Package 'metRology'

May 8, 2026

Version 0.9-29-2

Date 2025-01-20

Title Support for Metrological Applications

Depends R (>= 2.14.0), base, stats

Imports graphics, MASS, numDeriv, robustbase

Description Provides classes and calculation and plotting functions
for metrology applications, including measurement uncertainty estimation
and inter-laboratory metrology comparison studies.

License GPL (>= 2)

URL https://r-forge.r-project.org/projects/metrology/

NeedsCompilation no

Author Stephen L R Ellison. [aut],
Stephen L R Ellison [cre]

Maintainer Stephen L R Ellison <s.ellison@lgcgroup.com>

Repository CRAN

Date/Publication 2025-01-20 11:20:02 UTC

Contents

`metRology-package` *Support for Metrological Applications*

Description

Provides classes and calculation and plotting functions for metrology applications, including measurement uncertainty estimation and inter-laboratory metrology comparison studies.

Details

The metRology package includes functions for:

- Plotting for Key Comparisons (dot-and-bar, consistency)
- Uncertainty evaluation using algebraic or numeric differentiation, with support for correlation
- Monte Carlo evaluation of uncertainty (including correlation for normally distributed variables)
- Classes and functions for location estimates for metrology comparisons
- Mandel's h and k statistics and plots for interlaboratory studies
- Support functions for an excel interface

Changes in version 0.9-29-2 from version 0.9-28-4 include:

- A new set of functions for pair-difference chi-squared statistics; see `pdchisq` and `bootPDchisq`.
- Group label placement in `blockplot` has been extended to take `legend`-like character specification.
- Variable name checking in `uncert` and `uncertMC` has been harmonised and amended to fix a bug which caused apparent mismatches between the supplied variables in `x` and the arguments in function `obj` when the function included an argument "...".
- Minor changes in help files and some functions to clear new CRAN checks.

WARNING: The addition of a new pairwise statistic involves significant duplication of code used for the bootMSD class. Future versions are likely to use common bootstrap code and classes for both MSD and PDchisq classes. bootMSD.default is accordingly deprecated and may be removed or reduced to a wrapper for boot.mtr.pairwise, the current workhorse for bootPDchisq.

Changes in version 0.9-28-4 from version 0.9-28-3:

- Corrected bug in uncertMC which sometimes failed to include non-zero covariances in partial covariance treatment when negative.

Changes in version 0.9-28-3 from version 0.9-28-2 include:

- Corrected huber.estimate to use additional arguments in rlm call.

Changes in version 0.9-28-2 from version 0.9-28-0 include:

- Fixed an error occurring in `barplot.bootMSD` when `names.arg` was unspecified and the boot object had no labels.
- Fixed `blockplot` x-axis label, which was incorrect.

Changes in version 0.9-28-0 from version 0.9-27-2 include:

- A new plot, `blockplot`, added. A "block plot" is a histogram variant identifiying individual data points, which appear as "blocks" in the plot. `blockplot` provides for grouped data, which generates vertically separated subplots for each group. Fills and label colours can be specified for each data point.

Changes in version 0.9-27-2 from version 0.9-26-2 include:

- `pmsd` and related functions will now use fast interpolation by default, and provide exact values for both odd- and even-n data sets up to $n = 199$.
- `gplot` (called by `plot.mandel.kh`) now has a `spacing` parameter which allows finer control of vertical line spacing.

Changes in version 0.9-26-2 from version 0.9-26-1 include:

- Fix to a bug in `reml.loc` which failed to report the standard uncertainty `u` correctly.
- `cplot` now respects `cex.axis` as a plot parameter.

Changes in version 0.9-26-1 from version 0.9-26-0 include:

- Added plot and barplot methods for MSD class.
- Minor correction to code in `msd` to prevent over-replication of estimated `s` when `s` is a function and returns a vector.

Changes in version 0.9-26 from version 0.9-25 include:

- `msd` now returns an object of class `"MSD"` which includes the original data as attributes to permit subsequent bootstrapping.
- A new function, `bootMSD` that performs parametric bootstrapping for `MSD` objects to obtain critical values and *p*-values for the general case where standard uncertainties/standard errors differ appreciably.

Improvements in version 0.9-25 from version 0.9-23 include:

- `plot.d.ellipse` now takes default `xlab` and `ylab` from dimnames in the supplied `cov.dellipse`.

Improvements in version 0.9-23 from version 0.9-22 include:

- A wholly new Youden plot (see `yplot`), with many options for confidence ellipses
- A REML location estimate, `reml.loc`, in addition to `vr.mle`; `reml.loc` can use means and standard uncertainties/standard errors instead of raw data and when doing so does not require degrees of freedom.

- Incremental improvements in handling for the median scaled difference measure of anomalies. msd is faster and less memory-intensive, and pmsd now uses a beta formulation to extend to very high n (at least 1e6 - if you feel *very* patient).
- Support for log and log.p in dt.scaled.

Corrections and bugfixes include:

- amends plot.drop1.uncert to give a plot for each measure of change specified in which
- corrects a grep warning appearing in drop1.uncert;
- corrects an unnecessary 'missing u' error message in version 0.9-22's uncert() when cov was specified and u was not.

Author(s)

Stephen L R Ellison. [aut], Stephen L R Ellison [cre]

Maintainer: Stephen L R Ellison <s.ellison@lgcgroup.com>

---

algA *Robust estimation of location and scale using Algorithm A*

Description

Algorithm A is an implementation of Huber's location and scale estimate with iterated scale.

Usage

```
algA(x, k = 1.5, na.rm = FALSE, tol = .Machine$double.eps^0.25,
maxiter = 25, verbose = FALSE)
```

Arguments

x	numeric vector or array of values.
k	Tuning factor; Winsorisation occurs ar k standard deviations.
na.rm	a logical value indicating whether NA values should be removed before the computation proceeds.
tol	Convergence tolerance Iteration continues until the relative change in estimated sd drops below tol.
maxiter	Maximum number of iterations permitted.
verbose	Controls information displayed during iteration; see Details.

Details

Algorithm A is the robust estimate of location described in ISO 5725-5:1998. It proceeds by winsorisation and re-estimation of scale and location.

The argument k controls the point at which values are Winsorised and hence controls the efficiency. At k=1.5, the value chosen by ISO 5725, the estimator has asymptotic efficiency at the Normal of 0.964. With iterated estimate of scale and k=1.5, the estimator has a breakdown point of about 30

The convergence criterion for Algorithm A is not specified in ISO 5725-5:1998. The criterion chosen here is reasonably stringent but the results will differ from those obtained using other choices. Use verbose=2 to check the effect of different tolerance or maximum iteration count.

If verbose is non-zero, the current iteration number and estimate are printed; if verbose>1, the current set of truncated values w is also printed.

Value

mu	Robust estimate of location
s	Robust estimate of scale

Warning

Algorithm A uses the corrected median absolute deviation as the initial estimate of scale; an error is returned if the resulting scale estimate is zero, which can occur with over 50% of the data set equal. huberM in the robustbase package uses an alternative scale estimate in these circumstances.

Note

Algorithm A is identical to Huber's estimate with variable scale. The implementation here differs from hubers from MASS in:

- hubers allows prior specification of fixed scale (which provides higher breakdown if chosen properly) or location
- the option of verbose output in algA,
- a maximum iteration option in algA
- the convergence criterion; hubers converges on changes in mu, whilst this implementation of Algorithm A converges on changes in s.
- Internally, Algorithm A multiplies by a correction factor for standard deviation whilst hubers divides by a correction factor applied to the variance; the actual correction to s is identical.

The principal reasons for providing an implementation in the metRology package are i) to ensure a close implementation of the cited Standard irrespective of other package developments (though the MASS implementation has proved very stable) and ii) to make the implementation easy to recognise for users of the ISO standard.

Author(s)

S L R Ellison <s.ellison@lgcgroup.com>

References

ISO 5725-5:1998 Accuracy (trueness and precision) of measurement methods and results - Part 5: Alternative methods for the determination of the precision of a standard measurement method

Maronna R A, Martin R D, Yohai V J (2006) Robust statistics - theory and methods. Jhn Wiley and Sons, West Sussex, England.

See Also

algS, hubers, huberM

Examples

```
#Creosote example from ISO 5725-5:1998
#Means for each group are:
cm <-c(24.140, 20.155, 19.500, 20.300, 20.705, 17.570, 20.100, 20.940, 21.185)

algA(cm, verbose=TRUE)
#Iteration 4 corresponds very closely to the ISO 5725 answer
```

---

algS *'Algorithm S' - robust estimate of pooled standard deviation*

Description

'Algorithm S' calculates a robust estimate of pooled standard deviation from a set of standard deviations

Usage

```
algS(s, degfree, na.rm = FALSE, prob.eta = 0.9,
is.range = FALSE, tol = .Machine$double.eps^0.25,
maxiter = 25, verbose = FALSE)
```

Arguments

s	A vector of standard deviations or, if is.range is TRUE, ranges.
degfree	Scalar number of degrees of freedom associated with all values in s. If a vector is supplied, median(degfree) will be used.
na.rm	a logical value indicating whether 'NA' values should be stripped before the computation proceeds.
prob.eta	prob.eta is set to specify the lower tail area of the chi-squared distribution used as a cut-off.
is.range	if is.range is TRUE, s is interpreted as a vector of positive differences of duplcate observations and degfree is set to 1

`tol`	Convergence tolerance Iteration continues until the relative change in estimated pooled sd drops below `tol`.
`maxiter`	Maximum number of iterations permitted.
`verbose`	Controls information displayed during iteration; see Details.

Details

Algorithm S is suggested by ISO 5725-5:1998 as a robust estimator of pooled standard deviation s_{pool} from standard deviations of groups of size $\nu + 1$.

The algorithm calculates a 'limit factor', η, set to `qchisq(prob.eta, degfree)`. Following an initial estimate of s_{pool} as `median(s)`, the standard deviations s_i are replaced with $w_i = min(\eta * s_{pool}, s_i)$ and an updated value for s_{pool} calculated as

$$\xi * \sqrt{\frac{\sum_{i=1}^{p}(w_i)^2}{p}}$$

where p is the number of standard deviations and ξ is calculated as

$$\xi = \frac{1}{\sqrt{\chi^2_{p-1}\left(\nu\eta^2 + (1 - p_\eta)\,\eta^2\right)}}$$

If the s_i are ranges of two values, ISO 5725 recommends carrying out the above iteration on the ranges and then dividing by $\sqrt{\nu + 1}$; in the implementation here, this is done prior to returning the estimate.

If `verbose` is non-zero, the current iteration number and estimate are printed; if `verbose>1`, the current set of truncated values w is also printed.

Value

A scalar estimate of poooled standard deviation.

Author(s)

S L R Ellison <s.ellison@lgcgroup.com>

References

ISO 5725-5:1998 Accuracy (trueness and precision) of measurement methods and results - Part 5: Alternative methods for the determination of the precision of a standard measurement method

See Also

algA

Examples

```
#example from ISO 5725-5:1998 (cell ranges for percent creosote)

cdiff <- c(0.28, 0.49, 0.40, 0.00, 0.35, 1.98, 0.80, 0.32, 0.95)

algS(cdiff, is.range=TRUE)


#Compare with the sd of the two values (based on the range)
c.sd <- cdiff/sqrt(2)
algS(c.sd, degfree=1, verbose=TRUE)
```

---

apricot *Collaborative study results for fibre content in an apricot test material*

Description

A data frame containing reported duplicate results for dietary fibre from a collaborative study.

Usage

```
data(apricot)
```

Format

A data frame containing duplicate results from 9 laboratories:

lab Factor giving abbreviated laboratory identifier

fibre The reported fibre content.

Details

Replicate results appear on separate rows.

Source

B. W. Li, M. S. Cardozo (1994) Determination of total dietary fibre in foods and products with little or no starch, non-enzymatic gravimetric method: collaborative study. *J. AOAC Int.* **77**, 687-689, 1994

References

A. L. Ruhkin, C. J. Biggerstaff and M. G. Vangel (2000) Restricted maximum likelihood estimation of a common mean and the Mandel-Paule algorithm. *J. Stat. Planning an Inferences* **83**, 319-330, 2008

B. W. Li, M. S. Cardozo (1994) Determination of total dietary fibre in foods and products with little or no starch, non-enzymatic gravimetric method: collaborative study. *J. AOAC Int.* **77**, 687-689, 1994

`barplot.mandel.kh` *Barplot of Mandel's h or k statistics*

Description

`barplot.mandel.kh` produces a bar plot of Mandel's statistics, suitably grouped and with appropriate indicator lines for unusual values.

Usage

```
## S3 method for class 'mandel.kh'
barplot(height, probs = c(0.95, 0.99), main,
xlab = attr(height, "grouped.by"),
ylab = attr(height, "mandel.type"), separators = TRUE,
zero.line = TRUE, ylim, p.adjust = "none",
frame.plot = TRUE, ...,
col.ind = 1, lty.ind = c(2, 1), lwd.ind = 1,
col.sep = "lightgrey", lwd.sep = 1, lty.sep = 1,
lwd.zero = 1, col.zero = 1, lty.zero = 1)
```

Arguments

`height`	An object of class `'mandel.kh'` (the name is for consistency with `barplot`).
`probs`	Indicator lines are drawn for these probabilities. Note that `probs` is interpreted as specifying two-tailed probabilities for Mandel's h and one-sided (upper tail) probabilities for Mandel's k.
`main`	a main title for the plot. If missing, the default is `paste(deparse(substitute(x)), " - Mandel's", attr(x, "mandel.type"), if(attr(x, "mandel.method") == "robust") "(Robust variant)")`
`xlab`	a label for the x axis; defaults to the `grouped.by` attribute for `x`.
`ylab`	a label for the x axis; defaults to the `mandel.type` attribute for `x`.
`separators`	Logical; if `TRUE`, separator lines are drawn between groups of values.
`zero.line`	logical; if `TRUE` a horizontal line is drawn at zero.
`ylim`	the y limits of the plot. For Mandel's k, the default lower limit is zero.

`p.adjust` Correction method for probabilities. If not `"none"`, passed to `p.adjust` prior to calculating indicator lines. Usually, indicator lines are drawn without correction (that is, with `p.adjust="none"`); specifying a p-value correction effectively turns the Mandel's statistics into single outlier tests.

`frame.plot` Logical; If `TRUE` a box is drawn around the plot.

`...` Other (usually graphical) parameters passed to `barplot`. Note that some parameters appear after ... to prevent spurious argument matching inside `barplot.default`.

`col.ind`, `lty.ind`, `lwd.ind`
Graphical parameters used for the indicator lines, recyckled to `length(probs)`. For `attr(x, "mandel.type")=="h"` the graphical parameters are applied to negative as well as positive indicator lines, applied outwards from zero.

`col.sep`, `lwd.sep`, `lty.sep`
Graphical parameters used for the separator lines.

`lwd.zero`, `col.zero`, `lty.zero`
Graphical parameters used for the zero line.

Details

Mandel's statistics are traditionally plotted for inter-laboratory study data, grouped by laboratory, to give a rapid graphical view of laboratory bias and relative precision. This plot produces a grouped, side-by-side bar plot.

For classical Mandel statistics, indicator lines are drawn based on `qmandelh` or `qmandelk` as appropriate. For robust variants, indicator lines use `qnorm` for the h statistic and `qf(probs, n, Inf)` for the k statistic. Note that this corresponds to taking the robust estimates of location and scale as true values, so will be somewhat anticonservative.

Value

barplot.mandel.kh returns a numeric vector of mid-points of the groups along the x-axis.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

References

Accuracy (trueness and precision) of measurement methods and results – Part 2: Basic method for the determination of repeatability and reproducibility of a standard measurement method. ISO, Geneva (1994).

See Also

`mandel.h`, `mandel.k`, `mandel.kh`, `pmandelh`, `pmandelk` for probabilities, quantiles etc.

See `plot.mandel.kh` for the 'classic' Mandel plot.

Examples

```
data(RMstudy)

h <- with(RMstudy, mandel.h(RMstudy[2:9], g=Lab))
barplot(h, las=2) # Lab 4 shows consistent low bias;
               # Lab 23 several extreme values.

#Use colours to identify particular measurands:
barplot(h, las=2, col=1:8)
legend("bottomleft", legend=names(h), fill=1:8, cex=0.7, bg="white")

#Example of Mandel's k:
k <- with(RMstudy, mandel.k(RMstudy[2:9], g=Lab))
barplot(k, las=2) # Lab 8 looks unusually variable;
               # Lab 14 unusually precise
```

---

bkp *Draw block plots from block plot objects*

Description

bkp draws block plots based on block plot objects generated by blockplot and its methods. It is normally called from within blockplot but can be invoked directly.

Usage

```
bkp(x, labels = x$labels, xlim = NULL, ylim = NULL,
            main = NULL, xlab = NULL, ylab = "Frequency",
            square = FALSE, add = FALSE, offset = 0,
            grp.spacing = 2, grp.at = NA,
            fill = NA, border = NULL, density = NULL, angle = 45,
            lty = 1, lwd = 2, label.col = 1, cex = NA, adj = c(0.5, 0.4),
            uline = 2, uline.lwd = lwd, uline.lty = 1,
            uline.col = if (!is.null(border)) border else par("fg"),
            grp.labs = FALSE, grp.pos = 1, glab.control = list(),
            axes = c(TRUE, FALSE), asp = NA, frame.plot = any(axes),
            drop.unused = TRUE, unused.label="[Missing]", ...)
```

Arguments

x	An R object. For the default method, a vector of values for which the blockplot is desired. For the formula method, a valid formula, such as y ~ grp (see Details).
labels	Labels for data points, passed to text; in principle of any type acceptable to text. Labels are placed inside boxes so should be short for readability.
xlim	Limits (x1, x2) for the horizontal (x) range of the plot. The default is the range of breaks, after computation if necessary.

ylim	limits for the vertical range of the plot. Will be overridden if square=TRUE (see below).
main	Main title for the plot, passed to plot.
xlab, ylab	x- and y-axis labels for the plot. As usual, either can be expressions (see plotmath).
square	Logical: If square=TRUE, the aspect ratio is set (via asp) to make the individual blocks appear square when rendered. Note that this generally overrides ylim.
add	Logical: If TRUE, the plot will be added to an existing plot.
offset	Numeric scalar value. Vertical offset for the plot, in units of block height. offset is added to the vertical position of the boxplot and any subplots. offset can be considered as a number of vertical block heights to raise (or lower, if negative) the plot. This, with add, is useful for adding further groups manually to an existing plot.
grp.spacing	Numeric scalar, giving the minimum vertical spacing (in units of block height) between subplots when there is more than one group.
grp.at	Optional vector specifying explicit vertical locations for subplot baselines (including the first group). The default (grp.at=NA) is to use grp.spacing. If specified, grp.at overrides grp.spacing.
fill	Fill colour for the rectangles ("blocks") making up the plot. Recycled to length length(x). Passed to rect.
border	Border colour for the rectangles ("blocks") making up the plot. Recycled to length length(x). Passed to rect.
density	Shading line density for ("blocks") making up the plot. Recycled to length length(x). Passed to rect.
angle	Shading line angle for ("blocks") making up the plot. Recycled to length length(x). Passed to rect.
lty	Border line type for ("blocks") making up the plot. Recycled to length length(x). Passed to rect.
lwd	Border line width for ("blocks") making up the plot. Recycled to length length(x). Passed to rect.
label.col	Colour used for text labels in each ("block") making up the plot. Recycled to length length(x). Passed to text.
cex	Size of text labels in each ("block") making up the plot. Recycled to length length(x) and passed to text. The default is to fit text inside each block automatically.
adj	Vector of two values giving text location adjustment for all block labels. Passed to text
uline	Specification for the distance by which the baseline for each subplot extends beyond the data range for the group. See 'Graphical elements' for details. The default is two units either side.
uline.lwd	Scalar: Line width for the subplot baseline(s).
uline.lty	Scalar: Line type for the subplot baseline(s).
uline.col	Scalar: Line colour for the subplot baseline(s).

grp.labs	Logical, determining whether group labels are plotted, or a vector of labels. See 'Details'.
grp.pos	Integer or character specification for the position of group labels. See 'Details'.
glab.control	List of arguments to be passed to text when drawing group labels.
axes	Logical, indicating whether axes are drawn. Can be a vector of two logicals, specifying horizontal and vertical axes respectively. See 'Graphical elements' for details.
asp	Aspect ratio, passed to plot
frame.plot	Logical, controlling whether a frame (box) is drawn round the plot.
drop.unused	Logical specification for the treatment of empty groups. If TRUE, groups (identified by levels in x$groups) that contain no non-missing values will be omitted from the plot. If FALSE, a space is created for the missing subplot but no subplot is generated. If grp.labels requires group labelling, the group label is drawn in the space with unused.label appended.
unused.label	Character string appended to missing group labels.
...	Further parameters passed to plot

Details

bkp provides considerable control of graphical elements. The main elements, and the arguments controlling their location and appearance, are:

General appearance A block plot of a single group of data has the general appearance of a histogram. However, instead of vertical bars (of possibly variable width) indicating the number of data points within the bin interval, each bin is a stack of rectangles, each corresponding to a single data point and with an optional label identifying the datum.

Block plots of this kind are useful for data sets of modest size; typically 10-100 per group, as individual labels quickly become hard to identify in larger data sets.

By default, blockplot produces one such plot for a set of data. If a series of such plots is needed, this can be accommodated either by using blockplot with add=TRUE to build up a plot manually, setting xlim, ylim and breaks to accommodate all the required groups. Alternatively, a grouping factor can be provided (via argument groups) which will produce a series of subplots, laid out automatically. The use of groups and the corresponding layout options are detailed below (see "Groups").

The vertical position of a single block plot within the figure can be set using offset, which sets the baseline height, in units of block height, from the figure origin. This is useful for separating several groups that are added manually; just set offset appropriately for each separate plot. Note that setting offset has no effect on the automatic ylim setting, which means that ylim must be set manually to accommodate the vertical offset.

Blocks Each individual rectangle ("block" in the plot corresponds to a single data point. In this implementation, blocks appear in rank order from left to right *and* from bottom to top; that is, data are placed in vertical bins as in a normal histogram but, in addition, the vertical ordering of blocks corresponds to the data order within each bin, with blocks at the bottom corresponding to lower values.

Blocks are always 1 unit high, so the total vertical height of each bin corresponds directly to frequency (not density) in a histogram. The block width is the interval between `breaks`, which must be equispaced.

By default, the apparent aspect ratio for blocks depends primarily on `xlim` and `ylim` and the height and width of the plotting device. However, setting `square=TRUE` will cause the plot aspect ration (`asp`) to be set such that the bocks appear square in the current plot window.

Fill colour, border colour and style, fill effects and text colour of individual blocks can all be controlled using `fill`, `border`, `density`, `angle`, `lty`, `lwd` and `label.col`, as the relevant arguments can be vectors of length `length(x)`. This allows conditional formatting, for example to identify a particular data point or some secondary grouping variable.

Subplot baseline The baseline for each subplot is controlled by `uline`, as follows:

`TRUE`**:** The line extends the full width of the plot;

`FALSE`**:** No baseline is drawn;

numeric: If numeric (as for the default), `uline` specifies the distance that the baseline extends beyond each end of the data, in units of block width. `uline` can be length 2 numeric vector, which specifies the baseline extension on the left and right sides respectively.

Colour, line type, and line width for the subplot baseline(s) can be controlled with `uline.col`, `uline.lty`, and `uline.lwd` respectively.

Axes Axes can be controlled with the `axes` argument, which controls whether or not axes are drawn. If a vector of two logical values (as for the default), `axes` specifies drawing for horizontal and vertical axes respectively.

The horizontal axis is normally continuous for the plot. If a vertical (frequency) axis is requested (either by `axes=TRUE` or, for example, by `axes=c(TRUE, TRUE)`, a vertical axis is drawn for each subplot, starting at zero at the baseline and terminating at the highest vertical value in the subplot. Vertical axes, restarting at 0 at the next subplot baseline, are drawn if there is more than one group.

Groups `blockplot` provides a simple grouping mechanism to display separate subplots for different groups of data on the same figure. The default method provides for a grouping variable specified via `groups`. The formula method provides a somewhat more flexible interface, allowing specification of more than one grouping variable. Like `boxplot`, if there is more than one goruping variable in the formula, subplots are drawn for each (non-empty) level of the interaction term.

Subplots for different groups are arranged vertically. Vertical position can be specified explicitly via `grp.at` or, more simply, by setting `grp.spacing`. The latter sets `grp.at` to equal vertical spacing such that the smallest vertical gap is `grp.spacing`. Both `grp.at` and `grp.spacing` are in units of block height; that is, `grpspacing=2` (the default) means that the smallest vertical gap is equivalent to two blocks.

Group labels Labels can be added to each subplot. These are controlled by `grp.labs` .. `grp.labs` provides the specification for group labels, and can be a single logical or a vector of labels. Effects of `grp.labs` are as follows:

- `FALSE` (The default): No group labels are drawn.
- `TRUE` Labels are taken as `levels(groups)`, and set to `"1"` if there is only one group.
- Vector If a character vector (or expression) is provided, these are used as labels for the groups plotted.
  WARNING: If missing values in `x` cause group levels to be dropped, those groups will not be plotted. `grp.labs` must have the same length as the number of groups plotted.

An error is generated if the length of `labels` differs from the number of groups actually plotted.

`grp.pos` specifies the general positioning of group labels relative to each subplot. `grp.pos` can be a numeric or character specification.

If `grp.pos` is numeric, label position follows `pos` in `text`: Values of `1`, `2`, `3` and `4`, respectively indicate positions below, to the left of, above and to the right of the plotted data for the group. The four positions correspond to the midpoints of the corresponding edge of each subplot, where the 'edges' are the baseline, leftmost block, topmost block and rightmost block. Labels are placed a short distance outward from the midpoint of these edges. Labels are justified according to position; by default `grp.pos` is used as the `pos` argument to `text`.

`grp.pos` can also be one of `"bottom"`, `"bottomleft"`, `"left"`, `"topleft"`, `"top"`, `"topright"`, `"right"` or `"bottomright"`. These specify positions around the edges of the current subplot region, that is, the rectangle defined by the top and bottom of the current group plot and the left and right eges of the complete figure region. The label justification is set according to the edge; all the left and right placements use `pos=4` and `pos=2`, while `"bottom"` and `"top"` use `pos=1` and `pos=3`. (Note that this means that the vertical positions for labels differ slightly between `"top"` and `"topleft"` etc.)

The principal difference between the numeric and character specifications is therefore that the numeric codes place the labels close to the plotted data, while the character specifications place labels consistently relative to the sides of the figure region.

Further control of group label position is available via `grp.control`, which is a list (empty by default) of arguments passed to `text`. Ths can include arguments such as `pos` and `adj`, as well as appearance elements such as `col`, `cex` etc. If present in `grp.control`, `pos` and `adj`override the default label justification.

Value

`bxp` returns the original object `x` with additional elements:

`grp.at`	The vertical coordinated for the subplot baselines.
`blockwidth`	The width of the blocks in the plot.

Author(s)

Stephen L R Ellison <s.ellison@lgcgroup.com>.

References

ISO 5725-2:1994 *Accuracy (trueness and precision) of measurement methods and results – Part 2: Basic method for the determination of repeatability and reproducibility of a standard measurement method.* ISO, Geneva, 1994.

See Also

For constructing breaks and grouping: `blockplot`

For graphical elements: `text`, `rect`

Examples

```
#A simple blockplot
set.seed(55)
x<-rnorm(48, 15)
b <- blockplot(x)

#Aspect ratio control for square blocks
bkp(b, square=TRUE)

# Specifying groups
grp <- gl(3, 16)
bg <- blockplot(x~grp)

# Add vertical axes ( axes=TRUE asks for horizontal and vertical axes)
bkp(bg, axes=TRUE )

#Axes both left and right
par(mar=c(5,4,4,4)+0.1)
bkp(bg, axes=c(TRUE, TRUE, FALSE, TRUE) )
#Note that axes[3] is FALSE to suppress top axis

# Vectorised colour specification
bkp(bg, square=TRUE, fill=ifelse(1:48 %in% c(15, 23, 24), "gold", "white") )

# Group labelling
bkp(bg, square=TRUE, grp.labs=paste("Level", 1:3), grp.pos=2)

#Alternative (character) label positioning
bkp(bg, square=TRUE, grp.labs=paste("Level", 1:3), grp.pos="left")
                           #Note the difference from grp.pos=2

#Alternating label locations:
bkp(bg, square=TRUE, grp.labs=paste("Level", 1:3),
                     grp.pos=c("left", "right"))
```

`blockplot` *Generate a "block plot" - a histogram variant identifiying individual data points.*

Description

A "block plot" is a histogram variant identifiying individual data points. Histogram bars are replaced by a stack of rectangles ("blocks", each of which is labelled. `blockplot` provides for grouped data, which generates vertically separated subplots for each group. Fills and label colours can be specified for each data point.

Usage

```
blockplot(x, ...)

bplot(x, ...)

## Default S3 method:
blockplot(x, breaks = "23", labels = paste(1:length(x)), groups = NA,
xlim = NULL, ylim = NULL,
main = NULL, xlab = NULL, ylab = "Frequency", grp.labs = FALSE,
include.lowest = TRUE, right = TRUE, nclass = NULL,
plot = TRUE,  add=FALSE, ...)


## S3 method for class 'formula'
blockplot(x, data = NULL, ..., subset, main = NULL, xlab = NULL)


nclass.23(x)
```

Arguments

x	An R object. For the default method, a vector of values for which the blockplot is desired. For the formula method, a valid formula, such as y ~ grp (see Details).
data	For the formula method, a data frame or list from which the variables in formula should be taken.
subset	For the formula method, an optional vector specifying a subset of observations to be used for plotting.
breaks	Either a specification for choosing breakpoints for "binning" the data, or a vector giving the breakpoints themselves. The specification can be a single number, a function, or a character string identifying a function. See 'Details' for detailed specification.
labels	Labels for data points, passed to text; in principle of any type acceptable to text. Labels are placed inside boxes so should be short for readability.
groups	An optional grouping variable, coerced to factor. If present, one subplot is produced for each non-empty group.
xlim	Limits (x1, x2) for the horizontal (x) range of the plot. The default is the range of breaks, after computation if necessary.
ylim	limits for the vertical range of the plot. Will be overridden if square=TRUE (see below).
main	Main title for the plot, passed to plot.
xlab, ylab	x- and y-axis labels for the plot. As usual, either can be expressions (see plotmath).
grp.labs	Logical, determining whether group labels are plotted, or a vector of labels. See 'Details'.

include.lowest	Logical, indicating whether a value equal to the lowest (or highest, for right = FALSE) breaks value should be included in a given bin. Passed to cut.
right	Logical, indicating whether the bin intervals should be closed on the right (and open on the left) or vice versa. Passed to cut.
nclass	Suggested number of classes for breaks; equivalent to a single numerical value for breaks.
plot	If FALSE, no plot is produced. The return value is returned invisibly.
add	If TRUE, the plot is added to an existing figure.
...	Further parameters passed to other functions, in particular, bkp, which creates the plot, and plot

Details

blockplot produces a block plot - a histogram variant identifying individual data points. Histogram bars are replaced by a stack of rectangles ("blocks", each of which can be (and by default, is) labelled.

bplot is an alias for blockplot.

For the formula method, x is a formula, such as y ~ grp, in which y is a numeric vector of data values to be split into groups according to the grouping variable grp (usually a factor). More than one grouping variable can be specified, in which case subplots are produced for each level of the interaction between grouping factors.

The specification for breakpoints, breaks, is modelled closely on that for hist. breaks can be one of:

- a vector giving the (equally spaced) breakpoints between bins;
- a function to compute the vector of breakpoints;
- a single number giving the suggested number of bins for the blockplot;
- a character string naming an algorithm to compute the number of cells. Values of "23" (the default), "Sturges", "Scott", "FD" and "Freedman-Diaconis" are currently supported; see below for their effect
- a function to compute the number of bins.

In the last three cases the number is a suggestion only, as the breakpoints will be set to "pretty" values.

The different character string specifications correspond to "nclass" functions, including those used by hist; see nclass.FD for details of those. In addition, the default "23" corresponds to the function nclass.23. This is just a wrapper for the one-line expression

ceiling(length(x)^(2/3)),

which appears to provide good results for block plots.

Considerable control of graphical elements is provided by the plotting function bkp, which is called by blockplot. In particular, arguments passed through ... to bkp can control:

- The general shape of the plot, including the asbect ratio of the "blocks";
- whether a plot should be added to an existing figure (add)

- the fill colour and shading, the border width, type and colour, and the font size and colour of individual blocks;
- the vertical location of the plot in the figure region `offset`;
- the vertical spacing between multiple plots on the same figure when a grouping variable is provided (`grp.spacing` and `grp.at`;
- the presence, location and appearance of labels for individual subplots;
- whether axes are plotted on any of the four sides of the plot;
- the appearance or omission of empty groups.

See bkp for further details.

Value

Blockplot currently returns an object of class `blockplot`, which is a list with elements:

`x`	The original data
`groups`	If there is more than one group, a factor of groups for each data point, with additional attribute `"gname"` containing a default name for the grouping variable(s). `groups` is set to `NA` if there is only one group.
`x.left`	Vector of x-coordinates for the left side of each block
`x.height`	Vector of y-coordinates for each box, relative to the group baseline
`x.mid`	Vector of x-coordinates for the middle of each block (the text location)
`x.mid`	Vector of x-coordinates for the middle of each block (the text location)

Note

The name "block plot" may not be in general use, but the package author has been unable to identify either an alternative designation or an original source for this type of plot. An example - apparently hand drawn - was given in ISO 5725-2:1994 (referenced above).

Author(s)

S Ellison `<s.ellison@lgcgroup.com>`

References

ISO 5725-2:1994 *Accuracy (trueness and precision) of measurement methods and results – Part 2: Basic method for the determination of repeatability and reproducibility of a standard measurement method.* ISO, Geneva, 1994.

See Also

For plotting and control of plot appearance: `link{bkp}`

For graphical elements: `text`, `rect`

For specification of `breaks`: `link{nclass.Sturges}`, `link{nclass.Scott}`, `link{nclass.FD}`

Examples

```
#A simple blockplot
set.seed(55)
x<-rnorm(48, 15)
blockplot(x)

#Aspect ratio control for square blocks
blockplot(x, square=TRUE)

# Specifying groups
grp <- gl(3, 16)
blockplot(x, groups=grp)

#Formula interface
blockplot(x~grp)

#Vectorised colour specification
blockplot(x~grp, square=TRUE, fill=ifelse(1:48 %in% c(15, 23, 24), "gold", "white") )

#Group labelling
blockplot(x~grp, square=TRUE, grp.labs=paste("Level", 1:3), grp.pos=2)

#A missing group
xm <- x
xm[ grp == "2" ] <- NA
blockplot(xm~grp, square=TRUE, grp.labs=paste("Level", 1:3), grp.pos=2)

blockplot(xm~grp, square=TRUE, grp.labs=paste("Level", 1:3), grp.pos=2, drop.unused=FALSE)
```

---

boot.mtr.pairwise *Parametric bootstrap for pairwise comparison statistics.*

Description

Generates a parametric bootstrap for the pair-difference chi-squared statistic or MSD.

`boot.mtr.pairwise` is primarily intended to be called from other functions, notably `bootPDchisq`.

Usage

```
boot.mtr.pairwise(x, s=mad , B=3000, probs=c(0.95, 0.99),
cov=NULL, cor = NULL, stat=c("MSD", "PDchisq"),
method=c("rnorm", "lhs"), keep=FALSE, labels=names(x), ...)
```

Arguments

x	Vector of observations

s	Either a function returning an estimate of scale for x or a vector of length length(x) of standard errors or standard uncertainties in x.
B	Scalar number of bootstrap replicates.
probs	Vector of probabilities at which to calculate upper quantiles. Passed to quantile.
cov, cor	Covariance or correlation matrix, respectively, describing the covariance structure across x. Passed to pdchisq.
stat	The statistic to be bootstrapped. Either "MSD" or "PDchisq" for msd or pdchisq, respectively.
method	Character value describing the simulation method.
keep	If keep == TRUE the individual bootstrap replicates are retained.
labels	Character vector of labels for individual values.
...	Parameters passed to other methods.

Details

boot.mtr.pairwise calculates a parametric bootstrap simulation (or Monte carlo simulation) of the results of msd or pdchisq applied to data. This allows individual case-specific quantiles and *p*-values to be estimated that allow for different standard errors (or standard uncertainties) s.

The sampling method is currently sampling from rnorm.

Individual upper quantiles for probabilities probs and *p*-values are estimated directly from the bootstrap replicates. Quantiles use quantile. *p*-values are estimated from the proportion of replicates that exceed the observed values calculated by msd or pdchisq. Note that the print method for the summary object does not report zero proportions as identically zero.

Value

An object of class "bootMtrPairs", consisting of a list containing: of length length(x) of median scaled absolute deviations for each observation, with attributes:

t0	vector of values calculated by msd or pdchisq
labels	character vector of labels, by default taken from x
probs	vector of probabilities supplied and used for quantiles
critical.values	matrix of quantiles. Each row corresponds to a probability in probs and each column to an individual data point.
pvals	*p*-values estimated as the observed proportion of simulated values exceeding the calculated values t0.
B	Number of bootstrap replicates used.
method	The sampling method used by the parametric bootstrap.
stat	The statistic bootstrapped. Either "MSD" or "PDchisq" for msd or pdchisq, respectively.
t	If keep=TRUE, the individual bootstrap replicates generated by msd or pdchisq. t is set to NA if keep=FALSE

.

Summary, print and plot methods are provided for the class; see bootMtrPairs-class.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

See Also

msd, bootMSD-class, pdchisq, PDchisq-class,

Examples

```
data(Pb)
## Not run:
#Default method:
set.seed(1023)
boot.Pb.default <- boot.mtr.pairwise(Pb$value, Pb$u)  # Uses individual standard uncertainties
summary(boot.Pb.default)


#Method for pair-difference chi-squared object:
pwch.Pb<-pdchisq(Pb$value, Pb$u)  # Uses individual standard uncertainties
boot.Pb <- boot.mtr.pairwise(pwch.Pb, B=5000)
 #Increased replication compared to default
summary(boot.Pb)

# NOTE: The default summary gives individual observation p-values.
# To correct for multiple comparisons, apply
# a suitable p-value adjustment; for example:
summary(boot.Pb, p.adjust="holm")


## End(Not run)
```

---

bootMSD *Parametric bootstrap for median scaled difference*

Description

Generates a parametric bootstrap for the median of scaled differences from each point in a data set to all other points..

Usage

```
        bootMSD(x, ...)

        ## Default S3 method:
bootMSD(x, s = mad, B = 3000, probs = c(0.95, 0.99),
               method = c("rnorm", "lhs"), keep = FALSE, labels = names(x), ...)
```

```
        ## S3 method for class 'MSD'
bootMSD(x, B = 3000, probs = c(0.95, 0.99),
               method = c("rnorm", "lhs"), keep = FALSE, labels = names(x), ...)
```

Arguments

x	An R object. For the default method, a vector of observations. For the MSD method, an object of class "MSD". For print, summary and plot methods, an object of class "bootMSD".
s	Either a function returning an estimate of scale for x or a vector of length length(x) of standard errors or standard uncertainties in x.
B	Scalar number of bootstrap replicates.
probs	Vector of probabilities at which to calculate upper quantiles. Passed to quantile.
method	Character value describing the simulation method.
keep	If keep == TRUE the individual bootstrap replicates are retained.
labels	Character vector of labels for individual values.
...	Parameters passed to other methods.

Details

bootMSD calculates a parametric bootstrap simulation (or Monte carlo simulation) of the results of msd applied to data. This allows individual case-specific quantiles and *p*-values to be estimated that allow for different standard errors (or standard uncertainties) s.

The sampling method is currently either sampling from rnorm or by latin hypercube sampling using lhs.

Individual upper quantiles for probabilities probs and *p*-values are estimated directly from the bootstrap replicates. Quantiles use quantile. *p*-values are estimated from the proportion of replicates that exceed the observed MSD calculated by msd. Note that the print method for the summary object does not report zero proportions as identically zero.

Value

An object of class "bootMSD", consisting of a vector of length length(x) of median scaled absolute deviations for each observation, with attributes:

msd	vector of raw calculated MSD values calculated by msd
labels	character vector of labels, by default taken from x
probs	vextor of probabilities supplied and used for quantiles
critical.values	matrix of quantiles. Each row corresponds to a probability in probs and each column to an individual data point.
pvals	*p*-values estimated as the observed proportion of simulated values exceeding the MSD value calculated by msd.
B	Number of bootstrap replicates used.

method	The sampling method used by the parametric bootstrap.
t	If keep == TRUE, the individual bootstrap replicates generated by bootMSD. Set to NA if keep == FALSE.

Summary, print and plot methods are provided for the class; see bootMSD-class.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

Ellison, S. L. R. (2018) An outlier-resistant indicator of anomalies among inter-laboratory comparison data with associated uncertainty. _Metrologia_ (accepted 4 October 2018)

See Also

msd, bootMSD-class, print.bootMSD, plot.bootMSD, summary.bootMSD.

Examples

```
data(Pb)
## Not run:
#Default method:
set.seed(1023)
boot.Pb.default <- bootMSD(Pb$value, Pb$u)  # Uses individual standard uncertainties
summary(boot.Pb.default)


#Method for MSD object:
msd.Pb<-msd(Pb$value, Pb$u)  # Uses individual standard uncertainties
boot.Pb <- bootMSD(msd.Pb, B=5000)
 #Increased replication compared to default
summary(boot.Pb)

# NOTE: The default summary gives individual observation p-values.
# To correct for multiple comparisons, apply
# a suitable p-value adjustment:
summary(boot.Pb, p.adjust="holm")


## End(Not run)
```

bootMSD-class *Object returned by* bootMSD *and associated methods.*

Description

The object class returned by bootMSD and associated print, summary, and plotting classes.

Usage

```
       ## S3 method for class 'bootMSD'
print(x, ...)

       ## S3 method for class 'bootMSD'
plot(x, ...)

       ## S3 method for class 'bootMSD'
barplot(height, ylab="MSD", names.arg=height$labels,
crit.vals=TRUE, lty.crit=c(2,1), col.crit=2, lwd.crit=c(1,2), ylim=NULL, ... )

       ## S3 method for class 'bootMSD'
summary(object, p.adjust="none", ...)

       ## S3 method for class 'summary.bootMSD'
print(x, digits=3, ...,
signif.stars = getOption("show.signif.stars"),
signif.legend=signif.stars)
```

Arguments

x	An R object. For print.bootMSD and plot.bootMSD, an object of class "bootMSD". For print.summary.bootMSD, an object of class "summary.bootMSD".
height	An object of class "bootMSD".
object	An object of class "MSD".
p.adjust	Multiple correction method for calculated *p*-values, passed to p.adjust.
ylab	Label for vertical axis, passed to barplot
names.arg	Labels for individual bars in bar plot, passed to barplot. If names(height) is NULL, bars are numbered.
crit.vals	If TRUE, individual critical values based on observation-specific bootstrap quantiles are added to the plot. These are taken from critical.values in the supplied bootMSD object.
lty.crit, col.crit, lwd.crit	Vectors of line style parameters for plotted critical values, passed to segments. Recycled to the length of critical.values in the supplied bootMSD object.

ylim	Limits for plot y range, passed to barplot. The default ensures that the plotted bars and (if crit.vals=TRUE) the critical values are included in the figure region.
digits	integer; passed to print. The minimum number of significant digits to be printed in values. Change to NULL for default.
signif.stars	logical; if TRUE, P-values are additionally encoded visually as 'significance stars' in order to help scanning of long coefficient tables. Defaults to the show.signif.stars slot of options.
signif.legend	logical; if TRUE, a legend for the 'significance stars' is printed provided signif.stars == TRUE.
...	Parameters passed to other methods.

Details

The default plot method is an alias for the barplot method. For the plot methods, quantiles for each point are taken directly from the quantiles calulated by bootMSD and retained in the returned object.

For the summary method, *p*-values are initially calculated as the observed proportion of simulated values exceeding the MSD value calculated by msd. The summary method additionally returns *p*-values after adjustment for multiple comparisons using the adjustment method specified.

The print method for the summary.bootMSD object prints the summary as a data frame adjusted with columns for the calculated MSD values, data-specific upper quantiles (one column for each probability supplied to bootMSD and the *p*-values after adjustment for multiple comparisons based on the proportion of simulated values exceeding the observed MSD. Where that proportion is zero, the summary replaces the raw zero proportion with 1/B, corrects that proportion using the requested adjustment method, andreports the *p*-value as less than ("<") the resulting adjusted value.

Value

The print method returns the object, invisibly.

The plot and barplot methods return the values at the midpoint of each bar.

The summary method returns an object of class "summary.bootMSD" which is a list with members:

msd	Calculated MSD values from msd
labels	character vector of labels for individual data points
probs	Probabilities used for quantiles
critical.values	matrix of quantiles. Each row corresponds to a probability in probs and each column to an individual data point.
pvals	*p*-values estimated as the observed proportion of simulated values exceeding the MSD value calculated by msd.
p.adjust	Character value containing the name of the p-value adjustment method used.
p.adj	*p*-values adjusted using the given *p*-value adjustment method specified by p.adjust.
B	Number of bootstrap replicates used.
method	The sampling method used by the parametric bootstrap.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

See Also

msd, qmsd.

Examples

```
## Not run:
data(Pb)
msd.Pb<-msd(Pb$value, Pb$u)  # Uses individual standard uncertainties

set.seed(1023)
boot.Pb <- bootMSD(msd.Pb)
summary(boot.Pb)

# The default summary gives individual observation p-values. To
# avoid over-interpretation for the study as a whole,
# apply a sensible p-value adjustment:
summary(boot.Pb, p.adjust="holm")

plot(boot.Pb, crit=TRUE)


## End(Not run)
```

---

bootMtrPairs-class *Object returned by* bootMtrPairs *and associated methods.*

Description

The object class returned by boot.mtr.pairwise and associated print, summary, and plotting classes.

Usage

```
       ## S3 method for class 'bootMtrPairs'
print(x, ...)

       ## S3 method for class 'bootMtrPairs'
plot(x, ...)

       ## S3 method for class 'bootMtrPairs'
barplot(height, ylab=NULL, names.arg=height$labels,
crit.vals=TRUE, lty.crit=c(2,1), col.crit=2, lwd.crit=c(1,2), ylim=NULL, ... )

       ## S3 method for class 'bootMtrPairs'
```

```
summary(object, p.adjust="none", ...)

       ## S3 method for class 'summary.bootMtrPairs'
print(x, digits=3, ...,
signif.stars = getOption("show.signif.stars"),
signif.legend=signif.stars)
```

Arguments

x	An R object. For print.bootMtrPairs and plot.bootMtrPairs, an object of class "bootMtrPairs". For print.summary.bootMtrPairs, an object of class "summary.bootMtrPairs".
height	An object of class "bootMtrPairs".
object	An object of class "bootMtrPairs".
p.adjust	Multiple correction method for calculated *p*-values, passed to p.adjust.
ylab	Label for vertical axis, passed to barplot
names.arg	Labels for individual bars in bar plot, passed to barplot. If names(height) is NULL, bars are numbered.
crit.vals	If TRUE, individual critical values based on observation-specific bootstrap quantiles are added to the plot. These are taken from critical.values in the supplied bootMtrPairs object.
lty.crit, col.crit, lwd.crit	Vectors of line style parameters for plotted critical values, passed to segments. Recycled to the length of critical.values in the supplied bootMtrPairs object.
ylim	Limits for plot y range, passed to barplot. The default ensures that the plotted bars and (if crit.vals=TRUE) the critical values are included in the figure region.
digits	integer; passed to print. The minimum number of significant digits to be printed in values. Change to NULL for default.
signif.stars	logical; if TRUE, P-values are additionally encoded visually as 'significance stars' in order to help scanning of long coefficient tables. Defaults to the show.signif.stars slot of options.
signif.legend	logical; if TRUE, a legend for the 'significance stars' is printed provided signif.stars == TRUE.
...	Parameters passed to other methods.

Details

The default plot method is an alias for the barplot method. For the plot methods, quantiles for each point are taken directly from the quantiles calulated by boot.mtr.pairwise and retained in the returned object.

For the summary method, *p*-values are initially calculated as the observed proportion of simulated values exceeding the MSD value calculated by msd. The summary method additionally returns *p*-values after adjustment for multiple comparisons using the adjustment method specified.

The print method for the summary.bootMtrPairs object prints the summary as a data frame with columns for the calculated values (re-titled for the particular statstic), data-specific upper quantiles (one column for each probability supplied to bootMtrPairs and the *p*-values after adjustment for multiple comparisons based on the proportion of simulated values exceeding the observed MSD. Where that proportion is zero, the summary replaces the raw zero proportion with 1/B, corrects that proportion using the requested adjustment method, and reports the *p*-value as less than ("<") the resulting adjusted value.

Value

The print method returns the object, invisibly.

The plot and barplot methods return the values at the midpoint of each bar.

The summary method returns an object of class "summary.bootMtrPairs" which is a list with members:

t0	vector of values calculated by msd or pdchisq
labels	character vector of labels, by default taken from x
probs	vector of probabilities supplied and used for quantiles
critical.values	matrix of quantiles. Each row corresponds to a probability in probs and each column to an individual data point.
pvals	*p*-values estimated as the observed proportion of simulated values exceeding the calculated values t0.
p.adjust	Character value containing the name of the p-value adjustment method used.
p.adj	*p*-values adjusted using the given *p*-value adjustment method specified by p.adjust.
B	Number of bootstrap replicates used.
method	The sampling method used by the parametric bootstrap.
stat	The statistic subjected to bootstrapping. Either "MSD" or "PWchi2" for msd or pdchisq, respectively.
t	If keep == TRUE, the individual bootstrap replicates generated by msd or pdchisq. t is set to NA if keep == FALSE

.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

See Also

msd, qmsd, pdchisq

Examples

```
## Not run:
data(Pb)
pwch.Pb<-pdchisq(Pb$value, Pb$u)  # Uses individual standard uncertainties

set.seed(1023)
boot.Pb <- boot.mtr.pairwise(pwch.Pb)
summary(boot.Pb)

# The default summary gives individual observation p-values. To
# avoid over-interpretation for the study as a whole,
# apply a sensible p-value adjustment:
summary(boot.Pb, p.adjust="holm")

plot(boot.Pb, crit=TRUE)


## End(Not run)
```

---

boxplot.mandel.kh *Box plot of Mandel's h or k statistics*

Description

Produces a box plot of Mandel's statistics, with optional outlier labels and indicator lines for unusual values.

Usage

```
## S3 method for class 'mandel.kh'
boxplot(x, probs=c(0.95, 0.99), main,
xlab=attr(x, "grouped.by"), ylab=attr(x, "mandel.type"),
separators=FALSE, zero.line=TRUE, ylim,  p.adjust="none",
frame.plot = TRUE, horizontal=FALSE, at,
... ,
col.ind=1, lty.ind=c(2,1), lwd.ind=1,
col.sep="lightgrey", lwd.sep=1, lty.sep=1,
lwd.zero=1, col.zero=1, lty.zero=1,
outlier.labels=row.names(x), cex.lab=0.7, col.lab=1,
adj=NULL, pos=NULL, srt=0 )
```

Arguments

x	An object of class "mandel.kh"
probs	Indicator lines are drawn for these probabilities. Note that probs is interpreted as specifying two-tailed probabilities for Mandel's h and one-sided (upper tail) probabilities for Mandel's k.

main	a main title for the plot. If missing, the default is paste(deparse(substitute(x)), " - Mandel's", attr(x, "mandel.type"), if(attr(x, "mandel.method") == "robust") "(Robust variant)")
xlab	a label for the x axis; defaults to the grouped.by attribute for x.
ylab	a label for the x axis; defaults to the mandel.type attribute for x.
separators	Logical; if TRUE, separator lines are drawn between groups of values.
zero.line	logical; if TRUE a horizontal line is drawn at zero.
ylim	the y limits of the plot. For Mandel's k, the default lower limit for y is zero.
p.adjust	Correction method for probabilities. If not "none", passed to p.adjust prior to calculating indicator lines. Usually, indicator lines are drawn without correction (that is, with p.adjust="none"); specifying a p-value correction effectively turns the Mandel's statistics into single outlier tests.
frame.plot	Logical; If TRUE a box is drawn around the plot.
horizontal	if TRUE boxes are plotted horizontally and separators, indicators etc adjusted accordingly.
at	numeric vector giving the locations where the boxplots should be drawn; defaults to 1:n where n is the number of boxes.
...	Other (usually graphical) parameters passed to barplot. Note that some parameters appear after ... to prevent spurious argument matching inside barplot.default.
col.ind, lty.ind, lwd.ind	Graphical parameters used for the indicator lines, recyckled to length(probs). For attr(x, "mandel.type")=="h" the graphical parameters are applied to negative as well as positive indicator lines, applied outwards from zero.
col.sep, lwd.sep, lty.sep	Graphical parameters used for the separator lines.
lwd.zero, col.zero, lty.zero	Graphical parameters used for the zero line.
outlier.labels	Either a logical indicating whether outliers should be labelled or a character vector of length nrow(x) giving labels. Defaults to row.names(x).
cex.lab, col.lab	Character size and colour for outlier labels, passed to text as col and cex respectively.
adj, pos	Position of outlier labels relative to outliers; passed to text.
srt	Label rotation, in degrees, for outlier labels; passed to text.

Details

This plot produces a box plot (using boxplot.default) of the variables in an object of class "mandel.kh".

If labels are specified for outliers (the default), outliers are first located based on the locations given by boxplot.default. WARNING: ties may be mislabelled, as the label allocated will be the _first_ point at that location.

Indicator lines are, if requested, drawn as for plot.mandel.kh.

Vertical separators are drawn at midpoints of at. If

Value

boxplot.mandel.kh returns the box plot statistics returned by boxplot, invisibly.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

See Also

boxplot for box plot arguments, and text for outlier label location, colour and rotation. mandel.h, mandel.k, mandel.kh, pmandelh, pmandelk for probabilities, quantiles etc.

See plot.mandel.kh for the 'classic' Mandel plot.

Examples

```
data(RMstudy)

h <- with(RMstudy, mandel.h(RMstudy[2:9], g=Lab))
boxplot(h, las=2)
 #Recall that for normally distributed data mandel's h should
 #have the same dispersion (and IQR) for all groups. But outliers adversely
 #affect the estimate of dispersion, so the interquartile ranges differ.
 #The same effect also accounts for the many boxplot outliers visible
 #inside the classical Mandel indicator lines; the indicators also
 #assume normality.

#with separators:
boxplot(h, las=2, separators=TRUE)

#With different labels and label colours:
boxplot(h, las=2, outlier.labels=paste(1:nrow(h)), col.lab=1:5)

#... and a horizontal variant (note use of pos to change label positions)
par(omd=c(0.1,1,0,1)) #to make room for axis labels
boxplot(h, las=1, separators=TRUE, horizontal=TRUE, pos=1)
```

---

buildCor *Functions to build correlation and covariance matrices.*

Description

Functions to build and update correlation and covariance matrices using a compact specification of off-diagonal terms.

Usage

```
buildCor(n, cors)

updateCor(cor, cors, cor.names)

buildCov(s, covs, vars = s^2, cors, cov.names)

updateCov(cov, covs, cors, cov.names)
```

Arguments

n	scalar: number of rows/colums required in correlation matrix.
cors, covs	3-column matrix or data frame specification of individual correlation or covariance terms. Can also be a vector of length 3. See Details.
s, vars	vector of standard deviations or variances respectively. One of s or vars must be present.
cor.names, cov.names	vectors of names for the rows and columns of the returned matrix. cov.names defaults to names(s) if s is named.
cor, cov	correlation or covariance matrix requiring amendment.

Details

For buildCor, the size of the returned correlation matrix is set using n; an n by n correlation matrix is returned. For buildCov the size is set to length(s) by length(s).

Each row of cors specifies a correlation term r_{ij} in the form (i, j, r_{ij}). That is, the first two columns give the row and column in the desired correlation matrix, and the third gives the relevant correlation coefficient. On constructing or updating the correlation matrix, r_{ij} is set equal to r_{ji}, so it is only necessary to specify one of r_{ij} or r_{ji}.

covs specifies covariance terms in the same way except that the third column of covs must be a covariance.

If either cors or covs is a vector of length 3, it is coerced to a matrix of three columns.

If cor.names or cov.names are present, the matrix returned has dimnames set to the names supplied.

All four functions test for positive definite return values and generate a warning if not positive definite.

Value

A square symmetric correlation or covariance matrix.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None.

See Also

None.

Examples

```
#Duplicate correlation for example for uncert()
buildCor(4, cors=c(3, 4, 0.5))


#Multiple correlations
r<-buildCor(3, cors=rbind( c(1,2,0.5), c(2,3,0.25) ) )
r

updateCor(r, cors=c(1,3,0.13)) #perhaps more realistic

buildCov(1:3, cors=rbind( c(1,2,0.5), c(2,3,0.25),c(1,3,0.13) ) )
```

---

chromium *Chromium data for two different materials included in an interlaboratory study*

Description

Chromium data for two different materials included in an interlaboratory study intended to provide data for certification of a reference material.

Usage

```
data("chromium")
```

Format

A data frame with 28 observations on the following 2 variables.

`QC` Chromium concentrations (ug/kg) reported on a material used as a quality control material

`RM` Chromium concentrations (ug/kg) reported on a candidate reference material material used as a quality control material

Details

Chromium data for two different materials included in an interlaboratory study intended to provide data for certification of a crab tissue reference material. The study included a previously certified reference material (near end of stock) to serve as a quality control (QC) check. Laboratories were asked to report five replicate measurements on the candidate reference material and three for the QC material. Each row in the data set corresponds to the mean of replicate results reported by each laboratory.

Inspection of the data suggests that one laboratory interchanged or mislabelled the test materials; this is hard to see in univariate plots but relatively easy to see in a Youden plot (a type of pairwise scatter plot - see `youden.plot`).

Source

Private communication - Pending publication

Examples

```
data(chromium)
yplot(chromium)
```

`contribs` *Extract contributions from an 'uncert' object.*

Description

Extracts the individual nonzero contributions to the combined uncertainty in an 'uncert' object.

Usage

```
contribs(object, scope, as.sd = FALSE, keep.sign = TRUE,
simplify = TRUE, expand.dot=TRUE)
```

Arguments

`object`	An object of class `uncert` returned by `uncert` or `uncertMC`
`scope`	An expression, one-sided formula or character vector describing the particular variables for which contributions are desired. If missing, contributions for all variables are returned.
`as.sd`	logical; controls whether values are returned in the form of standard uncertainties or variance contributions. See Details.
`keep.sign`	logical; controls whether the sign of the cobntributions is appended to the return value when `as.sd=TRUE`. See Details.
`simplify`	logical. If `simplify=FALSE` the contribution matrix itself is returned. If `simplify=TRUE`, only the requested (by `scope`) nonzero elements of the contribution matrix are returned, as a vector. See Details for the treatment of off-diagonal terms
`expand.dot`	logical; if `TRUE`, '.' in a formula `scope` is expanded to all contributions including pairwise contributions. If `FALSE`, the dot operator implies only the single-variable terms. See Details.

Details

contribs calculates the contribution matrix C where $C_{i,j} = (c_i u_i)(c_j u_j) r_{i,j}$. In general, these values are possibly negative (co)variance contributions to the variance (squared standard uncertainty) in y. In GUM notation ('the GUM' is JCGM 100 (2008) - see references), the diagonal elements of C are squared standard uncertainties in y. The form of the return value depends on simplify, as.sd and keep.sign.

If as.sd is FALSE (the default), contributions C_{ij} are returned unchanged. For the diagonal elements of C (contributions for individual individual terms), this form corresponds to squared standard uncertainties $u_i^2(y)$ in GUM notation.

If as.sd=TRUE, the magnitude of the value returned is $\sqrt{|C_{ij}|}$. For the diagonal elements of C this corresponds to standard uncertainties $u[i]y$ in GUM notation.

If as.sd=TRUE, keep.sign controls whether the values are signed or returned as absolute values. If keep.sign=TRUE, the value returned is $sign(C_{ij}\sqrt{|C_{ij}|}sign(C[i,j]sqrt(abs(C[i,j]))$. If false, the absolute value is returned. Note that the sign is returned solely to indicate the direction of the original contribution. keep.sign has no effect if as.sd=FALSE.

If simplify=FALSE (the default), the requested elements of the contribution matrix C are returned as a matrix. If simplify=FALSE, the return value is a vector containing only those terms with nonzero values in the associated correlation matrix. The threshold for deciding a correlation is nonzero is that its magnitude is greater than 2*.Machine$double.eps.

Off-diagonal terms for the same pair of variables are summed, that is, for the pair $(C_{ij}, C_{ji}), j \neq i$ the (single) value returned is $C_{i,j} + C_{j,i} = 2C_{i,j}$.

The contributions returned can be limited to a chosen subset using scope; only the terms involving variables included in scope are returned. scope can be an expression, formula or character vector of variable names. If an expression or formula, only those contributions involving variables in the expression or formula are returned.

Any variable names in scope which are not present in row.names(object$budget) are silently ignored except for the formula specification which will return an error.

If simplify=FALSE, the matrix returned always contains all contributions involving individual variables in scope. If simplify=TRUE, however, specifying scope as a formula provides additional control over the returned contributions:

If a formula, scope accepts the usual model formula operators '.', '+', '-', '*' and '^', but the interpretation is not quite identical to lm.

First, if present, '.' is taken by default as 'all contributions', implying all single terms and all pairwise terms (like '.^2)' in other formula specifications). This can be disabled by specifying expand.dot=FALSE.

The negation operator '-' removes terms, but removing a single variable also removes any associated covariance contributions. For example, scope=~.-A is expanded to all single and pairwise contributions to the uncertanty budget that do not involve A.

Interaction-like terms of the form A:B are interpreted as indicating the *total* off-diagonal contribution, that is, A:B is equivalent to B:A and the associated value returned is based on $C_{i,j} + C_{j,i}$.

Cross-terms like ~A*B are supported and expand, as usual, to ~A+B+A:B.

Unlike the two other scope specifications, single terms in the formula do *not* automatically imply off-diagonal terms; A+B will not return the off-diagonal contribution for A and B. Use A*B or

`(A+B)^2` etc. to get off-diagonal contributions. Cross-terms of order above two are ignored so `A*B*C` safely returns only the set of individual and pairwise terms, but it is perhaps more precise to use `(A+B+C)^2`.

`I()` and other operators or functions are not supported.

Value

A named vector or matrix of contributions. Names for off-diagonal contributions in the vector format are constructed from the names of the two contributing variables.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

JCGM 100 (2008) *Evaluation of measurement data - Guide to the expression of uncertainty in measurement*. doi:10.59161/JCGM1002008E. (JCGM 100:2008 is a public domain copy of ISO/IEC *Guide to the expression of uncertainty in measurement* (1995)).

See Also

uncert-class, uncert.

Examples

```
#Example with negative correlation
x <- list(a=1, b=3, c=2, d=11)
u <- lapply(x, function(x) x/10)
u.cor<-diag(1,4)
u.cor[3,4]<-u.cor[4,3]<- -0.5
u.form.c<-uncert(~a+b*2+c*3+d/2, x, u, method="NUM", cor=u.cor)

contribs(u.form.c, simplify=FALSE)
contribs(u.form.c)
contribs(u.form.c, as.sd=TRUE)
contribs(u.form.c, as.sd=TRUE, keep.sign=FALSE)

contribs(u.form.c, scope=c("a", "c", "d") )

#Effects of formula specification for scope:
contribs(u.form.c, ~.)           #All contributions
contribs(u.form.c, ~(a+b+c+d)^2) #same as ~.
contribs(u.form.c, ~a+b+c+d )    #single-variable contributions only
contribs(u.form.c, ~., expand.dot=FALSE )    # as ~a+b+c+d
contribs(u.form.c, ~.-d)         #Drops d and c:d
contribs(u.form.c, ~.-c:d)
contribs(u.form.c, ~c+d)
contribs(u.form.c, ~c*d)
```

`cov.dellipse` *Constructs a covariance and location object for use in plotting data ellipses.*

Description

Constructs a covariance matrix and associated location using a variety of (possibly robust) estimators. The returned object is suitable for use by `plot.d.ellipse`.

Usage

```
cov.dellipse(x, y = NULL, cov.method = c("spearman", "kendall", "pearson",
                          "MCD", "OGK", "GK", "gk", "rgk", "mcd", "mve"),
                          scalefn = NULL, locfn = NULL, cov.control = list())
```

Arguments

`x`	An R `numeric` object. Can be a vector (in which case `y` must be specified and of the same length) or a two-column numeric matrix.
`y`	A numeric vector of the same length as `x`. It is an error to provide `y` in addition to a two-column matrix for `x`.
`cov.method`	A character value specifying the covariance method used.
`scalefn`	A function that computes univariate scale and (optionally) location estimates from a numeric vector. If provided, `scalefn()` should return a single numeric value containing a scale (standard deviation) estimate. For many covariance methods this can be a simple scale estimator. For cov.method "GK", scalefn must accept an additional argument `mu.too`. When mu.too is true, `scalefn()` should return a numeric vector of length 2 containing location and scale estimates. See `scaleTau2`, `s_Qn`,`s_mad`, or `s_IQR` for examples to be used as `scalefn` argument.
`locfn`	A function that computes univariate location estimates from a numeric vector. If used, `locfn()` should return a single numeric value containing a location (mean) estimate.
`cov.control`	A named list of arguments passed to the covariance calculation used. Note that this can override `scalefn` and `locfn`; see below for details.

Details

`cov.dellipse` is a wrapper for a range of covariance estimation methods found in various packages. Its operation and defaults depend on the particular covariance estimator specified by `cov.method`. Details for each are as follows.

`spearman`, `kendall` By default, the median and mad are used as location and scale respectively, and the covariance is calculated from the product of scale estimates and the Spearman rank correlation or Kendall's tau respectively. If either `scalefn` or `locfn`is supplied, `scalefn` is used for scale estimation and `locfn` for location. For both `spearman` and `kendall`, `scalefn` is only used as a scale estimator and need not take a `mu.too` argument.

`pearson` By default, the mean and sd are used as location and scale respectively, and the covariance is calculated from the product of scale estimates and the Pearson correlation. If either `scalefn` or `locfn`is supplied, `scalefn` is used for scale estimation and `locfn` for location, making it possible (if not very sensible) to use a combination of robust scale or location functions with the Pearson correlation coefficient. For this case, `scalefn` is only used as a scale estimator and need not take a `mu.too` argument.

`MCD`, `mcd` Both compute the Minimum Covariance Determinant (MCD) estimator, a robust multivariate location and scale estimate with a high breakdown point, via the 'Fast MCD' or 'Deterministic MCD' ("DetMcd") algorithm. `"MCD"` uses the implementation `covMcd` in the robustbase package; `"mcd"` uses `cov.mcd` in the MASS package. Neither require or use `scalefn` or `locfn`. Note that these MCD implementations differ appreciably for small samples (at least to n=60). `MCD` includes consistency and finite sample correction whereas `mcd` apparently does not apply a finite sample correction. As a result, the `mcd` scales can be considerably smaller for modest data set sizes.

`OGK` Computes the orthogonalized pairwise covariance matrix estimate described by Maronna and Zamar (2002), as implemented by the `covOGK` in the robustbase package. By default, scale and location use `scaleTau2` from robustbase. Alternatives can be specified either by providing **both** `scalefn` and `locfn` or by including an argument `sigmamu` in `cov.control`, which is passed to `covOGK`. See `covOGK` for a description of `sigmamu`. If `sigmamu` is not present in `cov.control` and both `scalefn` and `locfn` are provided, scale and location are constructed from `scalefn` and `locfn`. If only one of these is provided, a warning is issued and `[robustbase]{scaleTau2}` is used.

`GK` Computes a simple pairwise covariance estimate suggested by Gnanadesikan and Kettenring (1972), as implemented by the `covGK` in the robustbase package. By default, scale and location use `scaleTau2` from robustbase. Alternatives can be specified either by providing `scalefn` and `locfn` or by including an argument `scalefn` in `cov.control`, which is passed to `covGK`. See `covGK` for a description of `scalefn`. If `scalefn` is not present in `cov.control`, scale and location are constructed from `scalefn` and `locfn`. If `locfn` is omitted, `scalefn` is used if it takes an argument `mu.too` and the median is used otherwise.

`gk` As `GK`, except that the variables are scaled to unit (robust) sd (using `scalefn`) before calculating the covariance (which is then rescaled). This can prevent large scale differences from masking outliers in a variable with small scale.

`rgk` Implements Gnanadesikan and Kettenring's second covariance estimate based on scaled variables (Z_1, Z_2) and a robust correlation ρ^* calculated as

$$\rho^* = (\hat{\sigma}_+^{*2} - \hat{\sigma}_-^{*2})/(\hat{\sigma}_+^{*2} - \hat{\sigma}_-^{*2})$$

where $\hat{\sigma}_+^{*2}$ and $\hat{\sigma}_-^{*2}$ are robust variances of $(Z_1 + Z_2)$ and $(Z_1 - Z_2)$ respectively, calculated using `scalefn`. The advantage over `"gk"` and `"GK"` is that the correlation coefficient is guaranteed to be in $[-1, 1]$, making for a positive definite covariance matrix. Scaling also helps prevent large scale differences from masking outliers in a variable with small scale.

`mve` Uses uses `cov.mve` in the MASS package, which is based on the location and covariance matrix for a minimum volume ellipsoid. The method neither requires nor uses `scalefn` or `locfn`.

Value

An object of class `cov.dellipse`, which is a list with (at least) components

method Character string describing method; identical to `cov.method`

cov 2x2 covariance matrix

cor 2x2 correlation matrix

center vector (length 2) specifying centre of ellipse

scale vector, length 2, specifying scale estimates for each variable

n.obs number of points (rows) used in the covariance estimate

This list is intended to be consistent with that returned by `cov.wt`.

Author(s)

Stephen L R Ellison

References

Maronna, R.A. and Zamar, R.H. (2002) Robust estimates of location and dispersion of high-dimensional datasets; Technometrics **44(4)**, 307-317.

Gnanadesikan, R. and John R. Kettenring (1972) Robust estimates, residuals, and outlier detection with multiresponse data. Biometrics **28**, 81-124

See Also

`cov.rob` in MASS, `covMcd`, `covOGK` and `covGK` in robustbase.

Examples

```
data(potassium)
cov.dellipse(potassium) #Defaults to Spearman rank correlation

#With different method
cov.dellipse(potassium, cov.method="OGK")

#Same as above but specifying control parameters
library(robustbase) #For scaleTau2
cov.dellipse(potassium, cov.method="OGK", cov.control=list(sigmamu=scaleTau2))

#With individually specified (mad) scale
cov.dellipse(potassium, cov.method="GK", scalefn=mad)
#Defaults to median for location because mad()
#does not accept a mu.too argument

cov.dellipse(potassium, cov.method="GK", scalefn=scaleTau2)
#Defaults to specified scalefn for location because scaleTau2
#accepts mu.too=TRUE
```

cplot *Consistency plot for Key Comparisons*

Description

Produces a consistency plot for typical metrology comparison data.

Usage

```
cplot(x,u,labels=names(x), p.adjust.method="holm", ordered=TRUE,
      breaks=c(0,0.001,0.01, 0.05, 0.1,1),
      col=terrain.colors(length(breaks)-1), log.p=FALSE,
      main=paste("Consistency map -", deparse(substitute(x))),
      subtitle=NULL, key=FALSE,
      key.width=2.54, key.height=0.6,...)
```

Arguments

x	Vector of reported values
u	Vector of length length(x) of standard uncertainties
labels	Vector of of length length(x) labels for x-axis marks.
p.adjust.method	p-value adjustment method; passed to p.adjust.
ordered	If TRUE (the default) observations are arranged in ascending order of x.
breaks	Vector of breaks; passed to image.
col	Vector of colours of length length(breaks)-1. Passed to image.
log.p	If TRUE, the plot shows -log_10(p).
main, subtitle	Main and subtitle for plot.
key	If TRUE a key is added to the plot.
key.width, key.height	Width and height of key, if plotted. See details for specification.
...	Graphical parameters passed to image. If present, cex.axis is passed to axis for main figure axes only.

Details

Calculates the (square, symmetric matrix of) optionally adjusted p-values for a two-tailed z-test of |x[i]-x[j]|/sqrt(u[i]^2+u[j]^2) against zero and plots the p-values as an image.

p.adjust is called prior to plotting to correct for multiple comparisons. To suppress adjustment, set p.adjust.method="none".

key.height is a fraction of the figure region height. key.width is the width of the key area in cm, unless under 1, in which case it is interpreted as a fraction of the plot region width.

If log.p is TRUE and subtitle NULL, a subtitle indicating the use of log.p is added to the plot,

Value

Invisibly returns a matrix of pairwise test p-values or, if `log.p==TRUE`, matrix of `-log_10(p)`.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

See Also

`p.adjust`, `image`.

Examples

```
data(Pb)
cplot(Pb$value, Pb$u, key=TRUE)
```

`data.ellipse` *Construct data ellipses suitable for use with Youden plots.*

Description

Constructs and optionally plots a set of probability ellipses for a bivariate normal distribution with defined centre and covariance.

Usage

```
data.ellipse(cov, probs = 0.95, plot = TRUE, npoints = 100, ...)

## S3 method for class 'd.ellipse'
summary(object, ...)

## S3 method for class 'd.ellipse'
print(x,  ...)
```

Arguments

`cov`	Covariance and location object of class `cov.dellipse` as returned by `cov.dellipse()`
`probs`	A vector of probabilities at which ellipses will be constructed.
`plot`	Logical specifying whether the ellipses constructed will additionally be plotted. If `TRUE`, the result is plotted using `plot.d.ellipse()`
`npoints`	Integer number of points for each quadrant of the ellipses returned.
`object`	Object of class `d.ellipse` (for summary method).
`x`	Object of class `d.ellipse` (for print method.
`...`	Arguments passed to other methods, particularly `plot.d.ellipse` and, for `print`, `format` and `print.default`.

Details

`data.ellipse` constructs and returns one set of x, y coordinates for each value of `probs`, in a form that can be passed directly to `polygon`.

Ellipses are constructed from the upper `probs` quantile of the F distribution using

`T = sqrt( 2 * (n - 1) * qf(probs, 2, n - 1)/(n - 2))`

where `n` is the number of pairs used in forming the covariance matrix. If the number of points is missing or NA, `Inf` is substituted.

Summary and print methods are provided. The summary method returns a list with the same names as class `d.ellipse`, each containing a default summary of the respective member of the `d.ellipse` object. The print method returns its argument invisibly.

Value

An object of class `d.ellipse`, consisting of:

`ellipses`	A named list of ellipsoids named for each probability in `probs`. Each is a $4 * npoints \times 2$ matrix suitable for passing direct to `polygon`.
`probs`	Numeric vector of probabilities as supplied by `probs`
`cov`	Covariance object of class `cov.dellipse` as provided in `cov`

Author(s)

S L R Ellison

References

ISO 13528:2005 Statistical methods for use in proficiency testing by interlaboratory comparisons. (2005) International organization for Standardizaton, Geneva

Jackson, J. E. (1956) *Quality control methods for two related variables*. Industrial Quality Control, Vol. 7, pp. 2-6

Jackson, J. E. (1959) *Control Methods for Several Related Variables*. Technometrics, Vol. 1, pp. 359-377

See Also

`cov.dellipse`, `plot.d.ellipse`, `polygon`

Examples

```
data(chromium)
cov.Cr <- cov.dellipse(chromium)
dellipse.Cr <- data.ellipse(cov.Cr, plot=FALSE)
summary(dellipse.Cr)
```

derSimonian-Laird *derSimonian-Laird estimator*

Description

Calculates derSimonian-Laird estimate of location, with standard error, assuming a random-effects model

Usage

```
dsl(x, ..., na.rm = FALSE)

## Default S3 method:
dsl(x, s, n = NULL, groups = NULL, ..., na.rm = FALSE)
```

Arguments

x	numeric vector of mean values for groups, or (if groups is given) of individual observations
s	numeric vector of length length(x) of standard deviations or standard uncertainties associated with the values x.
n	integer giving the number of observations in each group. May be a vector of length length(x). If n is NULL, s is interpreted as a vector of standard uncertainties or standard errors. n is recycled to length(x)
groups	factor, or vetor which can be coerced to factor, of groups. If present, x is interpreted as a vector of individual observations and s and n ignored, if present, with a warning.
na.rm	logical: if TRUE, NA values are removed before processing.
...	Further parameters passed to other methods.

Details

dsl implements the derSimonian-Laird random-effects estimate of location, using the implementation described by Jackson (2010).

The estimator assumes a model of the form

$$x_i = \mu + b_i + e_i$$

in which b_i is drawn from $N(0, \tau^2)$ and e_i is drawn from $N(0, \sigma_i^2)$.

The estimator forms a direct calculation of τ, and uses this to form revised estimates of standard error $\sqrt{s_i^2 + \tau^2}$ in x, calculates weights as the inverse of these and in turn calculates a weighted mean, allowing for any calculated excess variance τ^2.

This implementation permits input in the form of:

- means x and standard errors s, in which case neither n nor groups are supplied;

- means `x`, standard deviations `s` and group size(s) `n`, standard errors then being calculated as `s/sqrt(n)`
- individual observations `x` with a groupinf factor `groups`, in which case standard errors are calculated from the groups using `tapply`.

Value

A loc.est object; see loc.est for details. In the returned object, individual values `xi` are always input means (calculated from groups and `n` as necessary); `method.details` is returned as a list containing:

mu The estimated location.

s The standard error in the location.

tau The excess variance (as a standard deviation).

Author(s)

S L R Ellison <s.ellison@lgcgroup.com>

References

Jackson et al. (2010) J Stat Plan Inf 140, 961-970

See Also

loc.est-class

Examples

```
#PCB measurements in a sediment from Key Comparison CCQM-K25
#s are reported standard uncertainties
pcb105 <- data.frame(x=c(10.21, 10.9, 10.94, 10.58, 10.81, 9.62, 10.8),
             s=c(0.381, 0.250, 0.130, 0.410, 0.445, 0.196, 0.093))

with( pcb105, dsl(x, s) )
```

---

drop1.uncert *Single variable deletions from the uncertainty budget.*

Description

`drop1` calculates revised combined uncertainty for single variable deletions from an object of class 'uncert'.

Usage

```
## S3 method for class 'uncert'
drop1(object, scope, simplify = TRUE,
which=c("% Change", "var", "u", "var.change", "u.change"), ...)

## S3 method for class 'uncertMC'
drop1(object, scope, simplify = TRUE,
which=c("% Change", "var", "u", "var.change", "u.change"), ...)

#Print and plot methods
## S3 method for class 'drop1.uncert'
print(x, ..., digits=2)

## S3 method for class 'drop1.uncert'
plot(x, ...,
which=c("% Change", "var", "u", "var.change", "u.change"))
```

Arguments

object	An object of class 'uncert' or 'uncertMC'.
scope	character vector, expression or formula containing the list of variables to be dropped. If missing, all variables in object$budget are taken as scope.
simplify	logical. If simplify=TRUE, the return value is simplified to a named vector. If FALSE, all forms available (see which) are returned in a data frame.
which	logical; controls the form of information returned when simplify=TRUE. Possible values are: "var" The modified values of $u(y)^2$. "u" The modified values of $u(y)$. "var.change" The signed changes in $u(y)^2$. "u.change" The signed changes in $u(y)$. "% Change" The percentage change in $u(y)$.
x	An object of class 'drop1.uncert' returned by drop1.uncert.
...	Further objects passed to other functions.
digits	number of digits used to format the output. See the digits argument of format.

Details

By analogy with drop1, drop1.uncert perfoms single variable deletions from the uncertainty budget in object, calculates the resulting uncertainty and returns the results in the form requested by simplify and which.

'Single variable deletion' of a variable x_i is equivalent to setting the uncertainty $u(x_i)$ to zero. Note that this also sets covariance terms involving x_i to zero. drop1.uncert does not support the deletion of single *terms* such as $cov(i, j)$.

In the case of 'uncertMC' objects, drop1 currently requires objectMCx to be present (i.e. uncertMC called with keep.x=TRUE). The uncertMC method does not support correlation.

For which="var.change", which="u.change" and which="% Change" the change on dropping a variable is negative if the uncertainty reduces on removing the variable.

The print method simply prints the output with a header formed from the expr attribute and with '%' appended to the "% Change" column.

The plot method produces a barplot of the chosen data column. A plot for each value in which is produced. Arguments in '...' are passed to barplot. If not already present in '...' a default main title and ylab are used. The expr attribute is shown as marginal text if not NA.

Value

If simplify=FALSE, an object of class 'drop1.uncert', consisting of a data frame with row names corresponding to row.names(object$budget), columns corresponding to all possible values of which in the order "var", "u", "var.change","u.change", "% Change", and an attribute expr containing a copy of the expr value of the 'uncert' object to which drop1.uncert is applied.

If simplify=TRUE, the column of the above data frame corresponding to which is returned as a vector with names row.names(object$budget).

Author(s)

S. L. R. Ellison, <s.ellison@lgcgroup.com>

References

None.

See Also

uncert, uncert-class, format for digits, barplot for available plot parameters.

Examples

```
#Continuing the example from plot.uncert:
require(graphics)

d1<-drop1(u.form.c, simplify=FALSE)
d1

plot(d1)

drop1(u.form.c)         #% change only
```

duewer.plot *Duewer concordance/apparent precision plot*

Description

Produces a Duewer concordance/apparent precision plot, showing relative precision or uncertainty plotted against (relative) deviation from assigned value.

Usage

```
dplot(x, ...)

duewer.plot(x, ...)

## Default S3 method:
duewer.plot(x,s,mu=median(x),sigma=mad(x), s0=median(s), labels=NA,
             radius=1:3, units=c("z","x"),
             main, xlab, ylab, xlim, ylim,
             at.xax=NULL, at.yax=NULL, aspect,
         col.contours="lightgrey", lty.contours=par("lty"), lwd.contours=par("lwd"),
             label.contours=T, format.clab="p=%4.3f",
             cex=par("cex"), cex.label=0.7, pos=3, adj=NULL,
             pos.clab="bottomright", col.clab=col.contours,
             cex.axis=par("cex.axis"), pch=par("pch"), las=par("las"),
             col=par("col"), bg=par("bg"), ...)
```

Arguments

x	Numeric vector of values to be plotted.
s	Numeric vector of standard deviations, standard errors or uncertainties of length length(x) associated with x.
mu	A single location against which to compare x.
sigma	A measure of dispersion against which deviations x-mu can be compared.
s0	A typical, expected or reference value for the standard uncertainties s
labels	An optional vector of point labels of length length(x). labels are coerced to character on use, so may be character, factor etc..
radius	A vector of radii for reference lines in the classic Duewer plot.
units	Controls scaling of the plot. If set to "z", a classic Duewer plot of s/s0 vs. (x-mu)/sigma is produced. If units=="x", the plot is drawn without scaling by sigma or s0.
main	Main title for the plot, passed to title().
xlab, ylab	x- and y-axis labels, passed to title().
xlim, ylim	x- and y-limits for the plot.
at.xax, at.yax	Locations for x- and yaxis tick marks, passed to axis()

aspect	The aspect ratio for the plot, passed to plot.window. This defaults to 1.0 for basis=="radius", giving semicircular countours, and NA otherwise.
col.contours, lty.contours, lwd.contours	Colour, line type and line width for contour lines.
label.contours	Logical, controlling whether countour lines are labelled with approximate probabilities.
format.clab	format string for contour labels; passed to sprintf.
cex	Expansion factor for plotted symbols.
cex.label	Expansion factor for point labels.
pos, adj	Specifies position/adjustment of point labels. Passed to text.
pos.clab	Specification for location of contour labels. Options are '"top"', '"topright"', '"right"', '"bottomright"', '"bottom"', '"bottomleft"', '"left"', '"topleft"'. A vector can be provided to give multiple labels. Contour labels for basis="prob" are placed approximately at the location indicated and adjusted outward appropriately. For basis="radius", "bottomright" and "bottomleft" are as for "right" and "left" but just below the x-axis, and "bottom" is replaced with c("bottomright", "bottomleft").
col.clab	Colour for contour labels.
cex.axis	Expansion factor for axis labels.
las	Axis label orientation, passed to axis.
pch, col, bg	Graphical parameters passed to points.
...	Other parameters passed to plotting functions. Currently unused.

Details

A Duewer plot is a plot of dispersion against location. Classically, this has been applied to multiple observations from laboratories. Locations x are mean results of the form (x-mu)/s and and dispersions s are the associated sd. The principle has also been applied to multiple results for different measurands per laboratory, by calculating z-scores for all observations relative to the assigned value and dispersion for each measurand and then plotting mean and sd of the scores. More recently the plot has been used to summarise reported values and (usually) standard uncertainties in metrology comparisons to allow quick assessment of anomalies within data sets.

The traditional plot includes visual guides in the form of semicircular contours at multiples of (x-mu)/sigma for the x-axis and s/s0 for the y-axis, s0 being a median or other estimate of the typical standard deviation.

Contours are, by default, labelled with probabilities corresponding to quantiles of the normal distribution.

dplot is an alias for duewer.plot.

Value

This function is called for its side effect, which is the production of a plot.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

References

Duewer, D, Probably in Anal. Chem. in about 1990

See Also

`axis` for axis control, `points`, `text` for plotting parameters; `sprintf` for contour label format.

`xs.plot` for a plot of location and scale data with probabilistic confidence regions.

Examples

```
require(metRology)
data(Pb)
Pb

duewer.plot(Pb$value, Pb$u)

duewer.plot(Pb$value, Pb$u, basis="prob", df=5)

#Illustrate contour labelling
duewer.plot(Pb$value, Pb$u, pos.clab="bottom")
```

Extract.ilab *The 'ilab' class.*

Description

Functions for manipulating interlaboratory study objects objects of class 'ilab'.

Usage

```
## S3 method for class 'ilab'
subset(x, subset, drop=FALSE, ...)

## S3 method for class 'ilab'
x[i, j]
```

Arguments

`x`	An object of class 'ilab'
`subset`	logical expression indicating elements or rows to keep: missing values are taken as false.
`drop`	passed on to '[' indexing operator.
`...`	Parameters passed to other functions
`i, j`	elements to extract. May be numeric or logical vectors.

Details

For the subset method, `subset` is an expression evaluated in the frame of `ilab$data` and in the parent environment if objects are not found in `ilab$data`. Note that since `ilab$distrib` and `ilab$distrib.pars` are not in `ilab$data`, any operation on these must be specified in full.

The indexing method `'['` operates on both rows and columns of the object. However, only the `$data` element can be addressed with the `j`; the `distrib` and `distrib.pars` elements are unaffected by `j` and will always be included in the returned object.

Value

An object of class 'ilab' with fewer rows and (if `j` is present) fewer columns.

Warning

Removing the standard columns from 'ilab' objects using `'['` may have unforeseen consequences for other functions; only the print method is likely to operate successfully.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None, yet.

See Also

ilab-class.

Examples

```
data(Pb)
il.pb<-construct.ilab(org=Pb$lab, x=Pb$value, measurand="Pb", item="none",
            u=Pb$u, k=Pb$k, U=Pb$U, title=c("CCQM K30", "Lead in wine"), method=Pb$method)

subset(il.pb, u < 0.03)

il.pb[1:6,]
```

gplot *Grouped plots of type "h"*

Description

gplot is primarily used by plot.mandel.kh to produce the underlying grouped data plot.

Usage

```
gplot(x, main = NULL, xlab = NULL, ylab = deparse(substitute(x)),
ylim = NULL, las = 1, axes = TRUE, cex.axis = 1,
frame.plot = axes, lwd = 1, lty = 1, col = par("col"),
separators = TRUE, col.sep = "lightgrey", lwd.sep = 1,
lty.sep = 1, zero.line = TRUE,
lwd.zero = 1, col.zero = 1, lty.zero = 1,
spacing=NA, ...)
```

Arguments

x	A matrix or data frame to be plotted.
main	Main title for the plot.
xlab, ylab	Labels for x and y axes.
ylim	the y limits of the plot.
las	the style of the axis labels; see par for details.
axes	a logical value indicating whether axes should be drawn on the plot.
cex.axis	The magnification to be used for axis annotation relative to the current setting of 'cex'.
frame.plot	Logical; If TRUE a box is drawn around the plot.
lwd, lty, col	Graphical parameters used for the plotted vertical lines corresponding to each value in x.
separators	Logical; if TRUE, separator lines are drawn between groups of values.
col.sep, lwd.sep, lty.sep	Graphical parameters used for the separator lines.
zero.line	logical; if TRUE a horizontal line is drawn at zero.
lwd.zero, col.zero, lty.zero	Graphical parameters used for the zero line.
...	Other graphical parameters passed to plot.
spacing	Spacing for data within each group, as a fraction of inter-group spacing. Defaults to 0.3 or less.

Details

gplot produces a plot of type="h", with values in x grouped by row and with optional vertical separators between groups. The plotting order (left to right) is in order of stack(as.data.frame(t(x))); each group corresoponds to a row in x.

Because gplot is primarily a supporting function for plot.mandel.kh, it assumes a suitable object will be provided and does minimal checking to ensure an appropriate object class. Error messages may not be very informative.

Value

A numeric vector of mid-points of the groups along the x-axis.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

References

Accuracy (trueness and precision) of measurement methods and results – Part 2: Basic method for the determination of repeatability and reproducibility of a standard measurement method. ISO, Geneva (1994).

See Also

plot.mandel.kh

Examples

```
data(RMstudy)
h <- with(RMstudy, mandel.h(RMstudy[2:9], g=Lab))
gplot(h, las=2)
 #Note the absence of indicator lines, title etc.
 #compared to plot(h)
```

GUM *Propagation of Measurement Uncertainty for Typical Metrology Applications Using the Methods Outlined in the GUM*

Description

A function for propagation of measurement uncertainty for typical metrology applications using the methods from the Joint Committee on Guides in Metrology (JCGM) *Guide to the Expression of Uncertainty in Measurement (GUM)*. This approach approximates the uncertainty of a function of random variables that define a measurement result by computing the uncertainty of the first-order Taylor series for the measurement function. This function also serves as the primary computational tool underlying the GUM uncertainty templates found in the metRology for Microsoft Excel user interface.

Usage

```
GUM(var.name, x.i, u.i, nu.i, measurement.fnc, correlation = diag(length(var.name)),
shared.u.i = var.name, cl = 0.95, cov.factor = "Student's t", sig.digits.U = 2, ...)
```

Arguments

var.name	Character vector of input variable names.
x.i	Vector of input variable values.
u.i	Vector of standard uncertainties (i.e. standard errors) for each input variable value.
nu.i	Degrees of freedom associated with each standard uncertainty.
measurement.fnc	Character string specifying the functional relationship between input variables that defines the output measurement result.
correlation	Matrix giving the correlation between the different input variable values. Default is to assume no correlation between input variable values.
shared.u.i	Character vector giving the relative relationship between the standard uncertainties for each variable value. Groups of variables based on a common shared standard uncertainty will all share the same variable name. The default is to assume all standard uncertainties are assessed independently, resulting a value of shared.u.i that is identical to var.name.
cl	Nominal confidence level to be used to compute the expanded uncertainty of the output measurement result. Default value is 0.95.
cov.factor	Type of coverage factor to be used. The default is to use the value from the Student's t distribution with confidence level specified above and nu.eff effective degrees of freedom.
sig.digits.U	Number of significant digits to be reported in the expanded uncertainty of the measurement result. The measurement result will be rounded to the same number of decimal places.
...	Arguments passed to other functions. Currently unimplemented.

Details

Whenever possible, sensitivity coefficients are obtained analytically using the gradient attribute of the deriv function. In situations where some part of the measurement function is not found in derivative table, sensitivity coefficients are obtained by numeric partial differentiation using the grad function from the package numDeriv.

Value

A list containing the 9 components:

y	Value of the measurement result obtained by evaluating the measurement function at the input variable values.
uc	The combined standard uncertainty of the measurement result, y.

nu.eff	The effective degrees of freedom associated with uc, computed using the Welch-Satterthwaite formula.
cl	The nominal confidence level used to obtain the coverage factor, k.
k	The coverage factor used to control the confidence level associated with the expanded uncertainty of the measurement result.
U	The expanded uncertainty of the measurement result, computed as U=k*uc.
contributions	Relative variance contributed to the standard uncertainty (uc) of the measurement result from each input variable.
sensitivities	Sensitivity coefficient associated with each input variable.
msgs	Error and warning messages that point out potential problems with the inputs to the GUM function or with the interpretation of the function's output.

Author(s)

Hung-kung Liu <hung-kung.liu@nist.gov> and Will Guthrie <will.guthrie@nist.gov>

References

Joint Committee on Guides in Metrology (JCGM), *Evaluation of Measurement Data Guide to the Expression of Uncertainty in Measurement*, http://www.bipm.org/utils/common/documents/jcgm/JCGM_100_2008_E.pdf, 2008.

See Also

GUM.validate a function to assess the statistical performance of GUM uncertainty intervals for the application of interest in terms of average attained coverage probability. uncert for a family of functions focused on the study and comparison of different approaches and numerical options in uncertainty analysis.

Examples

```
## a simple uncertainty analysis for the product of two quantities
GUM(c("x1","x2"),c(2.3,1.1),c(0.030,0.015),c(5,9999),"x1*x2")

## example of the difference in the measurements of two standards, each
## with a standard uncertainty based on a common value drawn from a control chart
## representative of the measurement process made using a check standard that
## is comparable to the two individual standards under study
GUM(c("s1","s2"),c(45.3,46.0),c(0.26,0.26),c(134,134),"s1-s2",shared.u.i=c("s1","s1"))

## compare with results for equivalent, alternative specification of shared.u.i
GUM(c("s1","s2"),c(45.3,46.0),c(0.26,0.26),c(134,134),"s1-s2",shared.u.i=c("s2","s2"))
```

GUM.H.1 *Example H.1 from the* Guide to the Expression of Uncertainty in Measurement

Description

Calibration of an end gauge for length measurement. Notation is based on the presentation in chapter 3 of *Data Modeling for Metrology and Testing in Measurement Science*.

Usage

```
data(GUM.H.1)
```

Format

A list containing 10 components that give the uncertainty budget, measurement function, and other quantities needed to carry out an uncertainty analysis using the methods from the Guide to the Expression of Uncertainty in Measurement:

var.name Character vector giving the name of each input variable included in the analysis.

unit Expression giving the units of each input variable

x.i Vector giving the reported value for each input variable.

u.i Vector giving the standard uncertainty associated with each reported value.

nu.i Vector giving the degrees of freedom associated with each standard uncertainty.

type Character vector indicating the method of evaluation (Type A or Type B) for each standard uncertainty.

distribution Character vector listing the probability distribution assumed to describe each variable value.

measurement.fnc Character string giving the measurement function for the output variable.

correlation Matrix giving the correlations between input variable values.

shared.u.i Character vector describing which standard uncertainties, if any, are based on a common underlying standard uncertainty. A vector that is the same as var.name indicates that all standard uncertainties are based on independent assessments of standard uncertainty. A vector with fewer names than in var.names indicates that one or more variables are derived from a common uncertainty assessment.

Details

Details can be found in the GUM and in Data Modeling for Metrology and Testing in Measurement Science.

Source

Joint Committee on Guides in Metrology (JCGM), *Evaluation of Measurement Data Guide to the Expression of Uncertainty in Measurement*, http://www.bipm.org/utils/common/documents/jcgm/JCGM_100_2008_E.pdf, 2008.

References

Guthrie, W.F. et. al., "Three Statistical Paradigms for Assessment and Interpretation of Measurement Uncertainty" in *Data Modeling for Metrology and Testing in Measurement Science*, F. Pavese and A.B. Forbes, eds., Birkhauser, Boston, 2009.

`GUM.validate` *Monte Carlo Check on the Statistical Performance of GUM Uncertainty Intervals Using Attained Coverage Probability*

Description

A function for assessing the statistical performance of measurement uncertainty intervals for particular metrology applications computed using the methods from the Joint Committee on Guides in Metrology (JCGM) *Guide to the Expression of Uncertainty in Measurement (GUM)*. The validation is carried out using the input values as true values in a simulation that directly checks the attained coverage probability of the uncertainty intervals produced using the `GUM` function.

Usage

```
GUM.validate(var.name, x.i, u.i, nu.i, type, distribution, measurement.fnc,
          correlation = diag(length(var.name)), shared.u.i = var.name, cl = 0.95,
             cov.factor = "Student's t", sig.digits.U = 2)
```

Arguments

`var.name`	Character vector of input variable names.
`x.i`	Vector of input variable values.
`u.i`	Vector of standard uncertainties (i.e. standard errors) for each input variable value.
`nu.i`	Degrees of freedom associated with each standard uncertainty.
`type`	Character vector of values "A" and "B" indicating the methods used to evaluate the standard uncertainty of each input value. Standard uncertainties evaluated using statistical methods are denoted Type A in the GUM, while standard uncertainties evaluated using other means are denoted Type B.
`distribution`	Character vector of probability distributions associated with the potential values taken on by each input variable. The current possible choices are "Normal" (i.e. Gaussian), "Triangular", or "Rectangular" (i.e. Uniform).
`measurement.fnc`	Character string specifying the functional relationship between input variables that defines the output measurement result.
`correlation`	Matrix giving the correlation between the different input variable values. Default is to assume no correlation between input variable values.

shared.u.i	Character vector giving the relative relationship between the standard uncertainties for each variable value. Groups of variables based on a common shared standard uncertainty share will all share the same variable name. The default is to assume all standard uncertainties are assessed independently, resulting a value of shared.u.i that is identical to var.name.
cl	Nominal confidence level to be used to compute the expanded uncertainty of the output measurement result. Default value is 0.95.
cov.factor	Type of coverage factor to be used. The default is to use the value from the Student's t distribution with confidence level specified above and nu.eff effective degrees of freedom.
sig.digits.U	Number of significant digits to be reported in the expanded uncertainty of the measurement result. The measurement result will be rounded to the same number of decimal places.

Details

Currently 1000 simulated sets of uncertainty data are used for the computation of the attained confidence level.

Value

A Monte Carlo assessment of the attained coverage of expanded uncertainty intervals like those produced using the GUM function for the application of interest.

Author(s)

Hung-kung Liu <hung-kung.liu@nist.gov> and Will Guthrie <will.guthrie@nist.gov>

References

Joint Committee on Guides in Metrology (JCGM), *Evaluation of Measurement Data Guide to the Expression of Uncertainty in Measurement*, http://www.bipm.org/utils/common/documents/jcgm/JCGM_100_2008_E.pdf, 2008.

See Also

GUM a function to compute GUM uncertainty intervals for general metrological applications.

Examples

```
## a simple uncertainty analysis for the product of two quantities
GUM.validate(c("x1","x2"), c(2.3,1.1), c(0.030,0.015), c(5,9999),
              c("A","B"),c("Normal","Rectangular"),"x1*x2")
```

ilab-class *The 'ilab' class.*

Description

The 'ilab' class and its constructor function.

Usage

```
construct.ilab(org, item, measurand, x, u, df, k, U, U.lower, U.upper,
           distrib=NULL, distrib.pars=NULL, study=NA, title=NA, p=0.95, ...)
```

Arguments

org	Character vector or factor of organisation names.
item	vector or factor of identifiers for test items. Coerced to factor on storage.
measurand	vector or factor identifying the measurand(s) involved in the study.
x	numeric vector of reported values.
u	numeric vector of reported standard uncertainties or standard errors associated with x.
df	optional numeric vector of degrees of freedom associated with each reported uncertainty.
k	numeric vector of coverage factors. The coverage factor is the factor multiplying u to obtain U.
U	numeric or character vector of expanded uncertainties or confidence interval half-widths. Coerced to numeric but may include a character representation of interval limits; see Details.
U.lower, U.upper	numeric vectors of lower and upper limits for the confidence interval around x, allowing asymmetric intervals. Defaults to U or to the limits specified by U. See Details.
distrib	A character vector of length length(x) or a named list of names of distribution functions associated with u. If a character vector, distrib is recycled to length length(x).
distrib.pars	A named list of lists of parameters describing the distributions associated with u to be passed to the relevant distribution function. If distrib is present but distrib.pars is not, an attempt is made to set defaults based on other parameters; see Details.
study	A character value or vector or a factor identifying different studies or study populations within the data set. Typically used, for example, for identifying participants in global and regional components of a combined study. Recycled to length length(x) if necessary.

title	An optional title for the study. May be a character vector, in which case each element is displayed on a separate line when printed.
p	Confidence level assumed to apply to k. Used only to set a default value for df when distrib indicates a t-distribution and df is unspecified.
...	Other *named* factors or character vectors used to group observations.

Details

If U is a character vector, it may contain character representations of range. Two forms are permitted:

"a-b" Interpreted as limits of a range from a to b. U.lower and U.upper are calculated from these limits and x

"+a[/ -b" (or "-a[/]+b")] U.upper is set to a in "+a", and U.lower is set to b in "-b".

If distrib.pars is missing, an attempt is made to deduce appropriate distribution parameters from x, u, df and distrib. In doing so, the following assumptions and values apply for the respective distributions:

norm mean=x$name, sd=u$name.

unif min=x-sqrt(3)*u, max=x+sqrt(3)*u.

tri min=x-sqrt(6)*u, max=x+sqrt(6)*u, mode=x.

t, t.scaled df=df, mean=x, sd=u.

In addition, if distrib contains "t" or "t.scaled", and df is NA, the corresponding degrees of freedom are chosen based on k and p.

Value

An object of class 'ilab' consisting of:

title	A character value or vector describing the study
subset	A character string describing any subset operation used to form the object.
data	A data frame with columns:

org	Factor of organisations submitting results in the study
item	Factor of test item identifiers.
measurand	Factor of measurands determined for each item
x	numeric vector of reported values.
u	numeric vector of reported standard uncertainties or standard errors associated with x.
df	numeric vector of degrees of freedom associated with each reported uncertainty. Set to NA if not provided.
k	numeric vector of coverage factors. The coverage factor is the factor multiplying u to obtain U.
U	numeric or character vector of expanded uncertainties or confidence interval half-widths. U is coerced to n
U.lower, U.upper	numeric vectors of lower and upper limits for the confidence interval around x.
study	Identifier for study groups (see Arguments above).
...	Other grouping factors (supplied in '...' in construct.ilab) which can be used for sub-categorisation.

distrib	An unnamed list of distribution names.
distrib.pars	An unnamed list of lists of parameters describing the distributions associated with u.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None, yet.

See Also

print.ilab, subset.ilab, plot.ilab

Examples

```
data(Pb)
construct.ilab(org=Pb$lab, x=Pb$value, measurand="Pb", item="none",
                u=Pb$u, k=Pb$k, U=Pb$U, title=c("CCQM K30", "Lead in wine"),
                method=Pb$method)

#Illustrate default for U and automatic distrib.pars
construct.ilab(org=Pb$lab, x=Pb$value, measurand="Pb", item="none",
                u=Pb$u, k=Pb$k, distrib="norm")

construct.ilab(org=Pb$lab, x=Pb$value, measurand="Pb", item="none",
                u=Pb$u, k=Pb$k, distrib="t.scaled")
```

---

kplot *Dot-and-bar plot for Key Comparisons*

Description

Produces a dot plot of typical metrology comparison data (value/uncertainty) with error bars, assigned value and uncertainty and optional percentage deviation axis or marginal density

Usage

```
kplot(x, ...)

## Default S3 method:
kplot(x,U=NULL, labels=names(x),  assigned=NULL, U.assigned=NULL,
U.lo=U, U.hi=U, k=2, strata=NULL,
do.percent=!is.null(assigned) && !do.pdf,
ordered=TRUE, order.strata=levels(strata),
xlim=c(0.5, length(x)+0.5), ylim,
main=NULL, xlab=NULL, ylab=NULL,
axis.main=2, axis.pct=4, at=1:length(x), at.main=NULL,
cex.axis=0.8, las=2, las.pct=1, ylab.line=2.5,
ylab.line.pct=2.1, ci.width=0.03, col.ci=par("fg"),
```

```
lty.ci=par("lty"), lwd.ci=par("lwd"), pch=21,
col=par("fg"), bg="white", add.outliers=FALSE,
outlier.offset=0.2, mar=NULL, box=TRUE,
do.pdf=FALSE, do.individual.pdf=do.pdf,
col.pdf=par("fg"), lwd.pdf=1, lty.pdf=1,
do.total.pdf=TRUE, col.total.pdf=col.pdf[1],
lwd.total.pdf=2, lty.total.pdf=1, n.pdf=200,
pdf.layout=c(4,1), pdf.scale=0.7, pdf.offset=0.05,
xlim.pdf, pdf.axis=FALSE, las.pdf=0,
mgp.pdf=c(3,0.5,0), ...)

## S3 method for class 'ilab'
kplot(x, ...)

kpoints(x,U=NULL, labels=names(x), U.lo=U, U.hi=U, k=2,
strata=NULL, ordered=TRUE, order.strata=levels(strata),
at=1:length(x), ci.width=0.03, col.ci=par("fg"),
lty.ci=par("lty"), lwd.ci=par("lwd"), pch=21,
col=par("fg"), bg="white", add.outliers=FALSE,
outlier.offset=0.2, ...)
```

Arguments

x	an R object. For the default method, a vector of reported values. For the ilab method, an object of class 'ilab'
U	Vector of length `length(x)` of expanded uncertainties
labels	Vector of of length `length(x)` labels for x-axis marks.
assigned	Assigned value for the comparison. Plotted as a horizontal line on the plot.
U.assigned	Expanded uncertainty for the assigned value
U.lo, U.hi	Vectors of of length `length(x)` of lower and upper limits for the uncertainty intervals around the reported values, to allow asymmetric intervals. Both default to U.
k	Coverage factor originally used in calculating U. Required only if `do.pdf=TRUE`, as k is used to calculate standard uncertainties from U. k can be a scalar (recycled to length `length(x)` if necessary) or a vector of length of length `length(x)`.
strata	A Factor identifying subsets of the data. Currently not implemented.
do.percent	Logical indicating whether percentage deviation should be plotted as a secondary axis. Defaults to `TRUE` if an assigned value is provided and `FALSE` if there is no assigned value or if a marginal density is required via `do.pdf=TRUE`.
ordered	If `TRUE`, values are plotted in ascending order.
order.strata	Character vector showing the order of plotting for strata. Currently not implemented.
xlim, ylim	Plot limits as in `plot.default`.
main, xlab, ylab	Titles; see `?title` for details.

axis.main, axis.pct
: Integers specifying on which side of the plot the relevant axis is to be drawn, passed to axis as side. The axis is placed as for follows: 1=below, 2=left, 3=above and 4=right. The main axis (axis.main) is provided in the same units as x; the percentage axis (axis.pct) shows corresponding percentage deviation.

at
: Vector of x-axis locations for the data points, x-axis tick marks and labels. Defaults to 1:length(x).

at.main
: The points at which tick-marks are to be drawn on the main (y) axis. Passed to axis via at.

cex.axis
: The magnification to be used for axis annotation relative to the current setting of 'cex'. Passed to axis.

las, las.pct
: Integers defining x- and y axis and percentage axis label orientation; see par(las).

ylab.line, ylab.line.pct
: Margin lines for main and percentage axis titles. Passed to axis.

ci.width
: Width of error bar terminators, passed to arrows.

col.ci, lty.ci, lwd.ci
: Graphical parameters for the error bars; passed to arrows.

pch, col, bg
: Graphical parameters for data points, passed to points. bg specifies the fill colour for pch from 21 to 25.

add.outliers
: If TRUE, points outside ylim are indicated as an arrow indicating the direction in which the omitted points lie, with a text label showing the reported value.

outlier.offset
: X-offset (in x-axis units) specifying lateral location of outlier tet labels relative to x-axis location of the outlier indicator.

mar
: Plot margins as in par(mar). The default varies depending on do.pct and do.pdf.

box
: If TRUE, a box is drawn round the plot region.

do.pdf
: If TRUE, a marginal density and/or individual densities for the individual reported values are plotted based on the reported values x and standard uncertainties calculated as U/k.

do.individual.pdf
: Logical controlling whether the individual densities are plotted as well as/instead of the combined density.

col.pdf, lwd.pdf, lty.pdf
: Graphical parameters controlling the appearance of the marginal density plot(s). Vectors are permitted, allowing different styles for each individual pdf.

do.total.pdf
: Logical controlling whether the sum of individual densities is plotted.

col.total.pdf, lwd.total.pdf, lty.total.pdf
: Graphical parameters controlling the appearance of the marginal density plot for the combined density.

n.pdf
: Number of points used to construct the marginal density.

pdf.layout
: Vector of length 2 specifying the relative sizes of the main plot and marginal density plot. See ?layout for details.

`pdf.scale`, `pdf.offset`
: Offset and scaling factor used to control the location and height of the marginal density plot(s).

`xlim.pdf`
: Controls the x-axis (i.e. the horizontal axis) for the marginal density plotting area.

`pdf.axis`
: If `TRUE` and no other axis has been plotted to the right of the main plot, an axis is plotted with the marginal density.

`las.pdf`, `mgp.pdf`
: Axis control parameters passed to `axis` to plot the axis for the marginal density.

`...`
: Parameters passed to other functions; currently unused.

Details

If `do.pdf=TRUE` a marginal density plot is added. This plot is constructed from a set of (currently) normal densities centred at `x` with standard deviation `U/k`.

If a marginal density is plotted, `par("layout")` is changed to `pdf.layout`; otherwise, `par("layout")` is set to `matrix(1)`. Both override any previously set layout. `par("layout")` is preserved on exit.

The 'ilab' method passes all parameters in '...' to the default method with default values for `x`, upper and lower bounds `U.lo` and `U.hi`, labels and title taken from the `ilab` object.

`kpoints` is a convenience function for adding points with confidence intervals to an existing plot. `kpoints` is not a generic function and requires a vector `x`. Note that `kpoints` does not check for the presence of a marginal density plot.

Value

Invisibly returns a list with components:

`order`
: The order for plotting the original data, as returned by `order`.

`at`
: x-axis locations used, in plotting order.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

See Also

`arrows`.

Examples

```
data(Pb)
kplot(Pb$value, Pb$U, assigned=2.99, U.assigned=0.06)
kplot(Pb$value, Pb$U, assigned=2.99, U.assigned=0.06, do.pdf=TRUE)

#Use of return value for annotation
kp<-kplot(Pb$value, Pb$U, assigned=2.99, U.assigned=0.06)
text(kp$at, Pb$value-Pb$U, Pb$lab, srt=90, pos=4, cex=0.7)
```

LCS *LCS: Largest consistent subset*

Description

Calculates a 'largest consistent subset' given values and associated uncertainty information.

Usage

```
LCS(x, u, p = 0.05, method = "enum", simplify = FALSE,
verbose = FALSE)
```

Arguments

x	Vector of observations.
u	Vector of standard errors or standard uncertainties associated with x.
p	Significance level at which consistency is tested.
method	Subset identification method. Currently only 'enum' is supported.
simplify	If simplify is TRUE, only the lowest-uncertainty subset is returned even if several are of the same size.
verbose	Logical: Controls the level of reporting during the search.

Details

LCS obtains the largest subset(s) of x which pass a chi-squared test for consistency, taking the uncertainties u into account.

method controls the search method used. Method "enum" uses complete enumeration of all subsets of size n, starting at n==length(x) and decreasing n until at least one consistent subset is found. No other method is currently supported; if a different method is specified, LCS provides a warning and continues with "enum".

There may be more than on consistent subset of size n. If so, LCS returns all such subsets unless simplify is TRUE, in which case LCS prints a short warning and returns the subset with smallest estimated uncertainty as estimated for the Graybill-Deal weighted mean assuming large degrees of freedom in u.

verbose controls the level of reporting. If TRUE, LCS prints the progress of the search.

The general idea of a Largest Consistent Subset as implemented here was suggested by Cox (2006), though at least one other related method has been suggested by Heydorn (2006). It has, however, been criticised as an estimator (Toman and Possolo (2009)) ; see Warning below.

Value

If there is only one subset of maximum size, or if simplify=TRUE, a vector of indices for x representing the largest consistent subset.

If there is more than one subset of maximum size and simplify=FALSE, a matrix of indices in which the rows contain the indices of each subset.

Warning

LCS methods are essentially equivalent to unsupervised outlier rejection. In general, this results in a possibly extreme low estimated variance for an arbitrarily small subset (in the limit of gross inconsistency, LCS will return subsets of size 1). The estimated uncertainty calculated for the Graybill-Deal weighted mean of the subset(s) does not generally take account of the subset selection process or the dispersion of the complete data set, so is *not* an estimate of sampling variance.

LCS is therefore not recommended for consensus value estimation. It is however, quite useful for identifying value/uncertainty outliers.

Author(s)

S. Ellison <s.ellison@lgcgroup.com>

References

Cox, M. G. (2007) The evaluation of key comparison data: determining the largest consistent subset. Metrologia **44**, 187-200 (2007)

Heydorn, K. (2006) The determination of an accepted reference value from proficiency data with stated uncertainties. Accred Qual Assur **10**, 479-484 (2006)

Toman, B. and Possolo, A. (2009) Laboratory effects models for interlaboratory comparisons. Accred. Qual. Assur. **14**, 553-563 (2009)

See Also

None.

Examples

```
data(Pb)
with(Pb, LCS(value, U/k))
```

---

loc.est-class *The location estimate class*

Description

The location estimate class contains output from a variety of estimators used in the metRology package.

A print method is provided.

Usage

```
## S3 method for class 'loc.est'
print(x, ...)
```

Arguments

x	An object of class 'loc.est'
...	Parameters passed to other functions. Currently unused.

Details

An object of class 'loc.est' is a list containing

x Scalar estimate of location

u Standard uncertainty (usually equivalent to standard error) of the location estimate.

df Degrees of freedom associated with the location estimate (may be NA)

xi Numeric vector of individual values contributing to the estimate

ui Numeric vector of uncertainties initially associated with `xi`.

dfi Numeric vector of degrees of freedom associated with `ui`.

u.eff Numeric vector of 'effective uncertainties' in `xi` after any additional terms or adjustments are added (see below).

w Numeric vector of weights associated with `xi` (see below).

method Character string describing the method used to obtain the estimate.

method.details An optional list of additional details provided by the particular method used.

The 'effective uncertainties' `u.eff` arise from some estimation methods (for example, Mandel-Paule). These typically involve either the estimation of an additional variance term, a scale adjustment to the output value uncertainty or (for example in the case of the arithmetic mean) replacement of the initial individual uncertainties with some single estimate based on the dispersion of values. These adjustments are usually equivalent to replacing the estimator used with a weighted mean using weights $1/u_{eff}^2$.

The weight vector `w` is *not* equivalent to $1/u_{eff}^2$. Rather, it gives the ratio of prior weights $1/u_{eff}^2$ to posterior weights, which combine prior weights with some additional weighting. Posterior weights arise in particular when using robust estimators, and are generally 1 otherwise. The returned location estimate in such cases can be calculated as `sum(w*x/(u^2))/sum(w/(u^2)))`.

`method.details` is an optional list that may contain anything from a short summary of a scale factor or additional variance to a complete object (e.g. an rlm object) returned by the function used to calculate the estimate.

Value

The print method is called for its side effect; no value is returned.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None, yet

See Also

mpaule

Examples

```
## Cd heat of vapourisation example (see ?mpaule)
x2<-c(27.044, 26.022, 26.340, 26.787, 26.796)
v<-c(3, 76, 464, 3, 14)*1e-3
mp<-mpaule(x2, sqrt(v))

print(mp)
```

M-estimators *M- and MM-estimators for location.*

Description

Functions for calculating M- and MM-estimators for location given values and associated standard errors or standard uncertainties.

Usage

```
MM.estimate(x, ...)

## Default S3 method:
MM.estimate(x, u, c = 4.685, ...)

huber.estimate(x, ...)

## Default S3 method:
huber.estimate(x, u, k= 1.345, ...)
```

Arguments

x	numeric vector of mean values for groups
u	numeric vector of standard deviations or standard uncertainties associated with the values x
c, k	Tuning parameters passed to other functions (see rlm)
...	Parameters passed to other functions.

Details

These functions are wrappers for robust estimation using rlm. All simply call rlm with the formula x~1 and weights 1/u^2.

Value

An object of class 'loc.est'.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None, yet.

See Also

rlm, loc.est-class

Examples

```
## Cd heat of vapourisation example
## from Paule, R. C.  and Mandel, J. (1982) - see ?mpaule
x2<-c(27.044, 26.022, 26.340, 26.787, 26.796)
v<-c(3, 76, 464, 3, 14)*1e-3

MM.estimate(x2, sqrt(v))

huber.estimate(x2, sqrt(v))
```

Mandel-h *Mandel's h statistic.*

Description

Density, distribution function, quantile function and random generation for Mandel's h statistic, a measure of relative deviation from a common mean.

Usage

```
dmandelh(x, g, log = FALSE)
pmandelh(q, g, lower.tail = TRUE, log.p = FALSE)
qmandelh(p, g, lower.tail = TRUE, log.p = FALSE)
rmandelh(B, g)
```

Arguments

x, q	vector of quantiles.
p	vector of probabilities.
g	number of means for which h is calculated.
B	Number of observations. If 'length(B) > 1', the length is taken to be the number required.
lower.tail	logical; if TRUE (default), probabilities are P[X <= x]; otherwise, P[X > x].
log, log.p	logical; if TRUE, probabilities p are given as log(p).

Details

Mandel's h is calculated for a particular mean value y[i] in a set of mean values y as h[i] = (y[i] - mean(y))/sd(y))

The density, probabilities and quantiles can be derived from the beta distribution: (1+h*sqrt(g)/(g-1))/2 is distributed as Beta((g-2)/2, (g-2)/2).

Value

dmandelh returns the density at x, pmandelh the cumulative probability, qmandelh the quantiles for probability p and rmandelh returns B random values drawn from the distribution.

Vector values of x, p, q and g are permitted, in which case the functions return vectors.

Warning

Note that rmandelh uses B and not n (as do most R random number functions) for number of random draws; this is for compatibility with the relevant functions for Mandel's k, for which n is conventionally used for the number of replicates per group. Be careful when using named parameters!

Author(s)

S. L. R. Ellison, <s.ellison@lgcgroup.com>

References

None.

See Also

pmandelk

Examples

```
#Generate the 95% and 99% quantiles for comparison with tables in
#ISO 5725:1996 Part 2:
n <- 3:30
round(qmandelh(0.975, n), 2) #95% 2-tailed

round(qmandelh(0.995, n), 2) #99% 2-tailed
```

Mandel-k *Mandel's k statistic.*

Description

Density, distribution function, quantile function and random generation for Mandel's k statistic, a measure of relative precision compared to a common variance.

Usage

```
dmandelk(x, g, n, log = FALSE)
pmandelk(q, g, n, lower.tail = TRUE, log.p = FALSE)
qmandelk(p, g, n, lower.tail = TRUE, log.p = FALSE)
rmandelk(B, g, n)
```

Arguments

x, q	vector of quantiles.
p	vector of probabilities.
g	number of groups for which k is calculated.
n	number of observations in each group of data for which k is calculated.
B	Number of observations. If 'length(B) > 1', the length is taken to be the number required.
lower.tail	logical; if TRUE (default), probabilities are P[X <= x]; otherwise, P[X > x].
log, log.p	logical; if TRUE, probabilities p are given as log(p).

Details

Mandel's k for one of a set of g standard deviations s is calculated as

$$k = \frac{s_{ij}^2}{\sum_{i=1}^{p} s_{ij}^2/p}$$

Since the numerator is chi-squared(n-1), or Gamma((n-1)/2, 2), and the denominator can be written as the sum of the same quantity and a pooled variance with distribution Gamma((g-1)*(n-1)/2, 2), k is distributed as Beta((n-1)/2, (g-1)(n-1)/2). Quantiles, probabilities, density and random numbers can therefore be generated from the Beta distribution. For example, qmandelk is calculated as `sqrt( g * qbeta( (n-1)/2, (g-1)*(n-1)/2))`.

Value

dmandelh returns the density at x, pmandelh the cumulative probability, qmandelh the quantiles for probability p and rmandelh returns B random values drawn from the distribution.

Vector values of x, p, q and g are permitted, in which case the functions return vectors.

Warning

Note that `rmandelk` uses `B` and not `n` (as do most R random number functions) for number of random draws; this is because `n` is conventionally used for the number of replicates per group. Be careful when using named parameters!

Author(s)

S. L. R. Ellison, <s.ellison@lgcgroup.com>

References

None.

See Also

`pmandelh`

Examples

```
#Generate the 95% and 99% quantiles for comparison with tables in
#ISO 5725:1996 Part 2:

round(qmandelk(0.95, g=3:30, n=3), 2) #95% upper tail

round(qmandelk(0.99, g=3:30, n=3), 2) #99% upper tail
```

Mandel-Paule *Mandel-Paule calculation*

Description

Calculate a weighted mean, between-group standard deviation and standard error on the weighted mean using the Mandel-Paule algorithm.

Usage

```
mpaule(x, ..., tol=.Machine$double.eps^0.25, maxiter=25)

## Default S3 method:
mpaule(x, u=NULL, n=NULL, groups=NULL,
                   tol=.Machine$double.eps^0.25, maxiter=25, ...)

mandel.paule(x, ..., tol=.Machine$double.eps^0.25, maxiter=25)
```

Arguments

x	numeric vector of mean values for groups, or (if groups is given) of individual observations
u	numeric vector of standard deviations or standard uncertainties associated with the values x
n	integer vector of numbers in each group. If NULL, u are interpreted as standard uncertainties or standard errors. n is recycled to length(x).
groups	factor, or vetor which can be coerced to factor, of groups. If present, x is interpreted as a vector of individual observations and u and n ignored.
...	Additional parameters passed to other functions.
tol	numeric tolerance; iteration stops when the variance step size drops below tol*var(x)
maxiter	numeric maximum number of iterations

Details

The Mandel-Paule algorithm finds the between-method variance by iteratively solving the equation relating the weighted mean to the weighting factor applied. The weighting factor is the inverse of the sum of the standard error in x and the between-group variance.

If the iterative procedure produces a negative estimate for the between-group variance, the between-group variance is set to zero.

For the default method, if u is present and n=NULL, u is interpreted as a vector of standard uncertainties or standard errors. If n is not NULL, u is interpreted as a vector of standard deviations and standard errors are calculated as u/sqrt(n). If groups is not NULL, x is interpreted as a vector of individual observations grouped by groups, and the algorithm is applied to the corresponding group means and standard errors.

If maxiter is set less than 1, no iterations are performed and the consensus mean is returned as NA.

mandel.paule is an alias for mpaule retained for backward compatibility.

Value

A loc.est object; see loc.est for details. In the returned object, df is set to $n - 1$ where n is the number of non-NA observations or group means as appropriate, and method.details is returned as :

var.between	the estimated between-group variance)
iter	the number of iterations taken
converged	converged indicates the convergence status. 0L indicates failure to converge (maxiter reached before step size drops below tolerance); 1L indicates normal convergence; 2L indicates that the final step size resulted in a negative between-group variance, at which point the variance and step size are set to 0.0

Author(s)

S. Cowen <simon.cowen@lgcgroup.com> with amendments by S. L. R. Ellison.

References

Paule, R. C. and Mandel, J. (1982), J Res Nat Bur Stand, **87**, (5) 377-385

Examples

```
## the second example in the paper cited above
x <- c(201.533, 216.55)
s <- c(0.154, 0.25)
n <- c(6, 2)

mpaule(x, s/sqrt(n))

## Cd heat of vapourisation example from the paper cited above
x2<-c(27.044, 26.022, 26.340, 26.787, 26.796)
v<-c(3, 76, 464, 3, 14)*1e-3
mpaule(x2, sqrt(v))
```

---

mandel.h *Calculate Mandel's h statistics for replicate observations*

Description

`mandel.h` calculates Mandel's h statistics for replicate observations. Mandel's h is an indication of relative deviation from the mean value.

Usage

```
mandel.h(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
method=c("classical", "robust"), n = NA, ...)

## Default S3 method:
mandel.h(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
method=c("classical", "robust"), n = NA, ...)

## S3 method for class 'ilab'
mandel.h(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
method=c("classical", "robust"), n = NA, ...)
```

Arguments

`x`	An R object (see Details below), which contains replicate observations or, if `g` is absent, means or standard deviations.
`g`	A primary grouping factor, usually corresponding to Laboratory in an inter-laboratory study. If not present, `x` is taken as a set of means or standard deviations (depending on whether `type` is `"h"` or `"k"`.
`m`	A secondary grouping factor, usually corresponding to test item or measured quantity. `m` is ignored if `x` has more than one column.

na.rm	A logical value indicating whether 'NA' values should be stripped before the computation proceeds. Passed to functions such as mean and sd.
rowname	A single character label for the primary grouping factor (e.g. "Lab", "Organisation").
method	Character scalar giving the calculation method. "classical" gives the traditional calculation; "robust" gives a robust variant (see Details).
n	scalar number of observations per group. Required only if x consists of calculated standard deviations.
...	Additional parameters passed to hubers when method="robust".

Details

mandel.h is a convenience wrapper for mandel.kh(..., type="h"). It is generic, with methods for numeric vectors, arrays, data frames, matrices and objects of class 'ilab'. All parameters are passed to mandel.kh.

Mandel's h is an indicators of relative deviation for grouped sets of observations. Given a set of observations x_{ijl} where i, j, l denotes observation l, $l = 1, 2, ...n$ for measurand or test item j and group (usually laboratory) i, $i = 1, 2, ...p$, Mandel's h is given by:

$$h = \frac{\bar{x_{ij}} - \bar{x_j}}{s_j}$$

where $s_j = \sqrt{\sum_{i=1}^{p} \frac{(\bar{x_{ij}} - \bar{x_j})}{p-1}}$

If x is a vector, one-dimensional array or single-column matrix, values are aggregated by g and, if present, by m. If x is a data frame or matrix, each column is aggregated by g and m silently ignored if present. In all cases, if g is NULL or missing, each row (or value, if a vector) in x is taken as a pre-calculated mean (for Mandel's h) or standard deviation (for Mandel's k).

If x is an object of class 'ilab', g defaults to '$org' and m to $measurand.

The returned object includes a label ('grouped.by') for the primary grouping factor. For the 'ilab' method, this is "Organisation". For other methods, If rowname is non-null, rowname is used. If rowname is NULL, the default is deparse(substitute(g)); if g is also NULL or missing, "Row" is used.

If method="robust", Mandel's h is replaced by a robust z score calculated by replacing $\bar{x_j}$ and s_j with the robust estimates of location and scale obtained using Huber's estimate with tuning constant k set to 1.5 (unless otherwise specified in ...).

Value

mandel.h returns an object of class "mandel.kh", which is a data frame consisting of the required Mandel's statistics and in which each row corresponds to a level of g and each column to a level of m or (if x was a matrix or data frame) to the corresponding column in x. In addition to the class, the object has attributes:

'mandel.type' "h" or "k"

'grouped.by' Character scalar giving the label used for the grouping factor g; see Details above for the defaults.

'n' Number of observations per group (n if specified

Author(s)

S Ellison <s.ellison@lgcgroup.com>

References

Accuracy (trueness and precision) of measurement methods and results – Part 2: Basic method for the determination of repeatability and reproducibility of a standard measurement method. ISO, Geneva (1994).

See Also

mandel.k, mandel.kh; pmandelh, pmandelk for probabilities, quantiles etc.; plot.mandel.kh, barplot.mandel.kh for plotting methods.

Examples

```
data(RMstudy)

#Data frame examples: note no secondary grouping factor
h <- with(RMstudy, mandel.h(RMstudy[2:9], g=Lab))
plot(h, las=2)

#Vector variant
RMstk <- stack(RMstudy[,2:9])
names(RMstk) <- c("x", "meas")
#names replace 'values' and 'ind'
RMstk$Lab <- rep(RMstudy$Lab, 8)
h2 <- with(RMstk, mandel.h(x, g=Lab, m=meas, rowname="Laboratory"))
#Note use of rowname to override g
plot(h2, las=2)

#ilab method
RM.ilab <- with(RMstk, construct.ilab(org=Lab, x=x, measurand=meas,
item=factor(rep("CRM", nrow(RMstk))) ) )

plot(mandel.h(RM.ilab))

#Robust variant
hrob <- with(RMstudy, mandel.kh(RMstudy[2:9], g=Lab, type="h", method="robust"))
plot(hrob, las=2)
```

---

mandel.k *Calculate Mandel's k statistics for replicate observations*

Description

mandel.k calculates Mandel's k statistics for replicate observations. Mandel's k an indicator of precision compared to the pooled standard deviation across all groups.

Usage

```
mandel.k(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
method=c("classical", "robust"), n = NA, ...)

## Default S3 method:
mandel.k(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
method=c("classical", "robust"), n = NA, ...)

## S3 method for class 'ilab'
mandel.k(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
method=c("classical", "robust"), n = NA, ...)
```

Arguments

x	An R object (see Details below), which contains replicate observations or, if g is absent, means or standard deviations.
g	A primary grouping factor, usually corresponding to Laboratory in an interlaboratory study. If not present, x is taken as a set of means or standard deviations (depending on whether type is "h" or "k".
m	A secondary grouping factor, usually corresponding to test item or measured quantity. m is ignored if x has more than one column.
na.rm	A logical value indicating whether 'NA' values should be stripped before the computation proceeds. Passed to functions such as mean and sd.
rowname	A single character label for the primary grouping factor (e.g. "Lab", "Organisation").
method	Character scalar giving the calculation method. "classical" gives the traditional calculation; "robust" gives a robust variant (see Details).
n	scalar number of observations per group. Required only if x consists of calculated standard deviations.
...	Additional parameters passed to other methods. Currently not implemented.

Details

mandel.k is a convenience wrapper for mandel.kh(..., type="k"). It is generic, with methods for numeric vectors, arrays, data frames, matrices and objects of class 'ilab'. All parameters are passed to mandel.kh.

Mandel's k is an indicator of relative dispersion for grouped sets of observations. Given a set of observations x_{ijl} where i, j, l denotes observation l, $l = 1, 2, ...n$ for measurand or test item j and group (usually laboratory) i, $i = 1, 2, ...p$, Mandel's k is given by:

$$k = \sqrt{\frac{s_{ij}^2}{\sum_{i=1}^{p} s_{ij}^2/p}}$$

where s_{ij} is the standard deviation of values x_{ijk} over $k = 1, 2, ..., n$.

If x is a vector, one-dimensional array or single-column matrix, values are aggregated by g and, if present, by m. If x is a data frame or matrix, each column is aggregated by g and m silently ignored

if present. In all cases, if g is NULL or missing, each row (or value, if a vector) in x is taken as a pre-calculated mean (for Mandel's h) or standard deviation (for Mandel's k).

If x is an object of class 'ilab', g defaults to '$org' and m to $measurand.

The returned object includes a label ('grouped.by') for the primary grouping factor. For the 'ilab' method, this is "Organisation". For other methods, If rowname is non-null, rowname is used. If rowname is NULL, the default is deparse(substitute(g)); if g is also NULL or missing, "Row" is used.

If method="robust", Mandel's k is calculated by replacing the classical pooled standard deviation with the robust pooled standard deviation calculated by algorithm S (see algS).

Value

mandel.k returns an object of class "mandel.kh", which is a data frame consisting of the required Mandel's statistics and in which each row corresponds to a level of g and each column to a level of m or (if x was a matrix or data frame) to the corresponding column in x. In addition to the class, the object has attributes:

'mandel.type' "h" or "k"

'grouped.by' Character scalar giving the label used for the grouping factor g; see Details above for the defaults.

'n' Number of observations per group (n if specified

Author(s)

S Ellison <s.ellison@lgcgroup.com>

References

Accuracy (trueness and precision) of measurement methods and results – Part 2: Basic method for the determination of repeatability and reproducibility of a standard measurement method. ISO, Geneva (1994).

See Also

mandel.h, mandel.kh; pmandelh, pmandelk for probabilities, quantiles etc.; plot.mandel.kh, barplot.mandel.kh for plotting methods.

Examples

```
data(RMstudy)

#Data frame examples: note no secondary grouping factor
h <- with(RMstudy, mandel.k(RMstudy[2:9], g=Lab))
plot(h, las=2)

#Vector variant
RMstk <- stack(RMstudy[,2:9])
names(RMstk) <- c("x", "meas")
#names replace 'values' and 'ind'
```

```
RMstk$Lab <- rep(RMstudy$Lab, 8)
h2 <- with(RMstk, mandel.k(x, g=Lab, m=meas, rowname="Laboratory"))
#Note use of rowname to override g
plot(h2, las=2)

#ilab method
RM.ilab <- with(RMstk, construct.ilab(org=Lab, x=x, measurand=meas,
item=factor(rep("CRM", nrow(RMstk))) ) )

plot(mandel.k(RM.ilab))

#Robust variant
krob <- with(RMstudy, mandel.kh(RMstudy[2:9], g=Lab, type="k", method="robust"))
plot(krob, las=2)
```

---

mandel.kh *Calculate Mandel's h and k statistics for replicate observations*

Description

mandel.kh calculates Mandel's h and k statistics for replicate observations. These are traditionally used to provide a rapid graphical summary of results from an inter-laboratory exercise in which each organisation provides replicate observations of one or more measurands on one or more test items. Mandel's h is an indication of relative deviation from the mean value; Mandel's k is an indicator of precision compared to the pooled standard deviation across all groups.

Usage

```
mandel.kh(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
type = c("h", "k"), method=c("classical", "robust"), n = NA, ...)

## Default S3 method:
mandel.kh(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
type = c("h", "k"), method=c("classical", "robust"), n = NA, ...)

## S3 method for class 'data.frame'
mandel.kh(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
type = c("h", "k"), method=c("classical", "robust"), n = NA, ...)

## S3 method for class 'matrix'
mandel.kh(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
type = c("h", "k"), method=c("classical", "robust"), n = NA, ...)

## S3 method for class 'array'
mandel.kh(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
type = c("h", "k"), method=c("classical", "robust"), n = NA, ...)

## S3 method for class 'ilab'
```

```
mandel.kh(x, g = NULL, m = NULL, na.rm = T, rowname = NULL,
type = c("h", "k"), method=c("classical", "robust"), n = NA, ...)
```

Arguments

x	An R object (see Details below), which contains replicate observations or, if g is absent, means or standard deviations.
g	A primary grouping factor, usually corresponding to Laboratory in an inter-laboratory study. If not present, x is taken as a set of means or standard deviations (depending on whether type is "h" or "k".
m	A secondary grouping factor, usually corresponding to test item or measured quantity. m is ignored if x has more than one column.
na.rm	A logical value indicating whether 'NA' values should be stripped before the computation proceeds. Passed to functions such as mean and sd.
rowname	A single character label for the primary grouping factor (e.g. "Lab", "Organisation").
type	Character denoting the statistic to be calculated; may be "h" or "k".
method	Character scalar giving the calculation method. "classical" gives the traditional calculation; "robust" gives a robust variant (see Details).
n	scalar number of observations per group. Required only if x consists of calculated standard deviations.
...	Additional parameters passed to hubers when method="robust" and type="h".

Details

mandel.kh can be called directly, but is usually intended to be called via convenience functions mandel.h or mandel.k.

mandel.kh is a generic, with methods for numeric vectors, arrays, data frames, matrices and objects of class 'ilab'.

Mandel's statistics are simple indicators of relative deviation or precision for grouped sets of observations. Given a set of observations x_{ijl} where i, j, l denotes observation l, $l = 1, 2, ...n$ for measurand or test item j and group (usually laboratory) i, $i = 1, 2, ...p$, Mandel's h and k are given by:

$$h = \frac{\bar{x_{ij}} - \bar{x_j}}{s_j}$$

where $s_j = \sqrt{\sum_{i=1}^{p} \frac{(\bar{x_{ij}} - \bar{x_j})}{p-1}}$

and

$$k = \sqrt{\frac{s_{ij}^2}{\sum_{i=1}^{p} s_{ij}^2/p}}$$

where s_{ij} is the standard deviation of values x_{ijk} over $k = 1, 2, ..., n$.

If x is a vector, one-dimensional array or single-column matrix, values are aggregated by g and, if present, by m. If x is a data frame or matrix, each column is aggregated by g and m silently ignored if present. In all cases, if g is NULL or missing, each row (or value, if a vector) in x is taken as a pre-calculated mean (for Mandel's h) or standard deviation (for Mandel's k).

If x is an object of class 'ilab', g defaults to '$org' and m to $measurand.

The returned object includes a label ('grouped.by') for the primary grouping factor. For the 'ilab' method, this is "Organisation". For other methods, If rowname is non-null, rowname is used. If rowname is NULL, the default is deparse(substitute(g)); if g is also NULL or missing, "Row" is used.

If method="robust", Mandel's h is replaced by a robust z score calculated by replacing $\bar{x}_j$ and s_j with the robust estimates of location and scale obtained using Huber's estimate with tuning constant k set to 1.5 (or as included in ...), and Mandel's k is calculated by replacing the classical pooled standard deviation in the denominator with the robust pooled standard deviation calculated by algorithm S (see algS).

Value

mandel.kh returns an object of class "mandel.kh", which is a data frame consisting of the required Mandel's statistics and in which each row corresponds to a level of g and each column to a level of m or (if x was a matrix or data frame) to the corresponding column in x. In addition to the class, the object has attributes:

'mandel.type' "h" or "k"

'grouped.by' Character scalar giving the label used for the grouping factor g; see Details above for the defaults.

'n' Number of observations per group (n if specified

Author(s)

S Ellison <s.ellison@lgcgroup.com>

References

Accuracy (trueness and precision) of measurement methods and results – Part 2: Basic method for the determination of repeatability and reproducibility of a standard measurement method. ISO, Geneva (1994).

See Also

mandel.h, mandel.k for convenience functions; pmandelh, pmandelk for probabilities, quantiles etc.; plot.mandel.kh, barplot.mandel.kh for plotting methods. algS and hubers for robust estimates used when method="robust".

Examples

```
data(RMstudy)

#Data frame examples: note no secondary grouping factor
h <- with(RMstudy, mandel.kh(RMstudy[2:9], g=Lab, type="h"))
```

```
plot(h, las=2)

k <- with(RMstudy, mandel.kh(RMstudy[2:9], g=Lab, type="k"))
plot(k, las=2)

#Vector variant
RMstk <- stack(RMstudy[,2:9])
names(RMstk) <- c("x", "meas")
#names replace 'values' and 'ind'
RMstk$Lab <- rep(RMstudy$Lab, 8)
h2 <- with(RMstk, mandel.kh(x, g=Lab, m=meas, rowname="Laboratory"))
#Note use of rowname to override g
plot(h2, las=2)

#ilab method
RM.ilab <- with(RMstk, construct.ilab(org=Lab, x=x, measurand=meas,
item=factor(rep("CRM", nrow(RMstk))) ) )

plot(mandel.kh(RM.ilab, type="h"))

#Robust variants
hrob <- with(RMstudy, mandel.kh(RMstudy[2:9], g=Lab, type="h", method="robust"))
plot(hrob, las=2)

krob <- with(RMstudy, mandel.kh(RMstudy[2:9], g=Lab, type="k", method="robust"))
plot(krob, las=2)
```

`methods.ilab` *Methods for the 'ilab' class.*

Description

Functions for printing and plotting interlaboratory study objects objects of class 'ilab'.

Usage

```
## S3 method for class 'ilab'
print(x, ..., digits = NULL, right = FALSE)

## S3 method for class 'ilab'
plot(x, ...)
```

Arguments

`x`	An object of class 'ilab'
`digits`	Number of digits to display in budget and (if present) distribution parameter lists. Passed to `format` for distribution parameter list and to `print.data.frame` for output.

right	If TRUE, strings in uncertainty budget are right-justified. This differs from the default in print.data.frame.
...	Parameters passed to other functions

Details

The print method uses print.data.frame to display the data after formatting the distrib and distrib.pars elements.

The plot method passes the object to kplot.

Value

The print and plot methods are called for their side effects.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None, yet.

See Also

ilab-class, subset.ilab kplot.

Examples

```
data(Pb)
il.pb<-construct.ilab(org=Pb$lab, x=Pb$value, measurand="Pb", item="none",
            u=Pb$u, k=Pb$k, U=Pb$U, title=c("CCQM K30", "Lead in wine"), method=Pb$method)

print(il.pb)

plot(il.pb)
```

---

mle.1wre *Vangel-Rukhin Maximum Likelihood Estimate*

Description

Calculate a weighted mean, between-group standard deviation and standard error on the weighted mean using the Maximum likelihood algorithm of Vangel-Rukhin.

Usage

```
mle.1wre(x, s2, n, init.mu = mean(x), init.sigma2 = var(x), labels = c(1:length(x)),
         max.iter = 200, tol = .Machine$double.eps^0.5, trace = FALSE)
```

Arguments

x	numeric vector of the sample mean values of each group
s2	numeric vector of the sample variances of each group
n	integer vector of sample size of each group
init.mu	numeric initial value for the mean
init.sigma2	numeric initial value for the between-group component of variance
labels	vector of group names. Coerced to character on use.
max.iter	numeric maximum number of iterations
tol	numeric tolerance; iteration stops when the relative step size drops below 'tol'
trace	when TRUE shows the sequence of intermediate results

Details

The Vangel-Rukhin MLE algorithm finds the between-method variance by iteratively solving the equation relating the weighted mean to the weighting factor applied. The weighting factor is the inverse of the sum of the standard error in 'x' and the between-method variance, scaled by the between-method variance.

For the default method, 's2' is interpreted as a vector of sample variances. 'x' is interpreted as a vector of sample means and the algorithm is applied to the corresponding group means, variances, and sample sizes.

The Vangel-Rukhin MLE algorithm shows an improvement in the number of iterations required to converge over the classical MLE based on the Score equations.

The function mle.1wre implements the MLE for the one way random effects based on the Fisher scoring equations and is provided for comparison purpose only.

Value

`mle.1wre` returns an object of class "summary.mle.1wre" which contains the following fields:

mu	the estimated mean
var.mu	the variance associated with the estimated mean
sigma2	the estimated between variance component
llh	the log likelihood of the estimates
tot.iter	the total number of iterations ran
cur.rel.abs.error	the current relative absolute error reached
sigmai2	a vector with the estimates of the within variance components

Author(s)

H. Gasca-Aragon

References

Vangel, M. G. and Rukhin, A. L. (1999), Biometrics, Vol 55, No. 1 pp 129-136

Searle, S. R., Cassella, G., and McCulloch, C. E. (1992). Variance Components. New York: Wiley.

See Also

vr.mle, loc.est-class

Examples

```
##=====================================================================
## the dietary fiber in apples example in the Vangel and Rukhin paper
##=====================================================================

m1 <- c(12.46, 13.035, 12.44, 12.87, 13.42, 12.08, 13.18, 14.335, 12.23)
s1 <- c(0.028, 0.233, 0.325, 0.071, 0.339, 0.325, 0.099, 0.064, 0.212)
n1 <- c(2, 2, 2, 2, 2, 2, 2, 2, 2)

mle.1wre(m1, s1^2, n1, tol=1e-6)


# output:
# 12.90585
# 0.2234490
# 0.4262122
# 12.46790 13.34380
# 6
```

---

msd *Median scaled difference*

Description

Generates median of scaled differences from each point in a data set to all other points..

Usage

```
msd(x, s=mad , ...)
```

Arguments

x	Vector of observations
s	Either a function returning an estimate of scale for x or a vector of length length(x) of standard errors or standard uncertainties in x.
...	Parameters passed to s if s is a function.

Details

For each observation `x[i]`, `msd` calculates the median of `|x[i]-x[j]|/sqrt(s[i]^2+s[j]^2)`, `j!=i`, that is, the median of differences divided by the estimated uncertainties of the distance.

If `s` is a function, it is applied to `x` and replicated to length `length(x)`; if a scalar, it is replicated to length `length(x)`.

The median scaled difference is a measure of how 'far' an individual observation is from the majority of the other values in the data set. As a rule of thumb, values above 2 are indicative of a suspect `(x[i], s[i])` data pair; that is, a value `x[i]` that is remote from a large fraction of the remaining data given its associated standard error or standard uncertainty `s[i]`.

Value

An object of class "MSD", consisting of a vector of length `length(x)` of median scaled absolute deviations for each observation, with attributes:

`names`	character vector of names, taken from `x`
`x`	values supplied as `x`
`s`	values supplied as `s`

Print and plotting methods are currently provided for the `"MSD"` class; see `MSD-class`.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

Ellison, S. L. R. (2018) An outlier-resistant indicator of anomalies among inter-laboratory comparison data with associated uncertainty. _Metrologia_ (accepted 4 October 2018)

See Also

`pmsd`, `qmsd`, `MSD-class`.

Examples

```
data(Pb)
msd(Pb$value)  # Uses mad(Pb$value) as scale estimate
msd(Pb$value, Pb$u)  # Scales differences using standard uncertainties
```

MSD-class *Methods for the object returned by* msd.

Description

Print and plotting methods for the MSD object class returned by msd.

Usage

```
       ## S3 method for class 'MSD'
print(x, ...)

       ## S3 method for class 'MSD'
plot(x, type="h", ylab="MSD", ylim=NULL, ...)

       ## S3 method for class 'MSD'
barplot(height, ylab="MSD", names.arg=names(height),
crit.vals=TRUE, lty.crit=c(2,1), col.crit=2, lwd.crit=c(1,2),
probs=c(0.95, 0.99), n=length(height), ylim=NULL, ... )
```

Arguments

x, height	Object of class "MSD".
type	The plot type. See plot.default.
ylab	Label for vertical axis, passed to barplot
names.arg	Labels for individual bars in bar plot, passed to barplot. If names(height) is NULL, bars are numbered.
crit.vals	If TRUE, horizontal lines at critical values are added to the plot. These are calculated by link{qmsd} based on supplied values of probs and n.
lty.crit, col.crit, lwd.crit	Vectors of line style parameters for plotted critical values, passed to abline.
probs	vector of probabilities at which critical values are drawn.
n	integer number of observations for critical value calculation; passed to qmsd.
ylim	Limits for y-axis. the default makes sure the axis begins at zero and includes all values
...	Parameters passed to other methods.

Details

See msd for the object description.

For the barplot method, critical values are 'single-observation' quantiles. For use as an outlier test, use probabilities adjusted for multiple comparison; for example, for the barplot method, consider raising the default probs to the power $1/n$.

Value

The `print` method returns the object, invisibly.

The `plot` method returns NULL, invisibly.

The `barplot` methods return the values at the midpoint of each bar.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

See Also

`msd`, `qmsd`.

Examples

```
data(Pb)
msd.Pb<-msd(Pb$value, Pb$u)  # Uses individual standard uncertainties
names(msd.Pb) <- as.character(Pb$lab)

plot(msd.Pb)

barplot(msd.Pb)
```

Pb *Lead in wine*

Description

A data frame containing reported results for lead (in mg/kg) from CCQM Key Comparison CCQM-K30.

Usage

```
data(Pb)
```

Format

A data frame containing 11 reported results with uncertainty data:

lab Factor giving abbreviated laboratory identifier

value The reported value for lead (mg/kg)

u Standard uncertainty (mg/kg). The values in `Pb` were calculated from the reported expanded uncertainty `U` and coverage factor `k` using `u=U/k`.

k Coverage factor. Conventionally, the coverage factor is set to a suitable quantile of Student's t based on the Welch-Satterthwaite effective degrees of freedom or simply set to 2 for approximately 95% confidence. In this data set, labs all quoted `k` for approximately 95% confidence.

U Expanded uncertainty as reported by labs.

method Factor indicating general measurement methodology:

IDMS Isotope dilution mass spectrometry

ICP Inductively coupled plasma spectrometry

GFAAS Graphite furnace atomic absorbtion spectrometry

include logical; Whether the reported result was included in the calculation of the Key Comparison Reference Value for the study.

Details

The study involved circulation of a homogeneous set of samples of wine for analysis for lead (Pb) content by a number of National Measurement Institutes.

The Key Comparison Reference Value, or assigned value for the lead content, was set at 2.99 mg/kg with expanded uncertainty 0.06 mg/kg.

Source

Hearn, R., Santamaria-Fernandez, R. and Sargent, M. (2008) Final report on key comparison CCQM-K30: Determination of lead in wine. *Metrologia* **45**, 08001, 2008

References

See source.

`pdchisq` *Pair-difference chi-squared statistic*

Description

Generates the pair-difference chi-squared statistic for each point in a data set, summing over scaled differences frm other points.

Usage

```
pdchisq(x, s=sd, cov=NULL, cor = NULL, na.rm=FALSE, ...)
```

Arguments

`x`	Vector of observations
`s`	Either a function returning an estimate of scale for `x` or a vector of length `length(x)` of standard errors or standard uncertainties in `x`.
`cov`, `cor`	Covariance or correlation matrix, respectively, describing the covariance structure across `x`.
`na.rm`	logical. Controls whether missing values should be removed. NOTE: `na.rm` is passed to `s` as well as `sum`; `s` must accordingly accept an argument `na.rm`.
`...`	Parameters passed to `s` if `s` is a function.

Details

For each observation x_j, pdchisq calculates the pairwise chi-squared statistic

$$\chi^2_{j\mathrm{PD}} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - x_j)^2/(s_i^2 + s_j^2 - 2\mathrm{cov}(x_i, x_j))$$

that is, the sum of squared differences divided by the estimated uncertainties of the distances.

If s is a function, it is applied to x and replicated to length length(x). Note that na.rm is passed to s, so s must accept an argument na.rm. For some scale functions that may require s to be defined as a wrapper to avoid unused argument warnings. For example, if foo is a scale function that does not accept an na.rm argument, use s=function(x, na.rm, ...) foo(x, ...) .

If s is a scalar, it is replicated to length length(x).

If cov is present, s is silently ignored. If cor is present, cov is constructed from cor and s

The pair-difference chi-squared statistic is a measure of how 'far' an individual observation is from all the other values in the data set, taking account of uncertainties.

Value

An object of class "PDchisq", consisting of a vector of length length(x) of median scaled absolute deviations for each observation, with attributes:

names	character vector of names, taken from x
x	values supplied as x
s	values supplied as s

Print and plotting methods are provided for the "PDchisq" class; see PDchisq-class.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

R J Douglas, A G Steele (2006) Pair-difference chi-squared statistics for Key Comparisons, Metrologia 43, 89-97

See Also

PDchisq-class, bootPDchisq.

Examples

```
data(Pb)
pdchisq(Pb$value)  # Uses mad(Pb$value) as scale estimate
pdchisq(Pb$value, Pb$u)  # Scales differences using standard uncertainties
```

PDchisq-class *Methods for the object returned by* pdchisq.

Description

Print and plotting methods for the PDchisq object class returned by pdchisq.

Usage

```
       ## S3 method for class 'PDchisq'
print(x, ...)

       ## S3 method for class 'PDchisq'
plot(x, type="h", ylab="Pair-difference chi-squared", ylim=NULL, ...)

       ## S3 method for class 'PDchisq'
barplot(height, ylab="Pair-difference chi-squared", names.arg=names(height),
crit.vals=TRUE, lty.crit=c(2,1), col.crit=2, lwd.crit=c(1,2),
probs=c(0.95, 0.99), n=length(height), ylim=NULL, ... )

bootPDchisq(x, B=3000, probs=c(0.95, 0.99),
method=c("rnorm", "lhs"), keep=FALSE, labels=names(x), ...)
```

Arguments

x, height	Object of class "PDchisq".
type	The plot type. See plot.default.
ylab	Label for vertical axis, passed to barplot
names.arg	Labels for individual bars in bar plot, passed to barplot. If names(height) is NULL, bars are numbered.
crit.vals	If TRUE, horizontal lines at critical values are added to the plot. These are calculated from link{qchisq} based on supplied values of probs and n.
lty.crit, col.crit, lwd.crit	Vectors of line style parameters for plotted critical values, passed to segments. Recycled to the length of critical.values in the supplied bootPDchisq object.
probs	vector of probabilities at which critical values are drawn (or, for the bootstrap, calculated and retained.
n	integer number of observations (NOT degrees of freedom) for critical value calculation; used (as $n-1$) by qchisq. See Details.
B	integer number of bootstrap replicates; passed to boot.mtr.pairwise.
ylim	Limits for y-axis. the default makes sure the axis begins at zero and includes all values
method	simulation method; passed to boot.mtr.pairwise.

keep	logical, indicating whether to retain simulated values. Passed to boot.mtr.pairwise.
labels	labels for returned object; passed to boot.mtr.pairwise.
...	Parameters passed to other methods.

Details

See pdchisq for the "PDchisq" object description.

The quantiles plotted by the barplot method are based on qchisq, divided by $n - 1$. Note that this assumes independence and is at best a guide; for more accurate quantiles, see bootPDchisq For the barplot method, the default critical values are 'single-observation' quantiles. For use as an outlier test, use probabilities adjusted for multiple comparison; for example, consider raising the default probs to the power $1/n$.

Specifying n directly is principally intended for when the plotted values are not the whole set of pair-difference chi-squared values calculated from a given data set. However, it can also be useful for limiting cases. Where a single value has very small uncertainty s, its pair-difference chi-squared is distributed approximately as $\chi^2(n-1)$. At the other extreme, if a single value has very large uncertainty s compared to others, its pair-difference chi-squared is distributed approximately as $\chi^2(1)$.

bootPDchisq generates a parametric bootstrap (Monte carlo simulation) from an object of class "PDchisq". This provides a case-specific test of the null that all mean values are equal, with the uncertainties and/or covariances accurate. bootPDchisq is a wrapper for boot.mtr.pairwise, which provides further information.

Value

The print method returns the object, invisibly.

The plot method returns NULL, invisibly.

The barplot method returns the values at the midpoint of each bar.

bootPDchisq returns an object of class "bootMtrPairs"; see bootMtrPairs-class for details.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

See Also

pdchisq, qchisq, bootMtrPairs-class.

Examples

```
data(Pb)
pdchisq.Pb<-pdchisq(Pb$value, Pb$u)  # Uses individual standard uncertainties
names(pdchisq.Pb) <- as.character(Pb$lab)

plot(pdchisq.Pb)

barplot(pdchisq.Pb)
```

`plot.d.ellipse` *Plot data ellipses*

Description

Plots a number of data ellipses specified by

Usage

```
## S3 method for class 'd.ellipse'
plot(x, col.ellipse = 1, lty.ellipse = 1, lwd.ellipse = 1,
fill = NA, density = NULL, angle = 45, add = FALSE, npoints = 100,
xlim = NA, ylim = NA,
prinax = FALSE, col.prinax = 1, lty.prinax = 1, lwd.prinax = 1,
xlab=NULL, ylab=NULL, ...)
```

Arguments

`x`	An object of class `d.ellipse`
`col.ellipse`, `lty.ellipse`, `lwd.ellipse`	Colour, line type and line width for the ellipse(s). Can be vectors, allowing different colour, line type etc. Recycled as necessary to length `length(x)`.
`fill`, `density`, `angle`	Fill colour, line density and line angle for each ellipse in `x`. See `polygon` for details. Can be vectors. Recycled as necessary to length `length(x)`.
`add`	If `TRUE`, ellipses are added to an existing plot. If `FALSE` a new plot is created.
`npoints`	Controls the number of points used to form each ellipse. See `data.ellipse` for details.
`xlim`, `ylim`	Plot limits. Ignored if `add == FALSE`
`prinax`	If `TRUE` (the default), the principal axes are drawn on the plot.
`col.prinax`, `lty.prinax`, `lwd.prinax`	Colour, line type and line width for principal axes.
`xlab`, `ylab`	Axis labels passed to `plot` if `add == FALSE`. Defaults to dimension names in `x` or, if those are `NULL`, to "X" and "Y".
`...`	Additional arguments, passed to `plot` if `add == TRUE`.

Details

A series of ellipses specified in `x` is plotted. The function is primarily used for adding ellipses to a Youden plot.

Value

The function is called for its side effect, which is the drawing of ellipses.

Author(s)

S L R Ellison

See Also

data.ellipse, youden.plot

Examples

```
data(chromium)
cov.Cr <- cov.dellipse(chromium)
dellipse.Cr <- data.ellipse(cov.Cr, plot=FALSE)
plot(dellipse.Cr)
```

---

plot.mandel.kh *Classical plots of Mandel's statistics.*

Description

plot.mandel.kh produces classic plots of Mandel's statistics, suitably grouped and with appropriate indicator lines for unusual values.

Usage

```
## S3 method for class 'mandel.kh'
plot(x, probs = c(0.95, 0.99), main, xlab = attr(x, "grouped.by"),
ylab = attr(x, "mandel.type"), ylim = NULL, las = 1,
axes = TRUE, cex.axis = 1, frame.plot = axes,
lwd = 1, lty = 1, col = par("col"),
col.ind = 1, lty.ind = c(2, 1), lwd.ind = 1,
separators = TRUE, col.sep = "lightgrey", lwd.sep = 1, lty.sep = 1,
zero.line = TRUE, lwd.zero = 1, col.zero = 1, lty.zero = 1,
p.adjust = "none", ...)
```

Arguments

x	An object of class 'mandel.kh'
probs	Indicator lines are drawn for these probabilities. Note that probs is interpreted as specifying two-tailed probabilities for Mandel's h and one-sided (upper tail) probabilities for Mandel's k.
main	a main title for the plot. If missing, the default is paste(deparse(substitute(x)), " - Mandel's", attr(x, "mandel.type"), if(attr(x, "mandel.method") == "robust") "(Robust variant)")
xlab	a label for the x axis; defaults to the grouped.by attribute for x.

ylab	a label for the x axis; defaults to the mandel.type attribute for x.
ylim	the y limits of the plot. For Mandel's k, the default lower limit is zero.
las	the style of the axis labels; see par for details.
axes	a logical value indicating whether axes should be drawn on the plot.
cex.axis	The magnification to be used for axis annotation relative to the current setting of 'cex'.
frame.plot	Logical; If TRUE a box is drawn around the plot.
lwd, lty, col	Graphical parameters used for the plotted vertical lines corresponding to each value of Mandel's statistics (the plot is of type "h"). All are recycled across the prinmary grouping factor, allowing different measurands/test items to be identified more clearly.
col.ind, lty.ind, lwd.ind	Graphical parameters used for the indicator lines, recyckled to length(probs). For attr(x, "mandel.type")=="h" the graphical parameters are applied to negative as well as positive indicator lines, applied outwards from zero.
separators	Logical; if TRUE, separator lines are drawn between groups of values.
col.sep, lwd.sep, lty.sep	Graphical parameters used for the separator lines.
zero.line	logical; if TRUE a horizontal line is drawn at zero.
lwd.zero, col.zero, lty.zero	Graphical parameters used for the zero line.
p.adjust	Correction method for probabilities. If not "none", passed to p.adjust prior to calculating indicator lines. Usually, indicator lines are drawn without correction (that is, with p.adjust="none"); specifying a p-value correction effectively turns the Mandel's statistics into single outlier tests.
...	Other (usually graphical) parameters passed to plot.

Details

Mandel's statistics are traditionally plotted for inter-laboratory study data, grouped by laboratory, to give a rapid graphical view of laboratory bias and relative precision. The traditional plot is a plot of type "h", that is, simple vertical lines from the x-axis.

For classical Mandel statistics, indicator lines are drawn based on qmandelh or qmandelk as appropriate. For robust variants, indicator lines use qnorm for the h statistic and qf(probs, n, Inf) for the k statistic. Note that this corresponds to taking the robust estimates of location and scale as true values, so will be somewhat anticonservative.

plot.mandel.kh uses gplot for the main plot.

Value

plot.mandel.kh returns a numeric vector of mid-points of the groups along the x-axis.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

References

Accuracy (trueness and precision) of measurement methods and results – Part 2: Basic method for the determination of repeatability and reproducibility of a standard measurement method. ISO, Geneva (1994).

See Also

mandel.h, mandel.k, mandel.kh, pmandelh, pmandelk for probabilities, quantiles etc.

See barplot.mandel.kh for an alternative plotting method. gplot for the underlying plotting function.

Examples

```
data(RMstudy)

h <- with(RMstudy, mandel.h(RMstudy[2:9], g=Lab))
plot(h, las=2) # Lab 4 shows consistent low bias;
               # Lab 23 several extreme values.

#Use colours to identify particular measurands:
plot(h, las=2, col=1:8)
legend("bottomleft", legend=names(h), col=1:8, lty=1, cex=0.7, bg="white")

#Example of Mandel's k:
k <- with(RMstudy, mandel.k(RMstudy[2:9], g=Lab))
plot(k, las=2) # Lab 8 looks unusually variable;
               # Lab 14 unusually precise
```

---

plot.uncert *Plot method for 'uncert' objects*

Description

Plots for uncertainty budgets produced by uncert calls.

Usage

```
## S3 method for class 'uncert'
plot(x, which = c(1,2,4,5), main = paste(deparse(substitute(x))),
    ask = prod(par("mfcol")) < length(which) && dev.interactive(),
    caption = list("Variance and covariance contributions",
    expression(sqrt(group("|", "Variance and covariance contributions", "|"))),
    expression("Contribution " * u[i](y) == c[i] * u[i]),
    "Combined contribution", "Correlation (x,y)",
    "Covariances (x,y)"), cex.caption = 1, ...)
```

Arguments

x	An object of class uncert produced by a call to uncert().
which	Integer in 1:6; the particular variant(s) of plot required. A vector is permitted, in which case plots are produced in ascending order.
main	Main title for the plot
ask	logical; if 'TRUE', the user is _ask_ed before each plot, see 'par("ask=")'
caption	A list of captions for *all* the different plots.
cex.caption	Text size for captions. Note that if the number of figures per page is over 2, captions are further scaled by 0.8
...	Further parameters passed to plot() (mostly barplot).

Details

For uncert objects created with methods other than MC, the plot types are:

which=1 A barplot of all non-zero contributions to the combined uncertainty. These are derived from the covariance matrix and the coefficients c_i. For terms on the diagonal of the covariance matrix, these are $(c_i * u_i)^2$; for off-diagonal terms (the correlation terms), $2 * (c_i * u_i) * (c_j * u_j) * r_i j$. The threshold for deciding an off-diagonal term is nonzero is that its magnitude is greater than 2*.Machine$double.eps. Note that off-diagonal contributions may be negative.

which=2 As for which=1 except that the square root of the absolute value is plotted. For the 'diagonal' terms, these are just $u_i(y)$ in the nomenclature used by the GUM.

which=3 A barplot of $u_i(y)^2$, without the correlation terms.

which=4 A barplot of the sum of all (co)variance contributions associated with each x_i, that is,

$$contrib(i) = (c_i u_i)^2 + \sum_{j \neq i} 2(c_i u_i)(c_j u_j) r_{i,j}$$

.

which=5 A barplot of the theoretical correlations $cov(x_i, y)/u(x_i)u_y$.

which=6 A barplot of the theoretical covariances $cov(x_i, y)$.

Values of which outside this range are silently ignored.

For the X-Y correlation and covariance plots, the covariances are calculated from the covariance matrix V (supplied to uncert() as cov or calculated as outer(u,u,"*")*cor) and sensitivity coefficients c_i as $cov(x_i, y) = \sum_j V_{j,i} c_j$. In fact the calculation used is simpler: cov.xy <- V %*% ci. The correlations are calculated in turn from these using $cov(x_i, y)/u(x_i)u_y$.

Perhaps the most informative plots are for which=1, which=2, which=4 and which=5. The first of these includes all nonzero signed contributions, making the negative contributions visible; the second (which=2) makes direct comparison of magnitudes easier. The combined contribution plot is the effect on the total variance of removing all terms associated with a particular variable; it shows how much u_y^2 would *reduce* if the uncertainty for x_i were reduced to zero. Note that in some cases with negative correlation the combined uncertainty can *increase*, on dropping a variable, shown by a negative reduction in the plot. (which=5) is among the most direct indications of the relative importance of individual parameters.

Objects created with the MC method are passed to plot.uncertMC.

Value

Invisibly returns the default return value for the last plot produced.

Author(s)

S. L. R. Ellison, <s.ellison@lgcgroup.com>

References

None.

See Also

uncert, barplot, plot.uncertMC.

Examples

```
#An example with negative correlation
x <- list(a=1, b=3, c=2, d=11)
u <- lapply(x, function(x) x/10)
u.cor<-diag(1,4)
u.cor[3,4]<-u.cor[4,3]<- -0.5
u.form.c<-uncert(~a+b*2+c*3+d/2, x, u, method="NUM", cor=u.cor)

par(mfrow=c(3,2))
plot(u.form.c, which=1:6, las=1, horiz=TRUE)  #Note use of barplot parameters
```

---

plot.uncertMC *Plot method for 'uncertMC' objects*

Description

Plots for uncertainty evaluations produced by uncertMC or calls to uncert with method=MC.

Usage

```
## S3 method for class 'uncertMC'
plot(x, which = 1:2,
    main=paste("Monte Carlo evaluation -",deparse(substitute(x))),
    ask = prod(par("mfcol")) < length(which) && dev.interactive(),
    caption = list("Histogram", "Q-Q plot", "Density",
    "Correlation x-y", "Covariance x-y"),
    xlab = paste(deparse(substitute(x)), "$y", sep = ""),
    ..., cex.caption = 1, cex.main = 1.25, lwd.y = 2, col.y = 2, lty.y,
    col.qqline = NULL, lty.qqline = NULL, lwd.qqline = NULL)
```

Arguments

x	An object of class uncertMC produced by uncertMC() or uncert() with method="MC".
which	Integer in 1:5; the particular variant(s) of plot required. A vector is permitted, in which case plots are produced in ascending order of which.
main	Main title for the plot
ask	logical; if 'TRUE', the user is _ask_ed before each plot, see 'par("ask=")'
caption	A list of captions for *all* the different plots.
xlab	x-axis label, currently passed only to the histogram plot.
...	Additional parameters passed to other functions. See details for which parameters are passed.
cex.caption	Expansion factor for individual plot captions; as cex in par.
cex.main	Expansion factor for main title; as cex.main in par.
lwd.y, col.y, lty.y	Line width and colour for the location line in the histogram and density plots. Setting lwd.y=0 suppresses the location line.
col.qqline, lty.qqline, lwd.qqline	Graphical parameters for the Q-Q line in the Q-Q plot.

Details

For uncert objects created with methods other than MC, the plot types are:

which=1 A histogram of the MC replicates in xMCy, with optional line for xMCy. The histogram is produced using hist.default

which=2 A Q-Q plot of the MC replicates in xMCy, with Q-Q line. The plot uses qqnorm.default. If datax is not present (in '...'), it is set to TRUE.

which=3 A density plot of the MC replicates in xMCy. The plot calls density.default to calculate the density and plot.density to produce the plot.

which=4 A bar plot of $cor(x_i, y)$ if x$y is present. Any correlation method supported by stats::cor may be included in '...' (e.g as method="pearson".

which=5 A bar plot of $cov(x_i, y)$ if x$y is present. Any correlation method supported by stats::cov may be included in '...' (e.g as method="pearson".

Values outside 1:5 are silently ignored.

Parameters in '...' are passed to the various plot methods or calculations called. Only those parameters relevant to a given plot are passed to each calculation or plotting function, so '...' can include any parameter accepted by any of the functions called.

For the x-y correlation and x-y covariance plot, values in x$cor.xy are used if available. If not, stats::cor or stats::cov is called on values in xMCy and xMCx if the latter is available (i.e. uncertMC was called with keep.x=TRUE). If neither x$cor.xy nor x$MC$x is present, or if method is unknown, the plot is skipped with a warning.

Value

plot.uncertMC invisibly returns NULL.

Author(s)

S. L. R. Ellison, <s.ellison@lgcgroup.com>

References

None.

See Also

uncertMC-class, hist, qqnorm, qqline, density, plot.density

Examples

```
expr <- expression(a/(b-c))
x <- list(a=1, b=3, c=2)
u <- lapply(x, function(x) x/20)
set.seed(403)
u.invexpr<-uncertMC(expr, x, u, distrib=rep("norm", 3), B=999, keep.x=TRUE )
par(mfrow=c(2,2))
plot(u.invexpr, which=1:4, pch=20, method="k")
                              # method="k" gives Kendall correlation
```

---

pmsd *Median scaled difference probabilities and quantiles*

Description

Cumulative lower tail probability and quantile for median of scaled differences.

Usage

```
dmsd(q, n, method=c('fast', 'exact', 'even', 'asymp'), max.odd=199)

pmsd(q, n, lower.tail = TRUE,
method=c('fast', 'exact', 'even', 'asymp'), max.odd=199)

qmsd(p, n, lower.tail = TRUE,
method=c('fast', 'exact', 'even', 'asymp'), max.odd=199)
```

Arguments

q	Vector of quantiles.
p	Vector of probabilities.
n	Number of observations from which msd was calculated. Unused (and can be missing) for method="asymp"
lower.tail	logical; if TRUE (the default), probabilities are P[X <= x]; otherwise, P[X > x].
method	Calculation method. See details.
max.odd	Highest odd n for which exact values are calculated.

Details

pmsd, dmsd and qmsd return probabilities, densities and quantiles, respectively, for the median scaled difference applied to a single observation in a standard normal distribution, where otehr values are also IID normal.

n is the number of observations in the data set of interest and *not* the degrees of freedom or number of differences (msd for a value x[i] in a set of n observations involves n-1 scaled differences).

n, p and q are recycled to the length of the longest, as necessary.

method determines the method of calculation. For method="fast", probabilities are calculated using monotonic spline interpolation on precalculated probabilities. qmsd with method="fast" is obtained by root-finding on the corresponding spline function using uniroot, and densities are estimated from the first derivative of the interpolating spline. This provides fast calculation, and values for most practical probabilities are within 10^-6 of exact calculations. For high probabilites and for low quantiles (below 0.48) at high n, fast quantile accuracy is poorer due to the very low function gradients in this regions, but is still guaranteed monotonic with p.

For method="exact", probabilities and densities are calculated using quadrature integration for an order statistic. For odd n, this requires a double integral. Values for odd n accordingly take about an order of magnitude longer to obtain than for even n. This can be slow (seconds for a vector of several hundred values of q on an Intel x86 machine running at 1-2GHz). qmsd with method="exact" is obtained by root-finding from pmsd(..., method="excat") using uniroot, and is over an order of magnitude slower than pmsd pmsd.

For method="exact", asymptotic (large n) probabilities, densities and quantiles are returned. n is unused and can be missing.

For method="exact", odd n above max.odd are replaced with the next lower even value. This provides a fair approximation for n above 30 (though the fast method is better) and a good approximation above the default of 199. Values of max.odd above 199 are not recommended as integration can become unstable at high odd n; a warning is issued if max.odd > 199.

For method="even", an exact calculation is performed with any odd n replaced with the next lower even value. This is equivalent to setting method="exact" and max.odd=0. This is provided for interest only; the method="fast" method provides a substantially better approximation for odd n than method="even" and is faster.

Note that these functions are appropriate for the distribution of single values. If seeking an outlier test in a data set of size N, either adjust p for N comparisons before applying qmsd to find a critical value, or adjust the returned p-values using, for example, Holm adjustment.

Value

A vector of length length(p) or length(q) (or, if longer, length(n)) of cumulative probabilities, densities or quantiles respectively.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

See Also

msd for calculation of MSD values, and bootMSD for a parametric bootstrap (MCS) method of obtaining p-values and quantiles for the more general non-IID case.

Examples

```
data(Pb)
msd(Pb$value)          # Uses mad(Pb$value) as scale estimate
msd(Pb$value, Pb$u)    # Scales differences using standard uncertainties
```

`potassium` *Potassium data for two different materials included in an interlaboratory study*

Description

Potassium data for two different materials included in an interlaboratory study intended to provide data for certification of a reference material.

Usage

```
data("potassium")
```

Format

A data frame with 25 observations on the following 2 variables.

`QC` Potassium concentrations (mg/kg) reported on a material used as a quality control material

`RM` Potassium concentrations (mg/kg) reported on a candidate reference material material used as a quality control material

Details

Potassium data for two different materials included in an interlaboratory study intended to provide data for certification of a crab tissue reference material. The study included a previously certified reference material (near end of stock) to serve as a quality control (QC) check. Laboratories were asked to report five replicate measurements on the candidate reference material and three for the QC material. Each row in the data set corresponds to the mean of replicate results reported by each laboratory.

Inspection of the data suggests that one laboratory interchanged or mislabelled the test materials. The anomalous results appear as an outlier for both QC and RM, as well as being visible as an off-diagonal outlier in a Youden plot - see `youden.plot`).

Source

Private communication - Pending publication

Examples

```
data(potassium)
yplot(potassium)
```

`rbind.ilab` *Combine 'ilab' objects*

Description

Functions to combine ilab objects.

Usage

```
rbind(..., deparse.level = 1)
## Default S3 method:
rbind(..., deparse.level = 1)
## S3 method for class 'ilab'
rbind(..., deparse.level = 1)

## S3 method for class 'ilab'
c(..., recursive=FALSE)

cbind(..., deparse.level = 1)
## Default S3 method:
cbind(..., deparse.level = 1)
## S3 method for class 'ilab'
cbind(..., deparse.level = 1)
```

Arguments

`...`	For `rbind` and `c`, objects of class "ilab" to be combined. For `cbind`, one "ilab" object and vectors, scalars or data frames to be appended to the "ilab" object's $data element. See Details.
`deparse.level`	integer controlling the construction of labels. Passed to `rbind.data.frame` or `base::cbind`.
`recursive`	logical, controlling recursion in lists. Included solely for consistency with `base::c`; no effect in `c.ilab`.

Details

The generic and default cbind and rbind functions defined by metRology use the first object in `'...'` to decide which method to apply. This differs from the behaviour of these functions in the base package, which dispatch based on inspection of all objects in `'...'` (see `rbind` in the base package for details). Control is, however, passed to the base package functions if an ilab object is not first in the list. See section "Warning" below for a work-round if this causes difficulty.

The `rbind` method for class "ilab" combines objects by applying `rbind` to the `$data` elements in turn and then concatenating the `$distrib` and `$distrib.pars` elements using the default `c` method. All objects combined by `rbind.ilab` must be of class "ilab".

Combination of the `$data` elements follows the rules of `rbind.data.frame`; in particular, names must match, but need not be in the same order and the return value column classes will be coerced to match the first if necessary.

`c.ilab` simply passes the objects to `rbind.ilab`, using the default value for `deparse.level`. `recursive` is ignored. An error is returned if any objects in `'...'` are not of class "ilab".

The `cbind` method for "ilab" objects combines objects of class "ilab" with atomic objects or data frames by applying `base::cbind` to `$data` in the (single) supplied ilab object and the items listed in `'...'`. `base::cbind` will extend scalars, vectors or columns in data frames to length `nrow(ilab$data)` if their length is an integer fraction of `nrow(ilab$data)`. Unlike `base::cbind`, `cbind.ilab` does not permit vectors or data frames longer than `nrow(ilab$data)` and will return an error in such cases. cbind.ilab will also return an error if any objects in `'...'` are not one of atomic, data frame or class "ilab", if more than one 'ilab' object is supplied or if none are.

Value

An object of class 'ilab'. The title for the returned object is the title for the first ilab object in the list.

For the `cbind` method, the returned object will have additional columns in the `$data` element, and the title will be unchanged.

Warning

Because of the unusual method dispatch behaviour of `base::cbind` and `base::rbind`, which use the data frame method if *any* objects in `'...'` are data frames, metRology masks the base package functions in order to guarantee correct dispatch when data frames are included in `'...'`. No adverse effects are currently known, but please report any to the package maintainer. Calling `base::cbind` directly will provide a work-around if necessary.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None, yet.

See Also

base package functions `cbind`, `c`.

Examples

```
data(Pb)
il1<-construct.ilab(org=Pb$lab, x=Pb$value, measurand="Pb", item="none",
                u=Pb$u, k=Pb$k, U=Pb$U, title=c("CCQM K30", "Lead in wine"),
                method=Pb$method)

rbind(il1, il1)
```

REML location estimate

Restricted maximum likelihood estimate of location

Description

Calculates REML estimate of location, with standard error, assuming a random-effects model

Usage

```
reml.loc(x, ..., na.rm = FALSE)

## Default S3 method:
reml.loc(x, s, n = NULL, groups = NULL, na.rm = FALSE,
          tol=.Machine$double.eps^0.5, REML=TRUE, ...)
```

Arguments

x	numeric vector of mean values for groups, or (if groups is given) of individual observations
s	numeric vector of length length(x) of standard deviations or standard uncertainties associated with the values x.
n	integer giving the number of observations in each group. May be a vector of length length(x). If n is NULL, s is interpreted as a vector of standard uncertainties or standard errors. n is recycled to length(x)
groups	factor, or vetor which can be coerced to factor, of groups. If present, x is interpreted as a vector of individual observations and s and n ignored, if present, with a warning.
na.rm	logical: if TRUE, NA values are removed before processing.
tol	numeric tolerance for convergence, used by optimize().
REML	logical: if TRUE (the default), the function optimises the REML criterion (see Details). If FALSE, the maximum likelihood criterion is used.
...	Further parameters passed to optimize().

Details

reml.loc finds an excess variance τ^2 and location μ that maximise the restricted maximum likelihood criterion.

The estimator assumes a model of the form

$$x_i = \mu + b_i + e_i$$

in which b_i is drawn from $N(0, \tau^2)$ and e_i is drawn from $N(0, \sigma_i^2)$.

By default the function maximises the data-dependent part of the negative log restricted likelihood:

$$\frac{1}{2}\left(\sum_{i=1}^{k}\frac{(x_i-mu)^2}{u_i^2}+\sum_{i=1}^{k}log(u_i^2)+log\left(\sum_{i=1}^{k}(1/u_i^2)\right)\right)$$

where $u_i = s_i^2 + \tau^2$ and k is the number of mean values. If `REML=FALSE`, the final term is omitted to give the maximum likelihood criterion.

This implementation permits input in the form of:

- means `x` and standard errors `s`, in which case neither `n` nor `groups` are supplied;
- means `x`, standard deviations `s` and group size(s) `n`, standard errors then being calculated as `s/sqrt(n)`
- individual observations `x` with a groupinf factor `groups`, in which case standard errors are calculated from the groups using `tapply`.

Value

A loc.est object; see loc.est for details. In the returned object, individual values `xi` are always input means (calculated from groups and `n` as necessary); `method.details` is returned as a list containing:

mu The estimated location.

s The standard error in the location.

tau The excess variance (as a standard deviation).

REML Logical, giving the value of REML used.

Author(s)

S L R Ellison <s.ellison@lgcgroup.com>

References

None, but see documentation for the metafor package for a more general implementation of REML.

See Also

loc.est-class

Examples

```
#PCB measurements in a sediment from Key Comparison CCQM-K25
#s are reported standard uncertainties
pcb105 <- data.frame(x=c(10.21, 10.9, 10.94, 10.58, 10.81, 9.62, 10.8),
             s=c(0.381, 0.250, 0.130, 0.410, 0.445, 0.196, 0.093))

with( pcb105, reml.loc(x, s) )
```

RMstudy *Collaborative study results for metals in a reference material certification study*

Description

A data frame containing reported replicate results, in mg/L, for elements reported in an inter-laboratory certification study for a candiate drinking reference material.

Usage

```
data(RMstudy)
```

Format

A data frame with 145 observations on the following 9 variables.

`Lab` a factor with levels `Lab1` - `Lab29`

`Arsenic` a numeric vector

`Cadmium` a numeric vector

`Chromium` a numeric vector

`Copper` a numeric vector

`Lead` a numeric vector

`Manganese` a numeric vector

`Nickel` a numeric vector

`Zinc` a numeric vector

Details

The data set includes results for eight of a total of 23 elements studied in an inter-laboratory study of a candidate reference material. All observations are reported in ug/l. Laboratories were asked to report 5 replicate observations for each element. Replicate observations appear on separate rows. Most but not all laboratories reported five replicates, and some laboratories reported no results for some elements. The eight elements included in the data set are those for which no more than three laboratories failed to report any results. Missing observations are coded `NA`.

Laboratories were coded anonymously in order of receipt of results.

Source

LGC limited, Teddington, UK (Private communication).

`Scaled t distribution` *Scaled and shifted t distribution.*

Description

Student's t distribution for 'df' degrees of freedom, shifted by 'mean' and scaled by 'sd'.

Usage

```
dt.scaled(x, df, mean = 0, sd = 1, ncp, log = FALSE)
pt.scaled(q, df, mean = 0, sd = 1, ncp, lower.tail = TRUE, log.p = FALSE)
qt.scaled(p, df, mean = 0, sd = 1, ncp, lower.tail = TRUE, log.p = FALSE)
rt.scaled(n, df, mean = 0, sd = 1, ncp)
```

Arguments

`x`, `q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`df`	degrees of freedom (> 0, maybe non-integer). `df = Inf` is allowed.
`mean`	mean value for the shifted, scaled distribution.
`sd`	Scale factor for the shifted, scaled distribution.
`ncp`	non-centrality parameter delta; currently except for `rt()`, only for `abs(ncp) <= 37.62`. If omitted, use the central t distribution.
`lower.tail`	logical; if TRUE (default), probabilities are P[X <= x]; otherwise, P[X > x].
`log`, `log.p`	logical; if TRUE, probabilities p are given as log(p).

Details

These are wrappers for the corresponding t distribution functions in package `stats`.

The scaled, shifted t distribution has mean `mean` and variance `sd^2 * df/(df-2)`

The scaled, shifted t distribution is used for Monte Carlo evaluation when a value x has been assigned a standard uncertainty u associated with with df degrees of freedom; the corresponding distribution function for that is then `t.scaled` with `mean=x`, `sd=u` and `df=df`.

Value

`dt.scaled` gives the density, `pt.scaled` gives the distribution function, `qt.scaled` gives the quantile function, and `rt.scaled` generates random deviates.

Invalid arguments will result in return value `NaN`, with a warning.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

See Also

TDist

Examples

```
u<-rt.scaled(20, df=5, mean=11, sd=0.7)

qt.scaled(c(0.025,0.975), Inf, mean=10, sd=1) #10 +- 1.96*sd

require(graphics)
hist(rt.scaled(10000, df=4, mean=11, sd=0.7), breaks=50, probability=TRUE)
x<-seq(0,25, 0.05)
lines(x,dnorm(x,mean=11, sd=0.7), col=2)
```

Triangular *The triangular distribution.*

Description

Density, distribution function, quantile function and random generation for the triangular distribution with range 'min' to 'max' and mode equal to 'mode'.

Usage

```
dtri(x, min=-sqrt(6), max=sqrt(6), mode = (min + max)/2,
log = FALSE)

ptri(q, min=-sqrt(6), max=sqrt(6), mode = (min + max)/2,
lower.tail = TRUE, log.p = FALSE)

qtri(p, min=-sqrt(6), max=sqrt(6), mode = (min + max)/2,
lower.tail = TRUE, log.p = FALSE)

rtri(n, min=-sqrt(6), max=sqrt(6), mode = (min + max)/2)
```

Arguments

x, q	Vector of quantiles.
p	Vector of quantiles.
n	Number of observations. If 'length(n) > 1', the length is taken to be the number required.
min	Vector of lower limits of distribution.
max	Vector of upper limits of distribution.
mode	Vector of modes
lower.tail	logical; if TRUE (default), probabilities are P[X <= x]; otherwise, P[X > x].
log, log.p	logical; if TRUE, probabilities p are given as log(p).

Details

The triangular distribution has density

$$f(x) = 2 * (x - min)/((max - min) * (mode - min))(min < x <= mode)$$

$$f(x) = 2 * (max - x)/((max - min) * (max - mode))(mode < x < max)$$

and 0 elsewhere.

The mean is

$$\frac{1}{3}(min + mode + max)$$

and the variance is

$$\frac{1}{18}(min^2 + mode^2 + max^2 - min * mode - min * max - mode * max)$$

The default values of `min`, `max` and `mode` give a distribution with mean 0 and unit variance.

If `min>max`, `min` amd `max` will be silently interchanged. If mode is not within `[min, max]`, the functions return `NA`, with a warning.

`rtri` calls `runif(n, 0, 1)` to generate probabilities which are passed to `qtri`.

Value

A vector of densities, probabilities, quantiles or random deviates. `dtri` gives the density, `ptri` gives the distribution function, `qtri` gives the quantile function, and `rtri` generates random deviates.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

See Also

runif

Examples

```
require(graphics)
x<-seq(-3,3,0.02)

par(mfrow=c(2,1))
plot(x, dtri(x), type="l", main="Density")
plot(x, ptri(x), type="l", main="p(X<x)")

u <- rtri(5000)

var(rtri(10000,-1,1))  # ~ = 1/6 = 0.167
```

uncert *Uncertainty estimation functions*

Description

Functions for estimating measurement uncertainty from standard uncertainties and either sensitivity coefficients or (for some methods) expressions or functions. Correlation is supported via either a correlation or covariance matrix.

Usage

```
   uncert(obj, ...)

   ## Default S3 method:
uncert(obj, c, method = c("GUM", "MC"), cor,
            cov, distrib=NULL, distrib.pars=NULL, B=200, x=NULL, keep.x = TRUE,
            u=obj, ...)

   ## S3 method for class 'expression'
uncert(obj, x, u, method=c("GUM", "NUM", "kragten", "k2", "MC"),
            cor, cov, distrib=NULL, distrib.pars=NULL,
            B=200, delta=0.01, keep.x = TRUE, ...)

   ## S3 method for class 'function'
uncert(obj, x, u, method=c("NUM", "kragten", "k2", "MC"),
            cor, cov, distrib=NULL, distrib.pars=NULL,
            B=200, delta=0.01, keep.x = TRUE, ...)

   ## S3 method for class 'formula'
uncert(obj, x, u, method=c("GUM", "NUM", "kragten", "k2", "MC"),
            cor, cov, distrib=NULL, distrib.pars=NULL,
            B=200, delta=0.01, keep.x = TRUE, ...)
```

Arguments

obj	An R object used for method dispatch; see below. Methods currently exist for numeric vector, expression, function, or formula objects.
u	For the default method, a numeric vector of standard uncertainties. For the formula or expression methods, a named list of standard uncertainties.Note that for the default method, u is set to the value of obj, allowing specification of either as the *first* argument
c	A numeric vector of senstivity coefficients.
x	For the expression or formula methods, an R object which can be used as an environment by eval. For the function method, a list of parameters supplied to FUN via do.call.

method	Method of uncertainty evaluation. The current list of methods is: **GUM** First-order error propagation (the "law of propagation of uncertainty") as implemented by the GUM. **NUM** Numerical differentiation using a simple small step size. **kragten** Numerical estimation of uncertainty following Kragten (Kragten (1994). **k2** A symmetric modification of Kragten's approach described by Ellison (Ellison (2005)). **MC** Monte Carlo simulation.
cor, cov	A (square, symmetric) correlation or covariance matrix, respectively. If neither is specified, cor is set to the identity matrix.
distrib	For method="MC", a character vector of length length(x) or a named list of names of distribution functions associated with u. See Details for defaults. The list format may include user-specified functions. Silently ignored for other methods.
distrib.pars	For method="MC", a list of lists of parameters describing the distributions associated with u to be passed to the relevant distribution function. If distrib is present but distrib.pars is not, neither are included in the return value unless method="MC". See Details for defaults when method="MC". Silently ignored for other methods.
B	Number of Monte Carlo replicates.
delta	Step size for numerical differentiation.
keep.x	For method="MC", if keep.x=TRUE, the simulated replicates of x are included in the return object.
...	Additional parameters to be passed to a function (for the function method) or used in an expression (for expression or formula method).

Details

The default "GUM" method applies first-order error propagation principles to estimate a combined standard uncertainty from a set of sensitivity coefficients and *either* a set of standard uncertainties and a correlation matrix (which defaults to an identity matrix) *or* a covariance matrix. Both options use the same calculation, which is simply (t(c) %*% cov) %*% c ; standard uncertainties are first combined with the correlation matrix provided to form the covariance matrix. Since the correlation matrix defaults to the identity matrix, the default is combination without correlation.

The default method takes obj as a vector of uncertainty contributions unless u is specified, in which case u is used. It is not necessary to specify both. The expression method requires obj to be a differentiable R expression which can be evaluated in the environment x to provide a numeric value. For the function method, obj must be an R function which takes parameters from x and returns a numeric value. For the formula method, obj must be a formula with no left-hand side (e.g. ~a*x+b*x^2) which can be evaluated in the environment x to provide a numeric value.

The formula and expression methods first calculate derivatives for the expression or formula, evaluate them using the supplied values of x and then pass the resulting sensitivity coefficients, with supplied u, cor or cov to uncert.default.

The derivatives for the "GUM" method (formula and expression methods only) are algorithmic derivatives (that is, algebraic or analytical derivatives) obtained using deriv applied to expr and formula.

Numerical derivatives are computed in different ways depending on the method specified:

- For `method="NUM"`, the derivatives are calculated as $(f(x+u\delta) - f(x-u\delta))/(2u\delta)$.
- For `method="kragten"`, derivatives are calculated as $(f(x+sign(\delta)u) - f(x))/u$.
- For `method="k2"`, derivatives are calculated as $(f(x+u) - f(x-u))/(2u)$.

`"NUM"` is likely to give a close approximation to analytical differentiation provided that `delta` is appreciably less than 1 but not so small as to give step sizes near machine precision. `"k2"` is equivalent to `"NUM"` with `delta=1.0`. Both will give zero coefficients at stationary points (e.g minima), leading to under-estimation of uncertainty if the curvature is large. `"kragten"` uses a *deliberately* one-sided (and large) step to avoid this problem; as a result, `"kragten"` is a poorer (sometimes much poorer) estimate of the analytical differential but likely a better approximation to the truth.

Since these methods rely on `u`, if `u` is unspecified and `cov` is provided, `u` is extracted from `cov` (using `sqrt(diag(cov))`). It is assumed that the row and column order in `cov` is identical to the order of named parameters in `x`.

Derivatives (and uncertainty contributions) are computed for all parameters in `x`. Additional parameters used in `FUN`, `expr` or `formula` may be included in `...`; these will be treated as constants in the uncertainty calculation.

If `distrib` is missing, or if it is a list with some members missing, the distribution is assumed Normal and `distrib$name` is set to `"norm"`. Similarly, if `distrib.pars` or a member of it is missing, the default parameters for `x$name` are `list(mean=x$name, sd=u$name)`. If the list is not named, `names(x)` are used (so the list must be in order of `names(x)`).

If `method="MC"`, `uncert` calls `uncertMC`. Distributions and distribution parameters are required and B must be present and >1. See `uncertMC` for details of distribution specification.

For other evaluation methods, the distributions are silently ignored.

Value

An object of class 'uncert' or, for `method="MC"` of class 'uncertMC'. See `uncert-class` and `uncertMC-class` for details.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

JCGM 100 (2008) *Evaluation of measurement data - Guide to the expression of uncertainty in measurement*. doi:10.59161/JCGM1002008E. (JCGM 100:2008 is a public domain copy of ISO/IEC *Guide to the expression of uncertainty in measurement* (1995)).

Kragten, J. (1994) Calculating standard deviations and confidence intervals with a universally applicable spreadsheet technique, Analyst, **119**, 2161-2166.

Ellison, S. L. R. (2005) Including correlation effects in an improved spreadsheet calculation of combined standard uncertainties, Accred. Qual. Assur. **10**, 338-343.

See Also

uncert-class, deriv

For method="MC" see uncertMC and uncertMC-class.

Examples

```
expr <- expression(a+b*2+c*3+d/2)
x <- list(a=1, b=3, c=2, d=11)
u <- lapply(x, function(x) x/10)
u.expr<-uncert(expr, x, u, method="NUM")
u.expr

#Compare with default:
uncert(u=c(0.1, 0.3, 0.2, 1.1), c=c(1.0, 2.0, 3.0, 0.5))

#... or with function method
f <- function(a,b,c,d) a+b*2+c*3+d/2
u.fun<-uncert(f, x, u, method="NUM")
u.fun

#.. or with the formula method
u.form<-uncert(~a+b*2+c*3+d/2, x, u, method="NUM")
u.form

#An example with correlation
u.cor<-diag(1,4)
u.cor[3,4]<-u.cor[4,3]<-0.5
u.formc<-uncert(~a+b*2+c*3+d/2, x, u, method="NUM", cor=u.cor)
u.formc

#A Monte Carlo example
#See uncertMC for a less linear example
u.formc.MC<-uncert(~a+b*2+c*3+d/2, x, u, method="MC", cor=u.cor, B=200)
u.formc.MC
```

---

uncert-class *The 'uncert' class*

Description

Object returned by uncert calls.

Usage

```
   ## S3 method for class 'uncert'
print(x, digits=NULL, right=FALSE, ..., simplify=TRUE)

   ## S3 method for class 'uncert'
summary(object, ..., simplify=TRUE)
```

Arguments

x, object	An object of class uncert
digits	Number of digits to display in budget and (if present) distribution parameter lists. Passed to format for distribution parameter list and to print.data.frame for output.
right	If TRUE, strings in uncertainty budget are right-justified. This differs from the default in print.data.frame.
...	Other parameters passed to print.data.frame
simplify	If TRUE, only the call, evaluation method, budget, value y and combined uncertainty (u.y) are printed.

Details

summary.uncert simply calls print.uncert.

An object of class "uncert" contains:

call The matched call

y The calculated value (for function, expression or formula methods) or NA

u.y The combined standard uncertainty

method The uncertainty evaluation method used.

budget A data frame consisting of:

- **x** x (if supplied; otherwise a vector of NA's).
- **u** The standard uncertainties in input quantities (originally provided as u)
- **df** The degrees of freedom asscociated with u
- **c** Sensitivity coefficients either provided as c or (for the formula, function and expression methods) as calculated.
- **u.c** The product of u and c. These are the contributions to the combined uncertainty for uncorrelated quantities.

additional Any relevant parameters other than those in $budget$x (typically addditional constants passed to function or expression methods)

distrib If available, a named list of the distributions associated with u. The list contains either root nams of distribution functions (e.g "norm" or function definitions.

distrib.pars If available, a list of lists of parameters describing the distributions associated with u.

cov The covariance matrix used

cor The correlation matrix used

cov.xy A data frame of covariances between x and y. Row names correspond to the correlation method used. For all uncertainty evaluation methods but MC, the only correlation calculation is "theoretical"; for MC row names include all methods supported by stats::cor at the time the object was created.

cor.xy A data frame of correlations between x and y, of the same form as cov.xy

deriv For the formula and expression methods, the result of a call to deriv; an expression which evaluates to the value with attributes corresponding to the derivatives (that is, an expression which can be evaluated to give the value and sensitivity coefficients)

Value

`print` and `summary` methods invisibly return the original object.

Methods

print The print method provides a formatted printout of the object. By default, `simplify=TRUE`; this displays a shortened listing. Columns in `$budget` are suppressed if all NA (typically df when not specified).

summary summary is currently an alias for the print method.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None.

See Also

`uncert`, especially for calculation methods; `plot.uncert`, `uncertMC-class`, `print.data.frame`, `format`.

Examples

```
expr <- expression(a+b*2+c*3+d/2)
x <- list(a=1, b=3, c=2, d=11)
u <- lapply(x, function(x) x/10)
u.expr<-uncert(expr, x, u, method="NUM")
print(u.expr)
```

`uncertMC` *Monte Carlo evaluation of measurement uncertainty.*

Description

`uncertMC` estimates measurement uncertainty from a function, expression or formula by Monte Carlo simulation.

Usage

```
uncertMC(expr, x, u, method = "MC", df, cor, cov, distrib, distrib.pars,
   B = 200, keep.x = TRUE, vectorized=TRUE, ...)
```

Arguments

expr	An expression, function, or formula with no left-hand side (e.g. ~a*x+b*x^2) which can be evaluated in the environment x to provide a numeric value.
x	A named list or vector of parameters supplied to expr.
u	A named list or named vector of length length(x) of standard uncertainties.
method	Method of uncertainty evaluation. The only method currently supported by uncertMC is "MC". If any other method is specified, control is passed to uncert.
df	A named list or named vector of degrees of freedom. df can be a partial named list if not all distributions (see below) use degrees of freedom.
cor, cov	Optional (square, symmetric) correlation or covariance matrices, respectively. If neither is specified, uncertMC assumes independent variables.
distrib	A character vector of length length(x) or a named list of names of distribution functions associated with u. See Details for defaults.
distrib.pars	A named list of lists of parameters describing the distributions associated with u to be passed to the relevant distribution function. If distrib is present but distrib.pars is not, an attempt is made to set defaults based on other parameters; see Details.
B	Number of Monte Carlo replicates.
keep.x	If TRUE, the simulated replicates of x are included in the return object.
vectorized	If TRUE, expr is assumed to take vector arguments. If FALSE, expr is treated as if it takes scalar arguments. See Details for the difference.
...	Additional parameters to be passed to a function (for the function method) or used in an expression (for expression or formula method).

Details

Although most likely to be called by uncert, uncertMC may be called directly.

If any of x, u, df, distrib or distrib.pars are not lists, they are coerced to lists. If x is not named, arbitrary names of the form 'Xn' are applied. If u, df, distrib or distrib.pars do not have names, the names will be set to names(x) if they are of length exactly length(x); if not, an error is returned.

For Monte Carlo evaluation, distributions and distribution parameters are needed but defaults are used if some or all are absent. If distrib is missing, or if it is a list with some members missing, the distribution is assumed Normal and any missing member of distrib is set to "norm".

Distributions are usually identified by the root of the distribution function name; for example to specify the Normal, distrib$name="norm". At present, only the random value generator (e.g. rnorm) is used. Names of user-specified distributions functions can also be used, provided they have a random value generator named r<dist> where <dist> is the abbreviated distribution. Parameters are passed to distribution functions using do.call, so the function must accept the parameters supplied in distrib.pars.

If distrib.pars or members of it are missing, an attempt is made to deduce appropriate distribution parameters from x, u, df and distrib. In doing so, the following assumptions and values apply for the respective distributions:

norm `mean=x$name, sd=u$name`.

unif `min=x-sqrt(3)*u, max=x+sqrt(3)*u`.

tri `min=x-sqrt(6)*u, max=x+sqrt(6)*u, mode=x`.

t, t.scaled `df=df, mean=x, sd=u`.

If either `cor` or `cov` are present, a test is made to see if off-diagonal elements are significant. If not, `uncertMC` treats the values as independent. The test simply checks whether the sum of off-diagonal elements of `cor` (calculated from `cov` if `cov` is present) is bigger than `.Machine.double.eps*nrow^2`.

Correlation is supported as long as all correlated variables are normally distributed. If correlation is present, `uncertMC` follows a two-stage simulation procedure. First, variables showing correlation are identified. Following a check that their associated `distrib` values are all `"norm"`, `mvrnorm` from the MASS library is called to generate the simulated `x` values for those variables. Second, any remaining (i.e. independent) variables are simulated from their respective `distrib` and `distrib.pars`.

Vectorisation makes a difference to execution speed. If `vectorize=TRUE`, MC evaluation uses `eval` using the simulated data as the evaluation environment; if not, `apply` is used row-wise on the simulated input matrix. This makes an appreciable difference to execution speed (typically `eval` is faster by a factor of 5 or more) so the default assumes vectorised expressions. However, not all functions and expressions take vector arguments, especially user functions involving complicated arithmetic or numerical solutions. Use `vectorize=FALSE` for functions or expressions that do not take vector arguments. Note: One common symptom of an expression that does not take vector arguments is an R warning indicating that only the first element (typically of a parameter in `x`) is used. uncertMC may also return NA for `u` on attempting to take the sd of a single simulated point.

Value

An object of class `uncertMC`. See `uncertMC-class` for details.

Author(s)

S. L. R. Ellison `<s.ellison@lgcgroup.com>`

References

JCGM 100 (2008) *Evaluation of measurement data - Guide to the expression of uncertainty in measurement*. doi:10.59161/JCGM1002008E. (JCGM 100:2008 is a public domain copy of ISO/IEC *Guide to the expression of uncertainty in measurement* (1995)).

Kragten, J. (1994) Calculating standard deviations and confidence intervals with a universally applicable spreadsheet technique, Analyst, **119**, 2161-2166.

Ellison, S. L. R. (2005) Including correlation effects in an improved spreadsheet calculation of combined standard uncertainties, Accred. Qual. Assur. **10**, 338-343.

See Also

`uncert`, `uncert-class`, `uncertMC-class`

Examples

```
expr <- expression(a+b*2+c*3+d/2)
x <- list(a=1, b=3, c=2, d=11)
u <- lapply(x, function(x) x/10)
u.MC<-uncertMC(expr, x, u, distrib=rep("norm", 4), method="MC")
print(u.MC, simplify=FALSE)

#An example with correlation
u.cor<-diag(1,4)
u.cor[3,4]<-u.cor[4,3]<-0.5
u.formc.MC<-uncertMC(~a+b*2+c*3+d/2, x, u, cor=u.cor, keep.x=TRUE)
u.formc.MC

#A non-linear example
expr <- expression(a/(b-c))
x <- list(a=1, b=3, c=2)
u <- lapply(x, function(x) x/20)
set.seed(403)
u.invexpr<-uncertMC(expr, x, u, distrib=rep("norm", 3), B=999, keep.x=TRUE )
u.invexpr

#Look at effect of vectorize
system.time(uncertMC(expr, x, u, distrib=rep("norm", 3), B=9999, keep.x=TRUE ))
system.time(uncertMC(expr, x, u, distrib=rep("norm", 3), B=9999, keep.x=TRUE, vectorize=FALSE))
```

---

uncertMC-class *The 'uncertMC' class*

Description

Object returned by `uncertMC` calls and by `uncertainty` with `method="MC"`.

`summary.uncertMC` is currently an alias for `print.uncertMC`.

Usage

```
   ## S3 method for class 'uncertMC'
print(x, digits=NULL, right=FALSE,
                      ..., simplify=TRUE, minimise=FALSE)

   ## S3 method for class 'uncertMC'
summary(object, digits=NULL, right=FALSE,
                      ..., simplify=TRUE, minimise=FALSE)
```

Arguments

x, object	An object of class "uncertMC"
digits	Number of digits to display in budget and (if present) distribution parameter lists. Passed to format for distribution parameter list and to print.data.frame for output.
right	If TRUE, strings in uncertainty budget are right-justified. This differs from the default in print.data.frame.
...	Other parameters passed to print.data.frame
simplify	If TRUE, only the call, evaluation method, budget, value y and combined uncertainty (u.y) are printed.
minimise	If TRUE, the header, call, expr and evaluation method are suppressed; this is the mode used when printing an uncertMC object as part of an uncert object.

Details

An object of class "uncertMC" inherits from class "uncert". In addition to the contents of the "uncert" object, it contains the results from the MC replication as a list MC. The complete description is:

call The matched call

expr The expression, formula or function supplied to uncertMC.

method The uncertainty evaluation method used (always 'MC').

B The number of Monte Carlo replicates used.

budget A data frame consisting of:

- **x** The starting values x.
- **u** The standard uncertainties in input quantities (originally provided as u)
- **df** The degrees of freedom asscociated with u
- **c** Sensitivity coefficients estimated from the MC output (see uncertMC for how this is done).

distrib If available, a named list of the distributions associated with u. The list contains either root nams of distribution functions (e.g "norm" or function definitions.

distrib.pars If available, a list of lists of parameters describing the distributions associated with u.

additional If supplied, any relevant parameters other than those in $budget$x (typically addditional constants passed to function or expression methods)

cov The covariance matrix used

cor The correlation matrix used

cov.xy A data frame of covariances between x and y. The Row names correspond to the correlation method used. For uncertMC objects only "pearson" is currently supported (because "kendall" and "spearman" take a very long time to compute)

cor.xy A data frame of correlations between x and y, of the same form as cov.xy

MC A list containing:

- **seed** The value of .Random.seed when uncertMC was called.
- **y** The B Monte Carlo replicates of the standard uncertainty calculated as sd(y).
- **x.mc** If uncertMC is called with keep.x=TRUE, a data frame whose columns are the Monte Carlo replicates of the variables in x.

Value

`print` and `summary` methods invisibly return the original object.

Methods

print The print method provides a formatted printout of the object. By default, `simplify=TRUE`; this displays a shortened listing. Columns in `$data` are suppressed if all NA.

summary summary is currently an alias for the print method.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None.

See Also

uncert, uncertMC, uncert-class, print.data.frame, format.

Examples

```
set.seed(13*17)
expr <- expression(a+b*2+c*3+d/2)
x <- list(a=1, b=3, c=2, d=11)
u <- lapply(x, function(x) x/10)
u.expr<-uncertMC(expr, x, u, distrib=rep("norm", 4), method="MC")
print(u.expr)
```

---

update.uncert *Update and recalculate an uncertainty estimate*

Description

'update' allows modification of components of 'uncert' or 'uncertMC' objects, including the uncertainty estimation method used, and will recalculate the estimate and return a new 'uncert' or 'uncertMC' object. Individual elements of most components can be amended.

Usage

```
## S3 method for class 'uncert'
update(object, expr = NULL, method = NULL, x = NULL, u = NULL, c=NULL,
df = NULL, cov = NULL, cor = NULL, distrib = NULL,
distrib.pars = NULL, delta = NULL, B = NULL, keep.x = NULL, ...)
```

Arguments

object	An object of class 'uncert'
expr	An expression, formula or function.
method	Uncertainty evaluation method. May be any of the methods listed for uncert.
x, u, df	Named list, vector or array of values to update elements of object uncert. See Details for options.
c	Update to uncert$budget$c No effect except for updates using uncert.default.
cov, cor	A covariance or correlation matrix. Only one of u and cov should be specified; if both are specified, cov will take precedence and a warning will be issued.
distrib	Named list or character vector of updated distribution names.
distrib.pars	Named list of updates for distribution names and parameters.
delta	Scalar value updating delta for numeric evaluation methods.
B	Updated number of Monte Carlo iterations for 'uncertMC' objects or specification of B when updating 'uncertMC' objects using method="MC".
keep.x	Update to keep.x passed to uncertMC for Monte Carlo updates.
...	Other values passed to uncert or uncertMC

Details

Update will use the values provided to update the object given, call the original function with the revised parameters and return the result as an object of class 'uncert' or 'uncert' depending on the uncertainty evaluation method used.

Note that updating with a different value of method may result in an object of different class. Updating an 'uncertMC' object with a method other than "MC" will return an object of class 'uncert'; similarly, updating an 'uncert' object using method="MC" will return an object of class 'uncertMC'.

Updates to vector or list elements of uncert such as x, u, df etc. can be specified as named lists, named vectors or arrays, with names corresponding to names of the input quantities in the uncertainty budget (that is, the names may correspond to one or more of row.names(uncert$budget)). If names are present, only the corresponding individual members are updated. If names are not present, the complete vector or list in uncert is replaced, and names added.

Matrix elements cor and cov must be specified completely; see buildCor, updateCor and associated functions for covariance matrices for compact update methods.

Value

An object of class 'uncert' or, for method="MC" of class 'uncertMC'. See uncert-class and uncertMC-class for details.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

None, yet.

See Also

uncert-class, uncert-class, uncertMC, uncertMC-class

Examples

```
#From uncert:
expr <- expression(a+b*2+c*3+d/2)
x <- list(a=1, b=3, c=2, d=11)
u <- lapply(x, function(x) x/10)
u.expr<-uncert(expr, x, u, method="NUM")
u.expr

update(u.expr, u=list(a=0.3))

update(u.expr, method="MC")
```

---

vr.mle *Vangel-Rukhin Maximum Likelihood Estimate*

Description

Calculate a weighted mean, between-group standard deviation and standard error on the weighted mean using the Maximum likelihood algorithm of Vangel-Rukhin.

Usage

```
vr.mle(x, s2, n, init.mu = mean(x), init.sigma2 = var(x), labels = c(1:length(x)),
       max.iter = 1000, tol = .Machine$double.eps^0.5, trace = FALSE)

## S3 method for class 'summary.vr.mle'
print(x, ..., digits=3)
```

Arguments

x	numeric vector of the sample mean values of each group
s2	numeric vector of the sample variances of each group
n	integer vector of sample size of each group
init.mu	numeric initial value for the mean
init.sigma2	numeric initial value for the between-group component of variance
labels	vector of group names. Coerced to character on use.
max.iter	numeric maximum number of iterations
tol	numeric tolerance; iteration stops when the relative step size drops below 'tol'
trace	when TRUE shows the sequence of intermediate results
..., digits	Passed to format to control printed output.

Details

The Vangel-Rukhin MLE algorithm finds the between-method variance by iteratively solving the equation relating the weighted mean to the weighting factor applied. The weighting factor is the inverse of the sum of the standard error in 'x' and the between-method variance, scaled by the between-method variance.

For the default method, 's2' is interpreted as a vector of sample variances. 'x' is interpreted as a vector of sample means and the algorithm is applied to the corresponding group means, variances, and sample sizes.

The Vangel-Rukhin MLE algorithm shows an improvement in the number of iterations required to converge over the classical MLE based on the Score equations.

The function mle.1wre implements the MLE for the one way random effects based on the Fisher scoring equations and is provided for comparison purpose only.

Value

`vr.mle` returns an object of class "summary.vr.mle" which contains the following fields:

`mu`	the estimated mean
`var.mu`	the variance associated with the estimated mean
`sigma2`	the estimated between variance component
`llh`	the log likelihood of the estimates
`tot.iter`	the total number of iterations ran
`cur.rel.abs.error`	the current relative absolute error reached
`gammai`	a vector with the estimates of the weights
`converged`	TRUE is convergence criteria was met, FALSE otherwise
`reduced.model`	TRUE implies that a reduced model, with no between-group effect, is suggested, based on `sigma2==0`; FALSE indicates `sigma2 > 0`.

Author(s)

H. Gasca-Aragon

References

Vangel, M. G. and Rukhin, A. L. (1999), Biometrics, Vol 55, No. 1 pp 129-136

Searle, S. R., Cassella, G., and McCulloch, C. E. (1992). Variance Components. New York: Wiley.

See Also

`mle.1wre`, , `loc.est-class`

Examples

```
##=====================================================================
## the dietary fiber in apples example in the Vangel and Rukhin paper
##=====================================================================

m1 <- c(12.46, 13.035, 12.44, 12.87, 13.42, 12.08, 13.18, 14.335, 12.23)
s1 <- c(0.028, 0.233, 0.325, 0.071, 0.339, 0.325, 0.099, 0.064, 0.212)
n1 <- c(2, 2, 2, 2, 2, 2, 2, 2, 2)

res<- vr.mle(m1, s1^2, n1, tol=1e-6)

res$mu
sqrt(res$var.mu)
res$sigma2
res$mu+c(-1,1)*qnorm(0.975)*sqrt(res$var.mu)
res$tot.iter
res$converged
res$reduced.model

# output
# 12.90585
# 0.2234490
# 0.4262122
# 12.46790 13.34380
# 6
# converged = TRUE
# reduced.model = FALSE
```

`welch.satterthwaite` *Welch-Satterthwaite effective degrees of freedom*

Description

Provides the Welch-Satterthwaite effective degrees of freedom given standard uncertainties and associated degrees of freedom.

w.s is an alias for welch.satterthwaite.

Usage

```
w.s(ui, df, ci = rep(1, length(ui)), uc=sqrt(sum((ci*ui)^2)))

welch.satterthwaite(ui, df, ci = rep(1, length(ui)),
                        uc=sqrt(sum((ci*ui)^2)))
```

Arguments

`ui`	Standard uncertainties

df	Degrees of freedom
ci	Sensitivity coefficients dy/dx_i
uc	Combined standard uncertainty

Details

Implements the Welch-Satterthwaite equation as provided in the ISO Guide to the expression of uncertainty in measurement (1995) (See JCGM 100:2008). This assumes that uc is the uncertainty in a measurement result y, where $y = f(x_1, x_2, \dots)$, ci are the partial derivatives $\partial y/\partial x_i$ and ui is the standard uncertainty associated with xi.

The implementation assumes that the combined uncertainty uc is equal to sqrt(sum((ci*ui)^2). An independent estimate of uc can be provided.

The ci are 'sensitivity coefficients'; the default is 1, so that the ui can be given either as standard uncertainties in the values of influence quantities x_i, together with the associated ci, or as contributions ci*ui to the uncertainty in y.

Correlation is not supported, because the Welch-Satterthwaite equation is only valid for independent variances.

Value

The calculated effective degrees of freedom associated with uc.

Author(s)

S. L. R. Ellison <s.ellison@lgcgroup.com>

References

JCGM 100 (2008) *Evaluation of measurement data - Guide to the expression of uncertainty in measurement*. doi:10.59161/JCGM1002008E. (JCGM 100:2008 is a public domain copy of ISO/IEC *Guide to the expression of uncertainty in measurement* (1995)).

Satterthwaite, F. E. (1946), An Approximate Distribution of Estimates of Variance Components., Biometrics Bulletin **2**, 110-114, doi:10.2307/3002019

Welch, B. L. (1947), The generalization of "Student's" problem when several different population variances are involved., Biometrika **34** 28-35

See Also

None, yet.

Examples

```
u <- c(0.1, 0.3, 0.2, 1.1)
ci <- c(1.0, 2.0, 3.0, 0.5)
degfree <- c(Inf,6,8,3)

w.s(ui=u,df=degfree, ci=ci)
```

`xs.plot` *Mean/Standard deviation plot with confidence region.*

Description

Produces a plot of standard deviations versus means with a confidence region based on either robust or classical estimates of location and scale.

Usage

```
xs.plot(x, ...)

## Default S3 method:
xs.plot(x,g,s, degfree, labels.arg=NA, mu, sigma,
        probs=c(0.5, 0.95, 0.99), basis=c("robust","classical"),
        method=c("chisq","density"), main=paste("X-S plot -", basis, "basis"),
        xlab=deparse(substitute(x)), ylab,
        contours=TRUE, col.contours="lightgrey", lty.contours=par("lty"),
        lwd.contours=par("lwd"),
        label.contours=contours, format.clab="p=%3.2f",
        pos.clab="bottomright", col.clab=col.contours, cex.clab=0.7,
        cex.label=0.7, pos=3, adj=NULL,
        pch=par("pch"), col=par("col"), bg=par("bg"), cex=par("cex"),
        add=FALSE, ...)

XSplot(x, ...)
```

Arguments

`x`	Numeric vector of values to be plotted.
`g`	grouping factor of length `length(x)`.
`s`	numeric vector of standard deviations of length `length(x)` associated with `x`.
`degfree`	A single value for degrees of freedom associated with *all* the standard deviations `s`. Ignored if `s` is not supplied.
`labels.arg`	An optional vector of point labels, coerced to character on use.
`mu`	A single location used to centre the confidence region. The default is specified by `"basis"`; see Details.
`sigma`	A measure of dispersion against which deviations x-mu can be compared.
`probs`	A vector of probabilities for confidence region contours.
`basis`	Controls the nature of the location and scale estimators used to produce the confidence contours drawn on the plot. See Details for specification.
`method`	The method used to calculate the confidence region. See Details.
`main`	Main title for the plot.
`xlab, ylab`	x- and y-axis labels,

contours	logical, specifying whether confidence contours should be drawn.
col.contours, lty.contours, lwd.contours	Colour, line type and line width for contour lines.
label.contours	Logical, controlling whether contour lines are labelled with approximate probabilities.
format.clab	Format string for contour labels, passed to sprintf.
pos.clab	Specification for location of contour labels. A vector can be provided to give multiple labels. See Details for further description.
col.clab, cex.clab	Colour and expansion for contour labels.
pch, col, bg, cex	Graphical parameters passed to points.
cex.label	Expansion factor for point labels, passed to text.
pos, adj	Specifies position/adjustment of point labels. Passed to text; see text for details.
add	If TRUE the plot region is not cleared before plotting; points and contours are added to the present plot. Use pch=NA to suppress symbols if only added contours are required.
...	Other parameters passed to plot.

Details

A plot of standard deviations against locations is produced, together with optional confidence region(s) calculated (by default) by a method suggested in ISO 13528:2005.

If s is supplied, x is taken as a vector of locations and s a vector of standard deviations. degfree must be supplied in this case.

If g is supplied and s is not, the locations and standard deviations used are the means and standard deviations for each group. degfree is calculated from the median group size. Groups should, of course, be of the same size for accurate inference; however, using the median group size allows for some groups with missing values.

If s and g are both supplied, g is ignored with a warning

If requested by contours=TRUE, confidence regions are drawn for each value of probs. Contour location and shape are controlled by basis which specifies the location and scale estimators used, and method, which specifies the method of calculation for the contours. Two methods are supported; one using the chi-squared distribution (method="chisq") and one based on equal density countours (method="density"). The default, and the method recommended by the cited Standard, is method="chisq" and basis="robust".

Both calculations for confidence regions require estimation of a location $\hat{\mu}$ and an estimate $\hat{\sigma}$ of the pooled within-group standard deviation or pooled estimate from s. If basis="robust", $\hat{\mu}$ and $\hat{\sigma}$ are calculated using algA and algS respectively. If basis="classical", $\hat{\mu}$ and $\hat{\sigma}$ are the mean of the group means and the classical pooled standard deviation respectively. If mu or sigma are given, these are used in place of the calculated $\hat{\mu}$ and $\hat{\sigma}$ respectively.

If method="chisq", contours for probability p are calculated as

$$s = \hat{\sigma} \exp\left(\pm \frac{1}{\sqrt{2(n-1)}} \sqrt{\chi^2_{2,p} - n\left(\frac{x-\hat{\mu}}{\hat{\sigma}}\right)^2} \right)$$

for x from $\hat{\mu} - \hat{\sigma}\sqrt{\frac{\chi^2_{2,p}}{n}}$ to $\hat{\mu} + \hat{\sigma}\sqrt{\frac{\chi^2_{2,p}}{n}}$.

If `method="density"`, contours for probability p are calculated using Helmert's distribution to provide constant likelihood contours round the chosen mean and standard deviation. In the present implementation, these are found using `uniroot` to find the mean corresponding to the required density at given standard deviations. The density chosen is $d_{max}(1-p)$ where p is the probability and d_{max} the maximum density for Helmert's distribution for the requisite nunber of degrees of freedom. (See Kruskal (1946) for a description of Helmert's distribution and, for example, Pawitan (2001) for the rationale behind the choice of density contour level.) This seems to give reasonably good results for $n \geq 3$ but is anticonservative (particularly to high s) for $n = 2$.

Contours are by default labelled. Label locations can be specified using `pos.clab`. Options are `"top"`, `"topright"`, `"right"`, `"bottomright"`, `"bottom"`, `"bottomleft"`, `"left"` and `"topleft"`. A vector can be specified to give labels at more than one such location. Contour labels are usually placed approximately at the location(s) indicated and adjusted outward appropriately. For the special case of `method="density"` and `degfree=1` (or where group sizes $n = 1$), for which the region is a maximu width at s=0, `"bottomright"` and `"bottomleft"` place labels immediately below the countour boundary at $s = 0$ and, if specified, `"bottom"` is replaced with `c("bottomright", "bottomleft")`.

`XSplot` is an alias for `xs.plot`.

Value

A list with components:

x, y respectively, the plotted locations and standard deviations. (the names allow a simple call to `plot()`)

mu The location and pooled SD estimates $\hat{\mu}$ and $\hat{\sigma}$ used to construct the confidence ellipsoids.

clist A list of sets of coordinates for each confidence region.

Author(s)

S Ellison <s.ellison@lgcgroup.com>

References

ISO 13528:2005, Statistical methods for use in proficiency testing by interlaboratory comparisons, International Organization for Standardization, Geneva (2005)

Y Pawitan, (2001) In all likelihood: Statistical Modelling and Inference Using Likelihood,Clarendon Press, Oxford, pp258-9

W Kruskal, American Mathematical Monthly 53, 435-438, (1946)

See Also

algA, algS

axis for axis control; points, text for plotting parameters; sprintf for contour label formatting.

duewer.plot for an alternative plot for locations and associated standard errors or standard uncertainties;

Examples

```
require(metRology)
set.seed(1017)
x <- rnorm(80)
g <- gl(20,4)

xs.plot(x,g)

#Identical plot with precalculated s:
X <- tapply(x,g,mean)
S <- tapply(x,g,sd)
xs.plot(X, s=S, degfree=3)

#Specify different location and within-group SD estimates:
xs.plot(X, s=S, degfree=3, mu=median(X), sigma=median(S))

#Illustrate multiple contour labelling, point labels and further embellishment
rv <- xs.plot(x,g, pos.clab=c("bottomleft", "bottomright"), labels=TRUE)
abline(v=rv$mu, h=rv$s, col=2)
```

---

youden.plot *Youden plots*

Description

A Youden plot is a bivariate scatter plot, named for its use by W. M Youden in interlaboratory studies. This implementation includes data ellipses based on Pearson, Spearman, Kendall or several robust covariance measures.

Usage

```
youden.plot(x, ...)

yplot(x, ...)

## Default S3 method:
youden.plot(x, y = NULL, type = c("points", "labels", "both", "outliers"),
labels, probs = c(0.95, 0.99), x0, y0,  pch = par("pch"), cex = par("cex"),
col = par("col"), bg = par("bg"), main, xlab, ylab,
```

```
xlim = c("data", "ellipse", "all"), ylim = c("data", "ellipse", "all"),
col.axes = 2, lwd.axes = 1, lty.axes = 1, cex.lab = 0.7, pos = 3,
out.method = c("F", "chisq", "n"), n.out, p.out = 0.99,
add = FALSE,  ...)
```

Arguments

x	An R numeric object. Can be a vector (in which case y must be specified and of the same length) or a two-column numeric matrix.
y	A numeric vector of the same length as x. It is an error to provide y in addition to a two-column matrix for x.
type	The type of plot produced. See Details.
labels	Character vector of text labels for data points. Defaults to row.names(x) if x is a matrix or data frame with row names, names(x) if x is a named vector, and to 1:length(x) or 1:nrow(x) as appropriate if x does not have names.
probs	Numeric vector of probabilities for data ellipses.
x0, y0	If specified, data ellipses will be centred on (x0, y0) instead of using the location calculated from the data.
pch, cex, col, bg	passed to points; see points for details.
main, xlab, ylab	Plot titles. If missing, titles are based on the names of the objects plotted.
xlim, ylim	Specifications for horizintal and vertical plot limits. Each can be either a length 2 numeric vector (as usual) or a character value matching one of "data", "ellipse" or "all". If "data", the relevant limits are set to include the range of the data. If "ellipse", the relevant limits are set to include the whole of the outermost ellipse. If "all", limits are set to include both the data and the outermost ellipse.
col.axes, lwd.axes, lty.axes	Colour, line width and line type for vertical and horizontal location markers drawn through the ellipse centre.
cex.lab	Size for data point labels; see text for details.
pos	a position specifier for data point labels; see text for details.
out.method	Character specifying outlier marking. See Details.
n.out	Number of outliers marked if out.method=="n".
p.out	Confidence level at which points are marked as outliers if out.method is one of "F" or "chisq".
add	If TRUE, ellipses and points are added to an existing plot.
...	Named arguments passed to other functions. In particular: • Arguments cov.method, cov.control, scalefn, and locfn will be passed to cov.dellipse to control the (optionally robust) location, scale and covariance estimates used for the ellipses. • Arguments sub, ann, axes, frame.plot, and asp are passed to plot to add a sub-title, control axes and annotation and, importantly, control the plot aspect ratio. • Arguments col.ellipse, lty.ellipse, lwd.ellipse, fill, density, angle, npoints, prinax, col.prinax, lty.prinax, and lwd.prinax are passed to plot.d.ellipse to control the appearance of ellipses and ellipse principal axes.

Details

type controls the type of plot produced. Allowed types and their effect are:

points Points only are drawn.

labels Point labels only are drawn

both Points are drawn with labels

outliers Points are drawn and outlying points are labelled (see below)

Ellipses are constructed based on a location and covariance matrix constructed from the data by the method specified by cov.method. probs specifies the approximate coverage. See data.ellipse for details of covariance methods and ellipse specification.

The outlier identification method, if any, is specified by out.method and controlled by one of n.out or p.out. If out.method is "F" or "chisq", points with Mahalanobis distance greater than an upper critical value with probability p.out are considered to be outliers. The critical values used are

"F" Mahalanobis distance greater than 2 * (n-1) * qf(p.out, 2, n-1) / (n-2)

"chisq" Mahalanobis distance greater than qchisq(p.out, 2) which(md > 2 * (n-1) * qf(p.out, 2, n-1) / (n-2)) #F dist

The Mahalanobis distance is calculated based on the covariance matrix used to consstruct plot ellipses.

If out.method is "n", the outermost n.out points (judged by Mahalanobis distance) are marked as outliers. Specifying out.method="n" and n.out=0 suppresses outlier identification. If outliers are marked, a list of marked outliers is included in the returned list (see Value, below).

yplot is an alias for youden.plot

Value

Invisibly returns the plotted data ellipses as an object of class d.ellipse.

Author(s)

S L R Ellison (s.ellison@lgcgroup.com)

References

Youden, W.J. and Steiner, E.H. (1975) *Statistical Manual of the AOAC*. AOAC International, Washington, US. ISBN 0-935584-15-3

ISO 13528:2005, Statistical methods for use in proficiency testing by interlaboratory comparisons, International Organization for Standardization, Geneva (2005)

See Also

d.ellipse

Examples

```
data(chromium)
data(potassium)

( yy <- youden.plot(chromium, main="Chromium") )

#With outlier ID (F based)
youden.plot(chromium, main="Chromium", xlim='a', ylim='a', type='o', p.out=0.95)
#Note use of xlim="a" etc. to ensure both ellipses and data are included.

#Top 5 most distant outliers (5 is also the default)
youden.plot(chromium, main="Chromium", xlim='a', ylim='a', type='o', out.method="n", n.out=5)

#With ellipse principal axes
#(useful to specify asp=1 or the axes will not always appear orthogonal)
youden.plot(chromium, main="Chromium",
xlim='a', ylim='a', type='o', p.out=0.99, prinax=TRUE, lty.prinax=2, asp=1.0)

youden.plot(potassium, main="Potassium")

#A different pairs plot ...
panel.youden <- function(x, y, ...) youden.plot(x, y, add=TRUE, type="o", cex=1, pos=1, p.out=0.95)
pairs(chromium, upper.panel=panel.youden)
```

Index