details.
alpha	The error rate for confidence intervals. Typically 0.05.
slope.test	The hypothesised value to be used, if testing for evidence that (S)MA slope(s) are significantly different from a hypothesised value.
elev.test	The hypothesised value to be used, if testing for evidence that (S)MA elevation(s) are significantly different from a hypothesised value.
robust	If TRUE, uses a new method of robust line fitting. (Currently available for single-sample testing only.)
V	The estimated variance matrix of measurement error. Average measurement error for Y is in the first row and column, and average measurement error for X is in the second row and column. The default is that there is no measurement error.

n_min	The minimum sample size for a group.
quiet	If TRUE, suppresses all messages.
multcomp	Logical. If TRUE, performs pair-wise comparisons between levels of the grouping variable.
multcompmethod	Whether to adjust the P values for multiple comparisons ('adjusted') or not ('default').
...	Further arguments passed to internal functions (none at the moment?)

Details

This is the key function in the smatr-package; all the key estimation and testing functionality in the package can all be accessed using this function, via different usages of the formula and other arguments, as described below.

One-sample testing The below options allow estimation of a (S)MA, confidence intervals for parameters, and hypothesis testing of parameters, from a single sample of two variables y and x. Use the sma function to fit a standardised major axis (SMA), or use ma in combination with the below options in order to fit major axis (MA) instead.

`sma(y~x)` Fits a SMA and constructs confidence intervals for the true slope and elevation. Replaces the line.cis function from previous versions of smatr.

`ma(y~x)` Fits a MA and constructs confidence intervals for the true slope and elevation. All the below functions also work for MA, if the ma function is called instead of the sma function.

`sma(y~x, slope.test=B)` Tests if the slope of a SMA equals B.

`sma(y~x, elev.test=A)` Tests if the elevation of a SMA equals A.

`sma(y~x, robust=T)` Fits a SMA using Huber's M estimation and constructs confidence intervals for the true slope and elevation. This offers robustness to outliers in estimation and inference, and can be used in combination with the slope.test and elev.test arguments.

`sma(y~x-1)` Fits a SMA where the line is forced through the origin, and constructs confidence intervals for the true slope. This type of formula can be used in combination with the slope.test argument.

For several samples: The below options allow estimation of several (S)MA lines, confidence intervals for parameters, and hypothesis testing of parameters, from two variables y and x for observations that have been classified into several different samples using the factor groups. Use the sma function to fit a standardised major axis (SMA), or use the ma in combination with the below options in order to fir major axis (MA) instead.

`sma(y~x*groups)` Test if several SMA lines share a common slope, and construct a confidence interval for the true common slope.

`sma(y~x+groups, type="elevation")` Test if several common slope SMA lines also share a common elevation, and construct a confidence interval for the true common elevation.

`sma(y~x+groups, type="shift")` Test if several groups of observations have no shift in location along common slope SMA lines.

`sma(y~x*groups, slope.test=B)` Test if several SMA lines share a common slope whose true value is exactly equal to B.

`sma(y~x+groups, elev.test=A)` Test if several common-slope SMA lines share a common elevation whose true value is exactly equal to `A`.

`sma(y~x*groups-1)` Test if several SMA lines forced through the origin share a common slope. This can also be used in combination with the `slope.test` argument or when testing for no shift along common (S)MA lines.

In all cases, estimates and confidence intervals for key parameters are returned, and if a hypothesis test is done, results will be returned and stored in the `slope.test` or `elev.test` output arguments.

The `plot` function can be applied to objects produced using the `sma` and `ma` functions, which is highly recommended to visualise results and check assumptions.

Multiple comparisons If multcomp=TRUE, pair-wise comparisons are made between levels of the grouping variable for differences in slope, elevation or shift, depending on the formula used. The P values can be adjusted for multiple comparisons (using the Sidak correction). See also `multcompmatrix` for visualization of the results. **Warning:** When using the multiple comparisons (multcomp=TRUE), you must specify a `data` statement. If your variables are not in a dataframe, simply combine them in a dataframe before calling `sma`.

Value

An object of class `sma` or `ma`, which contains the following output arguments:

coef The coefficients of the fitted (standardised) major axes. If several samples are being compared, this will return parameters from the alternative model, e.g. assuming separate slopes if testing for common slope, or assuming common slope but separate elevations if testing for common elevation.

nullcoef The coefficients under the null hypothesis

alpha As above.

method The method used in fitting lines: 'MA' or 'SMA'

intercept Whether or not (S)MA lines were forced through the origin (True or False).

call The call to the `ma` or `sma` function.

data As above.

log As above.

variables A list of the variables used in fitting (S)MA lines.

origvariables A list of the variables prior to transformation (if any) for use in fitting (S)MA lines.

groups Levels of the grouping variable, if present in the fit.

gt Type of grouptest ("slopecom","elevcom",or "shiftcom"), if it was carried out, or "none" if none.

gtr The result of that grouptest.

slopetest Output from the hypothesis test of slope(s), if any. Returned as a list of objects, including the P-value (`p`), the test statistic (`r` or `LR`), the (common) slope (`b`) and its confidence interval (`ci`).

elevtest Output from the hypothesis test of elevation(s), if any. Returned as a list of objects, including the P-value (`p`), the test statistic (`t` or `stat`), the (common) elevation (`a`) and its confidence interval (`ci`).

slopetestdone Whether a slopetest was actually carried out.

elevtestdone Whether an elevation test was actually carried out.

n Sample size(s).

r2 Squared correlation coefficient.

pval P-value of the test of correlation coefficient against zero.

from,to X values corresponding the maximum and minimum fitted values in each group. Used by plot.sma to determine endpoints for fitted lines).

groupsummary Neatly organized dataframe with coefficients by group.

Author(s)

Warton, D. I. <David.Warton@unsw.edu.au>, R.A. Duursma, D. Falster, S. Taskinen

References

Warton, D.I., R.A. Duursma, D.S. Falster and S. Taskinen (2012). smatr 3 - an R package for estimation and inference about allometric lines. *Methods in Ecology and Evolution*. **3**, 257-259.

Warton D. I. and Weber N. C. (2002) Common slope tests for bivariate structural relationships. *Biometrical Journal* **44**, 161–174.

Warton D. I., Wright I. J., Falster D. S. and Westoby M. (2006) A review of bivariate line-fitting methods for allometry. *Biological Reviews* **81**, 259–291.

Taskinen S. and Warton D. I. (in press) Robust estimation and inference for bivariate line-fitting in allometry. *Biometrical Journal*.

See Also

plot.sma

Examples

```
# Load leaf lifetime dataset:
data(leaflife)

### One sample analyses ###
# Extract only low-nutrient, low-rainfall data:
leaf.low <- subset(leaflife, soilp == 'low' & rain == 'low')

# Fit a MA for log(leaf longevity) vs log(leaf mass per area):
ma(longev ~ lma, log='xy', data=leaflife)

# Test if the MA slope is not significantly different from 1:
ma.test <- ma(longev ~ lma, log='xy', slope.test=1, data=leaflife)
summary(ma.test)

# Construct a residual plot to check assumptions:
plot(ma.test,type="residual")

### Several sample analyses ###
```

```
# Now consider low-nutrient sites (high and low rainfall):
leaf.low.soilp <- subset(leaflife, soilp == 'low')

# Fit SMA's separately at each of high and low rainfall sites,
# and test for common slope:
com.test <- sma(longev~lma*rain, log="xy", data=leaf.low.soilp)
com.test

# Plot longevity vs LMA separately for each group:
plot(com.test)

# Fit SMA's separately at each of high and low rainfall sites,
# and test if there is a common slope equal to 1:
sma(longev~lma*rain, log="xy", slope.test=1, data=leaf.low.soilp)

# Fit SMA's with common slope across each of high and low rainfall sites,
# and test for common elevation:
sma(longev~lma+rain, log="xy", data=leaf.low.soilp)

# Fit SMA's with common slope across each of high and low rainfall sites,
# and test for no shift along common SMA:
sma(longev~lma+rain, log="xy", type="shift", data=leaf.low.soilp)
```

`summary.sma` *Print a summary*

Description

Writes a summary of a (standardized) major axis fit.

Usage

```
## S3 method for class 'sma'
summary(object, ...)
```

Arguments

`object`	Object of class 'sma'.
`...`	Further arguments ignored.

Details

Does not add much to print.sma, at the moment, except when fit by multiple groups. See `sma` for examples.

Author(s)

Remko Duursma, Daniel Falster, David Warton

See Also

sma,print.sma

Index