Package 'MethComp'

January 19, 2020

Version 1.30.0

Date 2020-01-14

Title Analysis of Agreement in Method Comparison Studies

Depends R (>= 3.5)

Imports coda, nlme, rjags

Suggests lattice, lme4

Description Methods (standard and advanced) for analysis of agreement between measurement methods. These cover Bland-Altman plots, Deming regression, Lin's Total deviation index, and difference-on-average regression. See Carstensen B. (2010) ``Comparing Clinical Measurement Methods: A Practical Guide (Statistics in Practice)'' <doi:10.1002/9780470683019> for more information.

License GPL (>= 2)

Encoding UTF-8

URL http://BendixCarstensen.com/MethComp/

BugReports https://github.com/ekstroem/MethComp/issues

SystemRequirements JAGS (http://mcmc-jags.sourceforge.net)

NeedsCompilation no

Repository CRAN

ByteCompile true

RoxygenNote 7.0.2

Author Bendix Carstensen [aut],
Lyle Gurrin [aut],
Claus Thorn Ekstrøm [aut, cre],
Michal Figurski [aut]

Maintainer Claus Thorn Ekstrøm <ekstrom@sund.ku.dk>

Date/Publication 2020-01-19 18:30:10 UTC

R topics documented:

abconv *Derive linear conversion coefficients from a set of indeterminate coefficients*

Description

If a method comparison model is defined as $y_{mi} = \alpha_m + \beta_m \mu_i, m = 1,2$ y_mi = alpha_m + beta_m*mu_i, m=1,2 the coefficients of the linear conversion from method 1 to 2 are computed as: $\alpha_{2|1} = -\alpha_2 - \alpha_1 \beta_2 / \beta_1$ alpha_(2|1) = -alpha_2-alpha_1*beta_2/beta_1 $\beta_{2|1} = \beta_2 / \beta_1$ Morover the the point where the linear conversion function intersects the identity line is computed too.. The function is designed to work on numerical vectors of posterior samples from BUGS output.

Usage

```
abconv(
  a1,
  b1 = 1:4,
  a2 = NULL,
  b2 = NULL,
  col.names = c("alpha.2.1", "beta.2.1", "id.2.1")
)
```

Arguments

a1	Numerical vector of intercepts for first method. Alternatively a dataframe where the vectors are selected from.
b1	Numerical vector of slopes for first method. If a1 is a dataframe, b1 is assumed to be a numerical vector of length 4 pointing to the columns of a1 with the intercepts and slopes.
a2	Numerical vector of intercepts for second method.
b2	Numerical vector of slopes for second method.
col.names	Names for the resulting three vectors.

Value

A dataframe with three columns: intercept and slope for the conversion from method 1 to method 2, and the value where the conversion is the identity.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://BendixCarstensen.com

References

B Carstensen: Comparing and predicting between several methods of measurement, Biostatistics, 5, pp 399-413, 2004

See Also

BA.plot, MCmcmc

Examples

```
abconv( 0.3, 0.9, 0.8, 0.8 )
```

AltReg *Estimate in a method comparison model with replicates*

Description

Estimates in the general model for method comparison studies with replicate measurements by each method, allowing for a linear relationship between methods, using the method of alternating regressions.

Usage

```
AltReg(
  data,
  linked = FALSE,
  IxR = linked,
  MxI = TRUE,
  varMxI = FALSE,
  eps = 0.001,
  maxiter = 50,
  trace = FALSE,
  sd.lim = 0.01,
  Transform = NULL,
  trans.tol = 1e-06
)
```

Arguments

`data` Data frame with the data in long format, (or a `Meth` object) i.e. it must have columns `meth`, `item`, `repl` and `y`

`linked` Logical. Are the replicates linked across methods? If true, a random `item` by `repl` is included in the model, otherwise not.

`IxR` Logical, alias for linked.

`MxI` Logical, should the method by item effect (matrix effect) be in the model?

`varMxI` Logical, should the method by item effect have method-specific variances. Ignored if only two methods are compared. See details.

`eps` Convergence criterion, the test is the max of the relative change since last iteration in both mean and variance parameters.

`maxiter` Maximal number of iterations.

`trace` Should a trace of the iterations be printed? If `TRUE` iteration number, convergence criterion and current estimates of means and sds are printed.

`sd.lim` Estimated standard deviations below `sd.lim` are disregarded in the evaluation of convergence. See details.

`Transform` A character string, or a list of two functions, each other's inverse. The measurements are transformed by this before analysis. Possibilities are: "exp", "log", "logit", "pctlogit" (transforms percentages by the logit), "sqrt", "sq" (square), "cll" (complementary log-minus-log), "ll" (log-minus-log). For further details see `choose.trans`.

`trans.tol` The tolerance used to check whether the supplied transformation and its inverse combine to the identity. Only used if `Transform` is a list of two functions.

Details

When fitting a model with both IxR and MxI interactions it may become very unstable to have different variances of the MxI random effects for each method, and hence the default option is to have a constant MxI variance across methods. On the other hand it may be grossly inadequate to assume these variances to be identical.

If only two methods are compared, it is not possible to separate different variances of the MxI effect, and hence the `varMxI` is ignored in this case.

The model fitted is formulated as:

$$y_{mir} = \alpha_m + \beta_m(\mu_i + a_{ir} + c_{mi}) +$$

$$e_{mir}$$

and the relevant parameters to report are the estimates sds of a_{ir} and c_{mi} multiplied with the corresonidng β_m. Therefore, different values of the variances for MxI and IxR are reported also when `varMxI==FALSE`. Note that `varMxI==FALSE` is the default and that this is the opposite of the default in `BA.est`.

Value

An object of class `c("MethComp","AltReg")`, which is a list with three elements:

`Conv` A 3-way array with the 2 first dimensions named "To:" and "From:", with methods as levels. The third dimension is classifed by the linear parameters "alpha", "beta", and "sd".

`VarComp` A matrix with methods as rows and variance components as columns. Entries are the estimated standard deviations.

`data` The original data used in the analysis, with untransformed measurements (ys). This is needed for plotting purposes.

Moreover, if a transformation was applied before analysis, an attribute "Transform" is present; a list with two elements `trans` and `inv`, both of which are functions, the first the transform, the last the inverse.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <bendix.carstensen@regionh.dk>, http://BendixCarstensen.com.

References

B Carstensen: Comparing and predicting between several methods of measurement. Biostatistics (2004), 5, 3, pp. 399–413.

See Also

`BA.est`, `DA.reg`, `Meth.sim`, `MethComp`

Examples

```
data( ox )
ox <- Meth( ox )
## Not run:
ox.AR <- AltReg( ox, linked=TRUE, trace=TRUE, Transform="pctlogit" )
str( ox.AR )
ox.AR
# plot the resulting conversion between methods
plot(ox.AR,pl.type="conv",axlim=c(20,100),points=TRUE,xaxs="i",yaxs="i",pch=16)
# - or the rotated plot
plot(ox.AR,pl.type="BA",axlim=c(20,100),points=TRUE,xaxs="i",yaxs="i",pch=16)
## End(Not run)
```

`Ancona` *Data from a rating experiment of recorgnizing point counts.*

Description

At the course "Statsitical Analysis of Method Comparison Studies" ai the SISMEC conference in Ancona, on 28 September 2011, the participants on the course were used as raters of ten pictures of points. Pitures were shown 3 times each to the participants, and they assessed the number of points in each.

Format

A data frame with 510 observations on the following 4 variables.

`rater` a factor with 17 levels

`item` a numeric vector indicating the pictures shown. The value is the actual number of points.

`repl` a numeric vector, replicate number

`score` a numeric vector, the number of points in `item`

Source

The course "Statsitical Analysis of Method Comparison Studies" ai the SISMEC conference in Ancona, on 28 September 2011.

Examples

```
library( MethComp )
data( Ancona )
Anc <- Meth( Ancona, 1, 2, 3, 4 )
```

`BA.est` *Bias and variance components for a Bland-Altman plot.*

Description

A variance component model is fitted to method comparison data with replicate measurements in each method by item stratum. The purpose is to simplify the construction of a correct Bland-Altman-plot when replicate measurements are available, and to give the REML-estimates of the relevant variance components.

Usage

```
BA.est(
  data,
  linked = TRUE,
  IxR = has.repl(data),
  MxI = has.repl(data),
  corMxI = FALSE,
  varMxI = TRUE,
  IxR.pr = FALSE,
  bias = TRUE,
  alpha = 0.05,
  Transform = NULL,
  trans.tol = 1e-06,
  random.raters = FALSE,
  lmecontrol = lmeControl(msMaxIter = 300),
  weightfunction = c("mean", "median")
)
```

Arguments

`data` A `Meth` object representing method comparison data with replicate measurements, i.e. with columns `meth`, `item`, `repl` and `y`.

`linked` Logical. Are replicates linked within item across methods?

`IxR` Logical. Should an item by repl interaction be included in the model. This is needed when the replicates are linked within item across methods, so it is just another name for the `linked` argument. If `linked=` is given, this is ignored.

`MxI` Logical. Should the method by item interaction (matrix effect) be included in the model.

`corMxI` Logical. Should the method by item interaction allow coorelated effects within item. Ignored if only two methods are compared.

`varMxI` Logical. Should the method by item interaction have a variance that varies between methods. Ignored if only two methods are compared.

`IxR.pr` Logical. Should the item by repl interaction variation be included in the prediction standard deviation?

`bias` Logical. Should a systematic bias between methods be estimated? If `FALSE` no bias between methods are assumed, i.e. $\alpha_m = 0, m = 1, \dots M$.

`alpha` Numerical. Significance level. By default the value 2 is used when computing prediction intervals, otherwise the $1-\alpha/2$ t-quantile is used. The number of d.f. is taken as the number of units minus the number of items minus the number of methods minus 1 ($I - M - 1$).

`Transform` Transformation applied to data (y) before analysis. See `check.trans` for possible values.

`trans.tol` Numerical. The tolerance used to check whether the supplied transformation and its inverse combine to the identity.

`random.raters` Logical. Should methods/raters be considered as random. Defaults to `FALSE` which corresponds to a fixed effect of methods/raters.

`lmecontrol` A list of control parameters passed on to `lme`.

`weightfunction` Function to weigh variance components for random raters. Defaults to `mean` but can also be `median`.

Details

The model fitted is:

$$y = \alpha_m + \mu_i + c_{mi} + a_{ir} + e_{mir},$$
$$\text{var}(c_{mi}) = \tau_m^2,$$
$$\text{var}(a_{ir}) = \omega^2,$$
$$\text{var}(e_{mir}) = \sigma_m^2,$$

We can only fit separate variances for the τs if more than two methods are compared (i.e. `nM` > 2), hence varMxI is ignored when `nM`==2.

The function `VC.est` is the workhorse; `BA.est` just calls it. `VC.est` figures out which model to fit by `lme`, extracts results and returns estimates. `VC.est` is also used as part of the fitting algorithm in `AltReg`, where each iteration step requires fit of this model. The function `VC.est` is actually just a wrapper for the functions `VC.est.fixed` that handles the case with fixed methods (usually 2 or three) i.e. the classical method comparison problem, and `VC.est.random` that handles the situation where "methods" are merely a random sample of raters from some population of raters; and therefore are regarded as random.

Value

`BA.est` returns an object of class `c("MethComp","BA.est")`, a list with four elements `Conv`, `VarComp`, `LoA`, `RepCoef`; `VC.est` returns (invisibly!) a list with elements `Bias`, `VarComp`, `Mu`, `RanEff`. These list components are:

`Conv` 3-dimensional array with dimensions "To", "From" and unnamed. The first two dimensions have the methods compared as levels, the last one `c("alpha","beta","sd.pred","LoA: lower","upper")`. It represents the mean conversions between methods and the prediction standard deviation.

Where "To" and "From" take the same value the value of the "sd" component is $\sqrt{2}$ times the residual variation for the method. If `IxR.pr=TRUE` the variation between replicates are included too, i.e. $\sqrt{2(\sigma_m^2 + \omega^2)}$ sqrt[2(sigma_m^2+omega^2)].

`VarComp` A matrix of variance components (on the SD scale) with methods as rows and variance components "IxR", "MxI" and "res" as columns.

`LoA` Four-column matrix with mean difference, lower and upper limit of agreement and prediction SD. Each row in the matrix represents a pair of methods.

`RepCoef` Two-column matrix of repeatability SDs and repeatability coefficients. The SDs are the standard deviation of the difference between two measurements by the same method on the item under identical circumstances; the repeatability coefficient the numerical extent of the prediction interval for this difference, i.e. $2\sqrt{2}$ times the sd.

Mu Estimates of the item-specific parameters.

RanEff Estimates of the random effects from the model (BLUPS). This is a (possibly empty) list with possible elements named MxI and IxR according to whether these random effects are in the model.

The returned object has an attribute, Transform with the transformation applied to data before analysis, and its inverse — see choose.trans.

Author(s)

Bendix Carstensen

References

Carstensen, Simpson & Gurrin: Statistical models for assessing agreement in method comparison studies with replicate measurements, The International Journal of Biostatistics: Vol. 4 : Iss. 1, Article 16. http://www.bepress.com/ijb/vol4/iss1/16.

See Also

BA.plot, perm.repl

Examples

```
data( ox )
ox <- Meth( ox )
summary( ox )
BA.est( ox )
BA.est( ox, linked=FALSE )
BA.est( ox, linked=TRUE, Transform="pctlogit" )
## Not run:
data( sbp )
BA.est( sbp )
BA.est( sbp, linked=FALSE )
# Check what you get from VC.est
str( VC.est( sbp ) )
## End(Not run)
```

BA.plot *Bland-Altman plot of differences versus averages.*

Description

For two vectors of equal length representing measurements of the same quantity by two different methods, the differences are plotted versus the average. The limits of agreement (prediction limits for the differences) are plotted, optionally a regression of differences of means is given too. Works with `Meth` and `MethComp` objects too.

A plot method for the "PBreg" class object, that is a result of Passing-Bablok regression.

When a method comparison model i fitted and stored in a `MCmcmc` object, then the posterior distributions of the variance components are plotted, in separate displays for method.

Usage

```
BA.plot(
  y1,
  y2,
  meth.names = NULL,
  wh.comp = 1:2,
  pl.type = "BA",
  dif.type = "const",
  sd.type = "const",
  model = if (inherits(y1, "Meth") & has.repl(y1)) "exch" else NULL,
  eqax = FALSE,
  axlim = if (is.data.frame(y1)) range(y1$y) else range(c(y1, y2)),
  diflim = NULL,
  grid = TRUE,
  N.grid = 10,
  col.grid = grey(0.9),
  points = TRUE,
  col.points = "black",
  cex.points = 1,
  pch.points = 16,
  lwd = c(3, 1, 1),
  col.lines = "blue",
  repl.conn = FALSE,
  col.conn = "gray",
  lwd.conn = 1,
  xlab = NULL,
  ylab = NULL,
  eqn = FALSE,
  col.eqn = col.lines,
  font.eqn = 2,
  digits = 2,
  Transform = if (mult) "log" else NULL,
  mult = FALSE,
  alpha = NULL,
  ...
)

## S3 method for class 'PBreg'
```

```
plot(
  x,
  pch = 21,
  bg = "#2200aa33",
  xlim = c(0, max(x$model)),
  ylim = c(0, max(x$model)),
  xlab = x$meths[1],
  ylab = x$meths[2],
  subtype = 1,
  colors = list(CI = "#ccaaff50", fit = "blue", ref = "#99999955", bars = "gray", dens =
    "#8866aaa0", ref2 = c("#1222bb99", "#bb221299")),
  ...
)

## S3 method for class 'Meth'
plot(
  x,
  which = NULL,
  col.LoA = "blue",
  col.pt = "black",
  cex.name = 2,
  var.range,
  diff.range,
  var.names = FALSE,
  pch = 16,
  cex = 0.7,
  Transform,
  ...
)

## S3 method for class 'VarComp'
plot(
  x,
  which,
  lwd.line = rep(2, 4),
  col.line = c("red", "green", "blue", "black"),
  lty.line = rep(1, 4),
  grid = TRUE,
  col.grid = gray(0.8),
  rug = TRUE,
  probs = c(5, 50, 95),
  tot.var = FALSE,
  same.ax = TRUE,
  meth.names = TRUE,
  VC.names = "first",
  ...
)
```

Arguments

`y1` Numerical vector of measurements by 1st method. Can also be a `Meth` or a `MethComp` object, see details.

`y2` Numerical vector of measurements by 2nd method. Must of same length as `x`. Ignored if a `Meth` or a `MethComp` objects is given for `y1`.

`meth.names` Should the names of the methods be put on the plots?

`wh.comp` Which methods should be compared. Either numerical or character.

`pl.type` What type of plot should be made, `"BA"` for differences versus averages, `"conv"` for method 1 versus method 2.

`dif.type` How should difference depend on the averages. `"const"` or `"lin"`.

`sd.type` How should the standard deviation depend on the averages. `"const"` or `"lin"`.

`model` Should a variance component model be used to compute the limits of agreement? If `NULL` a simple analysis is made; other possibilities are `"exch"` or `"linked"` for exchangeable or linked replicates.

`eqax` Should the axes be identical? If a Bland-Altman plot is drawn, the axis for the differences will have the same extent as the axis for the averages, but centered on 0 (see `diflim`).

`axlim` The limits of the axes.

`diflim` The limits of the difference axis.

`grid` Logical. Should a vertical grid be set up? If numeric it is set up at the values specified. If `same.ax`, the range of the grid is taken to be the extent of the x-axis for all plots.

`N.grid` How many grid-lines should be drawn.

`col.grid` The color of the grid.

`points` Logical. Should the observed points be drawn?

`col.points` What color should they have?

`cex.points` How large should they be?

`pch.points` What plot character for the points

`lwd` Numerical vector of 3, giving the width of the conversion line (mean difference) and the limits of agreement.

`col.lines` What color should the lines have.

`repl.conn` Should replicate measurements be connected (within items)?

`col.conn` Color of connecting lines.

`lwd.conn` Width of connecting lines.

`xlab` Label on the x-axis.

`ylab` Label on the y-axis.

`eqn` Logical. Should the equations linking the methods be shown on the plot? If a Bland-Altman plot is made, both the equations linking the methods and the equation for the differences versus the averages are shown.

`col.eqn` Color for equations

`font.eqn` Font for equations

`digits` How many digits after the decimal point should be used when showing the equations.

`Transform` Transformation used to the measurements prior to plotting. Function or character, see `choose.trans` for possible values.

`mult` Logical. If TRUE, ratios of measurement instead of differences will be plotted in the Bland-Altman plot on a logarithmic axis, and limits of agreement will be given on this scale? This gives the same analysis as using `Transform="log"`, but a different plot. Using another transformation than the log is accommodated, but no LoA is shown on the axis.

`alpha` 1 minus the confidence level. If `NULL` a multiplier of 2 is used for constructing prediction limits, otherwise a t-quantile with d.f. equal th number of items minus 1.

`...` Parameters passed on the `density` furnction that does the smoothing of the posterior samples.

`x` A `MCmcmc` object.

`pch` Plot character for points.

`bg` Background colour for the plotting character.

`xlim` Limits for the x-axis.

`ylim` Limits for the y-axis.

`subtype` a numeric value or vector, that selects the desired plot subtype. Subtype **1** is an x-y plot of raw data with regression line and confidence boundaries for the fit as a shaded area. This is the default. Subtype **2** is a ranked residuals plot. Subtype **3** is the "Cusum" plot useful for assessing linearity of the fit. Plot subtypes 1 through 3 are standard plots from the 1983 paper by Passing and Bablok - see the reference. Plot subtype **4** is a histogram (with overlaid density line) of the individual slopes. The range of this plot is limited to 5 x IQR for better visibility.

`colors` A list of 6 elements allowing customization of colors of various plot elements. For plot subtype 1: "CI" is the color of the shaded confidence interval area; and "fit" is the color of fit line. For plot subtypes 2 & 3: "ref" is the color of the horizontal reference line. For plot subtype 4: "bars" is the bar background color, "dens" is the color of the density line, and "ref2" is a vector of two colors for lines indicating the median and confidence limits.

`which` For which of the compared methods should the plot be made?

`col.LoA` What color should be used for the limits of agreement.

`col.pt` What color should be used for the points.

`cex.name` Character expansion factor for plotting method names

`var.range` The range of both axes in the scatter plot and the x-axis in the Bland-Altman plot be?

`diff.range` The range of yaxis in the Bland-Altman plot. Defaults to a range as the x-axis, but centered around 0.

`var.names` If logical: should the individual panels be labelled with the variable names?. If character, then the values of the character will be used to label the methods.

`cex` Plot charcter expansion for points.

`lwd.line` Line width for drawing the density.

`col.line` Color for drawing the densities.

`lty.line` Line type for drawing the densities.

`rug` Should a small rug at the bottom show posterior quantiles?

`probs` Numeric vector with numbers in the range from 0 to 100, indicating the posterior percentiles to be shown in the rug.

`tot.var` Should the posterior of the total variance also be shown?

`same.ax` Should the same axes be used for all methods?

`VC.names` Should the names of the variance components be put on the first plot (`"first"`), the last (`"last"`), all (`"all"`) or none (`"none"`). Only the first letter is needed.

Details

A plot of the relationship between the methods is produced; either a Bland-Altman plot of the differences versus averages, or a 45 degree rotation as a conversion between the methods. If `model=NULL` a simple regression of averages on differences is made by calling `DA.reg`, and the specified conversion plotted.

The function generates a series of plots, one for each method compared in the `MCmcmc` object supplied (or those chosen by `which=`). Therefore the user must take care to set `mfrow` or `mfcol` to capture all the plots.

Value

An object of class `MethComp` and either `DA.reg` (if `model=NULL`) or `BA.est` (if `model` is character).

A plot as a side effect

A list with one element for each method. Each element of this is a list of densities, i.e. of objects of class `density`, one for each variance component.

Author(s)

Bendix Carstensen <`bendix.carstensen@regionh.dk`>, `http://BendixCarstensen.com`.

Michal J. Figurski <`mfigrs@gmail.com`>

Bendix Carstensen, <`bendix.carstensen@regionh.dk`>

References

JM Bland and DG Altman: Statistical methods for assessing agreement between two methods of clinical measurement, Lancet, i, 1986, pp. 307-310.

JM Bland and DG Altman. Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999.

B Carstensen: Comparing methods of measurement: Extending the LoA by regression. Stat Med. 2010 Feb 10;29(3):401-10.

Passing, H. and Bablok, W. (1983), A New Biometrical Procedure for Testing the Equality of Measurements from Two Different Analytical Methods. *Journal of Clinical Chemistry and Clinical Biochemistry*, **Vol 21**, 709–720

See Also

BA.est, DA.reg, MCmcmc.

PBreg,Deming.

Examples

```
data( ox )
ox <- Meth( ox )
# The simplest possible Bland-Altman plot
BA.plot( ox )

## With bells and whistles, comparing the naive and model
par( mfrow=c(2,2) )
BA.plot( ox, model=NULL, repl.conn=TRUE, col.lines="blue",
         axlim=c(0,100), diflim=c(-50,50), xaxs="i", yaxs="i",
         las=1, eqn=TRUE, dif.type="lin", pl.type="BA", sd.type="lin",
         grid=1:9*10, digits=3,font.eqn=1)
par(new=TRUE)
BA.plot( ox, model="linked", repl.conn=TRUE, col.lines="red",
         axlim=c(0,100), diflim=c(-50,50), xaxs="i", yaxs="i",
         las=1, eqn=FALSE, dif.type="lin", pl.type="BA", sd.type="lin",
        grid=1:0*10, digits=3)
BA.plot( ox, model=NULL, repl.conn=TRUE, col.lines="blue",
         axlim=c(0,100), diflim=c(-50,50), xaxs="i", yaxs="i",
         las=1, eqn=TRUE, dif.type="lin", pl.type="conv", sd.type="lin",
        grid=1:9*10, digits=3,font.eqn=1)
par(new=TRUE)
BA.plot( ox, model="linked", repl.conn=TRUE, col.lines="red",
         axlim=c(0,100), diflim=c(-50,50), xaxs="i", yaxs="i",
         las=1, eqn=FALSE, dif.type="lin", pl.type="conv", sd.type="lin",
         grid=1:9*10, digits=3)
# The same again, but now logit-transformed
BA.plot( ox, model=NULL, repl.conn=TRUE, col.lines="blue",
         axlim=c(0,100), diflim=c(-50,50), xaxs="i", yaxs="i",
         las=1, eqn=TRUE, dif.type="lin", pl.type="BA", sd.type="lin",
         grid=1:9*10, digits=3,font.eqn=1,Transform="pctlogit")
par(new=TRUE)
BA.plot( ox, model="linked", repl.conn=TRUE, col.lines="red",
         axlim=c(0,100), diflim=c(-50,50), xaxs="i", yaxs="i",
         las=1, eqn=FALSE, dif.type="lin", pl.type="BA", sd.type="lin",
         grid=1:0*10, digits=3,Transform="pctlogit")
BA.plot( ox, model=NULL, repl.conn=TRUE, col.lines="blue",
         axlim=c(0,100), diflim=c(-50,50), xaxs="i", yaxs="i",
         las=1, eqn=TRUE, dif.type="lin", pl.type="conv", sd.type="lin",
         grid=1:9*10, digits=3,font.eqn=1,Transform="pctlogit")
par(new=TRUE)
BA.plot( ox, model="linked", repl.conn=TRUE, col.lines="red",
         axlim=c(0,100), diflim=c(-50,50), xaxs="i", yaxs="i",
         las=1, eqn=FALSE, dif.type="lin", pl.type="conv", sd.type="lin",
         grid=1:9*10, digits=3,Transform="pctlogit")
```

```
## Model data frame generation
a <- data.frame(x=seq(1, 30)+rnorm(mean=0, sd=1, n=30),
                y=seq(1, 30)*rnorm(mean=1, sd=0.4, n=30))

## Call to PBreg
x <- PBreg(a)
print(x)
par(mfrow=c(2,2))
plot(x, s=1:4)

## Or the same using "Meth" object
a <- Meth(a, y=1:2)
x <- PBreg(a)
print(x)
par(mfrow=c(2,2))
plot(x, s=1:4)
```

`bothlines` *Add regression lines to a plot*

Description

Add the regression lines of y on x AND x on y to the plot. Optionally add the line obtained by allowing errors in both variables (Deming regression).

Usage

```
bothlines(x, y, Dem = FALSE, sdr = 1, col = "black", ...)
```

Arguments

`x`	Numeric vector
`y`	Numeric vector
`Dem`	Logical. Should the Deming regression line be added too?
`sdr`	Numeric. The assumed ratio of standard deviations used in the Deming regression.
`col`	Colour of the lines. Can be a vector of up to 3 elements, one for each line.
`...`	Additional arguments passed on to `abline`, which does the actual plotting.

Value

None.

Author(s)

Bendix Carstensen, Steno Diabetes Center, `http://BendixCarstensen.com`

See Also

abline.

Examples

```
data( ox )
oxw <- to.wide(ox)
attach( oxw )
plot( CO, pulse )
abline(0,1)
bothlines( CO, pulse, Dem=TRUE, col=rainbow(3), lwd=2 )
plot( CO, pulse,pch=16 )
abline(0,1, col=gray(0.7), lwd=2)
bothlines( CO, pulse, Dem=TRUE, col=c(rep("transparent",2),"black"), lwd=2 )
```

---

cardiac *Measurement of cardiac output by two different methods.*

Description

For each subject cardiac output is measured repeatedly (three to six times) by impedance cardiography (IC) and radionuclide ventriculography (RV).

Format

A data frame with 120 observations on the following 4 variables.

`meth` a factor with levels `IC` `RV`

`item` a numeric vector giving the item number.

`repl` a numeric vector with replicate number.

`y` the measuremnts of cardiac output.

Details

It is not entirely clear from the source whether the replicates are exchangeable within (method,item) or whether they represent pairs of measurements. From the description it looks as if replicates are linked between methods, but in the paper they are treated as if they were not.

Source

The dataset is adapted from table 4 in: JM Bland and DG Altman: Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999. Originally supplied to Bland \& Altman by Dr LS Bowling, see: Bowling LS, Sageman WS, O'Connor SM, Cole R, Amundson DE. Lack of agreement between measurement of ejection fraction by impedance cardiography versus radionuclide ventriculography. Critical Care Medicine 1993; 21: 1523-27.

Examples

```
data(cardiac)
cardiac <- Meth(cardiac)
summary(cardiac)
# Visually check exchangeability
plot( cardiac )
plot( perm.repl( cardiac ) )
BA.est(cardiac)
# Run MCmcmc using BRugs for an insufficient amount of iterations
## Not run: card.mi.ir <- MCmcmc( cardiac,
                               beta=FALSE, random=c("mi","ir"),
                               n.iter=100, trace=T )
print( card.mi.ir )
## End(Not run)
```

CardOutput *Measurements of Cardiac output.*

Description

Two different ways of measuring cardiac output and oxygen saturation in 15 critically ill persons.

Format

A data frame with 15 observations on the following 8 variables.

`Age` Patient age

`Diag` Diagnosis, a factor with levels `sepsis`, `cardiogenic`, `hypothermia`

`VO2` Oxygen consumption

`Svo2` Mixed venous O2 saturation

`Scvo2` Central venous oxygen saturation

`TCO` Thermodilution-derived cardiac output

`FCO` Fick-derived cardiac output.

`Sex` Sex, a factor with levels `F`, `M`

Source

Avi A. Weinbroum, Philippe Biderman, Dror Soffer, Joseph M. Klausner & Oded Szold:

Reliability of cardiac output calculation by the fick principle and central venous oxygen saturation in emergency conditions.

Journal of Clinical Monitoring and Computing (2008) 22: 361-366

Examples

```
data(CardOutput)
```

`check.trans` *Functions to handle transformations of measurement results.*

Description

Check whether two functions actually are each others inverse.

Usage

```
check.trans(trans, y, trans.tol = 1e-05)
```

Arguments

`trans`	A list of two functions, each other's inverse.
`y`	Vector of numerical values where the functions should be each other's inverse.
`trans.tol`	Numerical constant indication how precise the evaulation should be.

Value

`check.trans` returns nothing.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://bendixcarstensen.com/.

`choose.trans` *Functions to handle transformations of measurement results.*

Description

Choose a function and inverse based on a text string

Usage

```
choose.trans(tr)
```

Arguments

`tr`	A character string, or a list of two functions, they should be each other's inverse. Names of the list are ignored.

Value

`choose.trans` returns a named list with two elements "trans" and "inv", both functions which are each other's inverse. This is intended to be stored as an attribute `"Transform"` with the resulting object and used in plotting and reporting. All results will be on the transformed scale. If the `tr` argument to `choose.trans` is a character constant, the appropriate named list of two functions will be generated. Possibilities are: "exp", "log", "logit", "pctlogit" (transforms percentages by the logit), "sqrt", "sq" (square), "cll" (complementary log-minus-log), "ll" (log-minus-log). If there is no match `NULL` is returned, which will correspond to no transformation.

Author(s)

Bendix Carstensen, Steno Diabetes Center, `http://bendixcarstensen.com/`.

Examples

```
choose.trans( "logit" )
```

`corr.measures` *Classical association measures*

Description

A function that returns the values of some of the classical association measures proposed in the literature

Usage

```
corr.measures(x, y)
```

Arguments

`x`	A vector of numeric values of length N
`y`	A vector of numeric values of length N

Value

A vector of four association measures

DA.reg *Make a regression of differences on averages*

Description

For each pair of methods in data, a regression of the differences on the averages between methods is made and a linear relationship between methods with prediction standard deviations is derived.

Usage

```
DA.reg(
  data,
  Transform = NULL,
  trans.tol = 1e-06,
  print = TRUE,
  random.raters = FALSE,
  DA.slope = TRUE
)
```

Arguments

data	A Meth object. May also be a data frame with columns meth, item and y.
Transform	A character string, or a list of two functions, each other's inverse. The measurements are transformed by this before analysis. Possibilities are: "exp", "log", "logit", "pctlogit" (transforms percentages by the logit), "sqrt", "sq" (square), "cll" (complementary log-minus-log), "ll" (log-minus-log). For further details see choose.trans.
trans.tol	The tolerance used to check whether the supplied transformation and its inverse combine to the identity. Only used if Transform is a list of two functions.
print	Should the results be printed?
random.raters	If methods really are a random selection of raters, neither intercept nor slope different from 0 are sensible, so if this is TRUE, intercept and slope in the regression of difference on averages are fixed to 0. Meaning that we are essentially looking at the raw differences as residuals.
DA.slope	If this is TRUE, a slope of the differences in the verages is estimated, otherwise the relationship is assumed constant.

Details

If the input object contains replicate measurements these are taken as separate items in the order they appear in the dataset.

Value

DA.reg returns a MethComp object, i.e. a list with three components, Conv, VarComp, and data. Conv is a three-dimensional array, with dimensions To, From (both with levels equal to the methods in data) and an unnamed dimension with levels "alpha", "beta", "sd.pred", "beta=1", referring to the linear relationship of To to From, "int(t-f)", "slope(t-f)", "sd(t-f)", referring to the regression of the differences on the averages, and "int(sd)", "slope(sd)", and "s.d.=K", referring to the regression of the absoulte residuals on the averages, and LoA-lo, LoA-hi, the limits of agreement.

Converting from method l to method k using

$$y_{k|l} = \alpha + \beta y_l$$

with prediction standard deviation σ, just requires the entries [k,l,c("alpha","beta","sd.pred")], if we assume the s.d. is constant.

The next entry is the p-values for the hypothesis $\beta = 1$, intercept and slope of the SD of the differences as a linear function of the average and finally p-value of the hypothesis that standard errors are constant over the range. The latter three are derived by regressing the absolute values of the residuals on the averages, and can be used to produce LoA where the s.d. increases (or decreases) by the mean, using the function DA2y.

The VarComp element of the list is NULL, and only present for compatibility with the print method for MethComp objects.

The data element is the input dataframe. The measurements in y are left un-transformed, even if data are transformed (i.e. if the Transform attribute of the object is non-null).

DA2y returns a 2 by 3 matrix with rownames c("y1|2","y2|1") and columnnames c("int","slope","sd"), calculated under the assumption that the differences were formed as D <-y1 -y2.

y2DA returns a 3-component vector with names c("DA-int","DA-slope","DA-sd"), referring to differences D=y1-y2 as a linear function of A=(y1+y2)/2.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <bendix.carstensen@regionh.dk>, http://BendixCarstensen.com/MethComp

References

B. Carstensen: Comparing methods of measurement: Extending the LoA by regression. Stat Med, 29:401-410, 2010.

Examples

```
data( milk )
DA.reg( milk )
data( sbp )
print( DA.reg(sbp), digits=3 )
# Slope, intercept : y1 = 0.7 + 1.2*y2 (0.4)
A <- c(0.7,1.2,0.4)
( y2DA( A ) )
( DA2y( y2DA( A ) ) )
```

DA2y *Convert DA to (classical) regression*

Description

The functions DA2y and y2DA are convenience functions that convert the estimates of intercept, slope and sd from the regression of $D = y_1 - y_2$ on $A = (y_1 + y_2)/2$, back and forth to the resulting intercept, slope and sd in the relationship between y_1 and y_2, cf. Carstensen (2010), equation 6.

Usage

```
DA2y(a = 0, b = 0, s = NA)
```

Arguments

a	Intercept in the linear relation of the differences y1-y2 to the averages (y1+y2)/2. If a vector of length>1, this is used instead of a, b and s, and b and s are ignored.
b	Slope in the linear relstion of the differences to the averages.
s	SD from the regression of the differences in the averages. Can be NA.

Details

DA2y takes the intercept(a), slope(b) and sd(s) from the relationship (y1-y2)=a+b((y1+y2)/2)+e with sd(e)=s, and returns a two by 3 matrix with columns "int","slope","sd" and rows "y1|2","y2|1".

Value

DA2y returns a 2 by 3 matrix with rownames c("y1|2","y2|1") and columnnames c("int","slope","sd"), calculated under the assumption that the differences were formed as D <-y1 -y2.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <bendix.carstensen@regionh.dk>, http://BendixCarstensen.com/MethComp

References

B. Carstensen: Comparing methods of measurement: Extending the LoA by regression. Stat Med, 29:401-410, 2010.

Examples

```
data( milk )
DA.reg( milk )
data( sbp )
print( DA.reg(sbp), digits=3 )
# Slope, intercept : y1 = 0.7 + 1.2*y2 (0.4)
A <- c(0.7,1.2,0.4)
( y2DA( A ) )
( DA2y( y2DA( A ) ) )
```

`Deming` *Regression with errors in both variables (Deming regression)*

Description

The formal model underlying the procedure is based on a so called functional relationship:

$$x_i = \xi_i + e_{1i}, \qquad y_i = \alpha + \beta\xi_i + e_{2i}$$

with $\mathrm{var}(e_{1i}) = \sigma$, $\mathrm{var}(e_{2i}) = \lambda\sigma$, where λ is the known variance ratio.

Usage

```
Deming(
  x,
  y,
  vr = sdr^2,
  sdr = sqrt(vr),
  boot = FALSE,
  keep.boot = FALSE,
  alpha = 0.05
)
```

Arguments

`x` a numeric variable

`y` a numeric variable

`vr` The assumed known ratio of the (residual) variance of the `y`s relative to that of the `x`s. Defaults to 1.

`sdr` do. for standard deviations. Defaults to 1. `vr` takes precedence if both are given.

`boot` Should bootstrap estimates of standard errors of parameters be done? If `boot==TRUE`, 1000 bootstrap samples are done, if `boot` is numeric, `boot` samples are made.

`keep.boot` Should the 4-column matrix of bootstrap samples be returned? If `TRUE`, the summary is printed, but the matrix is returned invisibly. Ignored if `boot=FALSE`

`alpha` What significance level should be used when displaying confidence intervals?

Details

The estimates of the residual variance is based on a weighting of the sum of squared deviations in both directions, divided by $n - 2$. The ML estimate would use $2n$ instead, but in the model we actually estimate $n + 2$ parameters — α, β and the n ξs. This is not in Peter Sprent's book (see references).

Value

If `boot==FALSE` a named vector with components `Intercept`, `Slope`, `sigma.x`, `sigma.y`, where `x` and `y` are substituted by the variable names.

If `boot==TRUE` a matrix with rows `Intercept`, `Slope`, `sigma.x`, `sigma.y`, and colums giving the estimates, the bootstrap standard error and the bootstrap estimate and c.i. as the 0.5, $\alpha/2$ and $1-\alpha/2$ quantiles of the sample.

If `keep.boot==TRUE` this summary is printed, but a matrix with columns `Intercept`, `Slope`, `sigma.x`, `sigma.y` and `boot` rows is returned.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <`bendix.carstensen@regionh.dk`>, `http://BendixCarstensen.com`

References

Peter Sprent: Models in Regression, Methuen & Co., London 1969, ch.3.4.

WE Deming: Statistical adjustment of data, New York: Wiley, 1943.

Examples

```
# 'True' values
M <- runif(100,0,5)
# Measurements:
x <- M + rnorm(100)
y <- 2 + 3 * M + rnorm(100,sd=2)
# Deming regression with equal variances, variance ratio 2.
Deming(x,y)
Deming(x,y,vr=2)
Deming(x,y,boot=TRUE)
bb <- Deming(x,y,boot=TRUE,keep.boot=TRUE)
str(bb)
# Plot data with the two classical regression lines
plot(x,y)
abline(lm(y~x))
ir <- coef(lm(x~y))
abline(-ir[1]/ir[2],1/ir[2])
abline(Deming(x,y,sdr=2)[1:2],col="red")
abline(Deming(x,y,sdr=10)[1:2],col="blue")
# Comparing classical regression and "Deming extreme"
summary(lm(y~x))
```

```
Deming(x,y,vr=1000000)
```

`ends` *Function to identify the extremes of a vector*

Description

Function to identify the extremes of a vector

Usage

```
ends(w, rm = 1/3)
```

Arguments

`w`	A numeric vector of values
`rm`	A value between 0 and 1 giving the percentage of extreme observations to remove

Value

A logical vector of indices that a

`Enzyme` *Enzyme activity data*

Description

Three measurement of enzyme activity on 24 patients. The measurements is of the enzymes sucrase and alkaline phosphatase. The interest is to compare the 'homogenate' and 'pellet' methods.

Format

A data frame with 72 observations on the following 3 variables.

`meth` a factor with levels `SucHom` `SucPel` `Alkphos`, representing three different measurements, i.e. homogenate and pellet values of sucrase, as well as homogenate values of alkaline.

`item` a numeric vector, the person ID for the 24 patients

`y` a numeric vector, the measurements on the enzyme activity.

Source

R. L. Carter; Restricted Maximum Likelihood Estimation of Bias and Reliability in the Comparison of Several Measuring Methods; Biometrics, Dec., 1981, Vol. 37, No. 4, pp. 733-741.

Examples

```
data(Enzyme)
Enzyme <- Meth( Enzyme )
summary( Enzyme )
# plot( Enzyme )
```

`fat` *Measurements of subcutaneous and visceral fat*

Description

43 persons had Subcutaneous and Visceral fat thickness measured at Steno Diabetes Center in 2006 by two observers; all measurements were done three times. The interest is to compare the measurements by the two observers. Persons are items, observers are methods, the three replicates are exchangeable within (person,observer)=(item,method)

Format

A data frame with 258 observations on the following 6 variables.

`Id` Person id.

`Obs` Observers, a factor with levels `KL` and `SL`.

`Rep` Replicate — exchangeable within person and observer.

`Sub` Subcutaneous fat measured in cm.

`Vic` Visceral fat measured in cm.

Examples

```
data(fat)
str(fat)
vic <- Meth( fat, meth=2, item=1, repl="Rep", y="Vic" )
str(vic)
BA.est( vic, linked=FALSE )
```

glucose — *Glucose measurements by different methods*

Description

74 persons in 5 centres in Finland had blood glucose measured by 11 different methods, based on 4 different types of blood. Each person had blood sampled at 0, 30, 60 and 120 min after a 75 g glucose load.

Format

A data frame with 1302 observations on the following 6 variables.

- `meth` Method of measurement. A factor with 11 levels: `n.plas1` `n.plas2` `h.cap` `h.blood` `h.plas` `h.serum` `m.plas` `m.serum` `o.cap` `s.serum` `k.plas`.
- `type` Type of blood sample. A factor with 4 levels: `blood` `plasma` `serum` `capil`
- `item` Person id.
- `time` Time of blood sampling. Minutes since glucose load.
- `cent` Center of sampling. Except for the two first methods, `n.plas1` and `n.plas2`, samples were analyzed at the centres too
- `y` Glucose measurement in mmol/l.

Source

The study was conducted at the National Public Health Institute in Helsinki by Jaana Lindstrom.

References

B Carstensen, J Lindstrom, J Sundvall, K Borch-Johnsen1, J Tuomilehto & the DPS Study Group: Measurement of Blood Glucose: Comparison between different Types of Specimens. Annals of Clinical Biochemistry, to appear.

Examples

```
data( glucose )
str( glucose )
# Use only plasma and serum as methods and make a Bland-Altman plot
gluc <- subset( glucose, type %in% c("plasma","serum") )
gluc$meth <- gluc$type
gluc$repl <- gluc$time
BA.plot( gluc )
```

hba.MC *A MCmcmc object from the hba1c data*

Description

This object is included for illustrative purposes. It is a result of a 5-hour run using MCmcmc, with `n.iter=100000`.

Format

The format is a `MCmcmc` object.

Details

The data are the venous measurements from the `hba1c` dataset, using the day of analysis as replicate. Measurements are taken to be linked within replicate (=day of analysis).

Examples

```
data(hba.MC)
attr(hba.MC,"mcmc.par")
# print.MCmcmc(hba.MC)
# One of the chains is really fishy (it's the first one)
# trace.MCmcmc(hba.MC)
# trace.MCmcmc(hba.MC,"beta")
# Try to have a look, excluding the first chain
# hba.MCsub <- subset.MCmcmc(hba.MC,chains=-1)
# trace.MCmcmc(hba.MCsub)
# trace.MCmcmc(hba.MCsub,"beta")
# A MCmcmc object also has class mcmc.list, so we can use the
# coda functions for covergence diagnostics:
# acfplot( subset.MCmcmc(hba.MC, subset="sigma"))
```

hba1c *Measurements of HbA1c from Steno Diabetes Center*

Description

Three analysers (machines) for determination of HbA1c (glycosylated haemoglobin) were tested on samples from 38 individuals. Each had drawn a venous and capillary blood sample. These were analysed on five different days.

Format

A data frame with 835 observations on the following 6 variables.

`dev` Type of machine used. A factor with levels `BR.V2`, `BR.VC` and `Tosoh`.

`type` Type of blood analysed (capillary or venous). A factor with levels `Cap` `Ven`

`item` Person-id. A numeric vector

`d.samp` Day of sampling.

`d.ana` Day of laboratory analysis.

`y` The measured value of HbA1c.

Details

In the terminology of method comparison studies, methods is the cross-classification of `dev` and `type`, and replicate is `d.ana`. It may be of interest to look at the effect of time between `d.ana` and `d.samp`, i.e. the time between sampling and analysis.

Source

Bendix Carstensen, Steno Diabetes Center.

References

These data were analysed as example in: Carstensen: Comparing and predicting between several methods of measurement, Biostatistics 5, pp. 399–413, 2004.

Examples

```
data(hba1c)
str(hba1c)
hb1  <- with( hba1c,
              Meth( meth = interaction(dev,type),
                    item = item,
                    repl = d.ana-d.samp,
                       y = y, print=TRUE ) )
```

`MCmcmc` *Fit a model for method comparison studies using WinBUGS*

Description

A model linking each of a number of methods of measurement linearly to the "true" value is set up in BUGS and run via the function `bugs` from the `R2WinBUGS` package.

Usage

```
MCmcmc(
  data,
  bias = "linear",
  IxR = has.repl(data),
  linked = IxR,
  MxI = TRUE,
  matrix = MxI,
  varMxI = nlevels(factor(data$meth)) > 2,
  n.chains = 4,
  n.iter = 2000,
  n.burnin = n.iter/2,
  n.thin = ceiling((n.iter - n.burnin)/1000),
  bugs.directory = getOption("bugs.directory"),
  debug = FALSE,
  bugs.code.file = "model.txt",
  clearWD = TRUE,
  code.only = FALSE,
  ini.mult = 2,
  list.ini = TRUE,
  org = FALSE,
  program = "JAGS",
  Transform = NULL,
  trans.tol = 1e-06,
  ...
)
```

Arguments

`data` Data frame with variables `meth`, `item`, `repl` and `y`, possibly a `Meth` object. `y` represents a measurement on an `item` (typically patient or sample) by method `meth`, in replicate `repl`.

`bias` Character. Indicating how the bias between methods should be modelled. Possible values are `"none"`, `"constant"`, `"linear"` and `"proportional"`. Only the first three letters are significant. Case insensitive.

`IxR` Logical. Are the replicates linked across methods, i.e. should a random `item` by `repl` be included in the model.

`linked` Logical, alias for `IxR`.

`MxI` Logical, should a `meth` by `item` effect be included in the model?

`matrix` Logical, alias for `MxI`.

`varMxI` Logical, should the method by item effect have method-specific variances. Ignored if only two methods are compared.

`n.chains` How many chains should be run by WinBUGS — passed on to bugs.

`n.iter` How many total iterations — passed on to bugs.

`n.burnin` How many of these should be burn-in — passed on to bugs.

`n.thin` How many should be sampled — passed on to `bugs`.

`bugs.directory` Where is WinBUGS (>=1.4) installed — passed on to `bugs`. The default is to use a parameter from options(). If you use this routinely, this is most conveniently set in your `.Rprofile` file.

`debug` Should WinBUGS remain open after running — passed on to `bugs`.

`bugs.code.file` Where should the bugs code go?

`clearWD` Should the working directory be cleared for junk files after the running of WinBUGS — passed on to `bugs`.

`code.only` Should `MCmcmc` just create a bugs code file and a set of inits? See the `list.ini` argument.

`ini.mult` Numeric. What factor should be used to randomly perturb the initial values for the variance components, see below in details.

`list.ini` List of lists of starting values for the chains, or logical indicating whether starting values should be generated. If `TRUE` (the default), the function VC.est will be used to generate initial values for the chains. `list.ini` is a list of length `n.chains`. Each element of which is a list with the following vectors as elements:

`mu` - length I

`alpha` - length M

`beta` - length M

`sigma.mi` - length M - if M is 2 then length 1

`sigma.ir` - length 1

`sigma.mi` - length M

`sigma.res` - length M

If `code.only==TRUE`, `list.ini` indicates whether a list of initial values is returned (invisibly) or not. If `code.only==FALSE`, `list.ini==FALSE` is ignored.

`org` Logical. Should the posterior of the original model parameters be returned too? If `TRUE`, the `MCmcmc` object will have an attribute, `original`, with the posterior of the parameters in the model actually simulated.

`program` Which program should be used for the MCMC simulation. Possible values are `"BRugs"`, `"ob"`, `"winbugs"`, `"wb"` (WinBUGS), `"jags"` (JAGS). Case insensitive. Defaults to `"JAGS"` since: 1) JAGS is available on all platforms and 2) JAGS seems to be faster than BRugs on (some) windows machines.

`Transform` Transformation of data (`y`) before analysis. See `choose.trans`.

`trans.tol` The tolerance used to check whether the supplied transformation and its inverse combine to the identity.

`...` Additional arguments passed on to `bugs`.

Details

The model set up for an observation y_{mir} is:

$$y_{mir} = \alpha_m + \beta_m(\mu_i + b_{ir} + c_{mi}) +$$

$$e_{mir}$$

where b_{ir} is a random `item` by `repl` interaction (included if `"ir"` is in `random`) and c_{mi} is a random `meth` by `item` interaction (included if `"mi"` is in `random`). The μ_i's are parameters in the model but are not monitored — only the αs, βs and the variances of b_{ir}, c_{mi} and e_{mir} are monitored and returned. The estimated parameters are only determined up to a linear transformation of the μs, but the linear functions linking methods are invariant. The identifiable conversion parameters are:

$$\alpha_{m\cdot k} = \alpha_m - \alpha_k\beta_m/\beta_k,$$

$$\beta_{m\cdot k} = \beta_m/\beta_k$$

The posteriors of these are derived and included in the `posterior`, which also will contain the posterior of the variance components (the SDs, that is). Furthermore, the posterior of the point where the conversion lines intersects the identity as well as the prediction SDs between any pairs of methods are included.

The function `summary.MCmcmc` method gives estimates of the conversion parameters that are consistent. Clearly,

$$\text{median}(\beta_{1\cdot 2}) =$$

$$1/\text{median}(\beta_{2\cdot 1})$$

because the inverse is a monotone transformation, but there is no guarantee that

$$\text{median}(\alpha_{1\cdot 2}) = \text{median}(-\alpha_{2\cdot 1}/$$

$$\beta_{2\cdot 1})$$

and hence no guarantee that the parameters derived as posterior medians produce conversion lines that are the same in both directions. Therefore, `summary.MCmcmc` computes the estimate for $\alpha_{2\cdot 1}$ as

$$(\text{median}(\alpha_{1\cdot 2}) - \text{median}(\alpha_{2\cdot 1})$$

$$/\text{median}(\beta_{2\cdot 1}))/2$$

and the estimate of $\alpha_{1\cdot 2}$ correspondingly. The resulting parameter estimates defines the same lines.

Value

If `code.only==FALSE`, an object of class `MCmcmc` which is a `mcmc.list` object of the relevant parameters, i.e. the posteriors of the conversion parameters and the variance components transformed to the scales of each of the methods.

Furthermore, the object have the following attributes:

`random` Character vector indicating which random effects ("ir","mi") were included in the model.

`methods` Character vector with the method names.

`data` The data frame used in the analysis. This is used in `plot.MCmcmc` when plotting points.

`mcmc.par` A list giving the number of chains etc. used to generate the object.

`original` If `org=TRUE`, an `mcmc.list` object with the posterior of the original model parameters, i.e. the variance components and the unidentifiable mean parameters.

`Transform` The transformation used to the measurements before the analysis.

If `code.only==TRUE`, a list containing the initial values is generated.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://BendixCarstensen.com, Lyle Gurrin, University of Melbourne, http://www.epi.unimelb.edu.au/about/staff/gurrin-lyle.

References

B Carstensen: Comparing and predicting between several methods of measurement, Biostatistics, 5, pp 399-413, 2004

See Also

BA.plot, plot.MCmcmc, print.MCmcmc, check.MCmcmc

Examples

```
data( ox )
str( ox )
ox <- Meth( ox )
# Writes the BUGS program to your console
MCmcmc( ox, MI=TRUE, IR=TRUE, code.only=TRUE, bugs.code.file="" )

### What is written here is not necessarily correct on your machine.
# ox.MC <- MCmcmc( ox, MI=TRUE, IR=TRUE, n.iter=100, program="JAGS" )
# ox.MC <- MCmcmc( ox, MI=TRUE, IR=TRUE, n.iter=100 )
#  data( ox.MC )
#   str( ox.MC )
# print( ox.MC )
```

Meth *Create a Meth object representing a method comparison study*

Description

Creates a dataframe with columns `meth`, `item`, (`repl`) and `y`.

Usage

```
Meth(
  data = NULL,
  meth = "meth",
  item = "item",
  repl = NULL,
  y = "y",
  print = !is.null(data),
  keep.vars = !is.null(data)
)
```

Arguments

`data` A data frame

`meth` Vector of methods, numeric, character or factor. Can also be a number or character referring to a column in `data`.

`item` Vector of items, numeric, character or factor. Can also be a number or character referring to a column in `data`.

`repl` Vector of replicates, numeric, character or factor. Can also be a number or character referring to a column in `data`.

`y` Vector of measurements. Can also be a character or numerical vector pointing to columns in `data` which contains the measurements by different methods or a dataframe with columns representing measurements by different methods. In this case the argument `meth` is ignored, and the names of the columns are taken as method names.

`print` Logical: Should a summary result be printed?

`keep.vars` Logical. Should the remaining variables from the dataframe `data` be transferred to the `Meth` object.

Details

In order to perform analyses of method comparisons it is convenient to have a dataframe with classifying factors, `meth`, `item`, and possibly `repl` and the response variable `y`. This function creates such a dataframe, and gives it a class, `Meth`, for which there is a number of methods: `summary` - tabulation, `plot` - plotting and a couple of analysis methods.

If there are replicates in the values of `item` it is assumed that those observations represent replicate measurements and different replicate numbers are given to those.

Value

The `Meth` function returns a `Meth` object which is a dataframe with columns `meth`, `item`, (`repl`) and `y`. `summary.Meth` returns a table classified by method and no. of replicate measurements, extended with columns of the total number of items, total number of observations and the range of the measurements.

Examples

```
data(fat)
# Different ways of selecting columns and generating replicate numbers
Sub1 <- Meth(fat,meth=2,item=1,repl=3,y=4,print=TRUE)
Sub2 <- Meth(fat,2,1,3,4,print=TRUE)
Sub3 <- Meth(fat,meth="Obs",item="Id",repl="Rep",y="Sub",print=TRUE)
summary( Sub3 )
plot( Sub3 )

# Use observation in different columns as methods
data( CardOutput )
head( CardOutput )
sv <- Meth( CardOutput, y=c("Svo2","Scvo2") )
# Note that replicates are generated if a non-unique item-id is used
```

```
sv <- Meth( CardOutput, y=c("Svo2","Scvo2"), item="Age" )
str( sv )
# A summary is not created if the the first argument (data=) is not used:
sv <- Meth( y=CardOutput[,c("Svo2","Scvo2")], item=CardOutput$VO2 )
summary(sv)

# Sample items
ssv <- sample.Meth( sv, how="item", N=8 )

# More than two methods
data( sbp )
plot( Meth( sbp ) )
# Creating non-unique replicate numbers per (meth,item) creates a warning:
data( hba1c )
hb1  <- with( hba1c,
              Meth( meth=dev, item=item, repl=d.ana-d.samp, y=y, print=TRUE ) )
hb2  <- with( subset(hba1c,type=="Cap"),
              Meth( meth=dev, item=item, repl=d.ana-d.samp, y=y, print=TRUE ) )
```

Meth.sim *Simulate a dataframe containing replicate measurements on the same items using different methods.*

Description

Simulates a dataframe representing data from a method comparison study. It is returned as a Meth object.

Usage

```
Meth.sim(
  Ni = 100,
  Nm = 2,
  Nr = 3,
  nr = Nr,
  alpha = rep(0, Nm),
  beta = rep(1, Nm),
  mu.range = c(0, 100),
  sigma.mi = rep(5, Nm),
  sigma.ir = 2.5,
  sigma.mir = rep(5, Nm),
  m.thin = 1,
  i.thin = 1
)
```

Arguments

`Ni` The number of items (patient, animal, sample, unit etc.)

`Nm` The number of methods of measurement.

`Nr` The (maximal) number of replicate measurements for each (item,method) pair.

`nr` The minimal number of replicate measurements for each (item,method) pair. If `nr<Nr`, the number of replicates for each (meth,item) pair is uniformly distributed on the points `nr:Nr`, otherwise `nr` is ignored. Different number of replicates is only meaningful if replicates are not linked, hence `nr` is also ignored when `sigma.ir>0`.

`alpha` A vector of method-specific intercepts for the linear equation relating the "true" underlying item mean measurement to the mean measurement on each method.

`beta` A vector of method-specific slopes for the linear equation relating the "true" underlying item mean measurement to the mean measurement on each method.

`mu.range` The range across items of the "true" mean measurement. Item means are uniformly spaced across the range. If a vector length `Ni` is given, the values of that vector will be used as "true" means.

`sigma.mi` A vector of method-specific standard deviations for a method by item random effect. Some or all components can be zero.

`sigma.ir` Method-specific standard deviations for the item by replicate random effect.

`sigma.mir` A vector of method-specific residual standard deviations for a method by item by replicate random effect (residual variation). All components must be greater than zero.

`m.thin` Fraction of the observations from each method to keep.

`i.thin` Fraction of the observations from each item to keep. If both `m.thin` and `i.thin` are given the thinning is by their componentwise product.

Details

Data are simulated according to the following model for an observation y_{mir}:

$$y_{mir} = \alpha_m + \beta_m(\mu_i + b_{ir} + c_{mi}) + e_{mir}$$

where b_{ir} is a random `item` by `repl` interaction (with standard deviation for method m the corresponding component of the vector $\sigma_i r$), c_{mi} is a random `meth` by `item` interaction (with standard deviation for method m the corresponding component of the vector $\sigma_m i$) and e_{mir} is a residual error term (with standard deviation for method m the corresponding component of the vector $\sigma_m ir$). The μ_i's are uniformly spaced in a range specified by `mu.range`.

Value

A `Meth` object, i.e. dataframe with columns `meth`, `item`, `repl` and `y`, representing results from a method comparison study.

Author(s)

Lyle Gurrin, University of Melbourne, `http://www.epi.unimelb.edu.au/about/staff/gurrin-lyle`

Bendix Carstensen, Steno Diabetes Center, `http://BendixCarstensen.com`

See Also

summary.Meth, plot.Meth, MCmcmc

Examples

```
Meth.sim( Ni=4, Nr=3 )
xx <- Meth.sim( Nm=3, Nr=5, nr=2, alpha=1:3, beta=c(0.7,0.9,1.2), m.thin=0.7 )
summary( xx )
plot( xx )
```

MethComp — *Summarize conversion equations and prediction intervals between methods.*

Description

Takes the results from BA.est, DA.reg, AltReg or MCmcmc and returns a MethComp object, suitable for displaying the relationship between methods in print pr graphic form.

Usage

```
MethComp(obj)
```

Arguments

obj	A MethComp or MCmcmc object.

Details

Using MethComp on the results from BA.est or AltReg is not necessary, as these two functions already return objetcs of class MethComp.

Value

MethComp returns a MethComp object, which is a list with three elements, Conv, a three-way array giving the linear conversion equations between methods, VarComp, a two-way array classified by methods and variance components and data, a copy of the original Meth object supplied — see the description under BA.est.

A MethComp object has an attribute Transform, which is either NULL, or a named list with elements trans and inv, both of which are functions. The first is the transformation applied to measurements before analysis; the results are all given on the transformed scale. The second is the inverse transformation; this is only used when plotting the resulting relationship between methods.

The methods print, plot, lines and points return nothing.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <bendix.carstensen@regionh.dk >.

See Also

BA.est AltReg MCmcmc

Examples

```
data( ox )
BA.ox <- BA.est( ox, linked=TRUE )
print( BA.ox )
## Not run:
AR.ox <- AltReg( ox, linked=TRUE  )
print( AR.ox )
plot( AR.ox )
## End(Not run)
```

middle *Function to identify the middle of a vector*

Description

Function to identify the middle of a vector

Usage

```
middle(w, rm = 1/3)
```

Arguments

w	A numeric vector of values
rm	A value between 0 and 1 giving the percentage of extreme observations to remove

Value

A logical vector of indices that a

milk *Measurement of fat content of human milk by two different methods.*

Description

Fat content of human milk determined by measurement of glycerol released by enzymic hydrolysis of triglycerides (Trig) and measurement by the Standard Gerber method (Gerber). Units are (g/100 ml).

Format

A data frame with 90 observations on the following 3 variables.

`meth` a factor with levels `Gerber` `Trig`

`item` sample id

`y` a numeric vector

Source

The dataset is adapted from table 3 in: JM Bland and DG Altman: Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999. See: Lucas A, Hudson GJ, Simpson P, Cole TJ, Baker BA. An automated enzymic micromethod for the measurement of fat in human milk. Journal of Dairy Research 1987; 54: 487-92.

Examples

```
data(milk)
str(milk)
milk <- Meth(milk)
plot(milk)
abline(0,1)
```

ox *Measurement of oxygen saturation in blood*

Description

61 children had their blood oxygen content measured at the Children's Hospital in Melbourne, either with a chemical method analysing gases in the blood (`CO`) or by a pulse oximeter measuring transcutaneously (`pulse`). Replicates are linked between methods; i.e. replicate 1 for each of the two methods are done at the same time. However, replicate measurements were taken in quick succession so the pairs of measurements are exchangeable within person.

Format

A data frame with 354 observations on the following 4 variables.

`meth` Measurement methods, factor with levels `CO`, `pulse`

`item` Id for the child

`repl` Replicate of measurements. There were 3 measurements for most children, 4 had only 2 replicates with each method, one only 1

`y` Oxygen saturation in percent.

Examples

```
data(ox)
str(ox)
ox <- Meth(ox)
with( ox, table(table(item)) )
summary( ox )
# The effect of basing LoA on means over replicates:
par( mfrow=c(1,2), mar=c(4,4,1,4) )
BA.plot(      ox , diflim=c(-20,20), axlim=c(20,100), repl.conn=TRUE )
# BA.plot( mean(ox), diflim=c(-20,20), axlim=c(20,100) )
```

`ox.MC` *A MCmcmc object from the oximetry data.*

Description

This object is included for illustrative purposes. It is a result of using `MCmcmc`, with `n.iter=20000`.

Format

The format is a `MCmcmc` object.

Details

The data are the `ox` dataset, where measurements are linked within replicate (=day of analysis).

Examples

```
data(ox.MC)
attr(ox.MC,"mcmc.par")
## Not run:
print.MCmcmc(ox.MC)
trace.MCmcmc(ox.MC)
trace.MCmcmc(ox.MC,"beta")
 post.MCmcmc(ox.MC)
 post.MCmcmc(ox.MC,"beta")
## End(Not run)
# A MCmcmc object also has class mcmc.list, so we can use the
# coda functions for covergence diagnostics:
## Not run:  acfplot( subset.MCmcmc(ox.MC, subset="sigma"))
```

pairs.MCmcmc *Create a pairs plot for an MCmcmc object*

Description

Create a pairs plot for an MCmcmc object

Usage

```
## S3 method for class 'MCmcmc'
pairs(
  x,
  what = "sd",
  subset = NULL,
  col = NULL,
  pch = 16,
  cex = 0.2,
  scales = "free",
  ...
)
```

Arguments

x	An MCmcmc object.
what	Character indicating what parameters to plot. Possible values are "sd" or "var" which gives plots for the variance components (on the sd. scale), "beta" or "slope", which gives plots for slope parameters and "alpha" or "int", which gives plots for the intercept parameters.
subset	Character or numerical indicating the columns of the posterior that should be plotted by pairs.
col	Color of the lines points used for plotting of the posterior densities.
pch	Plot symbol for the points.
cex	Plot character size for points in pairs.
scales	Character vector of length two, with possible values "same" or "free", indicating whether x- and y-axes of the plots should be constrained to be the same across panels. For pairs only the first element is used to decide whether all panles should have the same axes.
...	Further aruments passed on to the Lattice function called: trace calls xyplot from the coda package, post calls densityplot from the coda package, calls pairs from the graphics package.

Value

A `Lattice` plot.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <`bendix.carstensen@regionh.dk` >, `http://BendixCarstensen.com`.

See Also

`MCmcmc`, `plot.MCmcmc`, `ox.MC`, `sbp.MC`

`PBreg` *Passing-Bablok regression*

Description

Implementation of the Passing-Bablok's procedure for assessing of the equality of measurements by two different analytical methods.

Usage

```
PBreg(x, y = NULL, conf.level = 0.05, wh.meth = 1:2)
```

Arguments

`x`	a `Meth` object, alternatively a numeric vector of measurements by method A, or a data frame of exactly two columns, first column with measurements by method A, second column with measurements by method B.
`y`	a numeric vector of measurements by method B - must be of the same length as `x`. If not provided, `x` must be the `Meth` object or a data frame of exactly 2 columns.
`conf.level`	confidence level for calculation of confidence boundaries - 0.05 is the default.
`wh.meth`	Which of the methods from the `Meth` object are used in the regression.

Details

This is an implementation of the original Passing-Bablok procedure of fitting unbiased linear regression line to data in the method comparison studies. It calcualtes the unbiased slope and intercept, along with their confidence intervals. However, the tests for linearity is not yet fully implemented.

It doesn't matter which results are assigned to "Method A" and "Method B", however the "Method A" results will be plotted on the x-axis by the `plot` method.

Value

`PBreg` returns an object of class `"PBreg"`, for which the `print`, `predict` and `plot` methods are defined.

An object of class `"PBreg"` is a list composed of the following elements:

`coefficients` a matrix of 3 columns and 2 rows, containing the estimates of the intercept and slope, along with their confidence boundaries.

`residuals` defined as in the `"lm"` class, as the response minus the fitted value.

`fitted.values` the fitted values.

`model` the model data frame used.

`n` a vector of two values: the number of observations read, and the number of observations used.

`S` A vector of all slope estimates.

`I` A vector of all intercept estimates.

`adj` A vector of fit parameters, where *Ss* is the number of estimated slopes (`length(S)`), *K* is the offset for slopes <(-1), *M1* and *M2* are the locations of confidence boundaries in `S`, and *l* and *L* are the numbers of points above and below the fitted line, used in cusum calculation.

`cusum` A vector of cumulative sums of residuals sorted by the D-rank.

`Di` A vector of D-ranks.

Note

Please note that this method can become very computationally intensive for larger numbers of observations. One can expect a reasonable computation times for datasets with fewer than 100 observations.

Author(s)

Michal J. Figurski <`mfigrs@gmail.com`>

References

Passing, H. and Bablok, W. (1983), A New Biometrical Procedure for Testing the Equality of Measurements from Two Different Analytical Methods. *Journal of Clinical Chemistry and Clinical Biochemistry*, Vol 21, 709–720

See Also

`plot.PBreg`,`predict.PBreg`,`Deming`.

Examples

```
## Model data frame generation
a <- data.frame(x=seq(1, 30)+rnorm(mean=0, sd=1, n=30),
                y=seq(1, 30)*rnorm(mean=1, sd=0.4, n=30))

## Call to PBreg
x <- PBreg(a)
print(x)

par(mfrow=c(2,2))
plot(x, s=1:4)

## A real data example
data(milk)
milk <- Meth(milk)
summary(milk)
PBmilk <- PBreg(milk)
par(mfrow=c(2,2))
plot(PBmilk, s=1:4)
```

PEFR — *Peak Expiratory Flow Rate (PEFR) measurements with Wright peak flow and mini Wright peak flow meter.*

Description

Measurement of PEFR with Wright peak flow and mini Wright peak flow meter on 17 individuals.

Format

A data frame with 68 observations on the following 3 variables.

- `meth` a factor with levels `Wright` and `Mini`, representing measurements by a Wright peak flow meter and a mini Wright meter respectively, in random order.
- `item` Numeric vector, the person ID.
- `y` Numeric vector, the measurements, i.e. PEFR for the two measurements with a Wright peak flow meter and a mini Wright meter respectively. The measurement unit is l/min.
- `repl` Numeric vector, replicate number. Replicates are exchangeable within item.

Source

J. M. Bland and D. G. Altman (1986) Statistical Methods for Assessing Agreement Between Two Methods of Clinical Measurement, Lancet. 1986 Feb 8;1(8476):307-10.

Examples

```
data(PEFR)
PEFR <- Meth(PEFR)
summary(PEFR)
plot(PEFR)
plot(perm.repl(PEFR))
```

---

perm.repl — *Manipulate the replicate numbering within (item,method)*

Description

Replicate numbers are generated within (item,method) in a dataframe representing a method comparison study. The function assumes that observations are in the correct order within each (item,method), i.e. if replicate observations are non-exchangeable within method, linked observations are assumed to be in the same order within each (item,method).

Usage

```
perm.repl(data)
```

Arguments

`data`	A `Meth` object or a data frame with columns `meth`, `item` and `y`.

Details

`make.repl` just adds replicate numbers in the order of the data.frame rows. `perm.repl` is designed to explore the effect of permuting the replicates within (item,method). If replicates are truly exchangeable within methods, the inference should be independent of this permutation.

Value

`make.repl` returns a dataframe with a column, `repl` added or replaced, whereas `has.repl` returns a logical indicating wheter a combination of (`meth`,`item`) wioth more that one valid y- value.

`perm.repl` returns a dataframe of class `Meth` where the rows (i.e. replicates) are randomly permuted within (`meth`,`item`), and subsequently ordered by (`meth`,`item`,`repl`).

Author(s)

Bendix Carstensen, Steno Diabetes Center, `http://bendixcarstensen.com/`

See Also

`perm.repl`

Examples

```
data(ox)
xx <- subset( ox, item<4 )[,-3]
cbind( xx, make.repl(xx) )
cbind( make.repl(xx), perm.repl(xx) )
data( ox )
xx <- subset( ox, item<4 )
cbind( xx, perm.repl(xx) )
# Replicates are linked in the oximetry dataset, so randomly permuting
# them clearly inflates the limits of agreement:
par( mfrow=c(1,2), mar=c(4,4,1,4) )
BA.plot(           ox , ymax=30, digits=1 )
BA.plot( perm.repl(ox), ymax=30, digits=1 )
```

plot.MCmcmc *Plot estimated conversion lines and formulae.*

Description

Plots the pairwise conversion formulae between methods from a MCmcmc object.

Usage

```
## S3 method for class 'MCmcmc'
plot(
  x,
  axlim = range(attr(x, "data")$y, na.rm = TRUE),
  wh.cmp,
  lwd.line = c(3, 1),
  col.line = rep("black", 2),
  lty.line = rep(1, 2),
  eqn = TRUE,
  digits = 2,
  grid = FALSE,
  col.grid = gray(0.8),
  points = FALSE,
  col.pts = "black",
  pch.pts = 16,
  cex.pts = 0.8,
  ...
)
```

Arguments

`x` A `MCmcmc` object

`axlim` The limits for the axes in the panels

`wh.cmp` Numeric vector or vector of method names. Which of the methods should be included in the plot?

`lwd.line` Numerical vector of length 2. The width of the conversion line and the prediction limits. If the second values is 0, no prediction limits are drawn.

`col.line` Numerical vector of length 2. The color of the conversion line and the prediction limits.

`lty.line` Numerical vector of length 2. The line types of the conversion line and the prediction limits.

`eqn` Should the conversion equations be printed on the plot?. Defaults to `TRUE`.

`digits` How many digits after the decimal point shoudl be used when printing the conversion equations.

`grid` Should a grid be drawn? If a numerical vector is given, the grid is drawn at those values.

`col.grid` What color should the grid have?

`points` Logical or character. Should the points be plotted. If `TRUE` or `"repl"` paired values of single replicates are plotted. If `"perm"`, replicates are randomly permuted within (item, method) befor plotting. If `"mean"`, means across replicates within item, method are formed and plotted.

`col.pts` What color should the observation have.

`pch.pts` What plotting symbol should be used.

`cex.pts` What scaling should be used for the plot symbols.

`...` Parameters to pass on. Currently not used.

Value

Nothing. The lower part of a (M-1) by (M-1) matrix of plots is drawn, showing the pairwise conversion lines. In the corners of each is given the two conversion equations together with the prediction standard error.

See Also

`MCmcmc`, `print.MCmcmc`

Examples

```
## Not run: data( hba1c )
## Not run: str( hba1c )
## Not run: hba1c <- transform( subset( hba1c, type=="Ven" ),
                    meth = dev,
                    repl = d.ana )
## End(Not run)
## Not run: hb.res <- MCmcmc( hba1c, n.iter=50 )
## Not run: data( hba.MC )
## Not run: str( hba.MC )
## Not run: par( ask=TRUE )
## Not run: plot( hba.MC )
## Not run: plot( hba.MC, pl.obs=TRUE )
```

plot.MethComp — *Summarize conversion equations and prediction intervals between methods.*

Description

plot.MethComp plots the conversion function with prediction limits; always using the original scale of measurements. It also sets the options "MethComp.wh.cmp" indicating which two methods are plotted and "MethComp.pl.type" indicating whether a plot of methods against each other or a Bland-Altman type plot of differences versus averages. By default the conversion lines are plotted.

Usage

```
## S3 method for class 'MethComp'
plot(
  x,
  wh.comp = 1:2,
  pl.type = "conv",
  dif.type = "lin",
  sd.type = "const",
  axlim = range(x$data$y, na.rm = TRUE),
  diflim = axlim - mean(axlim),
  points = FALSE,
  repl.conn = FALSE,
  col.conn = "gray",
  lwd.conn = 1,
  grid = TRUE,
  N.grid = 10,
  col.grid = grey(0.9),
  lwd = c(3, 1, 1),
  col.lines = "black",
  col.points = "black",
  pch.points = 16,
  eqn = is.null(attr(x, "Transform")),
  col.eqn = col.lines,
  font.eqn = 2,
  digits = 2,
  mult = FALSE,
  alpha = NULL,
  ...
)
```

Arguments

`x` A `MethComp` object.

`wh.comp` Numeric or character of length 2. Which two methods should be plotted.

`pl.type` Character. If "conv" it will be a plot of two methods against each other, otherwise it will be a plot of the 1st minus the 2nd versus the average; a Bland-Altman type plot.

`dif.type` Character. If "lin" (the default) a linear relationship between methods is allowed. Otherwise a constant difference is assumed and LoA can be indicated on the plot.

`sd.type` Should the estimated dependence of the SD (from `DA.reg` be used when plotting prediction limits?

`axlim` The extent of the axes of the measurements.

`diflim` The extent of the axis of the differences.

`points` Logical. Should the points be included in the plot.

`repl.conn` Logical. Should replcate measurements be connected; this assumes linked replicates.

`col.conn` Color of the lines connecting replicates.

`lwd.conn` Width of the connection lines.

`grid` Should there be a grid? If numerical, gridlines are drawn at these locations.

`N.grid` Numeric. How many gridlines? If a vector of length>1, it will be taken as the position of the gridlines.

`col.grid` Color of the gridlines.

`lwd` Numerical vector of length 3. Width of the conversion line and the prediction limits.

`col.lines` Color of the conversion lines.

`col.points` Color of the points.

`pch.points` Plot character for points.

`eqn` Logical. Should the conversion equation be printed on the plot.

`col.eqn` Color of the conversion formula

`font.eqn` font for the conversion formula

`digits` The number of digits after the decimal point in the conversion formulae.

`mult` Logical. Should ratios be plotted on a log-scale instead of differences on a linear scale? See description of the argument for `BA.plot`.

`alpha` 1 minus the confidence level for the prediction interval. If not given, the prediction interval is constructed as plus/minus twice the SD.

`...` Further arguments.

Details

`lines.MethComp` and `points.MethComp` adds conversion lines with prediction limits and points to a plot.

Value

MethComp returns a MethComp object, which is a list with three elements, Conv, a three-way array giving the linear conversion equations between methods, VarComp, a two-way array classified by methods and variance components and data, a copy of the original Meth object supplied — see the description under BA.est.

A MethComp object has an attribute Transform, which is either NULL, or a named list with elements trans and inv, both of which are functions. The first is the transformation applied to measurements before analysis; the results are all given on the transformed scale. The second is the inverse transformation; this is only used when plotting the resulting relationship between methods.

The methods print, plot, lines and points return nothing.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <bendix.carstensen@regionh.dk >.

See Also

BA.est AltReg MCmcmc

Examples

```
data( ox )
BA.ox <- BA.est( ox, linked=TRUE )
print( BA.ox )
## Not run:
AR.ox <- AltReg( ox, linked=TRUE  )
print( AR.ox )
plot( AR.ox )
## End(Not run)
```

---

plvol *Measurements of plasma volume measured by two different methods.*

Description

For each subject (item) the plasma volume is expressed as a percentage of the expected value for normal individuals. Two alternative sets of normal values are used, named Nadler and Hurley respectively.

Format

A data frame with 198 observations on the following 3 variables.

meth a factor with levels Hurley and Nadler

item a numeric vector

y a numeric vector

Source

The datset is adapted from table 2 in: JM Bland and DG Altman: Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999. Originally supplied to Bland \& Altman by C Dore, see: Cotes PM, Dore CJ, Liu Yin JA, Lewis SM, Messinezy M, Pearson TC, Reid C. Determination of serum immunoreactive erythropoietin in the investigation of erythrocytosis. New England Journal of Medicine 1986; 315: 283-87.

Examples

```
data(plvol)
str(plvol)
plot( y[meth=="Nadler"]~y[meth=="Hurley"],data=plvol,
      xlab="Plasma volume (Hurley) (pct)",
      ylab="Plasma volume (Nadler) (pct)" )
abline(0,1)
par( mar=c(4,4,1,4) )
BA.plot(plvol)
```

`predict.PBreg` *Predict results from PBreg object*

Description

A predict method for the "PBreg" class object, that is a result of Passing-Bablok regression.

Usage

```
## S3 method for class 'PBreg'
predict(
  object,
  newdata = object$model$x,
  interval = "confidence",
  level = 0.95,
  ...
)
```

Arguments

`object`	an object of class "PBreg".
`newdata`	an optional vector of new values of x to make predictions for. If omitted, the fitted values will be used.
`interval`	type of interval calculation - either confidence or none. The former is the default.
`level`	String. The type of interval to compute. Either "tolerance" or "confidence" (the default).
`...`	Not used

Value

If `interval` is `"confidence"` this function returns a data frame with three columns: "fit", "lwr" and "upr" - similarly to `predict.lm`.

If `interval` is `"none"` a vector of predicted values is returned.

Author(s)

Michal J. Figurski <`mfigrs@gmail.com`>

Examples

```
## Model data frame generation
a <- data.frame(x=seq(1, 30)+rnorm(mean=0, sd=1, n=30),
                y=seq(1, 30)*rnorm(mean=1, sd=0.4, n=30))

## Call to PBreg
x <- PBreg(a)
print(x)
predict(x, interval="none")

## Or the same using "Meth" object
a <- Meth(a, y=1:2)
x <- PBreg(a)
print(x)
predict(x)
```

`print.MCmcmc` *Print a MCmcmc object*

Description

Print a MCmcmc object

Usage

```
## S3 method for class 'MCmcmc'
print(x, digits = 3, alpha = 0.05, ...)
```

Arguments

`x`	an object used to select a method.
`digits`	Number of digits to print
`alpha`	Significance level
`...`	further arguments passed to or from other methods.

rainman *Perception of points in a swarm*

Description

Five raters were asked to guess the number of points in a swarm for 10 different figures (which - unknown to the raters - were each repeated three times).

Format

A data frame with 30 observations on the following 6 variables.

`SAND` The true number of points in the swarm. Each picture is replicated thrice

`ME` Ratings from judge 1

`TM` Ratings from judge 2

`AJ` Ratings from judge 3

`BM` Ratings from judge 4

`LO` Ratings from judge 5

Details

The raters had approximately 10 seconds to judge each picture, and they thought it were 30 different pictures. Before starting the experiment they were shown 6 (unrelated) pictures and were told the number of points in each of those pictures. The SAND column contains the picture id (which is also the true number of points in the swarm).

Source

Collected by Claus Ekstrom.

Examples

```
library(MethComp)
data( rainman )
str( rainman )
RM <- Meth( rainman, item=1, y=2:6 )
head( RM )
BA.est( RM, linked=FALSE )
library(lme4)
mf <- lmer( y ~ meth + item + (1|MI),
                data = transform( RM, MI=interaction(meth,item) ) )
summary( mf )
mr <- lmer( y ~ (1|meth) + (1|item) + (1|MI),
                data = transform( RM, MI=interaction(meth,item) ) )
summary( mr )
```

```
#
# Point swarms were generated by the following program
#
## Not run:
set.seed(2) # Original
npoints <- sample(4:30)*4
nplots <- 10
pdf(file="swarms.pdf", onefile=TRUE)

s1 <- sample(npoints[1:nplots])
print(s1)
for (i in 1:nplots) {
  n <- s1[i]
  set.seed(n)
  x <- runif(n)
  y <- runif(n)
  plot(x,y, xlim=c(-.15, 1.15), ylim=c(-.15, 1.15), pch=20, axes=F,
       xlab="", ylab="")
}
s1 <- sample(npoints[1:nplots])
print(s1)
for (i in 1:nplots) {
  n <- s1[i]
  set.seed(n)
  x <- runif(n)
  y <- runif(n)
  plot(y,x, xlim=c(-.15, 1.15), ylim=c(-.15, 1.15), pch=20, axes=F,
       xlab="", ylab="")
}
s1 <- sample(npoints[1:nplots])
print(s1)
for (i in 1:nplots) {
  n <- s1[i]
  set.seed(n)
  x <- runif(n)
  y <- runif(n)
  plot(-x,y, xlim=c(-1.15, .15), ylim=c(-.15, 1.15), pch=20, axes=F,
       xlab="", ylab="")
}
dev.off()

## End(Not run)
```

sample.Meth *Sample Meth object with replacement*

Description

Sample a Meth object with replacement. If how=="random", a random sample of the rows are sampled, the existing values of meth, item and y are kept but new replicate numbers are generated. If how=="linked", a random sample of the linked observations (i.e. observations with identical item and repl values) are sampled with replacement and replicate numbers are kept. If how=="item", items are sampled with replacement, and their observations are included the sampled numner of times.

Usage

```
sample.Meth(
  x,
  how = "random",
  N = if (how == "items") nlevels(x$item) else nrow(x)
)
```

Arguments

x	A Meth object.
how	Character. What sampling strategy should be used, one of "random", "linked" or "item". Only the first letter is significant. See details for explanation.
N	How many observations should be sampled?

Value

A meth object

Author(s)

Bendix Carstensen, <bendix.carstensen@regionh.dk>

Examples

```
data(fat)
# Different ways of selecting columns and generating replicate numbers
Sub1 <- Meth(fat,meth=2,item=1,repl=3,y=4,print=TRUE)
Sub2 <- Meth(fat,2,1,3,4,print=TRUE)
Sub3 <- Meth(fat,meth="Obs",item="Id",repl="Rep",y="Sub",print=TRUE)
summary( Sub3 )
plot( Sub3 )

# Use observation in different columns as methods
data( CardOutput )
head( CardOutput )
sv <- Meth( CardOutput, y=c("Svo2","Scvo2") )
# Note that replicates are generated if a non-unique item-id is used
sv <- Meth( CardOutput, y=c("Svo2","Scvo2"), item="Age" )
str( sv )
# A summary is not created if the the first argument (data=) is not used:
sv <- Meth( y=CardOutput[,c("Svo2","Scvo2")], item=CardOutput$VO2 )
summary(sv)

# Sample items
ssv <- sample.Meth( sv, how="item", N=8 )
```

```
# More than two methods
data( sbp )
plot( Meth( sbp ) )
# Creating non-unique replicate numbers per (meth,item) creates a warning:
data( hba1c )
hb1  <- with( hba1c,
              Meth( meth=dev, item=item, repl=d.ana-d.samp, y=y, print=TRUE ) )
hb2  <- with( subset(hba1c,type=="Cap"),
              Meth( meth=dev, item=item, repl=d.ana-d.samp, y=y, print=TRUE ) )
```

sbp *Systolic blood pressure measured by three different methods.*

Description

For each subject (item) there are three replicate measurements by three methods (two observers, J and R and the automatic machine, S). The replicates are linked within (method,item).

Format

A data frame with 765 observations on the following 4 variables:

- meth Methods, a factor with levels J(observer 1), R(observer 2) and S(machine)
- item Person id, numeric.
- repl Replicate number, a numeric vector
- y Systolic blood pressure masurement, a numeric vector

Source

The dataset is adapted from table 1 in: JM Bland and DG Altman: Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999. Originally supplied to Bland \& Altman by E. O'Brien, see: Altman DG, Bland JM. The analysis of blood pressure data. In O'Brien E, O'Malley K eds. Blood pressure measurement. Amsterdam: Elsevier, 1991: 287-314.

See Also

sbp.MC

Examples

```
data(sbp)
par( mfrow=c(2,2), mar=c(4,4,1,4) )
BA.plot( sbp, comp=1:2 )
BA.plot( sbp, comp=2:3 )
BA.plot( sbp, comp=c(1,3) )
## Not run:  BA.est( sbp, linked=TRUE )
```

sbp.MC *A MCmcmc object from the sbp data*

Description

This object is included for illustrative purposes. It is a result of using `MCmcmc`, with `n.iter=100000` on the dataset `sbp` from this package.

Format

The format is a `MCmcmc` object.

Details

The basic data are measurements of systolic blood pressure from the `sbp` dataset. Measurements are taken to be linked within replicate. The code used to generate the object was:

```
library(MethComp) data( sbp ) spb <- Meth( sbp ) sbp.MC <- MCmcmc( sbp,
linked=TRUE, n.iter=100000, program="JAGS" ) )
```

Examples

```
data(sbp.MC)
# How was the data generated
attr(sbp.MC,"mcmc.par")

# Traceplots
trace.MCmcmc(sbp.MC)
trace.MCmcmc(sbp.MC,"beta")

# A MCmcmc object also has class mcmc.list, so we can use the
# standard coda functions for convergence diagnostics:
# acfplot( subset.MCmcmc(sbp.MC,subset="sigma") )

# Have a look at the correlation between the 9 variance parameters
pairs( sbp.MC )

# Have a look at whether the MxI variance components are the same between methods:
## Not run:
pairs( sbp.MC, subset=c("mi"), eq=TRUE,
        panel=function(x,y,...)
              {
               abline(0,1)
               abline(v=median(x),h=median(y),col="gray")
               points(x,y,...)
              }
        )
## End(Not run)
```

`scint` *Relative renal function by Scintigraphy*

Description

Measurements of the relative kidney function (=renal function) for 111 patients. The percentage of the total renal function present in the left kidney is determined by one reference method, `DMSA` (static) and by one of two dynamic methods, `DTPA` or `EC`.

Format

A data frame with 222 observations on the following 5 variables:

`meth` Measurement method, a factor with levels `DMSA`, `DTPA`, `EC`.

`item` Patient identification.

`y` Percentage of total kidney function in the left kidney.

`age` Age of the patient.

`sex` Sex of the patient, a factor with levels `F`, `M`.

Source

F. C. Domingues, G. Y. Fujikawa, H. Decker, G. Alonso, J. C. Pereira, P. S. Duarte: Comparison of Relative Renal Function Measured with Either 99mTc-DTPA or 99mTc-EC Dynamic Scintigraphies with that Measured with 99mTc-DMSA Static Scintigraphy. International Braz J Urol Vol. 32 (4): 405-409, 2006

Examples

```
data(scint)
str(scint)
# Make a Bland-Altman plot for each of the possible comparisons:
par(mfrow=c(1,2),mgp=c(3,1,0)/1.6,mar=c(3,3,1,3))
BA.plot(scint,comp.levels=c(1,2),ymax=15,digits=1,cex=2)
BA.plot(scint,comp.levels=c(1,3),ymax=15,digits=1,cex=2)
```

`subset.MCmcmc` *Subset an MCmcmc object*

Description

Subset an MCmcmc object

Usage

```
## S3 method for class 'MCmcmc'
subset(x, subset = NULL, allow.repl = FALSE, chains = NULL, ...)
```

Arguments

`x`	object to be subsetted.
`subset`	Numerical, character or list giving the variables to keep. If numerical, the variables in the `MCmcmc` object with these numbers are selected. If character, each element of the character vector is "grep"ed against the variable names, and the matches are selected to the subset. If a list each element is used in turn, numerical and character elements can be mixed.
`allow.repl`	Logical. Should duplicate columns be allowed in the result?
`chains`	Numerical vector giving the number of the chains to keep.
`...`	further arguments to be passed to or from other methods.

`summary.MCmcmc` *Summary*

Description

Summary

Usage

```
## S3 method for class 'MCmcmc'
summary(object, alpha = 0.05, ...)
```

Arguments

`object`	An `MCmcmc` object
`alpha`	1 minus the the confidence level
`...`	Not used

summary.Meth *Summary for Meth obect*

Description

Summary for Meth obect

Usage

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

Arguments

object	A Meth object.
...	Parameters passed on to both the panel function plotting methods against each other, as well as to those plotting differences against means. rdname summary

TDI *Compute Lin's Total deviation index*

Description

This index calculates a value such that a certain fraction of difference between methods will be numerically smaller than this. The TDI is a measure which esentially is a number K such that the interval [-K,K] contains the limits of agreement.

Usage

```
TDI(y1, y2, p = 0.05, boot = 1000, alpha = 0.05)
```

Arguments

y1	Measurements by one method.
y2	Measurements by the other method.
p	The fraction of items with differences numerically exceeding the TDI
boot	If numerical, this is the number of bootstraps. If FALSE no confidence interval for the TDI is produced.
alpha	1 - confidende degree.

Details

If boot==FALSE a single number, the TDI is returned. If boot is a number, the median and the 1-alpha/2 central interval based on boot resamples are returned too, in a named vector of length 4.

Value

A list with 3 components. The names of the list are preceeded by the criterion percentage, i.e. the percentage of the population that the TDI is devised to catch.

`TDI` The numerically computed value for the TDI. If `boot` is numeric, a vector of median and a bootstrap c.i. is appended.

`TDI` The approximate value of the TDI

`Limits of Agreement`
Limits of agreement

Author(s)

Bendix Carstensen, `<bendix.carstensen@regionh.dk>`

References

LI Lin: Total deviation index for measuring individual agreement with applications in laboratory performance and bioequivalence, Statistics in Medicine, 19, 255-270 (2000)

Examples

```
data(plvol)
pw <- to.wide(plvol)
with(pw,TDI(Hurley,Nadler))
```

`to.long` *Functions to convert between long and wide representations of data*

Description

These functions are merely wrappers for `reshape`. Given the complicated syntax of `reshape` and the particularly simple structure of this problem, the functions facilitate the conversion enormously.

Usage

```
to.long(data, vars)
```

Arguments

`data` A `Meth` object.

`vars` The variables representing measurements by different methods. Either a character vector of names, or a numerical vector with the number of the variables in the dataframe.

Details

If data represents method comparisons with exchangeable replicates within method, the transformation to wide format does not necessarily make sense.

Value

A data frame with the reshaped data

Examples

```
data( milk )
str( milk )
mw <- to.wide( milk )
str( mw )
( mw <- subset( mw, as.integer(item) < 3 ) )
to.long( mw, 3:4 )
```

to.wide *Functions to convert between long and wide representations of data*

Description

These functions are merely wrappers for reshape. Given the complicated syntax of reshape and the particularly simple structure of this problem, the functions facilitate the conversion enormously.

Usage

```
to.wide(data, warn = TRUE)
```

Arguments

data A Meth object.

warn Logical. Should a warning be printed when replicates are taken as items?

Details

If data represents method comparisons with exchangeable replicates within method, the transformation to wide format does not necessarily make sense.

Value

A data frame with the reshaped data

Examples

```
data( milk )
str( milk )
mw <- to.wide( milk )
str( mw )
( mw <- subset( mw, as.integer(item) < 3 ) )
to.long( mw, 3:4 )
```

trace.MCmcmc *Functions to graphically assess the convergence of the MCMC-simulation in a MCmcmc object*

Description

These functions display traces for the relevant subset of the parameters in a MCmcmc object.

Usage

```
## S3 method for class 'MCmcmc'
trace(
  obj,
  what = "sd",
  scales = c("same", "free"),
  layout = "col",
  aspect = "fill",
  ...
)
```

Arguments

`obj` A `MCmcmc` object.

`what` Character indicating what parameters to plot. Possible values are `"sd"` or `"var"` which gives plots for the variance components (on the sd. scale), `"beta"` or `"slope"`, which gives plots for slope parameters and `"alpha"` or `"int"`, which gives plots for the intercept parameters.

`scales` Character vector of length two, with possible values "same" or "free", indicating whether x- and y-axes of the plots should be constrained to be the same across panels. For `pairs` only the first element is used to decide whether all panles should have the same axes.

`layout` Character. If `"col"` parameters are displayed columnwise by method, if `"row"` they are displayed row-wise.

`aspect` How should the panels be scaled. Default (`"fill"`) is to make a panels take up as much place as possible.

`...` Further aruments passed on to the `Lattice`

Details

A Lattice plot is returned, which means that it must printed when these functions are called in a batch program or inside another function or for-loop.

trace plots traces of the sampled chains, post plots posterior densities of the parameters and pairs plots a scatter-plot matrix of bivariate marginal posterior distributions.

Value

A Lattice plot.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <bendix.carstensen@regionh.dk >, http://BendixCarstensen.com.

See Also

MCmcmc, plot.MCmcmc, ox.MC, sbp.MC

Examples

```
# Load a provided MCmcmc object
data( ox.MC )
trace.MCmcmc( ox.MC, what="beta" )
pairs( ox.MC, what="sd" )
```

VitCap *Merits of two instruments designed to measure certain aspects of human lung function (Vital Capacity)*

Description

Measurement on certain aspects of human lung capacity for 72 patients on 4 instrument-operative combination, i.e. two different instruments and two different users, a skilled one and a new one.

Format

A data frame with 288 observations on the following 5 variables.

meth a factor with levels StNew, StSkil, ExpNew and ExpSkil, representing the instrument by user combinations. See below.

item a numeric vector, the person ID, i.e. the 72 patients

y a numeric vector, the measurements, i.e. vital capacity.

user a factor with levels New Skil, for the new user and the skilled user

instrument a factor with levels Exp and St, for the experimental instrument and the standard one.

Source

V. D. Barnett, Simultaneous Pairwise Linear Structural Relationships, Biometrics, Mar. 1969, Vol. 25, No. 1, pp. 129-142.

Examples

```
data(VitCap)
Vcap <- Meth( VitCap )
str( Vcap )
plot( Vcap )
```

---

y2DA *Convert DA to (classical) regression*

Description

The functions DA2y and y2DA are convenience functions that convert the estimates of intercept, slope and sd from the regression of $D = y_1 - y_2$ on $A = (y_1 + y_2)/2$, back and forth to the resulting intercept, slope and sd in the relationship between y_1 and y_2, cf. Carstensen (2010), equation 6.

Usage

```
y2DA(A = 0, B = 1, S = NA)
```

Arguments

A	Intercept in the linear relation of y1 on y2.
B	Slope in the linear relation of y1 on y2.
S	SD for the linear relation of y1 on y2. Can be NA.

Details

#' y2DA takes intercept(A), slope(B) and sd(S) from the relationship y1=A+B y2 + E with sd(E)=E, and returns a vector of length 3 with names "int(t-f)","slope(t-f)","sd(t-f)", where t refers to "to" (y1 and f to "from" y2.

Value

y2DA returns a 3-component vector with names c("DA-int","DA-slope","DA-sd"), referring to differences D=y1-y2 as a linear function of A=(y1+y2)/2.

Author(s)

Bendix Carstensen, Steno Diabetes Center, <bendix.carstensen@regionh.dk>, http://BendixCarstensen.com/MethComp

References

B. Carstensen: Comparing methods of measurement: Extending the LoA by regression. Stat Med, 29:401-410, 2010.

Examples

```
data( milk )
DA.reg( milk )
data( sbp )
print( DA.reg(sbp), digits=3 )
# Slope, intercept : y1 = 0.7 + 1.2*y2 (0.4)
A <- c(0.7,1.2,0.4)
( y2DA( A ) )
( DA2y( y2DA( A ) ) )
```

Index