Package ‘meta’

May 2, 2025

Title General Package for Meta-Analysis

Version 8.1-0

Date 2025-05-02

Depends R (>= 4.0.0), metadat

Imports metafor (>= 3.0-0), grid, ggplot2, lme4, CompQuadForm, xml2, methods, readr, dplyr, magrittr, purrr, stringr, tibble, scales

Suggests netmeta, BiasedUrn, pimeta, estmeansd, robvis, brglm2, writexl, rmarkdown, knitr, ggpubr, gridExtra

Author Guido Schwarzer [cre, aut] (ORCID: <https://orcid.org/0000-0001-6214-9087>)

Maintainer Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

URL https://github.com/guido-s/meta/
https://link.springer.com/book/10.1007/978-3-319-21416-0

Description User-friendly general package providing standard methods for meta-analysis and supporting Schwarzer, Carpenter, and Rücker <DOI:10.1007/978-3-319-21416-0>, ``Meta-Analysis with R'' (2015):
- common effect and random effects meta-analysis;
- several plots (forest, funnel, Galbraith / radial, L'Abbe, Baujat, bubble);
- three-level meta-analysis model;
- generalised linear mixed model;
- logistic regression with penalised likelihood for rare events;
- Hartung-Knapp method for random effects model;
- Kenward-Roger method for random effects model;
- prediction interval;
- statistical tests for funnel plot asymmetry;
- trim-and-fill method to evaluate bias in meta-analysis;
- meta-regression;
- cumulative meta-analysis and leave-one-out meta-analysis;
- import data from 'RevMan 5';
- produce forest plot summarising several (subgroup) meta-analyses.

License GPL (>= 2)

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2025-05-02 16:10:02 UTC

Contents

meta-package *meta: Brief overview of methods and general hints*

Description

R package **meta** is a user-friendly general package providing standard methods for meta-analysis and supporting Schwarzer et al. (2015), https://link.springer.com/book/10.1007/978-3-319-21416-0.

Details

R package **meta** (Schwarzer, 2007; Balduzzi et al., 2019) provides the following statistical methods for meta-analysis.

1. Common effect (also called fixed effect) and random effects model:
   - Meta-analysis of continuous outcome data (metacont)
   - Meta-analysis of binary outcome data (metabin)
   - Meta-analysis of incidence rates (metainc)
   - Generic inverse variance meta-analysis (metagen)
   - Meta-analysis of single correlations (metacor)
   - Meta-analysis of single means (metamean)
   - Meta-analysis of single proportions (metaprop)
   - Meta-analysis of single incidence rates (metarate)
2. Several plots for meta-analysis:
   - Forest plot (forest.meta, forest.metabind)
   - Funnel plot (funnel.meta)
   - Galbraith plot / radial plot (radial.meta)
   - L'Abbe plot for meta-analysis with binary outcome data (labbe.metabin, labbe.default)
   - Baujat plot to explore heterogeneity in meta-analysis (baujat.meta)
   - Bubble plot to display the result of a meta-regression (bubble.metareg)
3. Three-level meta-analysis model (Van den Noortgate et al., 2013)
4. Generalised linear mixed models (GLMMs) for binary and count data (Stijnen et al., 2010) (metabin, metainc, metaprop, and metarate)
5. Logistic regression with penalised likelihood for rare binary data (Evrenoglou et al., 2022) (metabin,
6. Various estimators for the between-study variance τ^2 in a random effects model (Veroniki et al., 2016); see description of argument `method.tau` below
7. Two methods to estimate the I-squared statistic (Higgins and Thompson, 2002); see description of argument `method.I2` below
8. Hartung-Knapp method for random effects meta-analysis (Hartung & Knapp, 2001a,b), see description of arguments `method.random.ci` and `adhoc.hakn.ci` below
9. Kenward-Roger method for random effects meta-analysis (Partlett and Riley, 2017), see description of arguments `method.random.ci` and `method.predict` below

10. Prediction interval for the treatment effect of a new study (Veroniki et al., 2019; Higgins et al., 2009; Partlett and Riley, 2017; Nagashima et al., 2019), see description of argument `method.predict` below
11. Statistical tests for funnel plot asymmetry (`metabias.meta`, `metabias.rm5`) and trim-and-fill method (`trimfill.meta`, `trimfill.default`) to evaluate bias in meta-analysis
12. Meta-regression (`metareg`)
13. Cumulative meta-analysis (`metacum`) and leave-one-out meta-analysis (`metainf`)
14. Import data from RevMan Web (`read.cdir`), RevMan 5 (`read.rm5`), see also `metacr` to conduct meta-analysis for a single comparison and outcome from a Cochrane review

R package **meta** provides two vignettes:

- `vignette("meta-workflow")` with an overview of main functions,
- `vignette("meta-tutorial")` with up-to-date commands for Balduzzi et al. (2019).

Additional statistical meta-analysis methods are provided by add-on R packages:

- Frequentist methods for network meta-analysis (R package **netmeta**)
- Statistical methods for sensitivity analysis in meta-analysis (R package **metasens**)
- Statistical methods for meta-analysis of diagnostic accuracy studies with several cutpoints (R package **diagmeta**)

In the following, more details on available and default statistical meta-analysis methods are provided and R function `settings.meta` is briefly described which can be used to change the default settings. Additional information on meta-analysis objects and available summary measures can be found on the help pages `meta-object` and `meta-sm`.

Estimation of between-study variance:

The following methods are available in all meta-analysis functions to estimate the between-study variance τ^2.

Argument	**Method**
`method.tau = "REML"`	Restricted maximum-likelihood estimator (Viechtbauer, 2005) (default)
`method.tau = "PM"`	Paule-Mandel estimator (Paule and Mandel, 1982)
`method.tau = "DL"`	DerSimonian-Laird estimator (DerSimonian and Laird, 1986)
`method.tau = "ML"`	Maximum-likelihood estimator (Viechtbauer, 2005)
`method.tau = "HS"`	Hunter-Schmidt estimator (Hunter and Schmidt, 2015)
`method.tau = "SJ"`	Sidik-Jonkman estimator (Sidik and Jonkman, 2005)
`method.tau = "HE"`	Hedges estimator (Hedges and Olkin, 1985)
`method.tau = "EB"`	Empirical Bayes estimator (Morris, 1983)

For GLMMs, only the maximum-likelihood method is available.

Historically, the DerSimonian-Laird method was the de facto standard to estimate the between-study variance τ^2 and is the default in some software packages including Review Manager 5 (RevMan 5) and R package **meta**, version 4 and below. However, its role has been challenged and especially the REML and Paule-Mandel estimators have been recommended (Veroniki et al., 2016; Langan et al., 2019). Accordingly, the currenct default in R package **meta** is the REML estimator.

The following R command could be used to employ the Paule-Mandel instead of the REML estimator in all meta-analyses of the current R session:

- `settings.meta(method.tau = "PM")`

Other estimators for τ^2 could be selected in a similar way.

Note, for binary outcomes, two variants of the DerSimonian-Laird estimator are available if the Mantel-Haenszel method is used for pooling. If argument `Q.Cochrane = TRUE` (default), the heterogeneity statistic Q is based on the Mantel-Haenszel instead of the inverse variance estimator under the common effect model. This is the estimator for τ^2 implemented in RevMan 5.

Estimation of I-squared statistic:

The following methods are available in all meta-analysis functions to estimate the I-squared statistic (Higgins and Thompson, 2002).

Argument	**Method**
`method.I2 = "Q"`	Based on heterogeneity statistic Q (default)
`method.I2 = "tau2"`	Based on between-study variance τ^2

Using `method.I2 = "Q"` (Higgins and Thompson, 2002, section 3.3), the value of I^2 does not change if the estimate of τ^2 changes. Furthermore, the value of I^2 and the test of heterogeneity based on the Q statistic are in agreement. R package **metafor** uses the second method (`method.I2 = "tau2"`) which is described in Higgins and Thompson (2002), section 3.2. This method is more general in the way that the value of I^2 changes with the estimate of τ^2.

Confidence interval for random effects estimate:

The following methods are available in all meta-analysis functions to calculate a confidence interval for the random effects estimate.

Argument	**Method**
`method.random.ci = "classic"`	Based on standard normal quantile (DerSimonian and Laird, 1986) (default)
`method.random.ci = "HK"`	Method by Hartung and Knapp (2001a/b)
`method.random.ci = "KR"`	Kenward-Roger method (Partlett and Riley, 2017)

DerSimonian and Laird (1986) introduced the classic random effects model using a quantile of the standard normal distribution to calculate a confidence interval for the random effects estimate. This method implicitly assumes that the weights in the random effects meta-analysis are not estimated but given. Particularly, the uncertainty in the estimation of the between-study variance τ^2 is ignored.

Hartung and Knapp (2001a,b) proposed an alternative method for random effects meta-analysis based on a refined variance estimator for the treatment estimate and a quantile of a *t*-distribution with *k-1* degrees of freedom where *k* corresponds to the number of studies in the meta-analysis.

The Kenward-Roger method is only available for the REML estimator (`method.tau = "REML"`) of the between-study variance τ^2 (Partlett and Riley, 2017). This method is based on an adjusted variance estimate for the random effects estimate. Furthermore, a quantile of a *t*-distribution with adequately modified degrees of freedom is used to calculate the confidence interval.

For GLMMs and three-level models, the Kenward-Roger method is not available, but a method similar to Knapp and Hartung (2003) is used if `method.random.ci = "HK"`. For this method, the variance estimator is not modified, however, a quantile of a *t*-distribution with *k-1* degrees of freedom is used; see description of argument `test` in `rma.glmm` and `rma.mv`.

Simulation studies (Hartung and Knapp, 2001a,b; IntHout et al., 2014; Langan et al., 2019) show improved coverage probabilities of the Hartung-Knapp method compared to the classic random effects method. However, in rare settings with very homogeneous treatment estimates, the Hartung-Knapp variance estimate can be arbitrarily small resulting in a very narrow confidence interval (Knapp and Hartung, 2003; Wiksten et al., 2016). In such cases, an *ad hoc* variance correction has been proposed by utilising the variance estimate from the classic random effects model with the Hartung-Knapp method (Knapp and Hartung, 2003; IQWiQ, 2022). An alternative *ad hoc* approach is to use the confidence interval of the classic common or random effects meta-analysis if it is wider than the interval from the Hartung-Knapp method (Wiksten et al., 2016; Jackson et al., 2017).

Argument `adhoc.hakn.ci` can be used to choose the *ad hoc* correction for the Hartung-Knapp (HK) method:

Argument	***Ad hoc* method**
`adhoc.hakn.ci = ""`	no *ad hoc* correction (default)
`adhoc.hakn.ci = "se"`	use variance correction if HK standard error is smaller than standard error from classic random effects meta-analysis (Knapp and Hartung, 2003)
`adhoc.hakn.ci = "IQWiG6"`	use variance correction if HK confidence interval is narrower than CI from classic random effects model with DerSimonian-Laird estimator (IQWiG, 2022)
`adhoc.hakn.ci = "ci"`	use wider confidence interval of classic random effects and HK meta-analysis (Hybrid method 2 in Jackson et al., 2017)

For GLMMs and three-level models, the *ad hoc* variance corrections are not available.

Prediction interval:

The following methods are available in all meta-analysis functions to calculate a prediction interval for the treatment effect in a single new study.

Argument	**Method**
`method.predict = "V"`	Based on *t*-distribution with *k-1* degrees of freedom (Veroniki et al., 2019) (default)
`method.predict = "HTS"`	Based on *t*-distribution with *k-2* degrees of freedom (Higgins et al., 2009)
`method.predict = "HK"`	Based on Hartung-Knapp standard error and *t*-distribution with *k-1* degrees of freedom
`method.predict = "HK-PR"`	Based on Hartung-Knapp standard error and *t*-distribution with *k-2* degrees of freedom (Partlett and Riley, 2017)
`method.predict = "KR"`	Based on Kenward-Roger standard error and *t*-distribution with approximate Kenward-Roger degrees of freedom

`method.predict = "KR-PR"`	Based on Kenward-Roger standard error and *t*-distribution with approximate Kenward-Roger degrees of freedom minus 1 (Partlett and Riley, 2017)
`method.predict = "NNF"`	Bootstrap approach (Nagashima et al., 2019)
`method.predict = "S"`	Based on standard normal quantile (Skipka, 2006)

By default (`method.predict = "V"`), the prediction interval is based on a *t*-distribution with *k-1* degrees of freedom where *k* corresponds to the number of studies in the meta-analysis (Veroniki et al., 2019). The method by Higgins et al., (2009), which is based on a *t*-distribution with *k-2* degrees of freedom, has been the default in R package **meta**, version 7.0-0 or lower.

The Hartung-Knapp prediction intervals are also based on a *t*-distribution, however, use a different standard error.

The Kenward-Roger method is only available for the REML estimator (`method.tau = "REML"`) of the between-study variance τ^2 (Partlett and Riley, 2017). This method is based on an adjusted variance estimate for the random effects estimate. Furthermore, a quantile of a *t*-distribution with adequately modified degrees of freedom is used to calculate the prediction interval.

The bootstrap approach is only available if R package **pimeta** is installed (Nagashima et al., 2019). Internally, the `pima` function is called with argument `method = "boot"`. Argument `seed.predict` can be used to get a reproducible bootstrap prediction interval and argument `seed.predict.subgroup` for reproducible bootstrap prediction intervals in subgroups.

The method of Skipka (2006) ignores the uncertainty in the estimation of the between-study variance τ^2 and thus has too narrow limits for meta-analyses with a small number of studies.

For GLMMs and three-level models, only the methods by Veroniki et al. (2019), Higgins et al. (2009) and Skipka (2006) are available. Argument `method.predict = "V"` in R package **meta** gives the same prediction intervals as R functions `rma.glmm` or `rma.mv` with argument `test = "t"`.

Note, in R package **meta**, version 7.0-0 or lower, the methods `method.predict = "HK-PR"` and `method.predict = "KR-PR"` have been available as `method.predict = "HK"` and `method.predict = "KR"`.

Argument `adhoc.hakn.pi` can be used to choose the *ad hoc* correction for the Hartung-Knapp method:

Argument	***Ad hoc* method**
`adhoc.hakn.pi = ""`	no *ad hoc* correction (default)
`adhoc.hakn.pi = "se"`	use variance correction if HK standard error is smaller

Confidence interval for the between-study variance:

The following methods are available in all meta-analysis functions to calculate a confidence interval for τ^2 and τ.

Argument	**Method**
`method.tau.ci = "J"`	Method by Jackson (2013)
`method.tau.ci = "BJ"`	Method by Biggerstaff and Jackson (2008)
`method.tau.ci = "QP"`	Q-Profile method (Viechtbauer, 2007)
`method.tau.ci = "PL"`	Profile-Likelihood method for three-level meta-analysis model (Van den Noortgate et al., 2013)
`method.tau.ci = ""`	No confidence interval

The first three methods have been recommended by Veroniki et al. (2016). By default, the Jackson method is used for the DerSimonian-Laird estimator of τ^2 and the Q-profile method for all other estimators of τ^2.

The Profile-Likelihood method is the only method available for the three-level meta-analysis model.

For GLMMs, no confidence intervals for τ^2 and τ are calculated.

Change default settings for R session:

R function `settings.meta` can be used to change the previously described and several other default settings for the current R session.

Some pre-defined general settings are available:

- `settings.meta("RevMan5")`
- `settings.meta("JAMA")`
- `settings.meta("BMJ")`
- `settings.meta("IQWiG5")`
- `settings.meta("IQWiG6")`
- `settings.meta("geneexpr")`

The first command can be used to reproduce meta-analyses from Cochrane reviews conducted with *Review Manager 5* (RevMan 5) and specifies to use a RevMan 5 layout in forest plots.

The second command can be used to generate forest plots following instructions for authors of the *Journal of the American Medical Association*. Study labels according to JAMA guidelines can be generated using `labels.meta`.

The third command can be used to generate forest plots in the current layout of the *British Medical Journal*.

The next two commands implement the recommendations of the Institute for Quality and Efficiency in Health Care (IQWiG), Germany accordinging to General Methods 5 and 6, respectively (`https://www.iqwig.de/en/about-us/methods/methods-paper/`).

The last setting can be used to print p-values in scientific notation and to suppress the calculation of confidence intervals for the between-study variance.

See `settings.meta` for more details on these pre-defined general settings.

In addition, `settings.meta` can be used to define individual settings for the current R session. For example, the following R command specifies the use of Hartung-Knapp and Paule-Mandel method, and the printing of prediction intervals for any meta-analysis generated after execution of this command:

- `settings.meta(method.random.ci = "HK", method.tau = "PM", prediction = TRUE)`

Data sets: The following data sets are available in R package **meta**.

Data set	**Description**
`Fleiss1993bin`	Aspirin after myocardial infarction
`Fleiss1993cont`	Mental health treatment on medical utilisation
`Olkin1995`	Thrombolytic therapy after acute myocardial infarction
`Pagliaro1992`	Prevention of first bleeding in cirrhosis
`amlodipine`	Amlodipine for work capacity
`caffeine`	Caffeine for daytime drowsiness (Cochrane Practice review)

cisapride	Cisapride in non-ulcer dispepsia
lungcancer	Smoking example
smoking	Smoking example
woodyplants	Elevated CO$_2$ and total biomass of woody plants

R package **metadat** has a large collection of meta-analysis data sets.

Note

Balduzzi et al. (2019) is the preferred citation in publications for **meta**. Type `citation("meta")` for a BibTeX entry of this publication.

Type `help(package = "meta")` for a listing of all R functions and datasets available in **meta**. For example, results of several meta-analyses can be combined with `metabind` which is useful to generate a forest plot with results of several subgroup analyses.

R package **meta** imports R functions from **metafor** (Viechtbauer, 2010) to

- estimate the between-study variance τ^2,
- conduct meta-regression,
- estimate three-level models,
- estimate generalised linear mixed models.

To report problems and bugs

- type `bug.report(package = "meta")` if you do not use RStudio,
- send an email to Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`> if you use RStudio.

The development version of **meta** is available on GitHub `https://github.com/guido-s/meta/`.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

References

Balduzzi S, Rücker G, Schwarzer G (2019): How to perform a meta-analysis with R: a practical tutorial. *Evidence-Based Mental Health*, **22**, 153–160

Biggerstaff BJ, Jackson D (2008): The exact distribution of Cochran's heterogeneity statistic in one-way random effects meta-analysis. *Statistics in Medicine*, **27**, 6093–110

DerSimonian R & Laird N (1986): Meta-analysis in clinical trials. *Controlled Clinical Trials*, **7**, 177–88

Evrenoglou T, White IR, Afach S, Mavridis D, Chaimani A. (2022): Network meta-analysis of rare events using penalized likelihood regression. *Statistics in Medicine*, **41**, 5203–19

Hartung J, Knapp G (2001a): On tests of the overall treatment effect in meta-analysis with normally distributed responses. *Statistics in Medicine*, **20**, 1771–82

Hartung J, Knapp G (2001b): A refined method for the meta-analysis of controlled clinical trials with binary outcome. *Statistics in Medicine*, **20**, 3875–89

Hedges LV & Olkin I (1985): *Statistical methods for meta-analysis*. San Diego, CA: Academic Press

Higgins JPT & Thompson SG (2002): Quantifying heterogeneity in a meta-analysis. *Statistics in Medicine*, **21**, 1539–58

Higgins JPT, Thompson SG, Spiegelhalter DJ (2009): A re-evaluation of random-effects meta-analysis. *Journal of the Royal Statistical Society: Series A*, **172**, 137–59

Hunter JE & Schmidt FL (2015): *Methods of Meta-Analysis: Correcting Error and Bias in Research Findings* (Third edition). Thousand Oaks, CA: Sage

IntHout J, Ioannidis JPA, Borm GF (2014): The Hartung-Knapp-Sidik-Jonkman method for random effects meta-analysis is straightforward and considerably outperforms the standard DerSimonian-Laird method. *BMC Medical Research Methodology*, **14**, 25

IQWiG (2022): General Methods: Version 6.1. https://www.iqwig.de/en/about-us/methods/methods-paper/

Jackson D (2013): Confidence intervals for the between-study variance in random effects meta-analysis using generalised Cochran heterogeneity statistics. *Research Synthesis Methods*, **4**, 220–229

Jackson D, Law M, Rücker G, Schwarzer G (2017): The Hartung-Knapp modification for random-effects meta-analysis: A useful refinement but are there any residual concerns? *Statistics in Medicine*, **36**, 3923–34

Knapp G & Hartung J (2003): Improved tests for a random effects meta-regression with a single covariate. *Statistics in Medicine*, **22**, 2693–710

Langan D, Higgins JPT, Jackson D, Bowden J, Veroniki AA, Kontopantelis E, et al. (2019): A comparison of heterogeneity variance estimators in simulated random-effects meta-analyses. *Research Synthesis Methods*, **10**, 83–98

Schwarzer G (2007): meta: An R package for meta-analysis. *R News*, **7**, 40–5

Schwarzer G, Carpenter JR and Rücker G (2015): *Meta-Analysis with R (Use-R!)*. Springer International Publishing, Switzerland

Skipka G (2006): The inclusion of the estimated inter-study variation into forest plots for random effects meta-analysis - a suggestion for a graphical representation [abstract]. *XIV Cochrane Colloquium, Dublin*, 23-26.

Stijnen T, Hamza TH, Ozdemir P (2010): Random effects meta-analysis of event outcome in the framework of the generalized linear mixed model with applications in sparse data. *Statistics in Medicine*, **29**, 3046–67

Veroniki AA, Jackson D, Viechtbauer W, Bender R, Bowden J, Knapp G, et al. (2016): Methods to estimate the between-study variance and its uncertainty in meta-analysis. *Research Synthesis Methods*, **7**, 55–79

Veroniki AA, Jackson D, Bender R, Kuss O, Higgins JPT, Knapp G, Salanti G (2019): Methods to calculate uncertainty in the estimated overall effect size from a random-effects meta-analysis. *Research Synthesis Methods*, **10**, 23–43

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

Viechtbauer W (2005): Bias and efficiency of meta-analytic variance estimators in the random-effects model. *Journal of Educational and Behavioral Statistics*, **30**, 261–93

Viechtbauer W (2007): Confidence intervals for the amount of heterogeneity in meta-analysis. *Statistics in Medicine*, **26**, 37–52

Viechtbauer W (2010): Conducting Meta-Analyses in R with the metafor Package. *Journal of Statistical Software*, **36**, 1–48

Wiksten A, Rücker G, Schwarzer G (2016): Hartung-Knapp method is not always conservative compared with fixed-effect meta-analysis. *Statistics in Medicine*, **35**, 2503–15

See Also

`meta-object`, `meta-sm`

`amlodipine` *Amlodipine for Work Capacity*

Description

Meta-analysis on the effect of amlodipine on work capacity.

This meta-analysis is used as a data example in Hartung and Knapp (2001).

Format

A data frame with the following columns:

study	study label
n.amlo	number of observations in amlodipine group
mean.amlo	estimated mean in amlodipine group
var.amlo	variance in amlodipine group
n.plac	number of observations in placebo group
mean.plac	estimated mean in placebo group
var.plac	variance in placebo group

Source

Hartung J & Knapp G (2001): On tests of the overall treatment effect in meta-analysis with normally distributed responses. *Statistics in Medicine*, **20**, 1771–82

See Also

`metacont`

Examples

```
data(amlodipine)

m <- metacont(n.amlo, mean.amlo, sqrt(var.amlo),
  n.plac, mean.plac, sqrt(var.plac),
  data = amlodipine, studlab = study,
  method.tau = "DL")
m.hk <- update(m, method.random.ci = "HK")

# Same results for mean difference as in Table III in Hartung and
# Knapp (2001)
#
vars.common <- c("TE.common", "lower.common", "upper.common")
vars.random <- c("TE.random", "lower.random", "upper.random")
#
res.common <- as.data.frame(m[vars.common])
names(res.common) <- vars.random
#
res.md <- rbind(res.common,
  as.data.frame(m[vars.random]),
  as.data.frame(m.hk[vars.random]))
#
res.md <- round(res.md, 5)
#
row.names(res.md) <- c("CE", "RE", "RE (HaKn)")
names(res.md) <- c("Absolute difference", "CI lower", "CI upper")
#
res.md
```

---

as.data.frame.meta *Coerce to a data frame*

Description

The `as.data.frame` method returns a data frame containing information on individual studies, e.g., estimated treatment effect and its standard error.

Usage

```
## S3 method for class 'meta'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	An object of class `meta`.
`row.names`	`NULL` or a character vector giving the row names for the data frame.
`optional`	logical. If `TRUE`, setting row names and converting column names (to syntactic names) is optional.
`...`	other arguments

Value

A data frame is returned by the function as.data.frame.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

metabin, metacont, metagen, forest.meta

Examples

```
data(Fleiss1993cont)
#
# Generate additional variable with grouping information
#
Fleiss1993cont$group <- c(1, 2, 1, 1, 2)
#
# Do meta-analysis without grouping information
#
m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD", studlab = paste(study, year))
#
# Update meta-analysis object and do subgroup analyses
#
update(m1, subgroup = group)

# Same result using metacont function directly
#
m2 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD", studlab = paste(study, year),
  subgroup = group)
m2

# Compare printout of the following two commands
#
as.data.frame(m1)
m1$data
```

barplot.rob *Produce weighted bar plot of risk of bias assessment*

Description

Produce weighted bar plot of risk of bias assessment

Usage

```
## S3 method for class 'rob'
barplot(
  height,
  overall = FALSE,
  weighted = TRUE,
  colour = ”cochrane”,
  quiet = FALSE,
  ...
)
```

Arguments

`height`	An object of class `rob`.
`overall`	A logical indicating whether to include a bar for overall risk of bias in the figure.
`weighted`	A logical indicating whether weights should be used in the bar plot.
`colour`	Specify colour scheme for the bar plot; see `rob_summary`.
`quiet`	A logical to suppress the display of the bar plot.
`...`	Additional arguments (ignored)

Details

This is a wrapper function for `rob_summary` of R package **robvis** to produce a weighted bar plot of risk of bias assessment.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

See Also

`rob`, `traffic_light`, `rob_summary`

Examples

```
# Use RevMan 5 settings
oldset <- settings.meta(”RevMan5”)

data(caffeine)

m1 <- metabin(h.caf, n.caf, h.decaf, n.decaf, sm = ”OR”,
  data = caffeine, studlab = paste(study, year))

# Add risk of bias assessment to meta-analysis
m2 <- rob(D1, D2, D3, D4, D5, overall = rob, data = m1, tool = ”rob2”)

# Print risk of bias assessment
rob(m2)
```

```
## Not run:
# Weighted bar plot (R package 'robvis' must be available)
if (requireNamespace("robvis", quietly = TRUE))
 barplot(rob(m2))

## End(Not run)

# Use previous settings
settings.meta(oldset)
```

baujat.meta *Baujat plot to explore heterogeneity in meta-analysis*

Description

Draw a Baujat plot to explore heterogeneity in meta-analysis.

Usage

```
## S3 method for class 'meta'
baujat(
  x,
  yscale = 1,
  xlim,
  ylim,
  xlab = "Contribution to overall heterogeneity",
  ylab = "Influence on overall result",
  pch = 21,
  cex = 1,
  col = "black",
  bg = "darkgray",
  studlab = TRUE,
  cex.studlab = 0.8,
  pos.studlab,
  offset = 0.5,
  xmin = 0,
  ymin = 0,
  grid = TRUE,
  col.grid = "lightgray",
  lty.grid = "dotted",
  lwd.grid = par("lwd"),
  pty = "s",
  pooled,
  ...
)
```

Arguments

`x` An object of class `meta`.

`yscale` Scaling factor for values on y-axis.

`xlim` The x limits (min,max) of the plot.

`ylim` The y limits (min,max) of the plot.

`xlab` A label for the x-axis.

`ylab` A label for the y-axis.

`pch` The plotting symbol used for individual studies.

`cex` The magnification to be used for plotting symbol.

`col` A vector with colour of plotting symbols.

`bg` A vector with background colour of plotting symbols (only used if `pch` in `21:25`).

`studlab` A logical indicating whether study labels should be printed in the graph. A vector with study labels can also be provided (must be of same length as `x$TE` then).

`cex.studlab` The magnification for study labels.

`pos.studlab` Position of study labels, see argument `pos` in `text`.

`offset` Offset for study labels (see `text`).

`xmin` A numeric specifying minimal value to print study labels (on x-axis).

`ymin` A numeric specifying minimal value to print study labels (on y-axis).

`grid` A logical indicating whether a grid is printed in the plot.

`col.grid` Colour for grid lines.

`lty.grid` The line type for grid lines.

`lwd.grid` The line width for grid lines.

`pty` A character specifying type of plot region (see `par`).

`pooled` A character string indicating whether a common effect or random effects model is used for pooling. Either missing (see Details), `"common"` or `"random"`, can be abbreviated.

`...` Graphical arguments as in `par` may also be passed as arguments.

Details

Baujat et al. (2002) introduced a scatter plot to explore heterogeneity in meta-analysis. On the x-axis the contribution of each study to the overall heterogeneity statistic (see list object `Q` of the meta-analysis object `x`) is plotted. On the y-axis the standardised difference of the overall treatment effect with and without each study is plotted; this quantity describes the influence of each study on the overal treatment effect.

Information from object `x` is utilised if argument `pooled` is missing. A common effect model is assumed (`pooled="common"`) if argument `x$common` is `TRUE`; a random effects model is assumed (`pooled="random"`) if argument `x$random` is `TRUE` and `x$common` is `FALSE`.

Internally, the `metainf` function is used to calculate the values on the y-axis.

Value

A data.frame with the following variables:

x	Coordinate on x-axis (contribution to heterogeneity statistic)
y	Coordinate on y-axis (influence on overall treatment effect)

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Baujat B, Mahé C, Pignon JP, Hill C (2002): A graphical method for exploring heterogeneity in meta-analyses: Application to a meta-analysis of 65 trials. *Statistics in Medicine*, **30**, 2641–52

See Also

metagen, metainf

Examples

```
data(Olkin1995)

# Only consider first ten studies
m1 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, sm = "OR", method = "I", studlab = paste(author, year),
  subset = 1:10)

# Generate Baujat plot
baujat(m1)

## Not run:
m1 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, sm = "OR", method = "I", studlab = paste(author, year))

# Do not print study labels if the x-value is smaller than 4 and
# the y-value is smaller than 1
baujat(m1, yscale = 10, xmin = 4, ymin = 1)

# Change position of study labels
baujat(m1, yscale = 10, xmin = 4, ymin = 1,
       pos = 1, xlim = c(0, 6.5))

# Generate Baujat plot and assign x- and y- coordinates to R object
# b1
b1 <- baujat(m1)

# Calculate overall heterogeneity statistic
sum(b1$x)
m1$Q
```

```
## End(Not run)
```

blup.meta — *Calculate best linear unbiased predictor for* `meta` *object*

Description

Calculate best linear unbiased predictors (BLUPs) for meta-analysis object created with R package **meta**.

Usage

```
## S3 method for class 'meta'
blup(x, level = x$level, backtransf = x$backtransf, ...)

## S3 method for class 'blup.meta'
print(
  x,
  backtransf = attr(x, "x")$backtransf,
  digits = gs("digits"),
  digits.se = gs("digits.se"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  big.mark = gs("big.mark"),
  se = FALSE,
  print.tau2 = gs("print.tau2"),
  print.tau = gs("print.tau"),
  details = gs("details"),
  ...
)

## S3 method for class 'blup.meta'
estimates(
  x,
  se = FALSE,
  backtransf = attr(x, "x")$backtransf,
  digits = gs("digits"),
  digits.se = gs("digits.se"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  writexl = !missing(path),
  path = "estimates_blup.xlsx",
  overwrite = FALSE,
  ...
)
```

```
## S3 method for class 'estimates.blup.meta'
print(x, big.mark = gs("big.mark"), details = gs("details"), ...)
```

Arguments

`x` An object of class `meta`, `blup.meta`, or `estimates.blup.meta`.

`level` The level used to calculate prediction intervals for BLUPs.

`backtransf` A logical indicating whether BLUPs should be back transformed. If `backtransf = TRUE`, results for `sm = "OR"` will be odds ratios rather than log odds ratios, for example.

`...` Additional arguments (passed on to `prmatrix`).

`digits` Minimal number of significant digits, see `print.default`.

`digits.se` Minimal number of significant digits for standard errors.

`digits.tau2` Minimal number of significant digits for between-study variance τ^2, see `print.default`.

`digits.tau` Minimal number of significant digits for τ, the square root of the between-study variance τ^2.

`big.mark` A character used as thousands separator.

`se` A logical indicating whether standard errors should be printed / extracted.

`print.tau2` A logical specifying whether between-study variance τ^2 should be printed.

`print.tau` A logical specifying whether τ, the square root of the between-study variance τ^2, should be printed.

`details` A logical specifying whether details on statistical methods should be printed.

`writexl` A logical indicating whether an Excel file should be created (R package **writexl** must be available).

`path` A character string specifying the filename of the Excel file.

`overwrite` A logical indicating whether an existing Excel file should be overwritten.

Value

Data frame with variables

`studlab` Study label

`blup` estimated best linear unbiased predictor

`se.blup` standard error (only if argument `backtransf = FALSE`)

`lower` lower prediction limits

`upper` upper prediction limits

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

See Also

`metabin`, `dat.bcg`

Examples

```
m1 <- metabin(tpos, tpos + tneg, cpos, cpos + cneg,
  data = dat.bcg, studlab = paste(author, year), method = "Inverse")
summary(m1)
blup(m1)

## Not run:
# Save estimates in Excel file (R package 'writexl' must be available)
if (requireNamespace("writexl", quietly = TRUE))
 estimates(blup(m1), path = "blup_m1.xlsx")

## End(Not run)
```

`bubble.metareg` *Bubble plot to display the result of a meta-regression*

Description

Draw a bubble plot to display the result of a meta-regression.

Usage

```
## S3 method for class 'metareg'
bubble(
  x,
  xlim,
  ylim,
  xlab,
  ylab,
  cex,
  min.cex = 0.5,
  max.cex = 5,
  pch = 21,
  col = "black",
  bg = "darkgray",
  lty = 1,
  lwd = 1,
  col.line = "black",
  studlab = FALSE,
  cex.studlab = 0.8,
  pos.studlab = 2,
  offset = 0.5,
  regline = TRUE,
  backtransf = x$.meta$x$backtransf,
  ref,
  col.ref = "lightgray",
  lty.ref = 1,
  lwd.ref = 1,
  pscale = x$.meta$x$pscale,
  irscale = x$.meta$x$irscale,
  axes = TRUE,
  box = TRUE,
  ...
)

bubble(x, ...)
```

Arguments

x An object of class metareg.

xlim The x limits (min,max) of the plot.

ylim The y limits (min,max) of the plot.

xlab A label for the x-axis.

ylab A label for the y-axis.

cex The magnification to be used for plotting symbols.

min.cex Minimal magnification for plotting symbols.

max.cex Maximal magnification for plotting symbols.

pch The plotting symbol(s) used for individual studies.

col A vector with colour of plotting symbols.

bg A vector with background colour of plotting symbols (only used if pch in 21:25).

lty The line type for the meta-regression line.

lwd The line width for the meta-regression line.

col.line Colour for the meta-regression line.

studlab A logical indicating whether study labels should be printed in the graph. A vector with study labels can also be provided (must be of same length as the numer of studies in the meta-analysis then).

cex.studlab The magnification for study labels.

pos.studlab Position of study labels, see argument pos in text.

offset Offset for study labels (see text).

regline A logical indicating whether a regression line should be added to the bubble plot.

backtransf A logical indicating whether results for relative summary measures (argument sm equal to "OR", "RR", "HR", or "IRR") should be back transformed. If backtransf=TRUE, results for sm="OR" are printed as odds ratios rather than log odds ratios, for example.

ref A numerical giving the reference value to be plotted as a line in the bubble plot. No reference line is plotted if argument ref is equal to NA.

col.ref Colour of the reference line.

lty.ref The line type for the reference line.

lwd.ref The line width for the reference line.

pscale A numeric giving scaling factor for printing of probabilities.

irscale A numeric defining a scaling factor for printing of incidence rates.

axes Either a logical or a character string equal to "x", "y" or "xy" indicating whether x- and y-axis should be printed.

box A logical indicating whether a box should be printed.

... Graphical arguments as in par may also be passed as arguments.

Details

A bubble plot can be used to display the result of a meta-regression. It is a scatter plot with the treatment effect for each study on the y-axis and the covariate used in the meta-regression on the x-axis. Typically, the size of the plotting symbol is inversely proportional to the variance of the estimated treatment effect (Thompson & Higgins, 2002).

Argument cex specifies the plotting size for each individual study. If this argument is missing the weights from the meta-regression model will be used (which typically is a random effects model). Use cex="common" in order to utilise weights from a common effect model to define the size of the plotted symbols (even for a random effects meta-regression). If a vector with individual study weights is provided, the length of this vector must be of the same length as the number of studies.

Arguments min.cex and max.cex can be used to define the size of the smallest and largest plotting symbol. The plotting size of the most precise study is set to max.cex whereas the plotting size of all studies with a plotting size smaller than min.cex will be set to min.cex.

For a meta-regression with more than one covariate. Only a scatter plot of the first covariate in the regression model is shown. In this case the effect of the first covariate adjusted for other covariates in the meta-regression model is shown.

For a factor or categorial covariate separate bubble plots for each group compared to the baseline group are plotted.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Thompson SG, Higgins JP (2002): How should meta-regression analyses be undertaken and interpreted? *Statistics in Medicine*, **21**, 1559–73

See Also

metagen, metainf

Examples

```
data(Fleiss1993cont)

# Add some (fictitious) grouping variables:
Fleiss1993cont$age <- c(55, 65, 52, 65, 58)
Fleiss1993cont$region <- c("Europe", "Europe", "Asia", "Asia", "Europe")

m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD")

mr1 <- metareg(m1, region)
mr1

bubble(mr1)
bubble(mr1, lwd = 2, col.line = "blue")

mr2 <- metareg(m1, age)
mr2

bubble(mr2, lwd = 2, col.line = "blue", xlim = c(50, 70))
bubble(mr2, lwd = 2, col.line = "blue", xlim = c(50, 70), cex = "common")

# Do not print regression line
#
bubble(mr2, lwd = 2, col.line = "blue", xlim = c(50, 70), regline = FALSE)
```

`caffeine` *Caffeine for daytime drowsiness*

Description

Caffeine for daytime drowsiness (Cochrane Practice review)

Format

A data frame with the following columns:

study	study label
year	year of publication
h.caf	Number of participants with headaches (caffeine group)
n.caf	Number of participants (caffeine group)
h.decaf	Number of participants with headaches (decaf group)
n.decaf	Number of participants (decaf group)
D1	Domain 1 of risk of bias 2 tool (RoB 2)
D2	Domain 2 (RoB 2)
D3	Domain 3 (RoB 2)
D4	Domain 4 (RoB 2)

D5	Domain 5 (RoB 2)
rob	Overall RoB 2 assessment

Details

Data come from the Cochrane Practice review on caffeine for daytime drowsiness. Eight fictitous studies evaluate the risk of headaches after drinking either caffeinated or decaffeinated coffee.

References

Higgins JPT, Savović J, Page MJ, Sterne JA on behalf of the RoB2 Development Group (2019): Revised Cochrane risk-of-bias tool for randomized trials. https://www.riskofbias.info/welcome/rob-2-0-tool

See Also

metabin, rob

Examples

```
oldset <- settings.meta("RevMan5")

data(caffeine)
head(caffeine)

m1 <- metabin(h.caf, n.caf, h.decaf, n.decaf, sm = "OR",
  data = caffeine, studlab = paste(study, year))

# Add risk of bias assessment to meta-analysis
m1 <- rob(D1, D2, D3, D4, D5, overall = rob, data = m1, tool = "rob2")

# Print risk of bias assessment
rob(m1)

# Forest plot with risk of bias assessment
forest(m1)

settings.meta(oldset)
```

ci — *Calculation of confidence intervals (based on normal approximation or t-distribution)*

Description

Calculation of confidence intervals; based on normal approximation or t-distribution.

Usage

```
ci(TE, seTE, level = 0.95, df = NULL, null.effect = 0)
```

Arguments

`TE`	Estimated treatment effect.
`seTE`	Standard error of treatment estimate.
`level`	The confidence level required.
`df`	Degrees of freedom (for confidence intervals based on t-distribution).
`null.effect`	A numeric value specifying the effect under the null hypothesis.

Value

List with components

`TE`	Estimated treatment effect
`seTE`	Standard error of treatment estimate
`lower`	Lower confidence limits
`upper`	Upper confidence limits
`statistic`	Test statistic (either z-score or t-score)
`p`	P-value of text.with null hypothesis `TE=0`
`level`	The confidence level required
`df`	Degrees of freedom (t-distribution)

Note

This function is primarily called from other functions of the library `meta`, e.g. `forest.meta`, `summary.meta`.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

Examples

```
data.frame(ci(170, 10))
data.frame(ci(170, 10, 0.99))
data.frame(ci(1.959964, 1))
data.frame(ci(2.2621571628, 1, df = 9))
```

cidprop.meta — *Calculate expected proportion of comparable studies with clinically important benefit or harm*

Description

Calculate expected proportion of comparable studies with clinically important benefit or harm which is derived from the prediction interval.

Usage

```
## S3 method for class 'meta'
cidprop(
  x,
  cid = NULL,
  cid.below.null = NULL,
  cid.above.null = NULL,
  label.cid = "",
  label.cid.below.null = NULL,
  label.cid.above.null = NULL,
  small.values = "desirable",
  ...
)

cidprop(x, ...)

## S3 method for class 'cidprop'
print(
  x,
  digits.cid = gs("digits.cid"),
  digits.percent = 1,
  big.mark = gs("big.mark"),
  details.methods = gs("details"),
  ...
)
```

Arguments

`x` An object of class `meta`.

`cid` A numeric value or vector specifying clinically important differences (CID) / decision thresholds used to calculate expected proportions of clinically important benefit or harm (see Details).

`cid.below.null` A numeric value or vector specifying CID limits below the null effect (see Details).

`cid.above.null` A numeric value or vector specifying CID limits above the null effect (see Details).

`label.cid` A character string or vector specifying labels for clinically important differences. Must be of same length as argument `cid`.

`label.cid.below.null`
A character string or vector specifying labels for clinically important differences below the null effect. Must be of same length as argument `cid.below.null` (or `cid`).

`label.cid.above.null`
A character string or vector specifying labels for clinically important differences above the null effect. Must be of same length as argument `cid.above.null` (or `cid`).

`small.values` A character string specifying whether small treatment effects indicate a beneficial (`"desirable"`) or harmful (`"undesirable"`), can be abbreviated.

`...` Additional arguments (ignored)

`digits.cid` Minimal number of significant digits for CIDs / decision thresholds, see `print.default`.

`digits.percent` Minimal number of significant digits for expected proportions, printed as percentages, see `print.default`.

`big.mark` A character used as thousands separator.

`details.methods`
A logical specifying whether details on statistical methods should be printed.

Details

Expected proportions of comparable studies with clinically important benefit or harm are derived from the prediction interval in the meta-analysis.

Clinically important benefit or harm can be defined using either argument `cid` or `cid.below.null` and `cid.above.null`. Input for the later arguments will be ignored if argument `cid` was specified. In this case, the values of `cid.below.null` and `cid.above.null` will be equal to

- `cid` and `1 / cid` for ratio measures,
- `cid` and `-cid` for difference measures.

Thresholds based on argument `cid` will always be symmetric. Asymmetric thresholds can be defined using arguments `cid.below.null` and `cid.above.null`.

Value

A list with elements

`prop.cid.below.null`
Expected proportion of comparable studies below lower CID(s)

`prop.cid.above.null`
Expected proportion of comparable studies above upper CID(s)

`prop.within.cid`
Expected proportion of comparable studies between lower and upper CID(s)

`cid`, `cid.below.null`, `cid.above.null`, `small.values`, `x`
As defined above

`label.cid`, `label.cid.below.null`, `label.cid.above.null`
As defined above

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

plot.cidprop

Examples

```
oldset <- settings.meta(digits.cid = 0)

m <- metagen(1:10 - 3, 1:10, sm = "MD")
#
pp1 <- cidprop(m, cid = 2)
pp1
#
pp2 <- cidprop(m, cid.below = 0.5, cid.above = 2)
pp2
#
pp3 <- cidprop(m, cid.below = 0.5, cid.above = 2, small.values = "u")
pp3

pp4 <- cidprop(m, cid = 1:2, label.cid = c("moderate", "large"))
pp4
#
pp5 <- cidprop(m, cid.below = -1.5, cid.above = 1:2,
  label.cid.below = "large", label.cid.above = c("moderate", "large"))
pp5

settings.meta(oldset)
```

---

cisapride *Cisapride in Non-Ulcer Dispepsia*

Description

Meta-analysis on cisapride in non-ulcer dispepsia.

This meta-analysis is used as a data example in Hartung and Knapp (2001).

Format

A data frame with the following columns:

study	study label
event.cisa	number of events in cisapride group
n.cisa	number of observations in cisapride group
event.plac	number of events in placebo group

n.plac number of observations in placebo group

Source

Hartung J & Knapp G (2001): A refined method for the meta-analysis of controlled clinical trials with binary outcome. *Statistics in Medicine*, **20**, 3875–89

See Also

metabin

Examples

```
data(cisapride)

m.or <- metabin(event.cisa, n.cisa, event.plac, n.plac,
  data = cisapride, sm = "OR",
  method = "Inverse", method.tau = "DL",
  studlab = study, method.incr = "all")

m.or.hk <- update(m.or, method.random.ci = "HK")
m.rr <- update(m.or, sm = "RR")
m.rr.hk <- update(m.or, sm = "RR", method.random.ci = "HK")

vars.common <- c("TE.common", "lower.common", "upper.common")
vars.random <- c("TE.random", "lower.random", "upper.random")
#
res.common.or <- as.data.frame(m.or[vars.common])
names(res.common.or) <- vars.random
#
res.common.rr <- as.data.frame(m.rr[vars.common])
names(res.common.rr) <- vars.random

# Results for log risk ratio - see Table VII in Hartung and Knapp (2001)
#
res.rr <- rbind(res.common.rr,
  as.data.frame(m.rr[vars.random]),
  as.data.frame(m.rr.hk[vars.random]))
#
row.names(res.rr) <- c("CE", "RE", "RE (HaKn)")
names(res.rr) <- c("Log risk ratio", "CI lower", "CI upper")
#
res.rr


# Results for log odds ratio (Table VII in Hartung and Knapp 2001)
#
res.or <- rbind(res.common.or,
  as.data.frame(m.or[vars.random]),
  as.data.frame(m.or.hk[vars.random]))
#
row.names(res.or) <- c("CE", "RE", "RE (HaKn)")
```

```
names(res.or) <- c("Log odds ratio", "CI lower", "CI upper")
#
res.or
```

drapery *Drapery plot*

Description

Draw a drapery plot with (scaled) p-value curves for individual studies and meta-analysis estimates.

Usage

```
drapery(
  x,
  type = "zvalue",
  layout = "grayscale",
  study.results = TRUE,
  lty.study = 1,
  lwd.study = 1,
  col.study = "darkgray",
  labels,
  col.labels = "black",
  cex.labels = 0.7,
  subset.labels,
  srt.labels,
  common = x$common,
  random = x$random,
  lty.common = 1,
  lwd.common = max(3, lwd.study),
  col.common = "blue",
  lty.random = 1,
  lwd.random = lwd.common,
  col.random = "red",
  sign = NULL,
  lty.sign = 1,
  lwd.sign = 1,
  col.sign = "black",
  prediction = random,
  col.predict = "lightblue",
  alpha = if (type == "zvalue") c(0.001, 0.01, 0.05, 0.1) else c(0.01, 0.05, 0.1),
  lty.alpha = 2,
  lwd.alpha = 1,
  col.alpha = "black",
  cex.alpha = 0.7,
  col.null.effect = "black",
  legend = TRUE,
  pos.legend = "topleft",
  bg = "white",
  bty = "o",
  backtransf = x$backtransf,
  xlab,
  ylab,
  xlim,
  ylim,
  lwd.max = 2.5,
  lwd.study.weight = if (random) "random" else "common",
  at = NULL,
  n.grid = if (type == "zvalue") 10000 else 1000,
  mar = c(5.1, 4.1, 4.1, 4.1),
  plot = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  fixed,
  lwd.fixed,
  lty.fixed,
  col.fixed,
  ...
)
```

Arguments

`x` An object of class `meta`.

`type` A character string indicating whether to plot test statistics (`"zvalue"`) or p-values (`"pvalue"`), can be abbreviated.

`layout` A character string for the line layout of individual studies: `"grayscale"`, `"equal"`, or `"linewidth"` (see Details), can be abbreviated.

`study.results` A logical indicating whether results for individual studies should be shown in the figure.

`lty.study` Line type for individual studies.

`lwd.study` Line width for individual studies.

`col.study` Colour of lines for individual studies.

`labels` A logical or character string indicating whether study labels should be shown at the top of the drapery plot; either `FALSE`, `"id"`, or `"studlab"`; see Details.

`col.labels` Colour of study labels.

`cex.labels` The magnification for study labels.

`subset.labels` A vector specifying which study labels should be shown in the drapery plot.

`srt.labels` A numerical vector or single numeric (between 0 and 90) specifying the angle to rotate study labels; see Details.

`common` A logical indicating whether to show result for the common effect model.

`random` A logical indicating whether to show result for the random effects model.

`lty.common` Line type for common effect meta-analysis.

lwd.common Line width for common effect meta-analysis.

col.common Colour of lines for common effect meta-analysis.

lty.random Line type for random effects meta-analysis.

lwd.random Line width for random effects meta-analysis.

col.random Colour of lines for random effects meta-analysis.

sign Significance level used to highlight significant values in curves.

lty.sign Line type for significant values.

lwd.sign Line width for significant values.

col.sign Line colour for significant values.

prediction A logical indicating whether to show prediction region.

col.predict Colour of prediction region

alpha Horizonal lines are printed for the specified alpha values.

lty.alpha Line type of horizonal lines for alpha values.

lwd.alpha Line width of horizonal lines for alpha values.

col.alpha Colour of horizonal lines for alpha values.

cex.alpha The magnification for the text of the alpha

col.null.effect
Colour of vertical line indicating null effect.

legend A logical indicating whether a legend should be printed.

pos.legend A character string with position of legend (see legend).

bg Background colour of legend (see legend).

bty Type of the box around the legend; either "o" or "n" (see legend).

backtransf A logical indicating whether results should be back transformed on the x-axis. For example, if backtransf = FALSE, log odds ratios instead of odds ratios are shown on the x-axis.

xlab A label for the x-axis.

ylab A label for the y-axis.

xlim The x limits (min, max) of the plot.

ylim The y limits (min, max) of the plot (ignored if type = "pvalue").

lwd.max The maximum line width (only considered if argument layout is equal to "linewidth").

lwd.study.weight
A character string indicating whether to determine line width for individual studies using weights from common effect ("common") or random effects model ("random"), can be abbreviated (only considered if argument layout is equal to "linewidth").

at Points at which tick-marks are to be drawn on the x-axis.

n.grid The number of grid points to calculate the p-value or test statistic functions.

mar Physical plot margin, see par.

plot A logical indicating whether to generate a figure.

`warn.deprecated`
: A logical indicating whether warnings should be printed if deprecated arguments are used.

`fixed` Deprecated argument (replaced by 'common').

`lwd.fixed` Deprecated argument (replaced by 'lwd.common').

`lty.fixed` Deprecated argument (replaced by 'lty.common').

`col.fixed` Deprecated argument (replaced by 'col.common').

`...` Graphical arguments as in `par` may also be passed as arguments.

Details

The concept of a p-value function, also called confidence curve, goes back to Birnbaum (1961). A drapery plot, showing p-value functions (or a scaled version based on the corresponding test statistics) for individual studies as well as meta-analysis estimates, is drawn in the active graphics window. Furthermore, a prediction region for a single future study is shown as a shaded area. In contrast to a forest plot, a drapery plot does not provide information for a single confidence level however for any confidence level.

Argument `type` can be used to either show p-value functions (Birnbaum, 1961) or a scaled version (Infanger, 2019) with test statistics (default).

Argument `layout` determines how curves for individual studies are presented:

- darker gray tones with increasing precision (`layout = "grayscale"`)
- thicker lines with increasing precision (`layout = "linewidth"`)
- equal lines (`layout = "equal"`)

Argument `labels` determines how curves of individual studies are labelled:

- number of the study in the (unsorted) forest plot / printout of a meta-analysis (`labels = "id"`)
- study labels provided by argument `studlab` in meta-analysis functions (`labels = "studlab"`)
- no study labels (`labels = FALSE`)

By default, study labels are used (`labels = "studlab"`) if no label has more than three characters; otherwise IDs are used (`labels = "id"`). The connection between IDs and study labels (among other information) is part of a data frame which is invisibly returned (if argument `study.results = TRUE`).

Argument `srt.labels` can be used to change the rotation of IDs or study labels. By default, study labels are rotated by +/- 45 degrees if at least one study label has more than three characters; otherwise labels are not rotated.

If `labels = "studlab"`, labels are rotated by -45 degrees for studies with a treatment estimate below the common effect estimate and otherwise by 45 degrees.

Author(s)

Gerta Rücker <`gerta.ruecker@uniklinik-freiburg.de`>, Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.`

References

Birnbaum A (1961): Confidence Curves: An Omnibus Technique for Estimation and Testing Statistical Hypotheses. *Journal of the American Statistical Association*, **56**, 246–9

Infanger D and Schmidt-Trucksäss A (2019): P value functions: An underused method to present research results and to promote quantitative reasoning *Statistics in Medicine*, **38**, 4189–97

See Also

forest, radial

Examples

```
data("lungcancer")
m1 <- metainc(d.smokers, py.smokers, d.nonsmokers, py.nonsmokers,
  data = lungcancer, studlab = study)

# Drapery plot
#
drapery(m1, xlim = c(0.5, 50))

## Not run:
data(Fleiss1993bin)
m2 <- metabin(d.asp, n.asp, d.plac, n.plac,
  data = Fleiss1993bin, studlab = paste(study, year),
  sm = "OR", random = FALSE)

# Produce drapery plot and print data frame with connection between
# IDs and study labels
#
(drapery(m2))

# For studies with a significant effect (p < 0.05), show
# study labels and print labels and lines in red
#
drapery(m2,
  labels = "studlab", subset.labels = pval < 0.05,
  srt.labels = 0, col.labels = "red",
  col.study = ifelse(pval < 0.05, "red", "darkgray"))

## End(Not run)
```

---

estimates.meta *Extract results from meta-analysis object*

Description

Extract study and meta-analysis results from meta-analysis object which can be stored in an Excel file.

Usage

```
## S3 method for class 'meta'
estimates(
  x,
  sortvar,
  study.results = TRUE,
  common = x$common,
  random = x$random,
  prediction = x$prediction,
  overall = x$overall,
  subgroup,
  prediction.subgroup = x$prediction.subgroup,
  se = FALSE,
  ci = TRUE,
  statistic = FALSE,
  pval = FALSE,
  n = TRUE,
  backtransf = x$backtransf,
  digits = gs("digits"),
  digits.se = gs("digits.se"),
  digits.stat = gs("digits.stat"),
  digits.pval = gs("digits.pval"),
  writexl = !missing(path),
  path = "estimates.xlsx",
  overwrite = FALSE,
  ...
)

estimates(x, ...)

## S3 method for class 'estimates.meta'
print(
  x,
  digits.tau = gs("digits.tau"),
  text.tau2 = gs("text.tau2"),
  text.tau = gs("text.tau"),
  big.mark = gs("big.mark"),
  details = TRUE,
  ...
)
```

Arguments

`x` A meta-analysis object of class `meta`.

`sortvar` An optional vector used to sort the individual studies (must be of same length as `x$TE`).

`study.results` A logical indicating whether study results should be extracted.

`common` A logical indicating whether results of common effect meta-analysis should be extracted.

`random` A logical indicating whether results of random effects meta-analysis should be extracted.

`prediction` A logical indicating whether prediction interval should be extracted.

`overall` A logical indicating whether overall summaries should be extracted. This argument is useful in a meta-analysis with subgroups if overall results should not be extracted.

`subgroup` A logical indicating whether subgroup results should be extracted.

`prediction.subgroup` A single logical or logical vector indicating whether / which prediction intervals should be extracted for subgroups.

`se` A logical indicating whether standard errors should be extracted.

`ci` A logical indicating whether confidence / prediction interval should be extracted.

`statistic` A logical indicating whether to extract statistic of test for overall effect.

`pval` A logical indicating whether to extract p-value of test for overall effect.

`n` A logical indicating whether sample sizes should be extracted (if available).

`backtransf` A logical indicating whether extracted results should be back transformed.

`digits` Minimal number of significant digits, see `print.default`.

`digits.se` Minimal number of significant digits for standard errors, see `print.default`.

`digits.stat` Minimal number of significant digits for z- or t-statistic for test of overall effect, see `print.default`.

`digits.pval` Minimal number of significant digits for p-value of overall treatment effect, see `print.default`.

`writexl` A logical indicating whether an Excel file should be created (R package **writexl** must be available).

`path` A character string specifying the filename of the Excel file.

`overwrite` A logical indicating whether an existing Excel file should be overwritten.

`...` Additional arguments passed on to `prmatrix`.

`digits.tau` Minimal number of significant digits for square root of between-study variance, see `print.default`.

`text.tau2` Text printed to identify between-study variance τ^2.

`text.tau` Text printed to identify τ, the square root of the between-study variance τ^2.

`big.mark` A character used as thousands separator.

`details` A logical specifying whether details on statistical methods should be printed.

Details

Extract study and meta-analysis results from meta-analysis object. By default, a data frame with the results is generated. Alternatively, an Excel file is created if argument `writexl = TRUE`.

The following information is extracted.

Variable	Content
studlab	Study label / descriptor of meta-analysis result
estimate	(Back transformed) estimate for individual studies / meta-analysis
se	Standard error
lower	Lower (back transformed) confidence / prediction interval limit
upper	Upper (back transformed) confidence / prediction interval limit
k	Number of studies
df	Degrees of freedom for confidence / prediction intervals
statistic	Statistic for test of effect
pval	P-value for test of effect
n	Total sample size
n.e	Sample size in first (experimental) group
n.c	Sample size in second (control) group

Some variables are only extracted if the corresponding logical argument se, ci, statistic or n is TRUE. Furthermore, (group) sample sizes are only extracted if available in the meta-analysis object.

The variables estimate, lower and upper contain the back transformed effects and confidence interval limits, e.g., odds ratio (argument sm = "OR" in metabin or metagen) or correlation (sm = "ZCOR" in metacor or metagen), if argument backtransf is TRUE. Otherwise, these variables contain the transformed values, e.g., log odds ratio (sm = "OR") or Fisher's Z transformed correlations (sm = "ZCOR"). See meta-sm for available summary measures and meta-transf for the corresponding transformations and back transformations.

Value

A data frame with additional class 'extract.meta' or an Excel file.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

meta-transf, meta-sm, as.data.frame.meta

Examples

```
m1 <- metacor(c(0.85, 0.7, 0.95), c(20, 40, 10))
summary(m1)
estimates(m1)
estimates(m1, backtransf = FALSE)
estimates(update(m1, common = FALSE, random = FALSE))
estimates(update(m1, prediction = TRUE))
estimates(update(m1, prediction = TRUE,
  level.ma = 0.99, level.predict = 0.9))

## Not run:
# Create Excel file with extracted results
# (R package 'writexl' must be available)
if (requireNamespace("writexl", quietly = TRUE)) {
 fname1 <- tempfile(fileext = ".xlsx")
 estimates(m1, path = fname1)
 # An existing Excel file is not overwritten but a warning is printed
 estimates(m1, path = fname1)
 # Overwrite an existing Excel file
 estimates(m1, path = fname1, overwrite = TRUE)
 # Suppress message on file creation and overwrite existing file
 suppressMessages(estimates(m1, path = fname1, overwrite = TRUE))
}

# Save the extracted results in a text file
fname2 <- tempfile(fileext = ".csv")
fname2
write.csv(estimates(m1), file = fname2, row.names = FALSE)

## End(Not run)
```

`Fleiss1993bin` *Aspirin after Myocardial Infarction*

Description

Meta-analysis on aspirin in preventing death after myocardial infarction.

Data example in Fleiss (1993) for meta-analysis with binary outcomes.

Format

A data frame with the following columns:

study	study label
year	year of publication
d.asp	number of deaths in aspirin group
n.asp	number of observations in aspirin group
d.plac	number of deaths in placebo group
n.plac	number of observations in placebo group

Source

Fleiss JL (1993): The statistical basis of meta-analysis. *Statistical Methods in Medical Research*, **2**, 121–45

See Also

`metabin`

Examples

```
data(Fleiss1993bin)
metabin(d.asp, n.asp, d.plac, n.plac, data = Fleiss1993bin,
  studlab = paste(study, year), sm = "OR", random = FALSE)
```

`Fleiss1993cont` *Mental Health Treatment*

Description

Meta-analysis on the Effect of Mental Health Treatment on Medical Utilisation.

Data example in Fleiss (1993) for meta-analysis with continuous outcomes.

Format

A data frame with the following columns:

study	study label
year	year of publication
n.psyc	number of observations in psychotherapy group
mean.psyc	estimated mean in psychotherapy group
sd.psyc	standard deviation in psychotherapy group
n.cont	number of observations in control group
mean.cont	estimated mean in control group
sd.cont	standard deviation in control group

Source

Fleiss JL (1993): The statistical basis of meta-analysis. *Statistical Methods in Medical Research*, **2**, 121–45

See Also

Fleiss1993bin

metacont

Examples

```
data(Fleiss1993cont)
# Note, the following command uses the bias-corrected version of
# Hedges' g. Accordingly, results differ from Fleiss (1993), section 3,
# using the uncorrected version of Hedges' g.
metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, studlab = paste(study, year),
  random = FALSE, sm = "SMD")
```

forest.meta *Forest plot to display the result of a meta-analysis*

Description

Draw a forest plot (using grid graphics system) in the active graphics window or store the forest plot in a file.

Usage

```
## S3 method for class 'meta'
forest(
  x,
  sortvar,
  studlab = TRUE,
  layout = gs("layout"),
  common = x$common,
  random = x$random,
  overall = x$overall,
  text.common = x$text.common,
  text.random = x$text.random,
  lty.common = gs("lty.common"),
  lty.random = gs("lty.random"),
  col.common = gs("col.common"),
  col.random = gs("col.random"),
  text.w.common = x$text.w.common,
  text.w.random = x$text.w.random,
  prediction = x$prediction,
  text.predict = x$text.predict,
  subgroup = TRUE,
  subgroup.hetstat = subgroup & (is.character(hetstat) || hetstat),
  print.subgroup.labels = TRUE,
  subgroup.name = x$subgroup.name,
  print.subgroup.name = x$print.subgroup.name,
  sep.subgroup = x$sep.subgroup,
  text.common.w = text.common,
  text.random.w = text.random,
  text.predict.w = text.predict,
  sort.subgroup = gs("sort.subgroup"),
  pooled.totals = common | random,
  pooled.events = gs("pooled.events"),
  pooled.times = gs("pooled.times"),
  study.results = gs("study.results"),
  rob = x$rob,
  rob.text = "Risk of Bias",
  rob.xpos = 0,
  rob.legend = TRUE,
  rob.only = FALSE,
  xlab = "",
  xlab.pos,
  smlab = NULL,
  smlab.pos,
  xlim,
  allstudies = TRUE,
  weight.study = NULL,
  pscale = x$pscale,
  irscale = x$irscale,
  irunit = x$irunit,
  file = NULL,
  width = gs("width"),
  rows.gr = NULL,
  func.gr = NULL,
  args.gr = NULL,
  dev.off = NULL,
  ref,
  cid = gs("cid"),
  cid.below.null = gs("cid.below.null"),
  cid.above.null = gs("cid.above.null"),
  lty.cid = gs("lty.cid"),
  col.cid = gs("col.cid"),
  fill.cid = gs("fill.cid"),
  fill.cid.below.null = fill.cid,
  fill.cid.above.null = rev(fill.cid),
  fill = gs("fill"),
  fill.equi = gs("fill.equi"),
  fill.lower.equi = fill.equi,
  fill.upper.equi = rev(fill.equi),
  leftcols = gs("leftcols"),
  rightcols = gs("rightcols"),
  leftlabs = gs("leftlabs"),
  rightlabs = gs("rightlabs"),
  label.e = x$label.e,
  label.c = x$label.c,
  label.e.attach = gs("label.e.attach"),
  label.c.attach = gs("label.c.attach"),
  label.left = x$label.left,
  label.right = x$label.right,
  bottom.lr = gs("bottom.lr"),
  lab.NA = gs("lab.NA"),
  lab.NA.effect = gs("lab.NA.effect"),
  lab.NA.weight = gs("lab.NA.weight"),
  lwd = gs("lwd"),
  at = NULL,
  label = TRUE,
  col.label = gs("col.label"),
  type.study = gs("type.study"),
  type.common = gs("type.common"),
  type.random = type.common,
  type.subgroup = ifelse(study.results, "diamond", "square"),
  type.subgroup.common = type.subgroup,
  type.subgroup.random = type.subgroup,
  col.study = gs("col.study"),
  col.square = gs("col.square"),
  col.square.lines = gs("col.square.lines"),
  col.circle = gs("col.circle"),
  col.circle.lines = col.circle,
  col.inside = gs("col.inside"),
  col.inside.common = col.inside,
  col.inside.random = col.inside,
  col.diamond = gs("col.diamond"),
  col.diamond.common = col.diamond,
  col.diamond.random = col.diamond,
  col.diamond.lines = gs("col.diamond.lines"),
  col.diamond.lines.common = col.diamond.lines,
  col.diamond.lines.random = col.diamond.lines,
  col.predict = gs("col.predict"),
  col.predict.lines = gs("col.predict.lines"),
  col.subgroup = gs("col.subgroup"),
  col.label.left = x$col.label.left,
  col.label.right = x$col.label.right,
  hetstat = common | random | overall.hetstat,
  overall.hetstat = x$overall.hetstat & !inherits(x, "metamerge"),
  hetlab = gs("hetlab"),
  resid.hetstat = gs("resid.hetstat"),
  resid.hetlab = gs("resid.hetlab"),
  print.I2 = gs("forest.I2"),
  print.I2.ci = gs("forest.I2.ci"),
  print.tau2 = gs("forest.tau2"),
  print.tau2.ci = gs("forest.tau2.ci"),
  print.tau = gs("forest.tau"),
  print.tau.ci = gs("forest.tau.ci"),
  print.Q = gs("forest.Q"),
  print.pval.Q = gs("forest.pval.Q"),
  print.Rb = gs("forest.Rb"),
  print.Rb.ci = gs("forest.Rb.ci"),
  text.subgroup.nohet = gs("text.subgroup.nohet"),
  LRT = gs("LRT"),
  test.overall = gs("test.overall"),
  test.overall.common = common & overall & test.overall,
  test.overall.random = random & overall & test.overall,
  label.test.overall.common,
  label.test.overall.random,
  print.stat = gs("forest.stat"),
  test.subgroup = x$test.subgroup,
  test.subgroup.common = test.subgroup & common,
  test.subgroup.random = test.subgroup & random,
  common.subgroup = common,
  random.subgroup = random,
  prediction.subgroup = x$prediction.subgroup,
  print.Q.subgroup = gs("forest.Q.subgroup"),
  label.test.subgroup.common,
  label.test.subgroup.random,
  test.effect.subgroup = gs("test.effect.subgroup"),
  test.effect.subgroup.common,
  test.effect.subgroup.random,
  label.test.effect.subgroup.common,
  label.test.effect.subgroup.random,
  text.addline1,
  text.addline2,
  details = gs("forest.details"),
  col.lines = gs("col.lines"),
  header.line,
  col.header.line = col.lines,
  col.jama.line = col.subgroup,
  data.pooled = NULL,
  fontsize = gs("fontsize"),
  fontfamily = gs("fontfamily"),
  fs.heading = fontsize,
  fs.common = gs("fs.common"),
  fs.random = gs("fs.random"),
  fs.predict = gs("fs.predict"),
  fs.common.labels = gs("fs.common.labels"),
  fs.random.labels = gs("fs.random.labels"),
  fs.predict.labels = gs("fs.predict.labels"),
  fs.study = fontsize,
  fs.study.labels = fs.study,
  fs.hetstat = gs("fs.hetstat"),
  fs.test.overall = gs("fs.test.overall"),
  fs.test.subgroup = gs("fs.test.subgroup"),
  fs.test.effect.subgroup = gs("fs.test.effect.subgroup"),
  fs.addline = gs("fs.addline"),
  fs.axis = fontsize,
  fs.smlab = fontsize,
  fs.xlab = fontsize,
  fs.lr = fontsize,
  fs.rob = fontsize,
  fs.rob.symbols = fontsize,
  fs.details = fontsize,
  ff.heading = "bold",
  ff.common = gs("ff.common"),
  ff.random = gs("ff.random"),
  ff.predict = gs("ff.predict"),
  ff.common.labels = gs("ff.common.labels"),
  ff.random.labels = gs("ff.random.labels"),
  ff.predict.labels = gs("ff.predict.labels"),
  ff.study = "plain",
  ff.study.labels = ff.study,
  ff.hetstat = gs("ff.hetstat"),
  ff.test.overall = gs("ff.test.overall"),
  ff.test.subgroup = gs("ff.test.subgroup"),
  ff.test.effect.subgroup = gs("ff.test.effect.subgroup"),
  ff.addline = gs("ff.addline"),
  ff.axis = gs("ff.axis"),
  ff.smlab = gs("ff.smlab"),
  ff.xlab = gs("ff.xlab"),
  ff.lr = gs("ff.lr"),
  ff.rob = "plain",
  ff.rob.symbols = "bold",
  ff.details = "plain",
  squaresize = if (layout == "BMJ") 0.9/spacing else 0.8/spacing,
  lwd.square = gs("lwd.square"),
  lwd.diamond = gs("lwd.diamond"),
  arrow.type = gs("arrow.type"),
  arrow.length = gs("arrow.length"),
  plotwidth = if (layout %in% c("BMJ", "JAMA")) "8cm" else "6cm",
  colgap = gs("colgap"),
  colgap.left = colgap,
  colgap.right = colgap,
  colgap.studlab = colgap.left,
  colgap.forest = gs("colgap.forest"),
  colgap.forest.left = colgap.forest,
  colgap.forest.right = colgap.forest,
  colgap.rob = "1mm",
  colgap.rob.overall = "2mm",
  calcwidth.pooled = (common | random) & (overall | !is.null(x$subgroup)),
  calcwidth.common = calcwidth.pooled,
  calcwidth.random = calcwidth.pooled,
  calcwidth.predict = gs("calcwidth.predict"),
  calcwidth.hetstat = gs("calcwidth.hetstat"),
  calcwidth.tests = gs("calcwidth.tests"),
  calcwidth.subgroup = gs("calcwidth.subgroup"),
  calcwidth.addline = gs("calcwidth.addline"),
  just = if (layout == "JAMA") "left" else "right",
  just.studlab = gs("just.studlab"),
  just.addcols = gs("just.addcols"),
  just.addcols.left = just.addcols,
  just.addcols.right = just.addcols,
  bmj.text = NULL,
  bmj.xpos = 0,
  bmj.sep = " / ",
  spacing = gs("spacing"),
  addrow = gs("addrow"),
  addrow.overall = gs("addrow.overall"),
  addrow.subgroups = gs("addrow.subgroups"),
  addrows.below.overall = gs("addrows.below.overall"),
  new = TRUE,
  backtransf = x$backtransf,
  digits = gs("digits.forest"),
  digits.se = gs("digits.se"),
  digits.stat = gs("digits.stat"),
  digits.pval = gs("digits.pval"),
  digits.pval.Q = gs("digits.pval.Q"),
  digits.Q = gs("digits.Q"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.I2 = gs("digits.I2"),
  digits.weight = gs("digits.weight"),
  digits.mean = gs("digits.mean"),
  digits.sd = gs("digits.sd"),
  digits.cor = digits,
  digits.time = digits,
  digits.n = 0,
  digits.event = 0,
  digits.TE = gs("digits.TE.forest"),
  digits.addcols = digits,
  digits.addcols.right = digits.addcols,
  digits.addcols.left = digits.addcols,
  scientific.pval = gs("scientific.pval"),
  big.mark = gs("big.mark"),
  zero.pval = if (layout == "JAMA") FALSE else gs("zero.pval"),
  JAMA.pval = if (layout == "JAMA") TRUE else gs("JAMA.pval"),
  warn.deprecated = gs("warn.deprecated"),
  ...
)

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

.forestArgs()
```

Arguments

`x` An object of class `meta`.

`sortvar` An optional vector used to sort the individual studies (must be of same length as `x$TE`).

`studlab` A logical indicating whether study labels should be printed in the graph. A vector with study labels can also be provided (must be of same length as the vector with estimates `x$TE`).

`layout` A character string specifying the layout of the forest plot (see Details).

`common` A logical indicating whether common effect estimate should be plotted.

`random` A logical indicating whether random effects estimate should be plotted.

`overall` A logical indicating whether overall summaries should be plotted. This argument is useful in a meta-analysis with subgroups if summaries should only be plotted on group level.

`text.common` A character string used in the plot to label the pooled common effect estimate.

`text.random` A character string used in the plot to label the pooled random effects estimate.

`lty.common` Line type of pooled common effect estimate.

`lty.random` Line type of pooled random effects estimate.

`col.common` Line colour of pooled common effect estimate.

`col.random` Line colour of pooled random effects estimate.

`text.w.common` A character string used to label weights of common effect model.

`text.w.random` A character string used to label weights of random effects model.

`prediction` A logical indicating whether a prediction interval should be printed.

`text.predict` A character string used in the plot to label the prediction interval.

`subgroup` A single logical or logical vector indicating whether / which subgroup results should be shown in forest plot. This argument is useful in a meta-analysis with subgroups if summaries should not be plotted for (some) subgroups.

`subgroup.hetstat`
A single logical or logical vector indicating whether / which information on heterogeneity in subgroups should be shown in forest plot. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should not be printed for (some) subgroups.

`print.subgroup.labels`
A logical indicating whether subgroup label should be printed.

`subgroup.name` A character string with a label for the grouping variable.

`print.subgroup.name`
A logical indicating whether the name of the grouping variable should be printed in front of the group labels.

`sep.subgroup` A character string defining the separator between label and levels of grouping variable.

`text.common.w` A character string to label the pooled common effect estimate within subgroups, or a character vector of same length as number of subgroups with corresponging labels.

`text.random.w` A character string to label the pooled random effect estimate within subgroups, or a character vector of same length as number of subgroups with corresponging labels.

`text.predict.w` A character string to label the prediction interval within subgroups, or a character vector of same length as number of subgroups with corresponging labels.

`sort.subgroup` A logical indicating whether groups should be ordered alphabetically.

`pooled.totals` A logical indicating whether total number of observations should be given in the figure.

`pooled.events` A logical indicating whether total number of events should be given in the figure.

`pooled.times` A logical indicating whether total person time at risk should be given in the figure.

`study.results` A logical indicating whether results for individual studies should be shown in the figure (useful to only plot subgroup results).

`rob` Risk of bias (RoB) assessment.

`rob.text` Column heading for RoB table.

`rob.xpos` A numeric specifying the horizontal position of the risk of bias label in RoB table heading. The value is a so called normalised parent coordinate in the horizontal direction (see `unit`).

`rob.legend` A logical specifying whether a legend with RoB domains should be printed.

`rob.only` A logical indicating whether the risk of bias assessment is the only information printed on the right side of the forest plot.

`xlab` A label for the x-axis.

`xlab.pos` A numeric specifying the center of the label on the x-axis.

`smlab` A label for the summary measure (printed at top of figure).

`smlab.pos` A numeric specifying the center of the label for the summary measure.

`xlim` The x limits (min,max) of the plot, or the character string "symmetric" to produce symmetric forest plots.

`allstudies` A logical indicating whether studies with inestimable treatment effects should be included in the forest plot.

`weight.study` A character string indicating weighting used to determine size of squares or diamonds (argument `type.study`) to plot individual study results. One of missing, `"same"`, `"common"`, or `"random"`, can be abbreviated. Plot symbols have the same size for all studies or represent study weights from common effect or random effects model.

`pscale` A numeric giving scaling factor for printing of single event probabilities or risk differences, i.e. if argument `sm` is equal to `"PLOGIT"`, `"PLN"`, `"PRAW"`, `"PAS"`, `"PFT"`, or `"RD"`.

`irscale` A numeric defining a scaling factor for printing of single incidence rates or incidence rate differences, i.e. if argument `sm` is equal to `"IR"`, `"IRLN"`, `"IRS"`, `"IRFT"`, or `"IRD"`.

`irunit` A character specifying the time unit used to calculate rates, e.g., person-years.

`file` File name.

`width` Width of graphics file.

`rows.gr` Additional rows in forest plot to change height of graphics file (e.g., in order to add a title at the top of the forest plot).

`func.gr` Name of graphics function, e.g., `pdf`.

`args.gr` List with additional graphical parameters passed on to graphics function (argument 'height' cannot be provided as the height is calculated internally; use instead argument 'rows.gr').

`dev.off` A logical to specify whether current graphics device should be shut down, i.e., whether file should be stored.

`ref` A numerical giving the reference value to be plotted as a line in the forest plot. No reference line is plotted if argument `ref` is equal to `NA`.

`cid` A numeric value or vector specifying clinically important differences (CID) / decision thresholds used to calculate probabilities of clinically important benefit or harm, or not important effects (see Details).

`cid.below.null` A numeric value or vector specifying CID limits below the null effect (see Details).

`cid.above.null` A numeric value or vector specifying CID limits above the null effect (see Details).

`lty.cid` Line type for CID lines.

`col.cid` Line colour for CID lines.

`fill.cid` Colour(s) for regions below or above CID limits.

`fill.cid.below.null`
Colour of CID regions below null effect / reference value. Can be equal to the number of lower limits or the number of limits plus 1 (in this case the region between minimum and smallest limit is also filled).

`fill.cid.above.null`
Colour of CID regions above null effect / reference value. Can be equal to the number of upper limits or the number of limits plus 1 (in this case the region between largest limit and maximum is also filled).

`fill` Colour for background of confidence interval plot (also used as colour for region between CID limits if argument `fill.equi` was not provided).

`fill.equi` Colour(s) for region between limits of equivalence defined by arguments `cid`, `cid.lower` or `cid.upper`.

`fill.lower.equi`
Colour of region between lower limit(s) and reference value. Can be equal to the number of lower limits or the number of limits plus 1 (in this case the the region between minimum and smallest limit is also filled).

`fill.upper.equi`
Colour of region between reference value and upper limit(s). Can be equal to the number of upper limits or the number of limits plus 1 (in this case the region between largest limit and maximum is also filled).

`leftcols` A character vector specifying (additional) columns to be printed on the left side of the forest plot or a logical value (see Details).

`rightcols` A character vector specifying (additional) columns to be printed on the right side of the forest plot or a logical value (see Details).

`leftlabs` A character vector specifying labels for (additional) columns on left side of the forest plot (see Details).

rightlabs A character vector specifying labels for (additional) columns on right side of the forest plot (see Details).

label.e Label to be used for experimental group in table heading.

label.c Label to be used for control group in table heading.

label.e.attach A character specifying the column name where label label.e should be attached to in table heading.

label.c.attach A character specifying the column name where label label.c should be attached to in table heading.

label.left Graph label on left side of null effect.

label.right Graph label on right side of null effect.

bottom.lr A logical indicating whether labels on right and left side should be printed at bottom or top of forest plot.

lab.NA A character string to label missing values.

lab.NA.effect A character string to label missing values in individual treatment estimates and confidence intervals.

lab.NA.weight A character string to label missing weights.

lwd The line width, see par.

at The points at which tick-marks are to be drawn, see grid.xaxis.

label A logical value indicating whether to draw the labels on the tick marks, or an expression or character vector which specify the labels to use. See grid.xaxis.

col.label The colour of labels on the x-axis.

type.study A character string or vector specifying how to plot treatment effects and confidence intervals for individual studies (see Details).

type.common A character string specifying how to plot treatment effect and confidence interval for common effect meta-analysis (see Details).

type.random A character string specifying how to plot treatment effect and confidence interval for random effects meta-analysis (see Details).

type.subgroup A character string specifying how to plot treatment effect and confidence interval for subgroup results (see Details).

type.subgroup.common

A character string specifying how to plot treatment effect and confidence interval for subgroup results (common effect model).

type.subgroup.random

A character string specifying how to plot treatment effect and confidence interval for subgroup results (random effects model).

col.study The colour for individual study results and confidence limits.

col.square The colour for squares reflecting study's weight in the meta-analysis.

col.square.lines

The colour for the outer lines of squares reflecting study's weight in the meta-analysis.

col.circle The colour for circles reflecting study weights in the meta-analysis.

`col.circle.lines`
The colour for the outer lines of circles reflecting study's weight in the meta-analysis.

`col.inside` The colour for individual study results and confidence limits if confidence limits are completely within squares.

`col.inside.common`
The colour for result of common effect meta-analysis if confidence limit lies completely within square.

`col.inside.random`
The colour for result of random effects meta-analysis if confidence limit lies completely within square.

`col.diamond` The colour of diamonds representing the results for common effect and random effects models.

`col.diamond.common`
The colour(s) of diamonds for common effect estimates.

`col.diamond.random`
The colour(s) of diamonds for random effects estimates.

`col.diamond.lines`
The colour of the outer lines of diamonds representing the results for common effect and random effects models.

`col.diamond.lines.common`
The colour(s) of the outer lines of diamond for common effect estimate.

`col.diamond.lines.random`
The colour(s) of the outer lines of diamond for random effects estimate.

`col.predict` Background colour(s) of prediction intervals.

`col.predict.lines`
Colour(s) of outer lines of prediction intervals.

`col.subgroup` The colour to print information on subgroups.

`col.label.left` The colour of the label on the left side of the null effect.

`col.label.right`
The colour of the label on the right side of the null effect.

`hetstat` Either a logical value indicating whether to print results for heterogeneity measures at all or a character string (see Details).

`overall.hetstat`
A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`hetlab` Label printed in front of results for heterogeneity measures.

`resid.hetstat` A logical value indicating whether to print measures of residual heterogeneity in a meta-analysis with subgroups.

`resid.hetlab` Label printed in front of results for residual heterogeneity measures.

`print.I2` A logical value indicating whether to print the value of the I-squared statistic.

`print.I2.ci` A logical value indicating whether to print the confidence interval of the I-squared statistic.

`print.tau2` A logical value indicating whether to print the value of the between-study variance τ^2.

`print.tau2.ci` A logical value indicating whether to print the confidence interval of τ^2.

`print.tau` A logical value indicating whether to print τ, the square root of the between-study variance τ^2.

`print.tau.ci` A logical value indicating whether to print the confidence interval of τ.

`print.Q` A logical value indicating whether to print the value of the heterogeneity statistic Q.

`print.pval.Q` A logical value indicating whether to print the p-value of the heterogeneity statistic Q.

`print.Rb` A logical value indicating whether to print the value of the I-squared statistic.

`print.Rb.ci` A logical value indicating whether to print the confidence interval of the I-squared statistic.

`text.subgroup.nohet`
A logical value or character string which is printed to indicate subgroups with less than two studies contributing to meta-analysis (and thus without heterogeneity). If FALSE, heterogeneity statistics are printed (with NAs).

`LRT` A logical value indicating whether to report Likelihood-Ratio or Wald-type test of heterogeneity for generalised linear mixed models.

`test.overall` A logical value indicating whether to print results of test for overall effect.

`test.overall.common`
A logical value indicating whether to print results of test for overall effect (common effect model).

`test.overall.random`
A logical value indicating whether to print results of test for overall effect (random effects model).

`label.test.overall.common`
Label printed in front of results of test for overall effect (common effect model).

`label.test.overall.random`
Label printed in front of results of test for overall effect (random effects model).

`print.stat` A logical value indicating whether z- or t-value for test of treatment effect should be printed.

`test.subgroup` A logical value indicating whether to print results of test for subgroup differences.

`test.subgroup.common`
A logical value indicating whether to print results of test for subgroup differences (common effect model).

`test.subgroup.random`
A logical value indicating whether to print results of test for subgroup differences (random effects model).

`common.subgroup`
A single logical or logical vector indicating whether / which common effect estimates should be printed for subgroups.

`random.subgroup`
A single logical or logical vector indicating whether / which random effects estimates should be printed for subgroups.

`prediction.subgroup`
A single logical or logical vector indicating whether / which prediction intervals should be printed for subgroups.

`print.Q.subgroup`
A logical value indicating whether to print the value of the heterogeneity statistic Q (test for subgroup differences).

`label.test.subgroup.common`
Label printed in front of results of test for subgroup differences (common effect model).

`label.test.subgroup.random`
Label printed in front of results of test for subgroup differences (random effects model).

`test.effect.subgroup`
A single logical or logical vector indicating whether / which tests for effect in subgroups should be printed.

`test.effect.subgroup.common`
A single logical or logical vector indicating whether / which tests for effect in subgroups should be printed (common effect model).

`test.effect.subgroup.random`
A single logical or logical vector indicating whether / which tests for effect in subgroups should be printed (random effects model).

`label.test.effect.subgroup.common`
Label printed in front of results of test for effect in subgroups (common effect model).

`label.test.effect.subgroup.random`
Label printed in front of results of test for effect in subgroups (random effects model).

`text.addline1` Text for first additional line (below meta-analysis results).

`text.addline2` Text for second additional line (below meta-analysis results).

`details` A logical specifying whether details on statistical methods should be printed.

`col.lines` The colour of lines.

`header.line` A logical value indicating whether to print a header line or a character string ("both", "below", "").

`col.header.line`
Colour of the header line(s).

`col.jama.line` Colour of the additional JAMA lines.

`data.pooled` Data set with information for line(s) with pooled results (see Details).

`fontsize` The size of text (in points), see `gpar`.

`fontfamily` The font family, see `gpar`.

`fs.heading` The size of text for column headings, see `gpar`.

`fs.common` The size of text for results of common effect model, see `gpar`.

`fs.random` The size of text for results of random effects model, see `gpar`.

`fs.predict` The size of text for results of prediction interval, see `gpar`.

`fs.common.labels`
The size of text for label of common effect model, see `gpar`.

`fs.random.labels`
The size of text for label of random effects model, see `gpar`.

`fs.predict.labels`
The size of text for label of prediction interval, see `gpar`.

`fs.study` The size of text for results of individual studies, see `gpar`.

`fs.study.labels`
The size of text for labels of individual studies, see `gpar`.

`fs.hetstat` The size of text for heterogeneity measures, see `gpar`.

`fs.test.overall`
The size of text of test for overall effect, see `gpar`.

`fs.test.subgroup`
The size of text of test of subgroup differences, see `gpar`.

`fs.test.effect.subgroup`
The size of text of test of effect in subgroups, see `gpar`.

`fs.addline` The size of text for additional lines, see `gpar`.

`fs.axis` The size of text on x-axis, see `gpar`.

`fs.smlab` The size of text of label for summary measure, see `gpar`.

`fs.xlab` The size of text of label on x-axis, see `gpar`.

`fs.lr` The size of text of label on left and right side of forest plot, see `gpar`.

`fs.rob` The size of text of risk of bias items in the legend, see `gpar`.

`fs.rob.symbols` The size of risk of bias symbols, see `gpar`.

`fs.details` The size of text for details on (meta-analysis) methods, see `gpar`.

`ff.heading` The fontface for column headings, see `gpar`.

`ff.common` The fontface of text for results of common effect model, see `gpar`.

`ff.random` The fontface of text for results of random effects model, see `gpar`.

`ff.predict` The fontface of text for results of prediction interval, see `gpar`.

`ff.common.labels`
The fontface of text for label of common effect model, see `gpar`.

`ff.random.labels`
The fontface of text for label of random effects model, see `gpar`.

`ff.predict.labels`
The fontface of text for label of prediction interval, see `gpar`.

`ff.study` The fontface of text for results of individual studies, see `gpar`.

`ff.study.labels`
The fontface of text for labels of individual studies, see `gpar`.

`ff.hetstat` The fontface of text for heterogeneity measures, see `gpar`.

`ff.test.overall`
The fontface of text of test for overall effect, see `gpar`.

`ff.test.subgroup`
The fontface of text for test of subgroup differences, see `gpar`.

`ff.test.effect.subgroup`
The fontface of text for test of effect in subgroups, see `gpar`.

`ff.addline` The fontface of text for additional lines, see `gpar`.

`ff.axis` The fontface of text on x-axis, see `gpar`.

`ff.smlab` The fontface of text of label for summary measure, see `gpar`.

`ff.xlab` The fontface of text of label on x-axis, see `gpar`.

`ff.lr` The fontface of text of label on left and right side of forest plot, see `gpar`.

`ff.rob` The fontface of text of risk of bias items, see `gpar`.

`ff.rob.symbols` The fontface of risk of bias symbols, see `gpar`.

`ff.details` The fontface for details on (meta-analysis) methods, see `gpar`.

`squaresize` A numeric used to increase or decrease the size of squares in the forest plot.

`lwd.square` The line width of the border around squares.

`lwd.diamond` The line width of the border around diamonds.

`arrow.type` A character string indicating whether arrows printed for results outside the forest plot should be `"open"`, or `"closed"`, can be abbreviated.

`arrow.length` The length of arrows in inches.

`plotwidth` Either a character string, e.g., "8cm", "60mm", or "3inch", or a `unit` object specifying width of the forest plot.

`colgap` Either a character string or a `unit` object specifying gap between columns printed on left and right side of forest plot.

`colgap.left` Either a character string or a `unit` object specifying gap between columns printed on left side of forest plot.

`colgap.right` Either a character string or a `unit` object specifying gap between columns printed on right side of forest plot.

`colgap.studlab` Either a character string or a `unit` object specifying gap between column with study labels and subsequent column.

`colgap.forest` Either a character string or a `unit` object specifying gap between column adjacent to forest plot and the forest plot.

`colgap.forest.left`
Either a character string or a `unit` object specifying gap between column on the left side of forest plot and the forest plot.

`colgap.forest.right`
Either a character string or a `unit` object specifying gap between column on the right side of forest plot and the forest plot.

`colgap.rob` Either a character string or a `unit` object specifying gap between risk of bias columns.

`colgap.rob.overall`
Either a character string or a `unit` object specifying gap before column with overall risk of bias assessment.

`calcwidth.pooled`
A logical indicating whether text for common effect and random effects model should be considered to calculate width of the column with study labels.

`calcwidth.common`
A logical indicating whether text given in arguments `text.common` and `text.common.w` should be considered to calculate width of the column with study labels.

`calcwidth.random`
A logical indicating whether text given in arguments `text.random` and `text.random.w` should be considered to calculate width of the column with study labels.

`calcwidth.predict`
A logical indicating whether text given in argument `text.predict` and `text.predict.w` should be considered to calculate width of the column with study labels.

`calcwidth.hetstat`
A logical indicating whether text for heterogeneity statistics should be considered to calculate width of the column with study labels.

`calcwidth.tests`
A logical indicating whether text for tests of overall effect or subgroup differences should be considered to calculate width of the column with study labels.

`calcwidth.subgroup`
A logical indicating whether text with subgroup labels should be considered to calculate width of the column with study labels.

`calcwidth.addline`
A logical indicating whether text for additional lines should be considered to calculate width of the column with study labels.

`just`
Justification of text in all columns but columns with study labels and additional variables (possible values: "left", "right", "center").

`just.studlab`
Justification of text for study labels (possible values: "left", "right", "center").

`just.addcols`
Justification of text for additional columns (possible values: "left", "right", "center").

`just.addcols.left`
Justification of text for additional columns on left side of forest plot (possible values: "left", "right", "center"). Can be of same length as number of additional columns on left side of forest plot.

`just.addcols.right`
Justification of text for additional columns on right side of forest plot (possible values: "left", "right", "center"). Can be of same length as number of additional columns on right side of forest plot.

`bmj.text`
A character string used in the plot with BMJ layout to label the group specific information.

`bmj.xpos`
A numeric specifying the horizontal position of the BMJ label. The value is a so called normalised parent coordinate in the horizontal direction (see `unit`).

`bmj.sep`
A character string used to separate sample sizes from number of events or means / standard deviations.

`spacing` A numeric determining line spacing in a forest plot.

`addrow` A logical value indicating whether an empty row is printed above study results.

`addrow.overall` A logical value indicating whether an empty row is printed above overall meta-analysis results.

`addrow.subgroups`
A logical value indicating whether an empty row is printed between results for subgroups.

`addrows.below.overall`
A numeric value indicating how many empty rows are printed between meta-analysis results and heterogeneity statistics and test results.

`new` A logical value indicating whether a new figure should be printed in an existing graphics window.

`backtransf` A logical indicating whether results should be back transformed in forest plots. If `backtransf = TRUE`, results for `sm = "OR"` are presented as odds ratios rather than log odds ratios and results for `sm = "ZCOR"` are presented as correlations rather than Fisher's z transformed correlations, for example.

`digits` Minimal number of significant digits for treatment effects, see `print.default`.

`digits.se` Minimal number of significant digits for standard errors.

`digits.stat` Minimal number of significant digits for z- or t-statistic for test of overall effect.

`digits.pval` Minimal number of significant digits for p-value of overall treatment effect.

`digits.pval.Q` Minimal number of significant digits for p-value of heterogeneity test.

`digits.Q` Minimal number of significant digits for heterogeneity statistic Q.

`digits.tau2` Minimal number of significant digits for between-study variance.

`digits.tau` Minimal number of significant digits for square root of between-study variance.

`digits.I2` Minimal number of significant digits for I-squared statistic.

`digits.weight` Minimal number of significant digits for weights.

`digits.mean` Minimal number of significant digits for means; only applies to `metacont` objects.

`digits.sd` Minimal number of significant digits for standard deviations; only applies to `metacont` objects.

`digits.cor` Minimal number of significant digits for correlations; only applies to `metacor` objects.

`digits.time` Minimal number of significant digits for times; only applies to `metainc` and `metarate` objects.

`digits.n` Minimal number of significant digits for sample sizes.

`digits.event` Minimal number of significant digits for event numbers.

`digits.TE` Minimal number of significant digits for list element 'TE'.

`digits.addcols` A vector or scalar with minimal number of significant digits for additional columns.

`digits.addcols.right`
A vector or scalar with minimal number of significant digits for additional columns on right side of forest plot.

`digits.addcols.left`
: A vector or scalar with minimal number of significant digits for additional columns on left side of forest plot.

`scientific.pval`
: A logical specifying whether p-values should be printed in scientific notation, e.g., 1.2345e-01 instead of 0.12345.

`big.mark`
: A character used as thousands separator.

`zero.pval`
: A logical specifying whether p-values should be printed with a leading zero.

`JAMA.pval`
: A logical specifying whether p-values for test of overall effect should be printed according to JAMA reporting standards.

`warn.deprecated`
: A logical indicating whether warnings should be printed if deprecated arguments are used.

`...`
: Additional graphical arguments.

Details

A forest plot, also called confidence interval plot, is drawn in the active graphics window. The forest functions in R package **meta** are based on the grid graphics system. Resize the graphics windows if the forest plot is too large or too small for the graphics window. Alternatively, save the forest plot in a file.

Saving forest plots:

A forest plot can be directly stored in a file using argument `file` or specifying the R function for the graphics device driver using argument `func.gr`, e.g., `pdf`. If only the filename is provided, the extension is checked and matched against the most common graphics device drivers.

Extension	**Graphics device**
`.pdf`	`pdf`
`.ps`	`postscript`
`.svg`	`svg`
`.bmp`	`bmp`
`.jpg` / `.jpeg`	`jpeg`
`.png`	`png`
`.tif` / `.tiff`	`tiff`

The height of the graphics device is automatically determined if the forest plot is saved to a file. Argument `rows.gr` can be used to increase or decrease the number of rows shown in the forest plot (either to show missing information or to remove whitespace). The width of the graphics device can be specified with argument `width`, see, for example, `pdf` or `jpeg`. Other arguments of graphics device functions can be provided as a list in argument `args.gr`.

Alternatively, the (resized) graphics window can be stored to a file using either `dev.copy2eps` or `dev.copy2pdf`. It is also possible to manually create a file using, for example, `pdf`, `png`, or `svg` and to specify the width and height of the graphic (see Examples).

Default layout for studies and pooled effects:

By default, treatment estimates and confidence intervals are plotted in the following way:

- For an individual study, a square with treatment estimate in the center and confidence interval as line extending either side of the square (`type.study = "square"`)
- For meta-analysis results, a diamond with treatment estimate in the center and right and left side corresponding to lower and upper confidence limits (`type.common = "diamond"`, `type.random = "diamond"`, and `type.subgroup = "diamond"`)

In a forest plot, size of the squares typically reflects the precision of individual treatment estimates based either on the common effect (`weight.study = "common"`) or random effects meta-analysis (`weight.study = "random"`). Information from meta-analysis object `x` is utilised if argument `weight.study` is missing. Weights from the common effect model are used if argument `x$common` is `TRUE`; weights from the random effects model are used if argument `x$random` is `TRUE` and `x$common` is `FALSE`. The same square sizes are used if `weight.study = "same"`.

A prediction interval for treatment effect of a new study (Higgins et al., 2009) is given in the forest plot if arguments `prediction` and `random` are `TRUE`. For graphical presentation of prediction intervals the approach by Guddat et al. (2012) is used.

Columns printed on left side of forest plot:

Argument `leftcols` can be used to specify columns which are printed on the left side of the forest plot. By default, i.e. if argument `leftcols` is `NULL` and `layout = "meta"`, and depending on the class of the meta-analysis object (which is defined by the R function used to generate the object) a different set of ***columns*** is printed ***on the left side of the forest plot***:

Function	Value of argument leftcols
`metabin`	`c("studlab", "event.e", "n.e", "event.c", "n.c")`
`metacont`	`c("studlab", "n.e", "mean.e", "sd.e", "n.c", "mean.c", "sd.c")`
`metacor`	`c("studlab", "n")`
`metagen`	`c("studlab", "TE", "seTE")`
`metainc`	`c("studlab", "event.e", "time.e", "event.c", "time.c")`
`metamean`	`c("studlab", "n", "mean", "sd")`
`metaprop`	`c("studlab", "event", "n")`
`metarate`	`c("studlab", "event", "time", "n")`

For three-level models, the cluster variable is printed next to the study labels (value `"cluster"` in argument `leftcols`).

By default, study labels and labels for pooled estimates and heterogeneity statistics will be printed in the first column on the left side of the forest plot. The character string `"studlab"` is used to identify study labels as this is the name of the list element of a meta-analysis object.

If the character string `"studlab"` is not provided in `leftcols` and `rightcols`, the first *additional* variable specified by the user is used as study labels (and labels for pooled estimates are printed in this column). Additional variables are any variables not mentioned in the section on predefined column names below. For example, `leftcols = "studlab"` and `leftcols = "study"` would result in the same forest plot if the variable `"study"` was used in the command to conduct the meta-analysis. If no additional variable is provided by the user, no study labels will be printed.

Overlapping information on left side of forest plot:

Depending on the number of columns printed on the left side of the forest plot, information on heterogeneity measures or statistical tests (see below) can be overlapping with the x-axis. Argument `addrows.below.overall` can be used to specify the number of empty rows that are printed between meta-analysis results and information on heterogeneity measures and statistical tests. By default, no additional rows are added to the forest plot. If `addrows.below.overall = NULL`, the function tries to add a sufficient number of empty rows to prevent overlapping text. Another possibility is to manually increase the space between the columns on the left side (argument `colgap.left`) or between the columns on the left side and the forest plot (argument `colgap.forest.left`).

Columns printed on right side of forest plot:

Argument `rightcols` can be used to specify columns which are printed on the right side of the forest plot. If argument `rightcols` is `FALSE`, no columns will be printed on the right side. By default, i.e. if argument `rightcols` is `NULL` and `layout = "meta"`, the following ***columns*** will be printed ***on the right side of the forest plot***:

Meta-analysis results	**Value of argument rightcols**
No summary	`c("effect", "ci")`
Only common effect model	`c("effect", "ci", "w.common")`
Only random effects model	`c("effect", "ci", "w.random")`
Both models	`c("effect", "ci", "w.common", "w.random")`

By default, estimated treatment effect and corresponding confidence interval will be printed. Depending on arguments `common` and `random`, weights of the common effect and/or random effects model will be given too.

Predefined columns and column labels:

The arguments `leftlabs` and `rightlabs` can be used to specify column headings which are printed on the left or right side of the forest plot. For certain columns predefined labels exist which are used by default, i.e., if arguments `leftlabs` and `rightlabs` are `NULL`:

Column:	`studlab`	`TE`	`seTE`	`cluster`	`n.e`	`n.c`
Label:	"Study"	"TE"	"seTE"	"Cluster"	"Total"	"Total"
Column:	`n`	`event.e`	`event.c`	`event`	`mean.e`	`mean.c`
Label:	"Total"	"Events"	"Events"	"Events"	"Mean"	"Mean"
Column:	`sd.e`	`sd.c`	`time.e`	`time.c`	`effect`	
Label:	"SD"	"SD"	"Time"	"Time"	`x$sm`	
Column:	`ci`	`effect.ci`	`w.common`	`w.random`	`cycles`	
Label:	`x$level`"%-CI"	*effect+ci*	"W(common)"	"W(random)"	"Cycles"	
Column:	`pval`	`tau2`	`tau`			
Label:	"P-value"	"Tau2"	"Tau"			

For other columns, the column name will be used as a label if no column label is defined. It is possible to only provide labels for new columns (see Examples). Otherwise the length of `leftlabs` and `rightlabs` must be the same as the number of printed columns. The value `NA` can be used to specify columns which should use default labels (see Examples).

In pairwise meta-analysis comparing two groups (i.e., `metabin`, `metacont`, `metainc`, and `metagen` depending on the outcome), arguments `label.e` and `label.c` are used to label columns belonging to the two treatment groups. By default, labels defined in the meta-analysis object are used. The columns where treatment labels are attached can be changed using arguments `label.e.attach` and `label.c.attach`.

Risk of bias assessment:

A risk of bias (RoB) assessment can be shown in the forest plot by either using a meta-analysis object with an RoB assessment as main input or providing a suitable object created with `rob`. Argument `rob = FALSE` can be used to suppress the print of the risk of bias information.

RoB assessments are shown as the only information on the right side of the forest plot. Thus, arguments `rightcols` and `rightlabs` should not be used. Predefined columns shown by default on the right side of a forest plot will be moved to the left side.

Information on heterogeneity and statistical tests:

Argument `hetstat` can be a character string to specify where to print heterogeneity information:

- row with results for common effect model (`hetstat = "common"`),
- row with results for random effects model (`hetstat = "random"`).

Otherwise, information on heterogeneity measures is printed below the meta-analysis results if argument `overall.hetstat = TRUE` (default). The heterogeneity measures to print can be specified (see list of arguments following `overall.hetstat`).

In addition, the following arguments can be used to print results for various statistical tests:

Argument	**Statistical test**
`test.overall.common`	Test for overall effect (common effect model)
`test.overall.random`	Test for overall effect (random effects model)
`test.effect.subgroup.common`	Test for effect in subgroup (CE model)
`test.effect.subgroup.random`	Test for effect in subgroup (RE model)
`test.subgroup.common`	Test for subgroup differences (CE model)
`test.subgroup.random`	Test for subgroup differences (RE model)

By default, these arguments are `FALSE` with exception of tests for subgroup differences which are `TRUE`. R function `settings.meta` can be used to change this default for the entire R session. For example, use the following command to always print results of tests for an overall effect: `settings.meta(test.overall = TRUE)`.

Highlight regions corresponding to minimal clinically important differences:

Regions corresponding to minimal clinically important differences can be added to the forest plot using either argument `cid` or `cid.below.null` and `cid.above.null`. Input for the later arguments will be ignored if argument `cid` was specified. In this case, the values of `cid.below.null` and `cid.above.null` will be equal to

- `cid` and `1 / cid` for ratio measures,
- `cid` and `-cid` for difference measures.

Thresholds based on argument `cid` will always be symmetric. Asymmetric thresholds can be defined using arguments `cid.below.null` and `cid.above.null`.

Flexible printing of subgroup results:

Argument `subgroup` determines whether summary results are printed for subgroups. A logical vector of length equal to the number of subgroups can be provided to determine which subgroup summaries are printed. By default, only subgroup results based on at least two studies are printed which is identical to use argument subgroup = k.w > 1. The order of the logical vector corresponds to the order of subgroups in list element 'subgroup.levels' of a meta-analysis object. Argument subgroup = k.w >= 1 can be used to show results for all subgroups (including those with a single study).

The following arguments can be used in a similar way:

- subgroup.hetstat (heterogeneity statistic in subgroups),
- common.subgroup (common effect estimates in subgroups),
- random.subgroup (random effects estimates in subgroups),
- prediction.subgroup (prediction interval in subgroups),
- test.effect.subgroup (test for effect in subgroups),
- test.effect.subgroup.common (test for effect in subgroups, common effect model),
- test.effect.subgroup.random (test for effect in subgroups, random effects model).

Data set with information to print with overall results:

Argument data.pooled can be used to provide information printed in the line(s) with overall results, i.e., for the common effect or random effects model. Input must be a data frame with variable names equal to those provided in arguments leftcols or rightcols. Only variables for additional variables are considered.

It is possible to provide a row in data set data.pooled for each common effect or random effects estimate. The order in the data set corresponds to the order of common effect and random effects estimates in the forest plot, i.e., common effect followed by random effects estimates. If the data set contains a single row, the value provided for a variable is considered for all printed common effect and random effects estimates

In meta-analyses with subgroups, a row must be provided in data set data.pooled for each overall common effect or random effects estimate, followed by common effect or random effects estimates within subgroups. The order for subgroup results is the same as see in the forest plot.

Additional general settings:

Arguments text.common, text.random, and text.predict can be used to change the label to identify overall results (common effect and random effects model as well as prediction interval). By default the following text is printed:

- "Common effect model" (argument text.common)
- "Random effects model" (text.random)
- "Prediction interval" (text.predict)

If confidence interval levels are different for individual studies, meta-analysis, and prediction interval (arguments level, level.ma, level.predict in meta-analysis functions, e.g., metabin), additional information is printed, e.g., " (99%-CI)" for a 99% confidence interval in the meta-analysis.

Argument pscale can be used to rescale single proportions or risk differences, e.g., pscale = 1000 means that proportions are expressed as events per 1000 observations. This is useful in situations with (very) low event probabilities.

Argument irscale can be used to rescale single rates or rate differences, e.g., irscale = 1000 means that rates are expressed as events per 1000 time units, e.g., person-years. This is useful in situations with (very) low rates. Argument irunit can be used to specify the time unit used in individual studies (default: "person-years"). This information is printed in summaries and forest plots if argument irscale is not equal to 1.

Forest plots in RevMan5 layout:

If argument `layout = "RevMan5"` (and arguments `leftcols` and `rightcols` are `NULL`), the layout for forest plots used for Cochrane reviews (which were generated with Review Manager 5) is reproduced:

1. All columns are printed on the left side of the forest plot (see arguments `leftcols` and `rightcols`)
2. Tests for overall effect and subgroup differences are printed (`test.overall`, `test.effect.subgroup`, `test.subgroup`)
3. Diamonds representing meta-analysis results are printed in black (`diamond.common`, `diamond.random`)
4. Colour of squares depends on the meta-analysis object (`col.square`, `col.square.lines`)
5. Information on effect measure and meta-analysis method is printed above the forest plot (`smlab`)
6. Label "Study or Subgroup" is printed for meta-analysis with subgroups (`leftlabs`)

Forest plots in JAMA layout:

If argument `layout = "JAMA"` (and arguments `leftcols` and `rightcols` are `NULL`), instructions for authors of the *Journal of the American Medical Association* are taken into account:

1. Graph labels on right and left side are printed in bold font at top of forest plot (see arguments `bottom.lr` and `ff.lr`)
2. Information on effect measure and level of confidence interval is printed at bottom of forest plot (`xlab`)
3. Tests for overall effect are printed (`test.overall`)
4. Diamonds representing meta-analysis results are printed in lightblue (`diamond.common`, `diamond.random`)
5. Squares representing individual study results are printed in darkblue (`col.square`, `col.square.lines`)
6. Between-study variance τ^2 is not printed
7. Empty rows are omitted (`addrow`, `addrow.overall`, `addrow.subgroups`)
8. Label "Source" is printed instead of "Study" (`leftlabs`)
9. P-values are printed without leading zeros (`zero.pval`)
10. P-values are rounded to three digits (for $0.001 < p \leq 0.01$) or two digits ($p > 0.01$) (`JAMA.pval`)

Study labels according to JAMA guidelines can be generated using `labels.meta`.

Forest plots showing results of subgroups:

The following changes are conducted if argument `layout = "subgroup"` (and arguments `leftcols` and `rightcols` are `NULL`) and a subgroup analysis was conducted:

1. Individual study results are omitted (see argument `study.results`)
2. Total number of observations is not printed (`pooled.totals`)
3. Label "Subgroup" is printed instead of "Study" (`leftlabs`)

Note

R function `.forestArgs` generates a character vector with all arguments of `forest.meta`.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Guddat C, Grouven U, Bender R, Skipka G (2012): A note on the graphical presentation of prediction intervals in random-effects meta-analyses. *Systematic Reviews*, **1**, 34

Higgins JPT, Thompson SG, Spiegelhalter DJ (2009): A re-evaluation of random-effects meta-analysis. *Journal of the Royal Statistical Society: Series A*, **172**, 137-59

See Also

metabin, metacont, metagen, forest.metabind, settings.meta, labels.meta

Examples

```
data(Olkin1995)
m1 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, subset = c(41, 47, 51, 59),
  sm = "RR", method = "I",
  studlab = paste(author, year))


## Not run:
# Do standard (symmetric) forest plot
#
forest(m1)

## End(Not run)

# Layout of forest plot similar to Review Manager 5
#
# Furthermore, add labels on both sides of forest plot and
# prediction interval
#
forest(m1, layout = "RevMan5", common = FALSE,
  label.left = "Favours experimental", col.label.left = "green",
  label.right = "Favours control", col.label.right = "red",
  prediction = TRUE)


## Not run:
# Create PDF files with the forest plot
#
# - specify filename (R function pdf() is used due to extension .pdf)
# - height of the figure is automatically determined
# - width is set to 10 inches
forest(m1, file = "forest-m1-1.pdf", width = 10)
#
# - specify graphics device function
#   (filename "Rplots.pdf" used, see help page of R function pdf())
# - height of the figure is automatically determined
# - width is set to 10 inches
# - set title for PDF file
# - set background of forest plot
```

```
forest(m1, func.gr = pdf, width = 10,
  args.gr = list(title = "My Forest Plot", bg = "green"))
#
# - manually specify the height of the figure
pdf("forest-m1-2.pdf", width = 10, height = 3)
forest(m1)
dev.off()

# Define equivalence limits: 0.75 and 1 / 0.75
#
forest(m1, layout = "RevMan5", common = FALSE, cid = 0.75,
  fill = "lightgray",
  fill.cid = "white")

# Fill regions with beneficial and detrimental effects
#
forest(m1, layout = "RevMan5", common = FALSE, cid = 0.75,
  fill = "lightgray",
  fill.cid.below.null = "green",
  fill.cid.above.null = "red")

# Define thresholds for small, moderate and large effects
# and use hcl.colors() to define colours to fill regions
#
thresholds <- c(0.25, 0.5, 0.75)
n.cols <- length(thresholds)
forest(m1, layout = "RevMan5", common = FALSE,
  label.left = "Desirable effect",
  label.right = "Undesirable effect",
  lty.cid = 3, col.cid = "darkgray",
  cid.below.null = thresholds, cid.above.null = 1 / rev(thresholds),
  fill.cid.below.null =
    hcl.colors(n.cols, palette = "Blues 2", alpha = 0.6),
  fill.cid.above.null =
    hcl.colors(n.cols, palette = "Oranges", alpha = 0.6, rev = TRUE))

# Conduct subgroup meta-analysis
#
m2 <- update(m1,
  subgroup = ifelse(year < 1987, "Before 1987", "1987 and later"),
  print.subgroup.name = FALSE)

# Show summary results for subgroups with at least two studies
#
forest(m2, sortvar = -TE, random = FALSE)

# Show results for all subgroups
#
forest(m2, sortvar = -TE, random = FALSE, subgroup = k.w >= 1)

# Forest plot specifying argument xlim
#
forest(m1, xlim = c(0.01, 10))
```

```
# Print results of test for overall effect
#
forest(m1, test.overall.common = TRUE, test.overall.random = TRUE)

# Forest plot with 'classic' layout used in R package meta,
# version < 1.6-0
#
forest(m1, col.square = ”black”, hetstat = FALSE)

# Change set of columns printed on left side of forest plot
# (resulting in overlapping text)
#
forest(m1, random = FALSE, leftcols = ”studlab”)

# Use argument 'calcwidth.hetstat' to consider text for heterogeneity
# measures in width of column with study labels
#
forest(m1, random = FALSE, leftcols = ”studlab”,
  calcwidth.hetstat = TRUE)

# Use argument 'addrows.below.overall' to manually add two empty
# rows
#
forest(m1, random = FALSE, leftcols = ”studlab”, addrows = 2)

# Do not print columns on right side of forest plot
#
forest(m1, rightcols = FALSE)

# Change study label to ”Author”
#
forest(m1, random = FALSE, leftlabs = c(”Author”, NA, NA, NA, NA))

# Just give effect estimate and 95% confidence interval on right
# side of forest plot (in one column)
#
forest(m1, rightcols = ”effect.ci”)

# Just give effect estimate and 95% confidence interval on right
# side of forest plot
#
forest(m1, rightcols = c(”effect”, ”ci”))

# 1. Change order of columns on left side
# 2. Attach labels to columns 'event.e' and 'event.c' instead of
#    columns 'n.e' and 'n.c'
#
forest(m1,
  leftcols = c(”studlab”, ”n.e”, ”event.e”, ”n.c”, ”event.c”),
  label.e.attach = ”event.e”, label.c.attach = ”event.c”)

# Specify column labels only for variables 'year' and 'author'
# (and define digits for additional variables)
#
forest(m1,
  leftcols = c("studlab", "event.e", "n.e", "event.c", "n.c", "author", "year"),
  leftlabs = c("Author", "Year of Publ"))

# Center text in all columns
#
forest(m1,
  leftcols = c("studlab", "event.e", "n.e", "event.c", "n.c",
               "author", "year"),
  leftlabs = c("Author", "Year of Publ"), hetstat = FALSE,
  just = "center", just.addcols = "center", just.studlab = "center")

# Same result
#
forest(m1,
  leftcols = c("studlab", "event.e", "n.e", "event.c", "n.c",
             "author", "year"),
  leftlabs = c("Author", "Year of Publ"), hetstat = FALSE,
  just = "c", just.addcols = "c", just.studlab = "c")

# Change some fontsizes and fontfaces
#
forest(m1,
  fs.study = 10, ff.study = "italic",
  fs.study.label = 11, ff.study.label = "bold",
  fs.axis = 5, ff.axis = "italic",
  ff.smlab = "bold.italic",
  ff.common = "plain", ff.hetstat = "plain")

# Change some colours
#
forest(m1,
  col.diamond = "green", col.diamond.lines = "red",
  col.study = c("green", "blue", "red", "orange"),
  col.square = "pink", col.square.lines = "black")

# Sort by weight in common effect model
#
forest(m1, sortvar = w.common, random = FALSE)

# Sort by decreasing weight in common effect model
#
forest(m1, sortvar = -w.common, random = FALSE)

# Sort by size of treatment effect
#
forest(m1, sortvar = TE, random = FALSE)

# Sort by size of treatment effect
#
forest(m1, sortvar = -TE, random = FALSE)
```

```
# Sort by decreasing year of publication
#
forest(m1, sortvar = -year, random = FALSE)

# Print results of test for subgroup differences (random effects
# model)
#
forest(m2, sortvar = -TE, common = FALSE)

# Print only subgroup results
#
forest(m2, layout = "subgroup")

# Print only subgroup results (and consider text for tests of
# subgroup differences in width of subgroup column)
#
forest(m2, layout = "subgroup", calcwidth.tests = TRUE)

# Print only subgroup results (and consider text for heterogeneity
# in width of subgroup column)
#
forest(m2, layout = "subgroup", calcwidth.hetstat = TRUE)

## End(Not run)
```

forest.metabind *Forest plot to display the result of a meta-analysis*

Description

Draws a forest plot in the active graphics window (using grid graphics system).

Usage

```
## S3 method for class 'metabind'
forest(
  x,
  leftcols,
  leftlabs,
  rightcols = c("effect", "ci"),
  rightlabs,
  common = x$common,
  random = x$random,
  overall = x$overall,
  subgroup = FALSE,
  hetstat = FALSE,
  overall.hetstat = x$overall.hetstat,
  prediction = x$prediction,
  lab.NA = "",
  col.square = gs("col.square"),
  col.square.lines = col.square,
  col.circle = gs("col.circle"),
  col.circle.lines = col.circle,
  col.diamond = gs("col.diamond"),
  col.diamond.common = col.diamond,
  col.diamond.random = col.diamond,
  col.diamond.lines = gs("col.diamond.lines"),
  col.diamond.lines.common = col.diamond.lines,
  col.diamond.lines.random = col.diamond.lines,
  col.predict = gs("col.predict"),
  col.predict.lines = gs("col.predict.lines"),
  type = NULL,
  type.common = NULL,
  type.random = NULL,
  type.predict = NULL,
  digits = gs("digits.forest"),
  digits.se = gs("digits.se"),
  digits.stat = gs("digits.stat"),
  digits.pval = max(gs("digits.pval") - 2, 2),
  digits.pval.Q = max(gs("digits.pval.Q") - 2, 2),
  digits.Q = gs("digits.Q"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.I2 = max(gs("digits.I2") - 1, 0),
  scientific.pval = gs("scientific.pval"),
  big.mark = gs("big.mark"),
  print.subgroup.labels = x$with.subgroups,
  addrow.subgroups = print.subgroup.labels,
  smlab,
  calcwidth.pooled = overall,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

Arguments

`x` An object of class `metabind`.

`leftcols` A character vector specifying (additional) columns to be plotted on the left side of the forest plot or a logical value (see Details).

`leftlabs` A character vector specifying labels for (additional) columns on left side of the forest plot (see Details).

`rightcols` A character vector specifying (additional) columns to be plotted on the right side of the forest plot or a logical value (see Details).

`rightlabs` A character vector specifying labels for (additional) columns on right side of the forest plot (see Details).

`common` A logical indicating whether common effect estimates should be plotted.

`random` A logical indicating whether random effects estimates should be plotted.

`overall` A logical indicating whether overall summaries should be plotted. This argument is useful in a meta-analysis with subgroups if summaries should only be plotted on group level.

`subgroup` A logical indicating whether subgroup results should be shown in forest plot. This argument is useful in a meta-analysis with subgroups if summaries should not be plotted on group level.

`hetstat` Either a logical value indicating whether to print results for heterogeneity measures at all or a character string (see Details).

`overall.hetstat`
A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`prediction` A logical indicating whether prediction interval(s) should be printed.

`lab.NA` A character string to label missing values.

`col.square` The colour for squares reflecting study's weight in the meta-analysis.

`col.square.lines`
The colour for the outer lines of squares reflecting study's weight in the meta-analysis.

`col.circle` The colour for circles reflecting study weights in the meta-analysis.

`col.circle.lines`
The colour for the outer lines of circles reflecting study's weight in the meta-analysis.

`col.diamond` The colour of diamonds representing the results for common effect and random effects models.

`col.diamond.common`
The colour of diamonds for common effect estimates.

`col.diamond.random`
The colour of diamonds for random effects estimates.

`col.diamond.lines`
The colour of the outer lines of diamonds representing the results for common effect and random effects models.

`col.diamond.lines.common`
The colour of the outer lines of diamond for common effect estimates.

`col.diamond.lines.random`
The colour of the outer lines of diamond for random effects estimates.

`col.predict` Background colour of prediction intervals.

`col.predict.lines`
Colour of outer lines of prediction intervals.

`type` A character string or vector specifying how to plot estimates.

`type.common` A single character string specifying how to plot common effect estimates.

`type.random` A single character string specifying how to plot random effects estimates.

`type.predict` A single character string specifying how to plot prediction intervals.

`digits` Minimal number of significant digits for treatment effects, see `print.default`.

`digits.se` Minimal number of significant digits for standard errors, see `print.default`.

`digits.stat` Minimal number of significant digits for z- or t-statistic for test of overall effect, see `print.default`.

`digits.pval` Minimal number of significant digits for p-value of overall treatment effect, see `print.default`.

`digits.pval.Q` Minimal number of significant digits for p-value of heterogeneity test, see `print.default`.

`digits.Q` Minimal number of significant digits for heterogeneity statistic Q, see `print.default`.

`digits.tau2` Minimal number of significant digits for between-study variance, see `print.default`.

`digits.tau` Minimal number of significant digits for square root of between-study variance, see `print.default`.

`digits.I2` Minimal number of significant digits for I-squared statistic, see `print.default`.

`scientific.pval` A logical specifying whether p-values should be printed in scientific notation, e.g., 1.2345e-01 instead of 0.12345.

`big.mark` A character used as thousands separator.

`print.subgroup.labels` A logical indicating whether subgroup label should be printed.

`addrow.subgroups` A logical value indicating whether an empty row is printed between results for subgroups.

`smlab` A label for the summary measurex (printed at top of figure).

`calcwidth.pooled` A logical indicating whether text for common effect and random effects model should be considered to calculate width of the column with study labels.

`warn.deprecated` A logical indicating whether warnings should be printed if deprecated arguments are used.

`...` Additional graphical arguments (passed on to `forest.meta`).

Details

A forest plot, also called confidence interval plot, is drawn in the active graphics window. The forest functions in R package **meta** are based on the grid graphics system. In order to print the forest plot, resize the graphics window and either use `dev.copy2eps` or `dev.copy2pdf`. Another possibility is to create a file using `pdf`, `png`, or `svg` and to specify the width and height of the graphic (see `forest.meta` examples).

The arguments `leftcols` and `rightcols` can be used to specify columns which are plotted on the left and right side of the forest plot, respectively.

The arguments `leftlabs` and `rightlabs` can be used to specify column headings which are plotted on left and right side of the forest plot, respectively. For certain columns predefined labels exist. For other columns, the column name will be used as a label. It is possible to only provide labels for new columns (see forest.meta examples). Otherwise the length of leftlabs and rightlabs must be the same as the number of printed columns, respectively. The value NA can be used to specify columns which should use default labels.

Argument hetstat can be a character string to specify where to print heterogeneity information:

- row with results for common effect model (hetstat = "common"),
- row with results for random effects model (hetstat = "random"),
- rows with 'study' information (hetstat = "study").

Otherwise, information on heterogeneity is printed in dedicated rows.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

forest.meta, metabin, metacont, metagen, metabind, settings.meta

Examples

```
data(Fleiss1993cont)

# Add some (fictitious) grouping variables:
#
Fleiss1993cont$age <- c(55, 65, 55, 65, 55)
Fleiss1993cont$region <- c("Europe", "Europe", "Asia", "Asia", "Europe")

m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD")

# Conduct two subgroup analyses
#
mu1 <- update(m1, subgroup = age, subgroup.name = "Age group")
mu2 <- update(m1, subgroup = region, subgroup.name = "Region")

# Combine subgroup meta-analyses and show forest plot with subgroup
# results
#
mb1 <- metabind(mu1, mu2)
mb1
forest(mb1)
```

`forest.metacum` *Forest plot to display the result of a cumulative meta-analysis*

Description

Draws a forest plot in the active graphics window (using grid graphics system).

Usage

```
## S3 method for class 'metacum'
forest(
  x,
  leftcols = NULL,
  leftlabs = NULL,
  rightcols = NULL,
  rightlabs = NULL,
  prediction = x$prediction,
  overall = x$overall,
  just.addcols = "right",
  smlab = "Cumulative Meta-Analysis",
  type = "square",
  lab.NA = ".",
  backtransf = x$backtransf,
  big.mark = gs("big.mark"),
  digits = gs("digits.forest"),
  digits.pval = gs("digits.pval"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.I2 = gs("digits.I2"),
  digits.cid = gs("digits.cid"),
  digits.percent = 1,
  col = gs("col.study"),
  col.bg = ifelse(type == "diamond", gs("col.diamond"), gs("col.square")),
 col.border = ifelse(type == "diamond", gs("col.diamond.lines"), gs("col.square.lines")),
  col.bg.predict = gs("col.predict"),
  col.border.predict = gs("col.predict.lines"),
  addrows.below.overall = 1L * details,
  details = gs("forest.details"),
  ...
)
```

Arguments

x	An object of class metacum.
leftcols	A character vector specifying (additional) columns to be plotted on the left side of the forest plot or a logical value.

`leftlabs` A character vector specifying labels for (additional) columns on left side of the forest plot.

`rightcols` A character vector specifying (additional) columns to be plotted on the right side of the forest plot or a logical value.

`rightlabs` A character vector specifying labels for (additional) columns on right side of the forest plot.

`prediction` A logical indicating whether prediction intervals should be printed.

`overall` A logical indicating whether overall results should be shown.

`just.addcols` Justification of text for additional columns (possible values: "left", "right", "center").

`smlab` A label for the summary measure (printed at top of figure).

`type` A character string or vector specifying how to plot treatment effects and confidence intervals for cumulative meta-analysis results.

`lab.NA` A character string to label missing values.

`backtransf` A logical indicating whether results should be back transformed in forest plots. If `backtransf = TRUE`, results for `sm = "OR"` are presented as odds ratios rather than log odds ratios, for example.

`big.mark` A character used as thousands separator.

`digits` Minimal number of significant digits for treatment effects, see `print.default`.

`digits.pval` Minimal number of significant digits for p-values.

`digits.tau2` Minimal number of significant digits for between-study variance.

`digits.tau` Minimal number of significant digits for square root of between-study variance.

`digits.I2` Minimal number of significant digits for I-squared statistic.

`digits.cid` Minimal number of significant digits for CID / decision thresholds, see `print.default`.

`digits.percent` Minimal number of significant digits for probabilities, printed as percentages, see `print.default`.

`col` The colour for cumulative meta-analysis results (only considered if `type = "square"`).

`col.bg` The background colour for squares and diamonds of cumulative meta-analysis results.

`col.border` The colour for the outer lines of squares and diamonds of cumulative meta-analysis results.

`col.bg.predict` The background colour for prediction intervals of cumulative meta-analysis results.

`col.border.predict`
The colour for the outer lines of prediction intervals of cumulative meta-analysis results.

`addrows.below.overall`
A numeric value indicating how many empty rows are printed between meta-analysis results and meta-analysis details.

`details` A logical specifying whether details on statistical methods should be printed.

`...` Additional graphical arguments (passed on to `forest.meta`).

Details

A forest plot, also called confidence interval plot, is drawn in the active graphics window. The forest functions in R package **meta** are based on the grid graphics system. In order to print the forest plot, resize the graphics window and either use `dev.copy2eps` or `dev.copy2pdf`. Another possibility is to create a file using `pdf`, `png`, or `svg` and to specify the width and height of the graphic (see `forest.meta` examples).

The arguments `leftcols` and `rightcols` can be used to specify columns which are plotted on the left and right side of the forest plot, respectively.

The arguments `leftlabs` and `rightlabs` can be used to specify column headings which are plotted on left and right side of the forest plot, respectively. For certain columns predefined labels exist. For other columns, the column name will be used as a label. It is possible to only provide labels for new columns (see `forest.meta` examples). Otherwise the length of `leftlabs` and `rightlabs` must be the same as the number of printed columns, respectively. The value `NA` can be used to specify columns which should use default labels.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

`forest.meta`, `metacum`, `settings.meta`

Examples

```
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac,
  data = Fleiss1993bin, studlab = study, sm = "RR", method = "I")
m1
metacum(m1)
metacum(m1, pooled = "random")

forest(metacum(m1))
forest(metacum(m1, pooled = "random"))
forest(metacum(m1, pooled = "random", prediction = TRUE))
```

`forest.metainf` *Forest plot to display the result of a leave-one-out meta-analysis*

Description

Draws a forest plot in the active graphics window (using grid graphics system).

Usage

```
## S3 method for class 'metainf'
forest(
  x,
  leftcols = NULL,
  leftlabs = NULL,
  rightcols = NULL,
  rightlabs = NULL,
  prediction = x$prediction,
  overall = x$overall,
  just.addcols = "right",
  smlab = "Leave-One-Out Meta-Analysis",
  type = "square",
  lab.NA = ".",
  backtransf = x$backtransf,
  big.mark = gs("big.mark"),
  digits = gs("digits.forest"),
  digits.pval = gs("digits.pval"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.I2 = gs("digits.I2"),
  digits.cid = gs("digits.cid"),
  digits.percent = 1,
  col = gs("col.study"),
  col.bg = ifelse(type == "diamond", gs("col.diamond"), gs("col.square")),
  col.border = ifelse(type == "diamond", gs("col.diamond.lines"), gs("col.square.lines")),
  col.bg.predict = gs("col.predict"),
  col.border.predict = gs("col.predict.lines"),
  addrows.below.overall = 1L * details,
  details = gs("forest.details"),
  ...
)
```

Arguments

`x` An object of class `metainf`.

`leftcols` A character vector specifying (additional) columns to be plotted on the left side of the forest plot or a logical value.

`leftlabs` A character vector specifying labels for (additional) columns on left side of the forest plot.

`rightcols` A character vector specifying (additional) columns to be plotted on the right side of the forest plot or a logical value.

`rightlabs` A character vector specifying labels for (additional) columns on right side of the forest plot.

`prediction` A logical indicating whether prediction intervals should be printed.

`overall` A logical indicating whether overall results should be shown.

`just.addcols` Justification of text for additional columns (possible values: "left", "right", "center").

`smlab` A label for the summary measure (printed at top of figure).

`type` A character string or vector specifying how to plot treatment effects and confidence intervals for cumulative meta-analysis results.

`lab.NA` A character string to label missing values.

`backtransf` A logical indicating whether results should be back transformed in forest plots. If `backtransf = TRUE`, results for `sm = "OR"` are presented as odds ratios rather than log odds ratios, for example.

`big.mark` A character used as thousands separator.

`digits` Minimal number of significant digits for treatment effects, see `print.default`.

`digits.pval` Minimal number of significant digits for p-values.

`digits.tau2` Minimal number of significant digits for between-study variance.

`digits.tau` Minimal number of significant digits for square root of between-study variance.

`digits.I2` Minimal number of significant digits for I-squared statistic.

`digits.cid` Minimal number of significant digits for CID / decision thresholds, see `print.default`.

`digits.percent` Minimal number of significant digits for probabilities, printed as percentages, see `print.default`.

`col` The colour for cumulative meta-analysis results (only considered if `type = "square"`).

`col.bg` The background colour for squares and diamonds of cumulative meta-analysis results.

`col.border` The colour for the outer lines of squares and diamonds of cumulative meta-analysis results.

`col.bg.predict` The background colour for prediction intervals of cumulative meta-analysis results.

`col.border.predict` The colour for the outer lines of prediction intervals of cumulative meta-analysis results.

`addrows.below.overall` A numeric value indicating how many empty rows are printed between meta-analysis results and meta-analysis details.

`details` A logical specifying whether details on statistical methods should be printed.

`...` Additional graphical arguments (passed on to `forest.meta`).

Details

A forest plot, also called confidence interval plot, is drawn in the active graphics window. The forest functions in R package **meta** are based on the grid graphics system. In order to print the forest plot, resize the graphics window and either use `dev.copy2eps` or `dev.copy2pdf`. Another possibility is to create a file using `pdf`, `png`, or `svg` and to specify the width and height of the graphic (see `forest.meta` examples).

The arguments `leftcols` and `rightcols` can be used to specify columns which are plotted on the left and right side of the forest plot, respectively.

The arguments `leftlabs` and `rightlabs` can be used to specify column headings which are plotted on left and right side of the forest plot, respectively. For certain columns predefined labels exist. For other columns, the column name will be used as a label. It is possible to only provide labels for new columns (see `forest.meta` examples). Otherwise the length of `leftlabs` and `rightlabs` must be the same as the number of printed columns, respectively. The value `NA` can be used to specify columns which should use default labels.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

`forest.meta`, `metainf`, `settings.meta`

Examples

```
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac,
  data = Fleiss1993bin, studlab = study, sm = "RR", method = "I")
m1
metainf(m1)
metainf(m1, pooled = "random")

forest(metainf(m1))
forest(metainf(m1, pooled = "random"))
forest(metainf(m1, pooled = "random", prediction = TRUE))
```

funnel.meta *Funnel plot*

Description

Draw a funnel plot which can be used to assess small study effects in meta-analysis. A contour-enhanced funnel plot can also be produced to assess causes of funnel plot asymmetry.

Usage

```
## S3 method for class 'meta'
funnel(
  x,
  type = "standard",
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  common = x$common,
  random = x$random,
  axes = TRUE,
  pch = if (!inherits(x, "trimfill")) 21 else ifelse(x$trimfill, 1, 21),
  text = NULL,
  cex = 1,
  lty.common = 2,
  lty.random = 9,
  lwd = 1,
  lwd.common = lwd,
  lwd.random = lwd,
  col = "black",
  bg = "darkgray",
  col.common = "black",
  col.random = "black",
  log,
  yaxis,
  contour.levels = if (type == "contour") c(0.9, 0.95, 0.99) else NULL,
  col.contour = if (type == "contour") c("gray80", "gray70", "gray60") else NULL,
  ref = ifelse(is_relative_effect(x$sm), 1, 0),
  level = if (common | random) x$level else NULL,
  studlab = FALSE,
  cex.studlab = 0.8,
  pos.studlab = 2,
  ref.triangle = FALSE,
  lty.ref = 1,
  lwd.ref = lwd,
  col.ref = "black",
  lty.ref.triangle = 5,
  backtransf = x$backtransf,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

setvals(x, vals = seq_along(unique(x)))
```

Arguments

`x` An object of class `meta`.

`type` A character string indicating type of funnel plot. Either `"standard"` or `"contour"`, can be abbreviated.

`xlim` The x limits (min,max) of the plot.

`ylim` The y limits (min,max) of the plot.

`xlab` A label for the x-axis.

`ylab` A label for the y-axis.

`common` A logical indicating whether the common effect estimate should be plotted.

`random` A logical indicating whether the random effects estimate should be plotted.

`axes` A logical indicating whether axes should be drawn on the plot.

`pch` The plotting symbol(s) used for individual studies.

`text` A character vector specifying the text to be used instead of plotting symbol.

`cex` The magnification to be used for plotting symbols.

`lty.common` Line type (common effect estimate).

`lty.random` Line type (random effects estimate).

`lwd` The line width for confidence intervals (if `level` is not `NULL`).

`lwd.common` The line width for common effect estimate (if `common` is not `NULL`).

`lwd.random` The line width for random effects estimate (if `random` is not `NULL`).

`col` A vector with colour of plotting symbols.

`bg` A vector with background colour of plotting symbols (only used if `pch` in `21:25`).

`col.common` Colour of line representing common effect estimate.

`col.random` Colour of line representing random effects estimate.

`log` A character string which contains `"x"` if the x-axis is to be logarithmic, `"y"` if the y-axis is to be logarithmic and `"xy"` or `"yx"` if both axes are to be logarithmic.

`yaxis` A character string indicating which type of weights are to be used. Either `"se"`, `"invvar"`, `"invse"`, `"size"`, `"invsqrtsize"`, or `"ess"`.

`contour.levels` A numeric vector specifying contour levels to produce contour-enhanced funnel plot.

`col.contour` Colour of contours.

`ref` Reference value (null effect) used to produce contour-enhanced funnel plot.

`level` The confidence level utilised in the plot. For the funnel plot, confidence limits are not drawn if `yaxis="size"` or `yaxis="invsqrtsize"`.

`studlab` A logical indicating whether study labels should be printed in the graph. A vector with study labels can also be provided (must be of same length as `x$TE` then).

`cex.studlab` Size(s) of study labels, see argument `cex` in `text`.

`pos.studlab` Position of study labels, see argument `pos` in `text`.

`ref.triangle` A logical indicating whether approximate confidence limits should be printed around reference value (null effect).

`lty.ref` Line type (reference value).

`lwd.ref` The line width for the reference value and corresponding confidence intervals (if `ref.triangle` is TRUE and `level` is not `NULL`).

`col.ref` Colour of line representing reference value.

`lty.ref.triangle`
Line type (confidence intervals of reference value).

`backtransf` A logical indicating whether results for relative summary measures (argument `sm` equal to `"OR"`, `"RR"`, `"HR"`, or `"IRR"`) should be back transformed in funnel plots. If `backtransf=TRUE`, results for `sm="OR"` are printed as odds ratios rather than log odds ratios, for example.

`warn.deprecated`
: A logical indicating whether warnings should be printed if deprecated arguments are used.

`...` Additional arguments (passed on to plot.default).

`vals` Vector with values used in `setvals` (see Examples).

Details

A funnel plot (Light & Pillemer, 1984) is drawn in the active graphics window. If `common` is TRUE, the estimate of the common effect model is plotted as a vertical line. Similarly, if `random` is TRUE, the estimate of the random effects model is plotted. If `level` is not NULL, the corresponding approximate confidence limits are drawn around the common effect estimate (if `common` is TRUE) or the random effects estimate (if `random` is TRUE and `common` is FALSE).

In the funnel plot, the standard error of the treatment estimates is plotted on the y-axis by default (`yaxis = "se"`) which is likely to be the best choice (Sterne & Egger, 2001). Only exception is meta-analysis of diagnostic test accuracy studies (Deeks et al., 2005) where the inverse of the square root of the *effective study size* is used (`yaxis = "ess"`). Other possible choices for `yaxis` are `"invvar"` (inverse of the variance), `"invse"` (inverse of the standard error), `"size"` (study size), and `"invsqrtsize"` (1 / sqrt(study size)).

If argument `yaxis` is not equal to `"size"`, `"invsqrtsize"` or `"ess"`, contour-enhanced funnel plots can be produced (Peters et al., 2008) by specifying the contour levels (argument `contour.levels`). By default (argument `col.contour` missing), suitable gray levels will be used to distinguish the contours. Different colours can be chosen by argument `col.contour`.

Note

R function `setvals` can be used to easily define the input for the arguments `pch`, `text`, `cex`, `col`, `bg`, and `cex.studlab`.

Author(s)

Guido Schwarzer `<guido.schwarzer@uniklinik-freiburg.de>`, Petra Graham `<pgraham@efs.mq.edu.au>`

References

Deeks JJ, Macaskill P, Irwig L (2005): The performance of tests of publication bias and other sample size effects in systematic reviews of diagnostic test accuracy was assessed. *Journal of Clinical Epidemiology*, **58**:882–93

Light RJ & Pillemer DB (1984): *Summing Up. The Science of Reviewing Research*. Cambridge: Harvard University Press

Peters JL, Sutton AJ, Jones DR, Abrams KR, Rushton L (2008): Contour-enhanced meta-analysis funnel plots help distinguish publication bias from other causes of asymmetry. *Journal of Clinical Epidemiology*, **61**, 991–6

Sterne JAC & Egger M (2001): Funnel plots for detecting bias in meta-analysis: Guidelines on choice of axis. *Journal of Clinical Epidemiology*, **54**, 1046–55

See Also

metabias, metabin, metagen, radial

Examples

```
data(Olkin1995)
m1 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, subset = c(41, 47, 51, 59),
  studlab = paste(author, year),
  sm = "RR", method = "I")

# Standard funnel plot
#
funnel(m1)

# Funnel plot with confidence intervals, common effect estimate and
# contours
#
fun <- funnel(m1, common = TRUE, level = 0.95, type = "contour")
legend("topleft", fun$text.contour, fill = fun$col.contour, bg = "white")

# Contour-enhanced funnel plot with user-chosen colours
#
funnel(m1, common = TRUE,
  level = 0.95, contour = c(0.9, 0.95, 0.99),
  col.contour = c("darkgreen", "green", "lightgreen"),
  lwd = 2, cex = 2, pch = 16, studlab = TRUE, cex.studlab = 1.25)
legend(0.05, 0.05,
  c("0.1 > p > 0.05", "0.05 > p > 0.01", "< 0.01"),
  fill = c("darkgreen", "green", "lightgreen"))

fun <- funnel(m1, common = TRUE,
  level = 0.95, contour = c(0.9, 0.95, 0.99),
  col.contour = c("darkgreen", "green", "lightgreen"),
  lwd = 2, cex = 2, pch = 16, studlab = TRUE, cex.studlab = 1.25)
legend(0.05, 0.05, fun$text.contour, fill = fun$col.contour)

# Use different colours for log risk ratios below and above 0
#
funnel(m1, bg = setvals(TE < 0, c("green", "red")))
```

gs *Get default for a meta-analysis setting.*

Description

Get default for a meta-analysis setting in R package **meta**.

Usage

```
gs(x = NULL, unname = NULL)
```

Arguments

x	A character string or vector with setting name(s).
unname	A logical whether to remove names from attributes.

Details

This function can be used to get the default for a meta-analysis setting defined using `settings.meta`.

This function is primarily used to define default settings in meta-analysis functions, e.g., `metabin` or `metacont`. A list of all arguments with current settings is printed using the command `settings.meta("print")`.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

`settings.meta`

Examples

```
# Get default setting for confidence interval of random effects
# model
#
gs("method.random.ci")

# Get default setting for summary measure in metabin()
#
gs("smbin")
```

JAMAlabels *Create study labels in JAMA layout (deprecated function)*

Description

Deprecated function to create study labels in JAMA layout (for forest plot). Replaced by `labels.meta`.

Usage

```
JAMAlabels(author, year, citation, data = NULL)
```

Arguments

`author`	A vector providing study authors.
`year`	A vector providing year of publication.
`citation`	A vector providing citation numbers.
`data`	An optional data frame containing the study information.

Details

This auxiliary function can be used to create study labels in JAMA layout which can be added to a forest plot using argument 'studlab'.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

labels.meta, forest.meta

Examples

```
data(Fleiss1993bin)

refs <- 20 + 1:7

Fleiss1993bin$mylabs <-
  JAMAlabels(study, year, refs, data = Fleiss1993bin)

m <- metabin(d.asp, n.asp, d.plac, n.plac, data = Fleiss1993bin,
  studlab = paste(study, year),
  sm = "OR", random = FALSE)

forest(m, studlab = mylabs, layout = "JAMA",
  fontfamily = "Times", fontsize = 10)
```

---

labbe.metabin *L'Abbé plot for meta-analysis with binary outcomes*

Description

Draw a L'Abbé plot for meta-analysis with binary outcomes.

Usage

```
## S3 method for class 'metabin'
labbe(
  x,
  xlim,
  ylim,
  xlab = NULL,
  ylab = NULL,
  TE.common = x$TE.common,
  TE.random = x$TE.random,
  common = x$common,
  random = x$random,
  backtransf = x$backtransf,
  axes = TRUE,
  pch = 21,
  text = NULL,
  cex = 1,
  col = "black",
  bg = "lightgray",
  lwd = 1,
  lwd.common = lwd,
  lwd.random = lwd,
  lty.common = 2,
  lty.random = 9,
  col.common = col,
  col.random = col,
  nulleffect = TRUE,
  lwd.nulleffect = lwd,
  col.nulleffect = "lightgray",
  sm = x$sm,
  weight,
  studlab = FALSE,
  cex.studlab = 0.8,
  pos.studlab = 2,
  label.e = x$label.e,
  label.c = x$label.c,
  warn.deprecated = gs("warn.deprecated"),
  TE.fixed,
  fixed,
  lwd.fixed,
  lty.fixed,
  col.fixed,
  ...
)

## Default S3 method:
labbe(
  x,
  y,
  xlim,
  ylim,
  xlab = NULL,
  ylab = NULL,
  TE.common = NULL,
  TE.random = NULL,
  common = !is.null(TE.common),
  random = !is.null(TE.random),
  backtransf = TRUE,
  axes = TRUE,
  pch = 21,
  text = NULL,
  cex = 1,
  col = "black",
  bg = "lightgray",
  lwd = 1,
  lwd.common = lwd,
  lwd.random = lwd,
  lty.common = 2,
  lty.random = 9,
  col.common = col,
  col.random = col,
  nulleffect = TRUE,
  lwd.nulleffect = lwd,
  col.nulleffect = "lightgray",
  sm = "",
  weight,
  studlab = FALSE,
  cex.studlab = 0.8,
  pos.studlab = 2,
  label.e = NULL,
  label.c = NULL,
  warn.deprecated = gs("warn.deprecated"),
  TE.fixed,
  fixed,
  lwd.fixed,
  lty.fixed,
  col.fixed,
  ...
)
```

Arguments

x	An object of class metabin. Alternatively, the x coordinates of points of the L'Abbé plot.
xlim	The x limits (min, max) of the plot.
ylim	The y limits (min, max) of the plot.

xlab A label for the x-axis.

ylab A label for the y-axis.

TE.common A numeric or vector specifying combined common effect estimate(s).

TE.random A numeric or vector specifying combined random effects estimate(s).

common A logical indicating whether the common effect estimate should be plotted.

random A logical indicating whether the random effects estimate should be plotted.

backtransf A logical indicating which values should be printed on x- and y-axis (see Details).

axes A logical indicating whether axes should be drawn on the plot.

pch The plotting symbol used for individual studies.

text A character vector specifying the text to be used instead of plotting symbol.

cex The magnification to be used for plotting symbol.

col A vector with colour of plotting symbols.

bg A vector with background colour of plotting symbols (only used if pch in 21:25).

lwd The line width.

lwd.common The line width(s) for common effect estimate(s) (if common is not NULL or FALSE).

lwd.random The line width(s) for random effects estimate(s) (if random is not NULL or FALSE).

lty.common Line type(s) for common effect estimate(s).

lty.random Line type(s) for random effects estimate(s).

col.common Colour of line(s) for common effect estimate(s).

col.random Colour of line(s) for random effects estimate(s).

nulleffect A logical indicating whether line for null effect should be added to the plot..

lwd.nulleffect Width of line for null effect.

col.nulleffect Colour of line for null effect.

sm A character string indicating underlying summary measure, i.e., "RD", "RR", "OR", or "ASD".

weight Either a numeric vector specifying relative sizes of plotting symbols or a character string indicating which type of plotting symbols is to be used for individual treatment estimates. One of missing (see Details), "same", "common", or "random", can be abbreviated. Plot symbols have the same size for all studies or represent study weights from common effect or random effects model.

studlab A logical indicating whether study labels should be printed in the graph. A vector with study labels can also be provided (must be of same length as x$event.e then).

cex.studlab Size of study labels.

pos.studlab Position of study labels, see argument pos in text.

label.e Label for experimental group.

label.c Label for control group.

`warn.deprecated`
A logical indicating whether warnings should be printed if deprecated arguments are used.

`TE.fixed` Deprecated argument (replaced by 'TE.common').

`fixed` Deprecated argument (replaced by 'common').

`lwd.fixed` Deprecated argument (replaced by 'lwd.common').

`lty.fixed` Deprecated argument (replaced by 'lty.common').

`col.fixed` Deprecated argument (replaced by 'col.common').

`...` Graphical arguments as in `par` may also be passed as arguments.

`y` The `y` coordinates of the L'Abbé plot, if argument `x` is not an object of class `metabin`.

Details

A L'Abbé plot is a scatter plot with the risk in the control group on the x-axis and the risk in the experimental group on the y-axis (L'Abbé et al., 1987). It can be used to evaluate heterogeneity in meta-analysis. Furthermore, this plot can aid to choose a summary measure (odds ratio, risk ratio, risk difference) that will result in more consistent results (Jiménez et al., 1997; Deeks, 2002).

If argument `backtransf` is TRUE (default), event probabilities will be printed on x- and y-axis. Otherwise, transformed event probabilities will be printed as defined by the summary measure, i.e., log odds of probabilities for odds ratio as summary measure (`sm = "OR"`), log probabilities for `sm = "RR"`, and arcsine-transformed probabilities for `sm = "ASD"`.

If `common` is TRUE, the estimate of the common effct model is plotted as a line. If `random` is TRUE, the estimate of the random effects model is plotted as a line.

Information from object `x` is utilised if argument `weight` is missing. Weights from the common effect model are used (`weight = "common"`) if argument `x$common` is `TRUE`; weights from the random effects model are used (`weight = "random"`) if argument `x$random` is `TRUE` and `x$common` is `FALSE`.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

References

Deeks JJ (2002): Issues in the selection of a summary statistic for meta-analysis of clinical trials with binary outcomes. *Statistics in Medicine*, **21**, 1575–600

Jiménez FJ, Guallar E, Martín-Moreno JM (1997): A graphical display useful for meta-analysis. *European Journal of Public Health*, **1**, 101–5

L'Abbé KA, Detsky AS, O'Rourke K (1987): Meta-analysis in clinical research. *Annals of Internal Medicine*, **107**, 224–33

See Also

`metabin`

Examples

```
data(Olkin1995)
m1 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, studlab = paste(author, year),
  sm = "RR", method = "I")

# L'Abbe plot for risk ratio
#
labbe(m1)

# L'Abbe plot for odds ratio
#
labbe(m1, sm = "OR")
# same plot
labbe(update(m1, sm = "OR"))

# L'Abbe plot for risk difference
#
labbe(m1, sm = "RD")

# L'Abbe plot on log odds scale
#
labbe(m1, sm = "OR", backtransf = FALSE)

# L'Abbe plot for odds ratio with coloured lines for various
# treatment effects (defined as log odds ratios)
#
mycols <- c("blue", "yellow", "green", "red", "green", "yellow", "blue")
labbe(m1, sm = "OR", random = FALSE,
  TE.common = log(c(1 / 10, 1 / 5, 1 / 2, 1, 2, 5, 10)),
  col.common = mycols, lwd.common = 2)

# L'Abbe plot on log odds scale with coloured lines for various
# treatment effects (defined as log odds ratios)
#
labbe(m1, sm = "OR", random = FALSE, backtransf = FALSE,
  TE.common = log(c(1 / 10, 1 / 5, 1 / 2, 1, 2, 5, 10)),
  col.common = mycols, lwd.common = 2)
```

labels.meta *Create study labels for forest plot*

Description

Create study labels for forest plot.

Usage

```
## S3 method for class 'meta'
labels(
  object,
  author = object$studlab,
  year = "",
  citation = NULL,
  layout = "JAMA",
  data = object$data,
  ...
)
```

Arguments

`object`	An object of class `meta`.
`author`	An optional vector providing study authors.
`year`	An optional vector providing year of publication.
`citation`	An optional vector providing citation numbers.
`layout`	A character string specifying layout. Either "JAMA" or "Lancet".
`data`	An optional data frame containing the study information.
`...`	Additional arguments (ignored at the moment).

Details

This auxiliary function can be used to create study labels in JAMA or Lancet layout which can be added to a forest plot using argument 'studlab'.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

`forest.meta`

Examples

```
data(Fleiss1993bin)

refs <- 20 + 1:7

m <- metabin(d.asp, n.asp, d.plac, n.plac, data = Fleiss1993bin,
 studlab = study, sm = "OR", random = FALSE)

forest(m,
  studlab = labels(m, year = year, citation = refs, layout = "JAMA"),
  layout = "JAMA", fontfamily = "Times", fontsize = 10)
```

```
forest(m,
  studlab = labels(m, year = year, citation = refs, layout = "Lancet"))
```

longarm *Transform data from pairwise comparisons to long arm-based format*

Description

This function transforms data from pairwise comparisons to a long arm-based format, i.e., two rows for a pairwise comparison.

Usage

```
longarm(
  treat1,
  treat2,
  event1,
  n1,
  event2,
  n2,
  mean1,
  sd1,
  mean2,
  sd2,
  time1,
  time2,
  data = NULL,
  studlab,
  id1 = NULL,
  id2 = NULL,
  append = TRUE,
  keep.duplicated = FALSE,
  keep.internal = FALSE
)
```

Arguments

treat1	Either label for first treatment or a meta-analysis or pairwise object (see Details).
treat2	Label for second treatment.
event1	Number of events (first treatment).
n1	Number of observations (first treatment).
event2	Number of events (second treatment).
n2	Number of observations (second treatment)
mean1	Estimated mean (first treatment).

`sd1` Standard deviation (first treatment).

`mean2` Estimated mean (second treatment).

`sd2` Standard deviation (second treatment).

`time1` Person time at risk (first treatment)

`time2` Person time at risk (second treatment)

`data` An optional data frame containing the study information.

`studlab` A vector with study labels (optional).

`id1` Last character(s) of variable names for additional variables with group specific information for first treatment.

`id2` Last character(s) of variable names for additional variables with group specific information for second treatment.

`append` A logical indicating if data frame provided in argument 'data' should be returned.

`keep.duplicated`
A logical indicating if duplicated rows should be returned (see Details).

`keep.internal` A logical indicating if variables generated internally should be returned (typically only relevant for data checking).

Details

This function transforms data given as one pairwise comparison per row to a long arm-based format with one row per treatment arm. The long arm-based format is, for example, the required input format for WinBUGS.

The function can be used to transform data with a binary, continuous or count outcome. The corresponding meta-analysis functions are `metabin`, `metacont` and `metainc`. Accordingly, a meta-analysis object created with one of these functions can be provided as argument `treat1`. It is also possible to use the longarm function with an R objected created with `pairwise`.

Otherwise, arguments `treat1` and `treat2` are mandatory to identify the individual treatments and, depending on the outcome, the following additional arguments are mandatory:

- event1, n1, event2, n2 (binary outcome);
- n1, mean1, sd1, n2, mean2, sd2 (continuous outcome);
- time1, n1, time2, n2 (count outcome).

Argument `studlab` must be provided if several pairwise comparisons come from a single study with more than two treatments.

The following variables will be returned:

studlab	study label
treat	treatment label
n	group sample size (count outcome only if provided)
events	number of events (binary or count outcome)
nonevents	number of non-events (binary outcome)
mean	estimated mean (continuous outcome)

sd	standard deviation (continuous outcome)
time	person time at risk (count outcome)

In addition, the data set provided in argument `data` will be returned if argument `append = TRUE` (default).

Argument `keep.duplicated` can be used to keep duplicated rows from the data set. Duplicated rows can occur, for example, in a three-arm study comparing treatments A and B with placebo. In this situation, the placebo arm will be returned twice in the data set in long arm-based format if `keep.duplicated = TRUE`. By default, duplicated rows with not be kept in the data set.

Value

A data frame in long arm-based format.

Note

R function `to.long` from R package **metafor** is called internally.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

See Also

`metabin`, `metacont`, `metainc`, `pairwise`

Examples

```
# Artificial example with three studies
m <- metabin(1:3, 100:102, 4:6, 200:202, studlab = LETTERS[1:3])
# Transform data to long arm-based format
longarm(m)
# Keep internal variables
longarm(m, keep.internal = TRUE)
```

`meta-object` *Description of R object of class "meta"*

Description

Detailed description of R objects of class "meta".

Details

The following R functions create an object of class "meta":

- metabin, metacont, metacor, metagen, metainc, metamean, metaprop, metarate, metacr, metamerge, trimfill

The following generic functions are available for an object of class "meta":

- as.data.frame.meta, labels.meta, print.meta, print.summary.meta, summary.meta, update.meta, weights.meta

An object of class "meta" is a list containing the following components.

studlab	Study labels
sm	Effect measure
null.effect	Effect under the null hypothesis
TE	Effect estimates (individual studies)
seTE	Standard error of effect estimates (individual studies)
statistic	Statistics for test of effect (individual studies)
pval	P-values for test of effect (individual studies)
df	Degrees of freedom (individual studies)
level	Level of confidence intervals for individual studies
lower	Lower confidence limits (individual studies)
upper	Upper confidence limits (individual studies)
three.level	Indicator variable for three-level meta-analysis model
cluster	Cluster variable (three-level meta-analysis model)
rho	Within-cluster correlation (three-level meta-analysis model)
k	Number of estimates combined in meta-analysis
k.study	Number of studies combined in meta-analysis
k.all	Number of all studies
k.TE	Number of studies with estimable effects
overall	Print meta-analysis results
overall.hetstat	Print overall heterogeneity statistics
common	Print results for common effect meta-analysis
random	Print results for random effects meta-analysis
prediction	Print prediction interval
backtransf	Back transform results in printouts and plots
method	Meta-analysis method (common effect model)
method.random	Meta-analysis method (random effects model)
w.common	Weights for common effect model (individual studies)
TE.common	Estimated overall effect (common effect model)
seTE.common	Standard error of overall effect (common effect model)
statistic.common	Statistic for test of overall effect (common effect model)
pval.common	P-value for test of overall effect (common effect model)
level.ma	Level of confidence interval for meta-analysis estimates
lower.common	Lower confidence limit (common effect model)
upper.common	Upper confidence limit (common effect model)
w.random	Weight for random effects model (individual studies)
TE.random	Estimated overall effect (random effects model)

`seTE.random`	Standard error of overall effect (random effects model)
`statistic.random`	Statistic for test of overall effect (random effects model)
`pval.random`	P-value for test of overall effect (random effects model)
`method.random.ci`	Confidence interval method (random effects model)
`df.random`	Degrees of freedom (random effects model)
`lower.random`	Lower confidence limit (random effects model)
`upper.random`	Upper confidence limit (random effects model)
`seTE.classic`	Standard error (classic random effects method)
`adhoc.hakn.ci`	*Ad hoc* correction for Hartung-Knapp method (confidence interval)
`df.hakn.ci`	Degrees of freedom for Hartung-Knapp method (if used in meta-analysis)
`seTE.hakn.ci`	Standard error for Hartung-Knapp method (not taking *ad hoc* variance correction into account)
`seTE.hakn.adhoc.ci`	Standard error for Hartung-Knapp method (taking *ad hoc* variance correction into account)
`df.kero`	Degrees of freedom for Kenward-Roger method (if used in meta-analysis)
`seTE.kero`	Standard error for Kenward-Roger method
`method.predict`	Method to calculate prediction interval
`adhoc.hakn.pi`	*Ad hoc* correction for Hartung-Knapp method (prediction interval)
`df.hakn.ci`	Degrees of freedom for Hartung-Knapp method (prediction interval)
`seTE.predict`	Standard error used to calculate prediction interval
`df.predict`	Degrees of freedom for prediction interval
`level.predict`	Level of prediction interval
`lower.predict`	Lower limit of prediction interval
`upper.predict`	Upper limit of prediction interval
`seTE.hakn.pi`	Standard error for Hartung-Knapp method (not taking *ad hoc* variance correction into account)
`seTE.hakn.adhoc.pi`	Standard error for Hartung-Knapp method (taking *ad hoc* variance correction into account)
`Q`	Heterogeneity statistic
`df.Q`	Degrees of freedom for heterogeneity statistic `Q`
`pval.Q`	P-value of heterogeneity test
`method.tau`	Method to estimate between-study variance τ^2
`control`	Additional arguments for iterative estimation of τ^2
`method.tau.ci`	Method for confidence interval of τ^2
`level.hetstat`	Level of confidence intervals for heterogeneity statistics
`tau2`	Between-study variance τ^2
`se.tau2`	Standard error of τ^2
`lower.tau2`	Lower confidence limit (τ^2)
`upper.tau2`	Upper confidence limit (τ^2)
`tau`	Square-root of between-study variance τ
`lower.tau`	Lower confidence limit (τ)
`upper.tau`	Upper confidence limit (τ)
`tau.preset`	Prespecified value for τ
`TE.tau`	Effect estimate used to estimate τ^2
`detail.tau`	Detail on between-study variance estimate

`phi`	Multiplicative heterogeneity parameter phi in penalised logistic regression
`H`	Heterogeneity statistic H
`lower.H`	Lower confidence limit (heterogeneity statistic H)
`upper.H`	Upper confidence limit (heterogeneity statistic H)
`I2`	Heterogeneity statistic I^2
`lower.I2`	Lower confidence limit (heterogeneity statistic I^2)
`upper.I2`	Upper confidence limit (heterogeneity statistic I^2)
`Rb`	Heterogeneity statistic R_b
`lower.Rb`	Lower confidence limit (heterogeneity statistic R_b)
`upper.Rb`	Upper confidence limit (heterogeneity statistic R_b)
`method.bias`	Method to test for funnel plot asymmetry
`text.common`	Label for common effect model
`text.random`	Label for random effects model
`text.predict`	Label for prediction interval
`text.w.common`	Label for weights (common effect model)
`text.w.random`	Label for weights (random effects model)
`title`	Title of meta-analysis / systematic review
`complab`	Comparison label
`outclab`	Outcome label
`label.e`	Label for experimental group
`label.c`	Label for control group
`label.left`	Graph label on left side of forest plot
`label.right`	Graph label on right side of forest plot
`keepdata`	Keep original data
`data`	Original data (set) used in function call (if `keepdata = TRUE`)
`subset`	Information on subset of original data used in meta-analysis (if `keepdata = TRUE`)
`exclude`	Studies excluded from meta-analysis
`warn`	Print warnings
`call`	Function call
`version`	Version of R package **meta** used to create object

For subgroup analysis (argument `subgroup`), the following additional components are added to the list.

`subgroup`	Subgroup information (for individual studies)
`subgroup.name`	Name of subgroup variable
`print.subgroup.name`	Print name of subgroup variable
`sep.subgroup`	Separator between name of subgroup variable and value
`test.subgroup`	Print test for subgroup differences
`prediction.subgroup`	Print prediction interval for subgroup(s)
`tau.common`	Assumption of common between-study variance in subgroups
`subgroup.levels`	Levels of grouping variable
`k.w`	Number of estimates combined in subgroups
`k.study.w`	Number of studies combined in subgroups
`k.all.w`	Number of studies in subgroups
`k.TE.w`	Number of studies with estimable effects in subgroups
`TE.common.w`	Estimated effect in subgroups (common effect model)

`seTE.common.w`	Standard error in subgroups (common effect model)
`statistic.common.w`	Statistic for test of effect in subgroups (common effect model)
`pval.common.w`	P-value for test of effect in subgroups (common effect model)
`lower.common.w`	Lower confidence limit in subgroups (common effect model)
`upper.common.w`	Upper confidence limit in subgroups (common effect model)
`w.common.w`	Total weight in subgroups (common effect model)
`TE.random.w`	Estimated effect in subgroups (random effect model)
`seTE.random.w`	Standard error in subgroups (random effects model)
`statistic.random.w`	Statistic for test of effect in subgroups (random effects model)
`pval.random.w`	P-value for test of effect in subgroups (random effects model)
`df.random.w`	Degrees of freedom in subgroups (random effects model)
`lower.random.w`	Lower confidence limit in subgroups (random effects model)
`upper.random.w`	Upper confidence limit in subgroups (random effects model)
`w.random.w`	Total weight in subgroups (random effects model)
`seTE.classic.w`	Standard error (classic random effects method)
`df.hakn.ci.w`	Degrees of freedom for Hartung-Knapp method in subgroups
`seTE.hakn.ci.w`	Standard error for Hartung-Knapp method in subgroups (not taking *ad hoc* variance correction into account)
`seTE.hakn.adhoc.ci.w`	Standard error for Hartung-Knapp method in subgroups
`df.kero.w`	Degrees of freedom for Kenward-Roger method in subgroups
`seTE.kero.w`	Standard error for Kenward-Roger method in subgroups
`seTE.predict.w`	Standard error for prediction interval in subgroups
`df.predict.w`	Degrees of freedom for prediction interval in subgroups
`lower.predict.w`	Lower limit of prediction interval in subgroups
`upper.predict.w`	Upper limit of prediction interval in subgroups
`seTE.hakn.pi.w`	Standard error for Hartung-Knapp method in subgroups (prediction intervals) (not taking *ad hoc* variance correction into account)
`seTE.hakn.adhoc.pi.w`	Standard error for Hartung-Knapp method in subgroups (prediction intervals)
`Q.w`	Heterogeneity statistic Q in subgroups
`pval.Q.w`	P-value for test of heterogeneity in subgroups
`tau2.w`	Between-study variance τ^2 in subgroups
`tau.w`	Square-root of between-study variance τ in subgroups
`H.w`	Heterogeneity statistic H in subgroups
`lower.H.w`	Lower confidence limit for H in subgroups
`upper.H.w`	Upper confidence limit for H in subgroups
`I2.w`	Heterogeneity statistic I^2 in subgroups
`lower.I2.w`	Lower confidence limit for I^2 in subgroups
`upper.I2.w`	Upper confidence limit for I^2 in subgroups
`Rb.w`	Heterogeneity statistic R_b in subgroups
`lower.Rb.w`	Lower confidence limit for R_b in subgroups
`upper.Rb.w`	Upper confidence limit for R_b in subgroups
`Q.w.common`	Within-group heterogeneity statistic Q (common effect model)
`Q.w.random`	Within-group heterogeneity statistic Q (random effects model) (only calculated if argument `tau.common = TRUE`)
`df.Q.w`	Degrees of freedom for `Q.w.common` and `Q.w.random`
`pval.Q.w.common`	P-value of test for residual heterogeneity (common effect model)
`pval.Q.w.random`	P-value of test for residual heterogeneity (random effects model)
`Q.b.common`	Between-groups heterogeneity statistic Q (common effect model)

`df.Q.b.common`	Degrees of freedom for `Q.b.common`
`pval.Q.b.common`	P-value of test for subgroup differences (common effect model)
`Q.b.random`	Between-groups heterogeneity statistic Q (random effects model)
`df.Q.b.random`	Degrees of freedom for `Q.b.random`
`pval.Q.b.random`	P-value of test for subgroup differences (random effects model)

An object created with `metabin` has the additional class `"metabin"` and the following components.

`event.e`	Events in experimental group (individual studies)
`n.e`	Sample size in experimental group (individual studies)
`event.e`	Events in control group (individual studies)
`n.e`	Sample size in control group (individual studies)
`incr`	Increment added to zero cells
`method.incr`	Continuity correction method
`sparse`	Continuity correction applied
`allstudies`	Include studies with double zeros
`doublezeros`	Indicator for studies with double zeros
`MH.exact`	Exact Mantel-Haenszel method
`RR.Cochrane`	Cochrane method to calculate risk ratio
`Q.Cochrane`	Cochrane method to calculate τ^2
`Q.CMH`	Cochran-Mantel-Haenszel statistic
`df.Q.CMH`	Degrees of freedom for `Q.CMH`
`pval.Q.CMH`	P-value of Cochran-Mantel-Haenszel test
`print.CMH`	Print results for Cochran-Mantel-Haenszel statistic
`incr.e`	Continuity correction in experimental group (individual studies)
`incr.c`	Continuity correction in control group (individual studies)
`k.MH`	Number of studies (Mantel-Haenszel method)

An object created with `metacont` has the additional class `"metacont"` and the following components.

`n.e`	Sample size in experimental group (individual studies)
`mean.e`	Estimated mean in experimental group (individual studies)
`sd.e`	Standard deviation in experimental group (individual studies)
`n.c`	Sample size in control group (individual studies)
`mean.c`	Estimated mean in control group (individual studies)
`sd.c`	Standard deviation in control group (individual studies)
`pooledvar`	Use pooled variance for mean difference
`method.smd`	Method for standardised mean difference (SMD)
`sd.glass`	Denominator in Glass' method
`exact.smd`	Use exact formulae for SMD
`method.ci`	Method to calculate confidence limits
`method.mean`	Method to approximate mean
`method.sd`	Method to approximate standard deviation

An object created with `metacor` has the additional class `"metacor"` and the following components.

cor	Correlation (individual studies)
n	Sample size (individual studies)

An object created with metainc has the additional class "metainc" and the following components.

event.e	Events in experimental group (individual studies)
time.e	Person time in experimental group (individual studies)
n.e	Sample size in experimental group (individual studies)
event.c	Events in control group (individual studies)
time.c	Person time in control group (individual studies)
n.c	Sample size in control group (individual studies)
incr	Increment added to zero cells
method.incr	Continuity correction method
sparse	Continuity correction applied
incr.e	Continuity correction in experimental group (individual studies)
incr.c	Continuity correction in control group (individual studies)
k.MH	Number of studies (Mantel-Haenszel method)

An object created with metamean has the additional class "metamean" and the following components.

n	Sample size (individual studies)
mean	Estimated mean (individual studies)
sd	Standard deviation (individual studies)
method.ci	Method to calculate confidence limits
method.mean	Method to approximate mean
method.sd	Method to approximate standard deviation

An object created with metaprop has the additional class "metaprop" and the following components.

event	Events (individual studies)
n	Sample size (individual studies)
incr	Increment added to zero cells
method.incr	Continuity correction method
sparse	Continuity correction applied
method.ci	Method to calculate confidence limits
incr.event	Continuity correction (individual studies)

An object created with metarate has the additional class "metarate" and the following components.

event	Events (individual studies)
time	Person time (individual studies)
n	Sample size (individual studies)
incr	Increment added to zero cells

method.incr	Continuity correction method
sparse	Continuity correction applied
method.ci	Method to calculate confidence limits
incr.event	Continuity correction (individual studies)

An object created with trimfill has the additional classes "trimfill" and "metagen" and the following components.

k0	Number of added studies
left	Studies missing on left side
ma.common	Use common effect or random effects model to estimate number of missing studies
type	Method to estimate missing studies
n.iter.max	Maximum number of iterations
n.iter	Number of iterations
trimfill	Filled studies (individual studies)
class.x	Primary class of meta-analysis object

An object created with metamerge has the additional class "metamerge". Furthermore, the following components have a different meaning:

k	Vector with number of estimates
k.study	Vector with number of studies
k.all	Vector with total number of studies
k.TE	Vector with number of studies with estimable effects
k.MH	Vector with number of studies combined with Mantel-Haenszel method
TE.common	Vector with common effect estimates
seTE.common	Vector with standard errors of common effect estimates
lower.common	Vector with lower confidence limits (common effect model)
upper.common	Vector with upper confidence limits (common effect model)
statistic.common	Vector with test statistics for test of overall effect (common effect model)
pval.common	Vector with p-value of test for overall effect (common effect model)
TE.random	Vector with random effects estimates
seTE.random	Vector with standard errors of random effects estimates
lower.random	Vector with lower confidence limits (random effects model)
upper.random	Vector with upper confidence limits (random effects model)
statistic.random	Vector with test statistics for test of overall effect (random effects model)
pval.random	Vector with p-value of test for overall effect (random effects model)
w.common	Vector or matrix with common effect weights
w.random	Vector or matrix with random effects weights

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

meta-package, meta-sm, print.meta, summary.meta, forest.meta

meta-sm *Description of summary measures available in R package **meta***

Description

Description of summary measures available in R package **meta**

Details

The following summary measures (argument sm) are recognised in R package **meta**.

Meta-analysis of binary outcome data (metabin):

Argument	**Summary measure**
sm = "OR"	Odds ratio (Fleiss, 1993)
sm = "RR"	Risk ratio (Fleiss, 1993)
sm = "RD"	Risk difference (Fleiss, 1993)
sm = "ASD"	Arcsine difference (Rücker et al., 2009)
sm = "DOR"	Diagnostic odds ratio (Moses et al., 1993)
sm = "VE"	Vaccine efficacy or vaccine effectiveness

Note, mathematically, odds ratios and diagnostic odds ratios are identical, however, the labels in printouts and figures differ. Furthermore, log risk ratio (logRR) and log vaccine ratio (logVR) are mathematical identical, however, back-transformed results differ as vaccine efficacy or effectiveness is defined as VE = 100 * (1 - RR).

A continuity correction is used for some summary measures in the case of a zero cell count (see metabin).

List elements TE, TE.common, TE.random, etc., contain transformed values, e.g., log odds ratios, log risk ratios or log vaccine ratios. In printouts and plots transformed values are back transformed if argument backtransf = TRUE (default), with exception of the arcsine difference where no backtransformation exists. Auxiliary function logVR2VE is used to back-transform log vaccine ratios to vaccine efficacy or effectiveness while exp is used to back-transform log odds or risk ratios.

Meta-analysis of continuous outcome data (metacont):

Argument	**Summary measure**
sm = "MD"	Mean difference
sm = "SMD"	Standardised mean difference
sm = "ROM"	Ratio of means

Three variants to calculate the standardised mean difference are available (see metacont).

For the ratio of means, list elements TE, TE.common, TE.random, etc., contain the log transformed ratio of means. In printouts and plots these values are back transformed using exp if argument backtransf = TRUE (default).

Meta-analysis of correlations (metacor):

Argument	Summary measure
sm = "ZCOR"	Fisher's z transformed correlation
sm = "COR"	Untransformed correlations

For Fisher's z transformed correlations, list elements TE, TE.common, TE.random, etc., contain the transformed correlations. In printouts and plots these values are back transformed using auxiliary function z2cor if argument backtransf = TRUE (default).

Meta-analysis of incidence rates (metainc):

Argument	Summary measure
sm = "IRR"	Incidence rate ratio
sm = "IRD"	Incidence rate difference
sm = "IRSD"	Square root transformed incidence rate difference
sm = "VE"	Vaccine efficacy or vaccine effectiveness

Note, log incidence rate ratio (logIRR) and log vaccine ratio (logVR) are mathematical identical, however, back-transformed results differ as vaccine efficacy or effectiveness is defined as VE = 100 * (1 - IRR).

List elements TE, TE.common, TE.random, etc., contain the transformed incidence rates. In printouts and plots these values are back transformed if argument backtransf = TRUE (default). For back-transformation, exp is used for the incidence rate ratio, power of 2 is used for square root transformed rates and logVR2VE is used for vaccine efficacy / effectiveness.

Meta-analysis of single means (metamean):

Argument	Summary measure
sm = "MRAW"	Raw, i.e. untransformed, means
sm = "MLN"	Log transformed means

Calculations are conducted on the log scale if sm = "MLN". Accordingly, list elements TE, TE.common, and TE.random contain the logarithm of means. In printouts and plots these values are back transformed using exp if argument backtransf = TRUE.

Meta-analysis of single proportions (metaprop):

The following transformations of proportions are implemented to calculate an overall proportion:

Argument	Summary measure
sm = "PLOGIT"	Logit transformation
sm = "PAS"	Arcsine transformation
sm = "PFT"	Freeman-Tukey Double arcsine transformation
sm = "PLN"	Log transformation
sm = "PRAW"	No transformation

List elements TE, TE.common, TE.random, etc., contain the transformed proportions. In printouts and plots these values are back transformed if argument backtransf = TRUE (default). For back-transformation, logit2p is used for logit transformed proportions, asin2p is used for (Freeman-Tukey) arcsine transformed proportions and exp is used for log transformed proportions.

Meta-analysis of single rates (metarate):
The following transformations of incidence rates are implemented to calculate an overall rate:

Argument	Summary measure
sm = "IRLN"	Log transformation
sm = "IRS"	Square root transformation
sm = "IRFT"	Freeman-Tukey Double arcsine transformation
sm = "IR"	No transformation

List elements TE, TE.common, TE.random, etc., contain the transformed incidence rates. In printouts and plots these values are back transformed if argument backtransf = TRUE (default). For back-transformation, exp is used for log transformed rates, power of 2 is used for square root transformed rates and asin2ir is used for Freeman-Tukey arcsine transformed rates.

Generic inverse variance method (metagen):
The following summary measures are recognised in addition to the above mentioned summary measures:

Argument	Summary measure
sm = "HR"	Hazard ratio
sm = "VE"	Vaccine efficacy or vaccine effectiveness

List elements TE, TE.common, TE.random, etc., contain transformed values, i.e., log hazard ratios and log vaccine ratios. In printouts and plots these values are back transformed if argument backtransf = TRUE (default); see also meta-transf.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Borenstein M, Hedges LV, Higgins JP, Rothstein HR (2010): A basic introduction to fixed-effect and random-effects models for meta-analysis. *Research Synthesis Methods*, **1**, 97–111

Fleiss JL (1993): The statistical basis of meta-analysis. *Statistical Methods in Medical Research*, **2**, 121–45

Moses LE, Shapiro D, Littenberg B (1993): Combining Independent Studies of a Diagnostic Test into a Summary Roc Curve: Data-Analytic Approaches and Some Additional Considerations. *Statistics in Medicine*, **12**, 1293–1316

Rücker G, Schwarzer G, Carpenter J, Olkin I (2009): Why add anything to nothing? The arcsine difference as a measure of treatment effect in meta-analysis with zero cells. *Statistics in Medicine*, **28**, 721–38

Stijnen T, Hamza TH, Ozdemir P (2010): Random effects meta-analysis of event outcome in the framework of the generalized linear mixed model with applications in sparse data. *Statistics in Medicine*, **29**, 3046–67

See Also

meta-package, meta-transf, meta-object, print.meta, summary.meta, forest.meta

meta-transf *Auxiliary functions for (back) transformations*

Description

Auxiliary functions to (back) transform effect estimates or confidence / prediction interval limit(s).

Usage

```
transf(x, sm, func = NULL, args = NULL)

cor2z(x)

p2asin(x)

p2logit(x)

VE2logVR(x)

backtransf(x, sm, n, time, func = NULL, args = NULL)

asin2ir(x, time = NULL)

asin2p(x, n = NULL)

logit2p(x)

logVR2VE(x)

z2cor(x)
```

Arguments

x	Numerical vector with effect estimates, lower or upper confidence / prediction interval limit(s).
sm	Summary measure.
func	User-specified function for (back) transformation.
args	Function arguments for user-specified function.
n	Sample size(s) to back transform Freeman-Tukey transformed proportions.
time	Time(s) to back transform Freeman-Tukey transformed incidence rates.

Details

Often in a meta-analysis, effect estimates are transformed before calculating a weighted average. For example, the log odds ratio and its standard error is used instead of the odds ratio in R function metagen. To report the results of a meta-analysis, effect estimates are typically back transformed to the original scale. R package **meta** provides some auxiliary functions for (back) transformations.

Transformations: The following auxiliary functions are provided by R package **meta** to transform effect estimates or confidence / prediction interval limits.

Function	Transformation
cor2z	Correlations to Fisher's Z transformed correlations
p2logit	Proportions to logit transformed proportions
p2asin	Proportions to arcsine transformed proportions
VE2logVR	Vaccine efficacy / effectiveness to log vaccine ratio

Note, no function for the Freeman-Tukey arcsine transformation is provided as this transformation is based on the number of events and sample sizes instead of the effect estimates.

R function transf is a wrapper function for the above and additional transformations, e.g., the log transformation using log for odds or risk ratios. Argument sm is mandatory to specify the requested transformation. It is also possible to specify a different function with arguments func and args.

Back transformations: The following auxiliary functions are available to back transform effect estimates or confidence / prediction interval limits.

Function	Transformation
asin2ir	Freeman-Tukey arcsine transformed rates to rates
asin2p	(Freeman-Tukey) arcsine transformed proportions to proportions
logit2p	Logit transformed proportions to proportions
logVR2VE	Log vaccine ratio to vaccine efficacy / effectiveness
z2cor	Fisher's Z transformed correlations to correlations

Argument time is mandatory in R function asin2ir.

If argument n is provided in R function asin2p, Freeman-Tukey arcsine transformed proportions are back transformed. Otherwise, arcsine transformed proportions are back transformed.

R function backtransf is a wrapper function for the above and additional transformations, e.g., the exponential transformation using exp for log odds or log risk ratios. Argument sm is mandatory to specify the requested transformation. For the Freeman-Tukey transformations, argument n or time is mandatory.

It is also possible to specify a different function with arguments func and args.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

meta-sm

Examples

```
logit2p(p2logit(0.5))
```

metaadd *Add pooled results from external analysis to meta-analysis*

Description

Add pooled results from external analysis to an existing meta-analysis object. This is useful, for example, to add results from a Bayesian meta-analysis which is not implemented in R package **meta**.

Usage

```
metaadd(
  x,
  type,
  TE,
  lower,
  upper,
  statistic = NA,
  pval = NA,
  text,
  data = NULL,
  method.common = "",
  method.random = "",
  method.tau = "",
  method.random.ci = "",
  method.predict = "",
  transf = gs("transf")
)
```

Arguments

`x`	Meta-analysis object.
`type`	A character string or vector indicating whether added results are from common effect, random effects model or prediction interval. Either `"common"`, `"random"` or `"prediction"`, can be abbreviated.
`TE`	Pooled estimate(s).
`lower`	Lower limit(s) of confidence or prediction interval.
`upper`	Upper limit(s) of confidence or prediction interval.
`statistic`	Test statistic(s).
`pval`	P-value(s).
`text`	A character string or vector used in printouts and forest plot to label the added results.
`data`	An optional data frame containing the new results or an object of class `meta`.
`method.common`	A character string or vector to describe the common effect method(s).

method.random A character string or vector to describe the random effects method(s).

method.tau A character string or vector to describe the estimator(s) of the between-study variance.

method.random.ci
A character string or vector to describe the method(s) to calculate confidence intervals under the random effects model.

method.predict A character string or vector to describe the method(s) used for prediction intervals.

transf A logical indicating whether inputs for arguments TE, lower and upper are already appropriately transformed to conduct the meta-analysis or on the original scale. If transf = TRUE (default), inputs are expected to be log odds ratios instead of odds ratios for sm = "OR" and Fisher's z transformed correlations instead of correlations for sm = "ZCOR", for example.

Details

In R package **meta**, objects of class "meta" contain results of both common effect and random effects meta-analyses. This function enables the user to add the pooled results of an additional analysis to an existing meta-analysis object. This is useful, for example, to add the result of a Bayesian meta-analysis.

If argument data is a meta-analysis object created with R package **meta**, arguments TE, lower, upper, statistic and pval are ignored as this information is extracted from the meta-analysis.

Otherwise, arguments TE, lower and upper have to be provided if type = "common" or type = "random". For type = "prediction", only arguments lower and upper are mandatory.

Note, R function metamerge can be used to add meta-analysis results of another meta-analysis object (see meta-object).

Value

An object of class "meta" with corresponding generic functions (see meta-object).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

metagen, metamerge

Examples

```
data(Fleiss1993bin)

# Common effect and random effects meta-analysis
m1 <- metabin(d.asp, n.asp, d.plac, n.plac, data = Fleiss1993bin,
  studlab = paste(study, year), sm = "OR")

# Naive pooling
```

```
m2 <- metabin(sum(d.asp), sum(n.asp), sum(d.plac), sum(n.plac),
  data = Fleiss1993bin, sm = "OR", text.common = "Naive pooling")

# Add results of second meta-analysis from common effect model
m12 <- metaadd(m1, data = m2, method.common = "Naive pooling")
m12

forest(m12)
```

metabias.meta *Test for funnel plot asymmetry*

Description

Test for funnel plot asymmetry, based on rank correlation or linear regression method.

Usage

```
## S3 method for class 'meta'
metabias(
  x,
  method.bias = x$method.bias,
  plotit = FALSE,
  correct = FALSE,
  k.min = 10,
  ...
)

## S3 method for class 'metabias'
print(
  x,
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = max(gs("digits.pval"), 2),
  digits.se = gs("digits.se"),
  digits.tau2 = gs("digits.tau2"),
  scientific.pval = gs("scientific.pval"),
  big.mark = gs("big.mark"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  text.tau2 = gs("text.tau2"),
  details.methods = gs("details"),
  ...
)

metabias(x, ...)
```

```
## Default S3 method:
metabias(
  x,
  seTE,
  method.bias = "Egger",
  plotit = FALSE,
  correct = FALSE,
  k.min = 10,
  ...
)
```

Arguments

`x` An object of class `meta` or estimated treatment effect in individual studies.

`method.bias` A character string indicating which test is to be used (see Details), can be abbreviated.

`plotit` A logical indicating whether a plot should be produced (see Details).

`correct` A logical indicating whether a continuity corrected statistic is used for rank correlation tests.

`k.min` Minimum number of studies to perform test for funnel plot asymmetry.

`...` Additional arguments passed on to `rma.uni`.

`digits` Minimal number of significant digits for estimates, see `print.default`.

`digits.stat` Minimal number of significant digits for z- or t-value of test for test of funnel plot asymmetry, see `print.default`.

`digits.pval` Minimal number of significant digits for p-value of test for test of funnel plot asymmetry, see `print.default`.

`digits.se` Minimal number of significant digits for standard errors, see `print.default`.

`digits.tau2` Minimal number of significant digits for residual heterogeneity variance, see `print.default`.

`scientific.pval` A logical specifying whether p-values should be printed in scientific notation, e.g., 1.2345e-01 instead of 0.12345.

`big.mark` A character used as thousands separator.

`zero.pval` A logical specifying whether p-values should be printed with a leading zero.

`JAMA.pval` A logical specifying whether p-values for test of overall effect should be printed according to JAMA reporting standards.

`text.tau2` Text printed to identify residual heterogeneity variance τ^2.

`details.methods` A logical specifying whether details on statistical methods should be printed.

`seTE` Standard error of estimated treatment effect (mandatory if `x` not of class `meta`).

Details

Functions to conduct rank correlation or linear regression tests for funnel plot asymmetry.

Classic generic tests: The following tests are generic tests for funnel plot asymmetry which only require estimates of the treatment effect and corresponding standard errors. Accordingly, these are the only tests provided by R function `metabias.default`.

If argument `method.bias` is `"Begg"`, the test statistic is based on the rank correlation between standardised treatment estimates and variance estimates of estimated treatment effects; Kendall's tau is used as correlation measure (Begg & Mazumdar, 1994). The test statistic follows a standard normal distribution. By default (if `correct` is FALSE), no continuity correction is utilised (Kendall & Gibbons, 1990).

If argument `method.bias` is `"Egger"`, the test statistic is based on a weighted linear regression of the treatment effect on its standard error (Egger et al., 1997). The test statistic follows a t distribution with `number of studies -2` degrees of freedom.

If argument `method.bias` is `"Thompson"`, the test statistic is based on a weighted linear regression of the treatment effect on its standard error using an additive between-study variance component denoted as methods (3a) - (3d) in Thompson & Sharp (1999). The test statistic follows a t distribution with `number of studies - 2` degrees of freedom.

Tests for meta-analysis with binary outcomes: The following tests for funnel plot asymmetry are only available for meta-analyses comparing two binary outcomes, i.e. meta-analyses generated with the `metabin` function. The only exception is the test by Peters et al. (2006) which can also be used in a meta-analysis of single proportions generated with `metaprop`.

If argument `method.bias` is `"Harbord"`, the test statistic is based on a weighted linear regression utilising efficient score and score variance (Harbord et al., 2006, 2009). The test statistic follows a t distribution with `number of studies - 2` degrees of freedom.

In order to calculate an arcsine test for funnel plot asymmetry (Rücker et al., 2008), one has to use the `metabin` function with argument `sm = "ASD"` as input to the `metabias` command. The three arcsine tests described in Rücker et al. (2008) can be calculated by setting `method.bias` to `"Begg"`, `"Egger"` and `"Thompson"`, respectively.

If argument `method.bias` is `"Macaskill"`, the test statistic is based on a weighted linear regression of the treatment effect on the total sample size with weights reciprocal to the variance of the average event probability (Macaskill et al., 2001, *method FPV*). The test statistic follows a t distribution with `number of studies - 2` degrees of freedom.

If argument `method.bias` is `"Peters"`, the test statistic is based on a weighted linear regression of the treatment effect on the inverse of the total sample size with weights reciprocal to the variance of the average event probability (Peters et al., 2006). The test statistic follows a t distribution with `number of studies - 2` degrees of freedom. Note, this test is a variant of Macaskill et al. (2001), *method FPV*, using the inverse sample size as covariate.

If argument `method.bias` is `"Schwarzer"`, the test statistic is based on the rank correlation between a standardised cell frequency and the inverse of the variance of the cell frequency; Kendall's tau is used as correlation measure (Schwarzer et al., 2007). The test statistic follows a standard normal distribution. By default (if `correct` is FALSE), no continuity correction is utilised (Kendall & Gibbons, 1990).

Finally, for meta-analysis of diagnostic test accuracy studies, if argument `method.bias` is `"Deeks"`, the test statistic is based on a weighted linear regression of the log diagnostic odds ratio on the inverse of the squared effective sample size using the effective sample size as weights (Deeks et al., 2005). The test statistic follows a t distribution with `number of studies -2` degrees of freedom.

Test for the standardised mean difference: If argument `method.bias` is `"Pustejovsky"`, the test statistic is based on a weighted linear regression of the treatment effect on the square root of the sum of the inverse group sample sizes using the treatment effect variance as weights (Pustejovsky & Rodgers, 2019). The test statistic follows a t distribution with `number of studies - 2` degrees of freedom.

Recommendations and default settings: Following recommendations by Sterne et al. (2011), by default, a test for funnel plot asymmetry is only conducted if the number of studies is ten or larger (argument `k.min = 10`). This behaviour can be changed by setting a smaller value for argument `k.min`. Note, the minimum number of studies is three.

If argument `method.bias` is missing, the Harbord test (`method.bias = "Harbord"`) is used in meta-analyses with a binary outcome for the odds ratio and Deeks' test (`method.bias = "Deeks"`) for the diagnostic odds ratios. In all other settings, the Egger test (`method.bias = "Egger"`) is used (Sterne et al., 2011).

No test for funnel plot asymmetry is conducted in meta-analyses with subgroups.

If argument `plotit = TRUE`, a scatter plot is shown if argument `method.bias` is equal to `"Begg"`, `"Egger"`, `"Thompson"`, `"Harbord"`, or `"Deeks"`.

Value

A list with class `metabias` containing the following components if a test for funnel plot asymmetry is conducted:

`statistic` Test statistic.

`df` The degrees of freedom of the test statistic in the case that it follows a t distribution.

`pval` The p-value for the test.

`estimate` Estimates used to calculate test statisic.

`method` A character string indicating what type of test was used.

`title` Title of Cochrane review.

`complab` Comparison label.

`outclab` Outcome label.

`var.model` A character string indicating whether none, multiplicative, or additive residual heterogeneity variance was assumed.

`method.bias` As defined above.

`x` Meta-analysis object.

`version` Version of R package **meta** used to create object.

Or a list with the following elements if test is not conducted due to the number of studies:

`k` Number of studies in meta-analysis.

`k.min` Minimum number of studies to perform test for funnel plot asymmetry.

`version` Version of R package **meta** used to create object.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Begg CB & Mazumdar M (1994): Operating characteristics of a rank correlation test for publication bias. *Biometrics*, **50**, 1088–101

Deeks JJ, Macaskill P, Irwig L (2005): The performance of tests of publication bias and other sample size effects in systematic reviews of diagnostic test accuracy was assessed. *Journal of Clinical Epidemiology*, **58**:882–93

Egger M, Smith GD, Schneider M & Minder C (1997): Bias in meta-analysis detected by a simple, graphical test. *British Medical Journal*, **315**, 629–34

Harbord RM, Egger M & Sterne J (2006): A modified test for small-study effects in meta-analyses of controlled trials with binary endpoints. *Statistics in Medicine*, **25**, 3443–57

Harbord RM, Harris RJ, Sterne JAC (2009): Updated tests for small-study effects in meta–analyses. *The Stata Journal*, **9**, 197–210

Kendall M & Gibbons JD (1990): *Rank Correlation Methods*. London: Edward Arnold

Macaskill P, Walter SD, Irwig L (2001): A comparison of methods to detect publication bias in meta-analysis. *Statistics in Medicine*, **20**, 641–54

Peters JL, Sutton AJ, Jones DR, Abrams KR & Rushton L (2006): Comparison of two methods to detect publication bias in meta-analysis. *Journal of the American Medical Association*, **295**, 676–80

Pustejovsky JE, Rodgers MA (2019): Testing for funnel plot asymmetry of standardized mean differences. *Research Synthesis Methods*, **10**, 57–71

Rücker G, Schwarzer G, Carpenter JR (2008): Arcsine test for publication bias in meta-analyses with binary outcomes. *Statistics in Medicine*, **27**, 746–63

Schwarzer G, Antes G & Schumacher M (2007): A test for publication bias in meta-analysis with sparse binary data. *Statistics in Medicine*, **26**, 721–33

Sterne, JAC et al. (2011): Recommendations for examining and interpreting funnel plot asymmetry in meta-analyses of randomised controlled trials. *BMJ (Clinical research ed.)*, **343**, 1

Thompson SG & Sharp, SJ (1999): Explaining heterogeneity in meta-analysis: a comparison of methods, *Statistics in Medicine*, **18**, 2693–708

See Also

`funnel`, `funnel.meta`, `metabin`, `metacont`, `metagen`

Examples

```
data(Olkin1995)
m1 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, subset = 1:10, sm = "RR", method = "I")

metabias(m1)
metabias(m1, plotit = TRUE)
metabias(m1, method.bias = "Begg")
metabias(m1, method.bias = "Begg", correct = TRUE)

metabias(m1, method.bias = "Schwarzer")
metabias(m1, method.bias = "Egger")$pval

# Arcsine test (based on linear regression)
#
m1.as <- update(m1, sm = "ASD")
metabias(m1.as)
# Same result (using function metabias.default)
metabias(m1.as$TE, m1.as$seTE)

# No test for funnel plot asymmetry calculated
#
m2 <- update(m1, subset = 1:5)
metabias(m2)

m3 <- update(m1, subset = 1:2)
metabias(m3)

# Test for funnel plot asymmetry calculated (use of argument k.min)
#
metabias(m2, k.min = 5)
```

---

metabias.rm5 *Cochrane review: Test for funnel plot asymmetry*

Description

Conduct a test for funnel plot asymmetry for all outcomes in a Cochrane review of intervention studies

Usage

```
## S3 method for class 'rm5'
metabias(
  x,
  comp.no,
  outcome.no,
  method.bias = "linreg",
  method.bias.binary = method.bias,
  method.bias.or = "score",
  k.min = 10,
  ...
)

## S3 method for class 'cdir'
```

```
metabias(
  x,
  comp.no,
  outcome.no,
  method.bias = "linreg",
  method.bias.binary = method.bias,
  method.bias.or = "score",
  k.min = 10,
  ...
)
```

Arguments

`x` An object of class `rm5` or `cdir`.

`comp.no` Comparison number.

`outcome.no` Outcome number.

`method.bias` A character string indicating which test for small-study effects is to be used for all outcomes. Either `"rank"`, `"linreg"`, or `"mm"`, can be abbreviated. See function `metabias`

`method.bias.binary` A character string indicating which test is to be used for binary outcomes. Either `"rank"`, `"linreg"`, `"mm"`, `"count"`, `"score"`, or `"peters"`, can be abbreviated. See function `metabias`

`method.bias.or` A character string indicating which test is to be used for binary outcomes with odds ratio as summary measure. Either `"rank"`, `"linreg"`, `"mm"`, `"count"`, `"score"`, or `"peters"`, can be abbreviated. See function `metabias`

`k.min` Minimum number of studies to perform test for small-study effects.

`...` Additional arguments (ignored at the moment)

Details

This function can be used to conduct a test for funnel plot asymmetry for all or selected meta-analyses in a Cochrane review of intervention studies (Higgins et al, 2023).

The R function `metacr` is called internally.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Higgins JPT, Thomas J, Chandler J, Cumpston M, Li T, Page MJ, Welch VA (editors) (2023): *Cochrane Handbook for Systematic Reviews of Interventions Version 6.4 (updated August 2023)*. Available from https://training.cochrane.org/handbook/

See Also

`metabias`, `metacr`, `read.rm5`, `read.cdir`, `summary.rm5`, `summary.cdir`

Examples

```
# Locate export data file "Fleiss1993_CR.csv" in sub-directory of
# package "meta"
#
filename <- system.file("extdata", "Fleiss1993_CR.csv", package = "meta")
Fleiss1993_CR <- read.rm5(filename)

# Print results for all tests of small-study effects
#
metabias(Fleiss1993_CR, k.min = 5)

# Print result of test of small-study effects for second outcome in
# first comparison
#
metabias(Fleiss1993_CR, comp.no = 1, outcome.no = 2, k.min = 5)
```

metabin *Meta-analysis of binary outcome data*

Description

Calculation of common effect and random effects estimates (risk ratio, odds ratio, risk difference, arcsine difference, or diagnostic odds ratio) for meta-analyses with binary outcome data. Mantel-Haenszel, inverse variance, Peto method, generalised linear mixed model (GLMM), logistic regression with penalised likelihood and sample size method are available for pooling. For GLMMs, the rma.glmm function from R package **metafor** (Viechtbauer, 2010) is called internally. For penalised logistic regression, R package **brglm2** must be available.

Usage

```
metabin(
  event.e,
  n.e,
  event.c,
  n.c,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  cluster = NULL,
  rho = 0,
  method = ifelse(tau.common, "Inverse", gs("method")),
 sm = ifelse(!is.na(charmatch(tolower(method), c("peto", "glmm", "lrp", "ssw"), nomatch
    = NA)), "OR", gs("smbin")),
  incr = gs("incr"),
  method.incr = gs("method.incr"),
  allstudies = gs("allstudies"),
  incr.e = if (length(incr) > 1) incr else NULL,
  incr.c = if (length(incr) > 1) incr else NULL,
  level = gs("level"),
  MH.exact = gs("MH.exact"),
  RR.Cochrane = gs("RR.Cochrane"),
  Q.Cochrane = gs("Q.Cochrane") & method == "MH" & method.tau == "DL",
  model.glmm = gs("model.glmm"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  overall = common | random,
  overall.hetstat = if (is.null(gs("overall.hetstat"))) common | random else
    gs("overall.hetstat"),
  prediction = gs("prediction") | !missing(method.predict),
  method.tau,
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.preset = NULL,
  TE.tau = NULL,
  tau.common = gs("tau.common"),
  detail.tau = NULL,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  method.bias = ifelse(sm == "OR", "Harbord", ifelse(sm == "DOR", "Deeks",
    gs("method.bias"))),
  backtransf = gs("backtransf"),
  pscale = 1,
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title = gs("title"),
  complab = gs("complab"),
  outclab = "",
  label.e = gs("label.e"),
  label.c = gs("label.c"),
  label.left = gs("label.left"),
  label.right = gs("label.right"),
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  subgroup,
  subgroup.name = NULL,
  print.subgroup.name = gs("print.subgroup.name"),
  sep.subgroup = gs("sep.subgroup"),
  test.subgroup = gs("test.subgroup"),
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  byvar,
  hakn,
  adhoc.hakn,
  print.CMH = gs("print.CMH"),
  keepdata = gs("keepdata"),
  warn = gs("warn"),
  warn.deprecated = gs("warn.deprecated"),
  control = NULL,
  ...
)
```

Arguments

`event.e` Number of events in experimental group, or true positives in diagnostic study, or an R object created with `pairwise`.

`n.e` Number of observations in experimental group or number of ill participants in diagnostic study.

`event.c` Number of events in control group or false positives in diagnostic study.

`n.c` Number of observations in control group or number of healthy participants in diagnostic study.

`studlab` An optional vector with study labels.

`data` An optional data frame containing the study information, i.e., event.e, n.e, event.c, and n.c.

`subset` An optional vector specifying a subset of studies to be used.

`exclude` An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots.

`cluster` An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.

`rho` Assumed correlation of estimates within a cluster.

`method` A character string indicating which method is to be used for pooling of studies. One of `"Inverse"`, `"MH"`, `"Peto"`, `"GLMM"`, `"LRP"`, or `"SSW"`, can be abbreviated.

`sm` A character string indicating which summary measure (`"RR"`, `"OR"`, `"RD"`, `"ASD"`, `"DOR"`, or `"VE"`) is to be used for pooling of studies, see Details.

`incr` Could be either a numerical value which is added to cell frequencies for studies with a zero cell count, the character string `"TACC"` which stands for treatment arm continuity correction, or a numeric vector with the continuity correction for each study, see Details.

`method.incr` A character string indicating which continuity correction method should be used (`"only0"`, `"if0all"`, `"all"`, or `"user"`), see Details.

`allstudies` A logical indicating if studies with zero or all events in both groups are to be included in the meta-analysis (applies only if `sm` is equal to `"RR"`, `"OR"`, or `"DOR"`).

`incr.e` Continuity correction in experimental group, see Details.

`incr.c` Continuity correction in control group, see Details.

`level` The level used to calculate confidence intervals for individual studies.

`MH.exact` A logical indicating if `incr` is not to be added to cell frequencies for studies with a zero cell count to calculate the pooled estimate based on the Mantel-Haenszel method.

`RR.Cochrane` A logical indicating if 2*`incr` instead of 1*`incr` is to be added to `n.e` and `n.c` in the calculation of the risk ratio (i.e., `sm="RR"`) for studies with a zero cell. This is used in RevMan 5, the program for preparing and maintaining Cochrane reviews.

`Q.Cochrane` A logical indicating if the Mantel-Haenszel estimate is used in the calculation of the heterogeneity statistic Q which is implemented in RevMan 5, the program for preparing and maintaining Cochrane reviews.

`model.glmm` A character string indicating which GLMM should be used. One of `"UM.FS"`, `"UM.RS"`, `"CM.EL"`, and `"CM.AL"`, see Details.

`common` A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`overall` A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

`overall.hetstat`
A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`prediction` A logical indicating whether a prediction interval should be printed.

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`tau.preset` Prespecified value for the square root of the between-study variance τ^2.

`TE.tau` Overall treatment effect used to estimate the between-study variance tau-squared.

`tau.common` A logical indicating whether tau-squared should be the same across subgroups.

`detail.tau` Detail on between-study variance estimate.

`method.I2` A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either `"Q"` or `"tau2"`, can be abbreviated (see `meta-package`).

`level.ma` The level used to calculate confidence intervals for meta-analysis estimates.

`method.random.ci`
A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

`adhoc.hakn.ci` A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see `meta-package`).

`level.predict` The level used to calculate prediction interval for a new study.

`method.predict` A character string indicating which method is used to calculate a prediction interval (see `meta-package`).

`adhoc.hakn.pi` A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see `meta-package`).

`seed.predict` A numeric value used as seed to calculate bootstrap prediction interval (see `meta-package`).

`method.bias` A character string indicating which test for funnel plot asymmetry is to be used. Either `"Begg"`, `"Egger"`, `"Thompson"`, `"Schwarzer"`, `"Harbord"`, `"Peters"`, or `"Deeks"`, can be abbreviated. See function `metabias`.

`backtransf` A logical indicating whether results for odds ratio (`sm="OR"`), risk ratio (`sm="RR"`), or diagnostic odds ratio (`sm="DOR"`) should be back transformed in printouts and plots. If TRUE (default), results will be presented as odds ratios and risk ratios; otherwise log odds ratios and log risk ratios will be shown.

`pscale` A numeric defining a scaling factor for printing of risk differences.

`text.common` A character string used in printouts and forest plot to label the pooled common effect estimate.

`text.random` A character string used in printouts and forest plot to label the pooled random effects estimate.

`text.predict` A character string used in printouts and forest plot to label the prediction interval.

`text.w.common` A character string used to label weights of common effect model.

`text.w.random` A character string used to label weights of random effects model.

`title` Title of meta-analysis / systematic review.

`complab` Comparison label.

`outclab` Outcome label.

`label.e` Label for experimental group.

`label.c` Label for control group.

`label.left` Graph label on left side of null effect in forest plot.

`label.right` Graph label on right side of null effect in forest plot.

`col.label.left` The colour of the graph label on the left side of the null effect.

`col.label.right`
The colour of the graph label on the right side of the null effect.

`subgroup` An optional vector to conduct a meta-analysis with subgroups.

`subgroup.name` A character string with a name for the subgroup variable.

`print.subgroup.name`
: A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

`sep.subgroup`
: A character string defining the separator between name of subgroup variable and subgroup label.

`test.subgroup`
: A logical value indicating whether to print results of test for subgroup differences.

`prediction.subgroup`
: A logical indicating whether prediction intervals should be printed for subgroups.

`seed.predict.subgroup`
: A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

`byvar`
: Deprecated argument (replaced by 'subgroup').

`hakn`
: Deprecated argument (replaced by 'method.random.ci').

`adhoc.hakn`
: Deprecated argument (replaced by 'adhoc.hakn.ci').

`print.CMH`
: A logical indicating whether result of the Cochran-Mantel-Haenszel test for overall effect should be printed.

`keepdata`
: A logical indicating whether original data (set) should be kept in meta object.

`warn`
: A logical indicating whether warnings should be printed (e.g., if `incr` is added to studies with zero cell frequencies or if estimation problems exist in fitting a GLMM).

`warn.deprecated`
: A logical indicating whether warnings should be printed if deprecated arguments are used.

`control`
: An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to `rma.uni` or `rma.glmm`.

`...`
: Additional arguments passed on to `rma.glmm` function and to catch deprecated arguments.

Details

Calculation of common and random effects estimates for meta-analyses with binary outcome data.

The following measures of treatment effect are available (Rücker et al., 2009):

- Risk ratio (`sm = "RR"`)
- Odds ratio (`sm = "OR"`)
- Risk difference (`sm = "RD"`)
- Arcsine difference (`sm = "ASD"`)
- Diagnostic Odds ratio (`sm = "DOR"`)
- Vaccine efficacy or vaccine effectiveness (`sm = "VE"`)

Note, mathematically, odds ratios and diagnostic odds ratios are identical, however, the labels in printouts and figures differ. Furthermore, log risk ratio (logRR) and log vaccine ratio (logVR) are mathematical identical, however, back-transformed results differ as vaccine efficacy or effectiveness is defined as `VE = 100 * (1 - RR)`.

A three-level random effects meta-analysis model (Van den Noortgate et al., 2013) is utilised if argument `cluster` is used and at least one cluster provides more than one estimate. Internally, `rma.mv` is called to conduct the analysis and `weights.rma.mv` with argument `type = "rowsum"` is used to calculate random effects weights.

Default settings are utilised for several arguments (assignments using `gs` function). These defaults can be changed for the current R session using the `settings.meta` function.

Furthermore, R function `update.meta` can be used to rerun a meta-analysis with different settings.

Meta-analysis method:

By default, both common effect (also called common effect) and random effects models are considered (see arguments `common` and `random`). If `method` is `"MH"` (default), the Mantel-Haenszel method (Greenland & Robins, 1985; Robins et al., 1986) is used to calculate the common effect estimate; if `method` is `"Inverse"`, inverse variance weighting is used for pooling (Fleiss, 1993); if `method` is `"Peto"`, the Peto method is used for pooling (Yusuf et al., 1985); if `method` is `"SSW"`, the sample size method is used for pooling (Bakbergenuly et al., 2020).

While the Mantel-Haenszel and Peto method are defined under the common effect model, random effects variants based on these methods are also implemented in `metabin`. Following RevMan 5, the Mantel-Haenszel estimator is used in the calculation of the between-study heterogeneity statistic Q which is used in the DerSimonian-Laird estimator (DerSimonian and Laird, 1986). Accordingly, the results for the random effects meta-analysis using the Mantel-Haenszel or inverse variance method are typically very similar. For the Peto method, Peto's log odds ratio, i.e. `(O-E) / V` and its standard error `sqrt(1 / V)` with `O-E` and `V` denoting "Observed minus Expected" and its variance, are utilised in the random effects model. Accordingly, results of a random effects model using `sm = "Peto"` can be different to results from a random effects model using `sm = "MH"` or `sm = "Inverse"`. Note, the random effects estimate is based on the inverse variance method for all methods discussed so far.

A distinctive and frequently overlooked advantage of binary endpoints is that individual patient data (IPD) can be extracted from a two-by-two table. Accordingly, statistical methods for IPD, i.e., logistic regression and generalised linear mixed models, can be utilised in a meta-analysis of binary outcomes (Stijnen et al., 2010; Simmonds et al., 2016).

R package **brglm2** must be available to fit a one-stage logistic regression model with penalised likelihood (Evrenoglou et al., 2022). The estimation of the summary odds ratio relies on the maximisation of the likelihood function, penalised using a Firth-type correction. This penalisation aims to reduce bias in cases with rare events and a small number of available studies. However, this method is not restricted to only such cases and can be applied more generally to binary data. Note, with this type of penalisation, all studies can be included in the analysis, regardless of the total number of observed events. This allows both single and double zero studies to be included without any continuity correction. The random effects model uses a multiplicative heterogeneity parameter ϕ, added to the model as an *ad hoc* term. The estimation of this parameter relies on a modified expression of Pearson's statistic, which accounts for sparse data. An estimate of ϕ equal to 1 indicates the absence of heterogeneity.

Generalised linear mixed models are available (argument `method = "GLMM"`) for the odds ratio as summary measure for the common effect and random effects model by calling the `rma.glmm` function from R package **metafor** internally.

Four different GLMMs are available for meta-analysis with binary outcomes using argument `model.glmm` (which corresponds to argument `model` in the `rma.glmm` function):

1. Logistic regression model with common study effects (default)
   (`model.glmm = "UM.FS"`, i.e., **U**nconditional **M**odel - **F**ixed **S**tudy effects)
2. Mixed-effects logistic regression model with random study effects
   (`model.glmm = "UM.RS"`, i.e., **U**nconditional **M**odel - **R**andom **S**tudy effects)
3. Generalised linear mixed model (conditional Hypergeometric-Normal)
   (`model.glmm = "CM.EL"`, i.e., **C**onditional **M**odel - **E**xact **L**ikelihood)
4. Generalised linear mixed model (conditional Binomial-Normal)
   (`model.glmm = "CM.AL"`, i.e., **C**onditional **M**odel - **A**pproximate **L**ikelihood)

Details on these four GLMMs as well as additional arguments which can be provided using argument '`...`' in `metabin` are described in `rma.glmm` where you can also find information on the iterative algorithms used for estimation. Note, regardless of which value is used for argument `model.glmm`, results for two different GLMMs are calculated: common effect model (with fixed treatment effect) and random effects model (with random treatment effects).

Continuity correction:

Four approaches are available to apply a continuity correction:

- Only studies with a zero cell count (`method.incr = "only0"`)
- All studies if at least one study has a zero cell count (`method.incr = "if0all"`)
- All studies irrespective of zero cell counts (`method.incr = "all"`)
- Use values provided in arguments `incr.e` and `incr.c` (`method.incr = "user"`)

By default, a continuity correction is only applied to studies with a zero cell count (`method.incr = "only0"`). This method showed the best performance for the odds ratio in a simulation study under the random effects model (Weber et al., 2020).

The continuity correction method is used both to calculate individual study results with confidence limits and to conduct meta-analysis based on the inverse variance method. For the risk difference, the method is only considered to calculate standard errors and confidence limits. For Peto method and GLMMs no continuity correction is used in the meta-analysis. Furthermore, the continuity correction is ignored for individual studies for the Peto method.

For studies with a zero cell count, by default, 0.5 (argument `incr`) is added to all cell frequencies for the odds ratio or only the number of events for the risk ratio (argument `RR.Cochrane = FALSE`, default). The increment is added to all cell frequencies for the risk ratio if argument `RR.Cochrane = TRUE`. For the risk difference, `incr` is only added to all cell frequencies to calculate the standard error. Finally, a treatment arm continuity correction is used if `incr = "TACC"` (Sweeting et al., 2004; Diamond et al., 2007).

For odds ratio and risk ratio, treatment estimates and standard errors are only calculated for studies with zero or all events in both groups if `allstudies = TRUE`.

For the Mantel-Haenszel method, by default (if `MH.exact` is FALSE), `incr` is added to cell frequencies of a study with a zero cell count in the calculation of the pooled risk ratio or odds ratio as well as the estimation of the variance of the pooled risk difference, risk ratio or odds ratio. This approach is also used in other software, e.g. RevMan 5 and the Stata procedure metan. According to Fleiss (in Cooper & Hedges, 1994), there is no need to add 0.5 to a cell frequency of zero to calculate the Mantel-Haenszel estimate and he advocates the exact method (`MH.exact` = TRUE). Note, estimates based on exact Mantel-Haenszel method or GLMM are not defined if the number of events is zero in all studies either in the experimental or control group.

Subgroup analysis:

Argument subgroup can be used to conduct subgroup analysis for a categorical covariate. The metareg function can be used instead for more than one categorical covariate or continuous covariates.

Exclusion of studies from meta-analysis:

Arguments subset and exclude can be used to exclude studies from the meta-analysis. Studies are removed completely from the meta-analysis using argument subset, while excluded studies are shown in printouts and forest plots using argument exclude (see Examples in metagen). Meta-analysis results are the same for both arguments.

Presentation of meta-analysis results:

Internally, both common effect and random effects models are calculated regardless of values choosen for arguments common and random. Accordingly, the estimate for the random effects model can be extracted from component TE.random of an object of class "meta" even if argument random = FALSE. However, all functions in R package **meta** will adequately consider the values for common and random. E.g. function print.meta will not print results for the random effects model if random = FALSE.

A prediction interval will only be shown if prediction = TRUE.

Value

An object of class c("metabin", "meta") with corresponding generic functions (see meta-object).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Bakbergenuly I, Hoaglin DC, Kulinskaya E (2020): Methods for estimating between-study variance and overall effect in meta-analysis of odds-ratios. *Research Synthesis Methods*, **11**, 426–42

Cooper H & Hedges LV (1994): *The Handbook of Research Synthesis*. Newbury Park, CA: Russell Sage Foundation

Diamond GA, Bax L, Kaul S (2007): Uncertain Effects of Rosiglitazone on the Risk for Myocardial Infarction and Cardiovascular Death. *Annals of Internal Medicine*, **147**, 578–81

DerSimonian R & Laird N (1986): Meta-analysis in clinical trials. *Controlled Clinical Trials*, **7**, 177–88

Evrenoglou T, White IR, Afach S, Mavridis D, Chaimani A. (2022): Network meta-analysis of rare events using penalized likelihood regression. *Statistics in Medicine*, **41**, 5203–19

Fleiss JL (1993): The statistical basis of meta-analysis. *Statistical Methods in Medical Research*, **2**, 121–45

Greenland S & Robins JM (1985): Estimation of a common effect parameter from sparse follow-up data. *Biometrics*, **41**, 55–68

Review Manager (RevMan) [Computer program]. Version 5.4. The Cochrane Collaboration, 2020

Robins J, Breslow N, Greenland S (1986): Estimators of the Mantel-Haenszel Variance Consistent in Both Sparse Data and Large-Strata Limiting Models. *Biometrics*, **42**, 311–23

Rücker G, Schwarzer G, Carpenter J, Olkin I (2009): Why add anything to nothing? The arcsine difference as a measure of treatment effect in meta-analysis with zero cells. *Statistics in Medicine*, **28**, 721–38

Simmonds MC, Higgins JP (2016): A general framework for the use of logistic regression models in meta-analysis. *Statistical Methods in Medical Research*, **25**, 2858–77

StataCorp. 2011. *Stata Statistical Software: Release 12*. College Station, TX: StataCorp LP.

Stijnen T, Hamza TH, Ozdemir P (2010): Random effects meta-analysis of event outcome in the framework of the generalized linear mixed model with applications in sparse data. *Statistics in Medicine*, **29**, 3046–67

Sweeting MJ, Sutton AJ, Lambert PC (2004): What to add to nothing? Use and avoidance of continuity corrections in meta-analysis of sparse data. *Statistics in Medicine*, **23**, 1351–75

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

Viechtbauer W (2010): Conducting meta-analyses in R with the metafor package. *Journal of Statistical Software*, **36**, 1–48

Weber F, Knapp G, Ickstadt K, Kundt G, Glass Ä (2020): Zero-cell corrections in random-effects meta-analyses. *Research Synthesis Methods*, **11**, 913–9

Yusuf S, Peto R, Lewis J, Collins R, Sleight P (1985): Beta blockade during and after myocardial infarction: An overview of the randomized trials. *Progress in Cardiovascular Diseases*, **27**, 335–71

See Also

meta-package, update.meta, forest, funnel, metabias, metacont, metagen, metareg, print.meta

Examples

```
# Calculate odds ratio and confidence interval for a single study
#
metabin(10, 20, 15, 20, sm = "OR")

# Different results (due to handling of studies with double zeros)
#
metabin(0, 10, 0, 10, sm = "OR")
metabin(0, 10, 0, 10, sm = "OR", allstudies = TRUE)

# Use subset of Olkin (1995) to conduct meta-analysis based on
# inverse variance method (with risk ratio as summary measure)
#
data(Olkin1995)
m1 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, subset = c(41, 47, 51, 59),
  studlab = paste(author, year),
  method = "Inverse")
m1
# Show results for individual studies
summary(m1)

# Use different subset of Olkin (1995)
```

```
#
m2 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, subset = year < 1970,
  studlab = paste(author, year),
  method = "Inverse")
m2
forest(m2)

# Meta-analysis with odds ratio as summary measure
#
m3 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, subset = year < 1970,
  studlab = paste(author, year),
  sm = "OR", method = "Inverse")
# Same meta-analysis result using 'update.meta' function
m3 <- update(m2, sm = "OR")
m3

# Meta-analysis based on Mantel-Haenszel method (with odds ratio as
# summary measure)
#
m4 <- update(m3, method = "MH")
m4

# Meta-analysis based on Peto method (only available for odds ratio
# as summary measure)
#
m5 <- update(m3, method = "Peto")
m5

## Not run:
# Meta-analyses using generalised linear mixed models (GLMM)

# Logistic regression model with (k = 4) fixed study effects
# (default: model.glmm = "UM.FS")
m6 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  studlab = paste(author, year),
  data = Olkin1995, subset = year < 1970, method = "GLMM")
# Same results:
m6 <- update(m2, method = "GLMM")
m6

# Mixed-effects logistic regression model with random study effects
m7 <- update(m6, model.glmm = "UM.RS")
#
# Use additional argument 'nAGQ' for internal call of 'rma.glmm'
# function
#
m7 <- update(m6, model.glmm = "UM.RS", nAGQ = 1)
m7

# Generalised linear mixed model (conditional Hypergeometric-Normal)
# (R package 'BiasedUrn' must be available)
```

```
if (requireNamespace("BiasedUrn", quietly = TRUE)) {
 m8 <- update(m6, model.glmm = "CM.EL")
 m8
}

# Generalised linear mixed model (conditional Binomial-Normal)
m9 <- update(m6, model.glmm = "CM.AL")
m9

# Logistic regression model with (k = 70) fixed study effects
m10 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
   studlab = paste(author, year),
   data = Olkin1995, method = "GLMM")
m10

# Mixed-effects logistic regression model with random study effects
update(m10, model.glmm = "UM.RS")

# Conditional Hypergeometric-Normal GLMM (with long computation time)
system.time(m11 <- update(m10, model.glmm = "CM.EL"))
m11

# Generalised linear mixed model (conditional Binomial-Normal)
update(m10, model.glmm = "CM.AL")

## End(Not run)
```

metabind *Combine and summarize meta-analysis objects*

Description

This function can be used to combine meta-analysis objects and is, for example, useful to summarize results of various meta-analysis methods or to generate a forest plot with results of several subgroup analyses.

Usage

```
metabind(
  ...,
  subgroup = NULL,
  name = NULL,
  common = NULL,
  random = NULL,
  prediction = NULL,
  backtransf = NULL,
  outclab = NULL,
  pooled = NULL,
  warn.deprecated = gs("warn.deprecated")
)
```

Arguments

`...` Any number of meta-analysis objects or a single list with meta-analyses.

`subgroup` An optional variable to generate a forest plot with subgroups.

`name` An optional character vector providing descriptive names for the meta-analysis objects.

`common` A logical vector indicating whether results of common effect model should be considered.

`random` A logical vector indicating whether results of random effects model should be considered.

`prediction` A logical vector indicating whether results of prediction intervals should be considered.

`backtransf` A logical indicating whether results should be back transformed in printouts and plots. If `backtransf=TRUE` (default), results for `sm="OR"` are printed as odds ratios rather than log odds ratios, for example.

`outclab` Outcome label for all meta-analyis objects.

`pooled` Deprecated argument (replaced by `common` and `random`.

`warn.deprecated` A logical indicating whether warnings should be printed if deprecated arguments are used.

Details

This function can be used to combine any number of meta-analysis objects which is useful, for example, to summarize results of various meta-analysis methods or to generate a forest plot with results of several subgroup analyses (see Examples).

Individual study results are not retained with `metabind` as the function allows to combine meta-analyses from different data sets (e.g., with randomised or observational studies). Individual study results are retained with R function `metamerge` which can be used to combine results of meta-analyses of the same dataset.

Value

An object of class `c("metabind", "meta")` with corresponding generic functions (see `meta-object`).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

`metagen`, `forest.metabind`, `metamerge`

Examples

```
data(Fleiss1993cont)

# Add some (fictitious) grouping variables:
#
Fleiss1993cont$age <- c(55, 65, 55, 65, 55)
Fleiss1993cont$region <- c("Europe", "Europe", "Asia", "Asia", "Europe")

m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD")

# Conduct two subgroup analyses
#
mu1 <- update(m1, subgroup = age, subgroup.name = "Age group")
mu2 <- update(m1, subgroup = region, subgroup.name = "Region")

# Combine random effects subgroup meta-analyses and show forest
# plot with subgroup results
#
mb1 <- metabind(mu1, mu2, common = FALSE)
mb1
forest(mb1)

# Use various estimation methods for between-study heterogeneity
# variance
#
m1.pm <- update(m1, method.tau = "PM")
m1.dl <- update(m1, method.tau = "DL")
m1.ml <- update(m1, method.tau = "ML")
m1.hs <- update(m1, method.tau = "HS")
m1.sj <- update(m1, method.tau = "SJ")
m1.he <- update(m1, method.tau = "HE")
m1.eb <- update(m1, method.tau = "EB")

# Combine meta-analyses and show results
#
taus <- c("Restricted maximum-likelihood estimator",
  "Paule-Mandel estimator",
  "DerSimonian-Laird estimator",
  "Maximum-likelihood estimator",
  "Hunter-Schmidt estimator",
  "Sidik-Jonkman estimator",
  "Hedges estimator",
  "Empirical Bayes estimator")
#
m1.taus <- metabind(m1, m1.pm, m1.dl, m1.ml, m1.hs, m1.sj, m1.he, m1.eb,
  name = taus, common = FALSE)
m1.taus
forest(m1.taus)
```

metacont *Meta-analysis of continuous outcome data*

Description

Calculation of common and random effects estimates for meta-analyses with continuous outcome data; inverse variance weighting is used for pooling.

Usage

```
metacont(
  n.e,
  mean.e,
  sd.e,
  n.c,
  mean.c,
  sd.c,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  cluster = NULL,
  rho = 0,
  median.e,
  q1.e,
  q3.e,
  min.e,
  max.e,
  median.c,
  q1.c,
  q3.c,
  min.c,
  max.c,
  method.mean = "Luo",
  method.sd = "Shi",
  approx.mean.e,
  approx.mean.c = approx.mean.e,
  approx.sd.e,
  approx.sd.c = approx.sd.e,
  sm = gs("smcont"),
  method.ci = gs("method.ci.cont"),
  level = gs("level"),
  pooledvar = gs("pooledvar"),
  method.smd = gs("method.smd"),
  sd.glass = gs("sd.glass"),
  exact.smd = gs("exact.smd"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  overall = common | random,
  overall.hetstat = if (is.null(gs("overall.hetstat"))) common | random else
    gs("overall.hetstat"),
  prediction = gs("prediction") | !missing(method.predict),
  method.tau = gs("method.tau"),
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.preset = NULL,
  TE.tau = NULL,
  tau.common = gs("tau.common"),
  detail.tau = NULL,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  method.bias = gs("method.bias"),
  backtransf = gs("backtransf"),
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title = gs("title"),
  complab = gs("complab"),
  outclab = "",
  label.e = gs("label.e"),
  label.c = gs("label.c"),
  label.left = gs("label.left"),
  label.right = gs("label.right"),
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  subgroup,
  subgroup.name = NULL,
  print.subgroup.name = gs("print.subgroup.name"),
  sep.subgroup = gs("sep.subgroup"),
  test.subgroup = gs("test.subgroup"),
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  byvar,
  id,
  adhoc.hakn,
  keepdata = gs("keepdata"),
  warn = gs("warn"),
  warn.deprecated = gs("warn.deprecated"),
  control = NULL,
  ...
)
```

Arguments

n.e Number of observations in experimental group or an R object created with pairwise.

mean.e Estimated mean in experimental group.

sd.e Standard deviation in experimental group.

n.c Number of observations in control group.

mean.c Estimated mean in control group.

sd.c Standard deviation in control group.

studlab An optional vector with study labels.

data An optional data frame containing the study information.

subset An optional vector specifying a subset of studies to be used.

exclude An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots.

cluster An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.

rho Assumed correlation of estimates within a cluster.

median.e Median in experimental group (used to estimate the mean and standard deviation).

q1.e First quartile in experimental group (used to estimate the mean and standard deviation).

q3.e Third quartile in experimental group (used to estimate the mean and standard deviation).

min.e Minimum in experimental group (used to estimate the mean and standard deviation).

max.e Maximum in experimental group (used to estimate the mean and standard deviation).

median.c Median in control group (used to estimate the mean and standard deviation).

q1.c First quartile in control group (used to estimate the mean and standard deviation).

q3.c Third quartile in control group (used to estimate the mean and standard deviation).

min.c Minimum in control group (used to estimate the mean and standard deviation).

max.c Maximum in control group (used to estimate the mean and standard deviation).

method.mean A character string indicating which method to use to approximate the mean from the median and other statistics (see Details).

method.sd A character string indicating which method to use to approximate the standard deviation from sample size, median, interquartile range and range (see Details).

approx.mean.e Approximation method to estimate means in experimental group (see Details).

approx.mean.c Approximation method to estimate means in control group (see Details).

approx.sd.e Approximation method to estimate standard deviations in experimental group (see Details).

approx.sd.c Approximation method to estimate standard deviations in control group (see Details).

sm A character string indicating which summary measure ("MD", "SMD", or "ROM") is to be used for pooling of studies.

method.ci A character string indicating which method is used to calculate confidence intervals for individual studies (see Details).

level The level used to calculate confidence intervals for individual studies.

pooledvar A logical indicating if a pooled variance should be used for the mean difference (sm="MD") or ratio of means (sm="ROM").

method.smd A character string indicating which method is used to estimate the standardised mean difference (sm="SMD"). Either "Hedges" for Hedges' g (default), "Cohen" for Cohen's d, or "Glass" for Glass' delta, can be abbreviated.

sd.glass A character string indicating which standard deviation is used in the denominator for Glass' method to estimate the standardised mean difference. Either "control" using the standard deviation in the control group (sd.c) or "experimental" using the standard deviation in the experimental group (sd.e), can be abbreviated.

exact.smd A logical indicating whether exact formulae should be used in estimation of the standardised mean difference and its standard error (see Details).

common A logical indicating whether a common effect meta-analysis should be conducted.

random A logical indicating whether a random effects meta-analysis should be conducted.

overall A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

overall.hetstat

A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

prediction A logical indicating whether a prediction interval should be printed.

method.tau A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see meta-package).

method.tau.ci A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see meta-package).

level.hetstat The level used to calculate confidence intervals for heterogeneity statistics.

tau.preset Prespecified value for the square root of the between-study variance τ^2.

TE.tau Overall treatment effect used to estimate the between-study variance tau-squared.

tau.common A logical indicating whether tau-squared should be the same across subgroups.

detail.tau Detail on between-study variance estimate.

method.I2 A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either "Q" or "tau2", can be abbreviated (see meta-package).

level.ma The level used to calculate confidence intervals for meta-analysis estimates.

method.random.ci
A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see meta-package).

adhoc.hakn.ci A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see meta-package).

level.predict The level used to calculate prediction interval for a new study.

method.predict A character string indicating which method is used to calculate a prediction interval (see meta-package).

adhoc.hakn.pi A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see meta-package).

seed.predict A numeric value used as seed to calculate bootstrap prediction interval (see meta-package).

method.bias A character string indicating which test is to be used. Either "Begg", "Egger", "Thompson", or "Pustejovsky" (see metabias), can be abbreviated.

backtransf A logical indicating whether results for ratio of means (sm="ROM") should be back transformed in printouts and plots. If TRUE (default), results will be presented as ratio of means; otherwise log ratio of means will be shown.

text.common A character string used in printouts and forest plot to label the pooled common effect estimate.

text.random A character string used in printouts and forest plot to label the pooled random effects estimate.

text.predict A character string used in printouts and forest plot to label the prediction interval.

text.w.common A character string used to label weights of common effect model.

text.w.random A character string used to label weights of random effects model.

title Title of meta-analysis / systematic review.

complab Comparison label.

outclab Outcome label.

label.e Label for experimental group.

label.c Label for control group.

label.left Graph label on left side of null effect in forest plot.

label.right Graph label on right side of null effect in forest plot.

col.label.left The colour of the graph label on the left side of the null effect.

col.label.right
The colour of the graph label on the right side of the null effect.

subgroup An optional vector to conduct a meta-analysis with subgroups.

subgroup.name A character string with a name for the subgroup variable.

print.subgroup.name
A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

sep.subgroup A character string defining the separator between name of subgroup variable and subgroup label.

test.subgroup A logical value indicating whether to print results of test for subgroup differences.

prediction.subgroup
A logical indicating whether prediction intervals should be printed for subgroups.

seed.predict.subgroup
A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

byvar Deprecated argument (replaced by 'subgroup').

id Deprecated argument (replaced by 'cluster').

adhoc.hakn Deprecated argument (replaced by 'adhoc.hakn.ci').

keepdata A logical indicating whether original data (set) should be kept in meta object.

warn A logical indicating whether warnings should be printed (e.g., if studies are excluded from meta-analysis due to zero standard deviations).

warn.deprecated
A logical indicating whether warnings should be printed if deprecated arguments are used.

control An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to rma.uni.

... Additional arguments (to catch deprecated arguments).

Details

Calculation of common and random effects estimates for meta-analyses with continuous outcome data; inverse variance weighting is used for pooling.

A three-level random effects meta-analysis model (Van den Noortgate et al., 2013) is utilised if argument cluster is used and at least one cluster provides more than one estimate. Internally, rma.mv is called to conduct the analysis and weights.rma.mv with argument type = "rowsum" is used to calculate random effects weights.

Default settings are utilised for several arguments (assignments using gs function). These defaults can be changed for the current R session using the settings.meta function.

Furthermore, R function update.meta can be used to rerun a meta-analysis with different settings.

Three different types of summary measures are available for continuous outcomes:

- mean difference (argument sm = "MD")

- standardised mean difference (sm = "SMD")
- ratio of means (sm = "ROM")

Standardised mean difference:
For the standardised mean difference three methods are implemented:

- Hedges' g (default, method.smd = "Hedges") - see Hedges (1981)
- Cohen's d (method.smd = "Cohen") - see Cohen (1988)
- Glass' delta (method.smd = "Glass") - see Glass (1976)

Hedges (1981) calculated the exact bias in Cohen's d which is a ratio of gamma distributions with the degrees of freedom, i.e. total sample size minus two, as argument. By default (argument exact.smd = FALSE), an accurate approximation of this bias provided in Hedges (1981) is utilised for Hedges' g as well as its standard error; these approximations are also used in RevMan 5. Following Borenstein et al. (2009) these approximations are not used in the estimation of Cohen's d. White and Thomas (2005) argued that approximations are unnecessary with modern software and accordingly promote to use the exact formulae; this is possible using argument exact.smd = TRUE. For Hedges' g the exact formulae are used to calculate the standardised mean difference as well as the standard error; for Cohen's d the exact formula is only used to calculate the standard error. In typical applications (with sample sizes above 10), the differences between using the exact formulae and the approximation will be minimal.

For Glass' delta, by default (argument sd.glass = "control"), the standard deviation in the control group (sd.c) is used in the denominator of the standard mean difference. The standard deviation in the experimental group (sd.e) can be used by specifying sd.glass = "experimental".

Ratio of means:
Meta-analysis of ratio of means – also called response ratios – is described in Hedges et al. (1999) and Friedrich et al. (2008). Calculations are conducted on the log scale and list elements TE, TE.common, and TE.random contain the logarithm of the ratio of means. In printouts and plots these values are back transformed if argument backtransf = TRUE.

Approximate means from sample sizes, medians and other statistics:
Missing means in the experimental group (analogously for the control group) can be derived from

1. sample size, median, interquartile range and range (arguments n.e, median.e, q1.e, q3.e, min.e, and max.e),
2. sample size, median and interquartile range (arguments n.e, median.e, q1.e, and q3.e), or
3. sample size, median and range (arguments n.e, median.e, min.e, and max.e).

By default, methods described in Luo et al. (2018) are utilised (argument method.mean = "Luo"):

- equation (15) if sample size, median, interquartile range and range are available,
- equation (11) if sample size, median and interquartile range are available,
- equation (7) if sample size, median and range are available.

Instead the methods described in Wan et al. (2014) are used if argument method.mean = "Wan":

- equation (10) if sample size, median, interquartile range and range are available,
- equation (14) if sample size, median and interquartile range are available,
- equation (2) if sample size, median and range are available.

The following methods are also available to estimate means from quantiles or ranges if R package **estmeansd** is installed:

- Method for Unknown Non-Normal Distributions (MLN) approach (Cai et al. (2021), argument `method.mean = "Cai"`),
- Quantile Estimation (QE) method (McGrath et al. (2020), argument `method.mean = "QE-McGrath"`)),
- Box-Cox (BC) method (McGrath et al. (2020), argument `method.mean = "BC-McGrath"`)).

By default, missing means are replaced successively using interquartile ranges and ranges (if available), interquartile ranges (if available) and finally ranges. Arguments `approx.mean.e` and `approx.mean.c` can be used to overwrite this behaviour for each individual study and treatment arm:

- use means directly (entry `""` in argument `approx.mean.e` or `approx.mean.c`);
- median, interquartile range and range (`"iqr.range"`);
- median and interquartile range (`"iqr"`);
- median and range (`"range"`).

Approximate standard deviations from sample sizes, medians and other statistics:

Missing standard deviations in the experimental group (analogously for the control group) can be derived from

1. sample size, median, interquartile range and range (arguments `n.e`, `median.e`, `q1.e`, `q3.e`, `min.e`, and `max.e`),
2. sample size, median and interquartile range (arguments `n.e`, `median.e`, `q1.e` and `q3.e`), or
3. sample size, median and range (arguments `n.e`, `median.e`, `min.e` and `max.e`).

Wan et al. (2014) describe methods to estimate the standard deviation from the sample size, median and additional statistics. Shi et al. (2020) provide an improved estimate of the standard deviation if the interquartile range and range are available in addition to the sample size and median. Accordingly, equation (11) in Shi et al. (2020) is the default (argument `method.sd = "Shi"`), if the median, interquartile range and range are provided. The method by Wan et al. (2014) is used if argument `method.sd = "Wan"` and, depending on the sample size, either equation (12) or (13) is used. If only the interquartile range or range is available, equations (15) / (16) and (7) / (9) in Wan et al. (2014) are used, respectively.

The following methods are also available to estimate standard deviations from quantiles or ranges if R package **estmeansd** is installed:

- Method for Unknown Non-Normal Distributions (MLN) approach (Cai et al. (2021), argument `method.mean = "Cai"`),
- Quantile Estimation (QE) method (McGrath et al. (2020), argument `method.mean = "QE-McGrath"`)),
- Box-Cox (BC) method (McGrath et al. (2020), argument `method.mean = "BC-McGrath"`)).

By default, missing standard deviations are replaced successively using these method, i.e., interquartile ranges and ranges are used before interquartile ranges before ranges. Arguments `approx.sd.e` and `approx.sd.c` can be used to overwrite this default for each individual study and treatment arms:

- sample size, median, interquartile range and range (`"iqr.range"`);
- sample size, median and interquartile range (`"iqr"`);
- sample size, median and range (`"range"`).

Confidence intervals for individual studies:

For the mean difference (argument `sm = "MD"`), the confidence interval for individual studies can be based on the

- standard normal distribution (method.ci = "z", default), or
- t-distribution (method.ci = "t").

Note, this choice does not affect the results of the common effect and random effects meta-analysis.

Subgroup analysis:

Argument subgroup can be used to conduct subgroup analysis for a categorical covariate. The metareg function can be used instead for more than one categorical covariate or continuous covariates.

Exclusion of studies from meta-analysis:

Arguments subset and exclude can be used to exclude studies from the meta-analysis. Studies are removed completely from the meta-analysis using argument subset, while excluded studies are shown in printouts and forest plots using argument exclude (see Examples in metagen). Meta-analysis results are the same for both arguments.

Presentation of meta-analysis results:

Internally, both common effect and random effects models are calculated regardless of values choosen for arguments common and random. Accordingly, the estimate for the random effects model can be extracted from component TE.random of an object of class "meta" even if argument random = FALSE. However, all functions in R package **meta** will adequately consider the values for common and random. E.g. function print.meta will not print results for the random effects model if random = FALSE.

A prediction interval will only be shown if prediction = TRUE.

Value

An object of class c("metacont", "meta") with corresponding generic functions (see meta-object).

Note

The function metagen is called internally to calculate individual and overall treatment estimates and standard errors.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Borenstein M, Hedges LV, Higgins JPT, Rothstein HR (2009): *Introduction to Meta-Analysis*. Chichester: Wiley

Cai S, Zhou J, Pan J (2021): Estimating the sample mean and standard deviation from order statistics and sample size in meta-analysis. *Statistical Methods in Medical Research*, **30**, 2701–2719

Cohen J (1988): *Statistical Power Analysis for the Behavioral Sciences (second ed.)*. Lawrence Erlbaum Associates

Friedrich JO, Adhikari NK, Beyene J (2008): The ratio of means method as an alternative to mean differences for analyzing continuous outcome variables in meta-analysis: A simulation study. *BMC Medical Research Methodology*, **8**, 32

Glass G (1976): Primary, secondary, and meta-analysis of research. *Educational Researcher*, **5**, 3–8

Hedges LV (1981): Distribution theory for Glass's estimator of effect size and related estimators. *Journal of Educational and Behavioral Statistics*, **6**, 107–28

Hedges LV, Gurevitch J, Curtis PS (1999): The meta-analysis of response ratios in experimental ecology. *Ecology*, **80**, 1150–6

Luo D, Wan X, Liu J, Tong T (2018): Optimally estimating the sample mean from the sample size, median, mid-range, and/or mid-quartile range. *Statistical Methods in Medical Research*, **27**, 1785–805

McGrath S, Zhao X, Steele R, et al. and the DEPRESsion Screening Data (DEPRESSD) Collaboration (2020): Estimating the sample mean and standard deviation from commonly reported quantiles in meta-analysis. *Statistical Methods in Medical Research*, **29**, 2520–2537

Review Manager (RevMan) [Computer program]. Version 5.4. The Cochrane Collaboration, 2020

Shi J, Luo D, Weng H, Zeng XT, Lin L, Chu H, Tong T (2020): Optimally estimating the sample standard deviation from the five-number summary. *Research Synthesis Methods*, **11**, 641–54

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

Wan X, Wang W, Liu J, Tong T (2014): Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range. *BMC Medical Research Methodology*, **14**, 135

White IR, Thomas J (2005): Standardized mean differences in individually-randomized and cluster-randomized trials, with applications to meta-analysis. *Clinical Trials*, **2**, 141–51

See Also

`meta-package`, `update.meta`, `metabin`, `metagen`, `pairwise`

Examples

```
data(Fleiss1993cont)

# Meta-analysis with Hedges' g as effect measure
#
m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD")
m1
forest(m1)

# Use Cohen's d instead of Hedges' g as effect measure
#
update(m1, method.smd = "Cohen")

# Use Glass' delta instead of Hedges' g as effect measure
#
update(m1, method.smd = "Glass")

# Use Glass' delta based on the standard deviation in the experimental group
#
```

```
update(m1, method.smd = "Glass", sd.glass = "experimental")

# Calculate Hedges' g based on exact formulae
#
update(m1, exact.smd = TRUE)

data(amlodipine)
m2 <- metacont(n.amlo, mean.amlo, sqrt(var.amlo),
  n.plac, mean.plac, sqrt(var.plac),
  data = amlodipine, studlab = study)
m2

# Use pooled variance
#
update(m2, pooledvar = TRUE)

# Meta-analysis of response ratios (Hedges et al., 1999)
#
data(woodyplants)
m3 <- metacont(n.elev, mean.elev, sd.elev, n.amb, mean.amb, sd.amb,
  data = woodyplants, sm = "ROM")
m3
print(m3, backtransf = FALSE)
```

metacor *Meta-analysis of correlations*

Description

Calculation of common effect and random effects estimates for meta-analyses with correlations; inverse variance weighting is used for pooling.

Usage

```
metacor(
  cor,
  n,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  cluster = NULL,
  rho = 0,
  sm = gs("smcor"),
  level = gs("level"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  overall = common | random,
  overall.hetstat = if (is.null(gs("overall.hetstat"))) common | random else
    gs("overall.hetstat"),
  prediction = gs("prediction") | !missing(method.predict),
  method.tau = gs("method.tau"),
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.preset = NULL,
  TE.tau = NULL,
  tau.common = gs("tau.common"),
  detail.tau = NULL,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  null.effect = 0,
  method.bias = gs("method.bias"),
  backtransf = gs("backtransf"),
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title = gs("title"),
  complab = gs("complab"),
  outclab = "",
  label.left = gs("label.left"),
  label.right = gs("label.right"),
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  subgroup,
  subgroup.name = NULL,
  print.subgroup.name = gs("print.subgroup.name"),
  sep.subgroup = gs("sep.subgroup"),
  test.subgroup = gs("test.subgroup"),
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  byvar,
  adhoc.hakn,
  keepdata = gs("keepdata"),
  warn.deprecated = gs("warn.deprecated"),
  control = NULL,
  ...
)
```

Arguments

`cor` Correlations.

`n` Number of observations.

`studlab` An optional vector with study labels.

`data` An optional data frame containing the study information, i.e., cor and n.

`subset` An optional vector specifying a subset of studies to be used.

`exclude` An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots.

`cluster` An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.

`rho` Assumed correlation of estimates within a cluster.

`sm` A character string indicating which summary measure (`"ZCOR"` or `"COR"`) is to be used for pooling of studies.

`level` The level used to calculate confidence intervals for individual studies.

`common` A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`overall` A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

`overall.hetstat`

A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`prediction` A logical indicating whether a prediction interval should be printed.

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`tau.preset` Prespecified value for the square root of the between-study variance τ^2.

`TE.tau` Overall treatment effect used to estimate the between-study variance tau-squared.

`tau.common` A logical indicating whether tau-squared should be the same across subgroups.

`detail.tau` Detail on between-study variance estimate.

`method.I2` A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either `"Q"` or `"tau2"`, can be abbreviated (see `meta-package`).

`level.ma` The level used to calculate confidence intervals for meta-analysis estimates.

`method.random.ci`

A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

adhoc.hakn.ci A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see meta-package).

level.predict The level used to calculate prediction interval for a new study.

method.predict A character string indicating which method is used to calculate a prediction interval (see meta-package).

adhoc.hakn.pi A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see meta-package).

seed.predict A numeric value used as seed to calculate bootstrap prediction interval (see meta-package).

null.effect A numeric value specifying the effect under the null hypothesis.

method.bias A character string indicating which test is to be used. Either "Begg", "Egger", or "Thompson", can be abbreviated. See function metabias.

backtransf A logical indicating whether results for Fisher's z transformed correlations (sm = "ZCOR") should be back transformed in printouts and plots. If TRUE (default), results will be presented as correlations; otherwise Fisher's z transformed correlations will be shown.

text.common A character string used in printouts and forest plot to label the pooled common effect estimate.

text.random A character string used in printouts and forest plot to label the pooled random effects estimate.

text.predict A character string used in printouts and forest plot to label the prediction interval.

text.w.common A character string used to label weights of common effect model.

text.w.random A character string used to label weights of random effects model.

title Title of meta-analysis / systematic review.

complab Comparison label.

outclab Outcome label.

label.left Graph label on left side of null effect in forest plot.

label.right Graph label on right side of null effect in forest plot.

col.label.left The colour of the graph label on the left side of the null effect.

col.label.right
The colour of the graph label on the right side of the null effect.

subgroup An optional vector to conduct a meta-analysis with subgroups.

subgroup.name A character string with a name for the subgroup variable.

print.subgroup.name
A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

sep.subgroup A character string defining the separator between name of subgroup variable and subgroup label.

test.subgroup A logical value indicating whether to print results of test for subgroup differences.

`prediction.subgroup`
: A logical indicating whether prediction intervals should be printed for subgroups.

`seed.predict.subgroup`
: A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

`byvar` Deprecated argument (replaced by 'subgroup').

`adhoc.hakn` Deprecated argument (replaced by 'adhoc.hakn.ci').

`keepdata` A logical indicating whether original data (set) should be kept in meta object.

`warn.deprecated`
: A logical indicating whether warnings should be printed if deprecated arguments are used.

`control` An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to `rma.uni`.

`...` Additional arguments (to catch deprecated arguments).

Details

This function conducts common effect and random effects meta-analysis of correlations based either on Fisher's z transformation of correlations (`sm = "ZCOR"`) or direct combination of (untransformed) correlations (`sm = "COR"`) (see Cooper et al., 2009, p264-5 and p273-4). Note, the input to argument `cor` is always correlations and not Fisher's z transformed correlations if `sm = "ZCOR"`.

Only few statisticians would advocate the use of untransformed correlations unless sample sizes are very large (see Cooper et al., 2009, p265). The artificial example given below shows that the smallest study gets the largest weight if correlations are combined directly because the correlation is closest to 1.

A three-level random effects meta-analysis model (Van den Noortgate et al., 2013) is utilised if argument `cluster` is used and at least one cluster provides more than one estimate. Internally, `rma.mv` is called to conduct the analysis and `weights.rma.mv` with argument `type = "rowsum"` is used to calculate random effects weights.

Default settings are utilised for several arguments (assignments using `gs` function). These defaults can be changed for the current R session using the `settings.meta` function.

Furthermore, R function `update.meta` can be used to rerun a meta-analysis with different settings.

Subgroup analysis:

Argument `subgroup` can be used to conduct subgroup analysis for a categorical covariate. The `metareg` function can be used instead for more than one categorical covariate or continuous covariates.

Exclusion of studies from meta-analysis:

Arguments `subset` and `exclude` can be used to exclude studies from the meta-analysis. Studies are removed completely from the meta-analysis using argument `subset`, while excluded studies are shown in printouts and forest plots using argument `exclude` (see Examples in `metagen`). Meta-analysis results are the same for both arguments.

Presentation of meta-analysis results:

Internally, both common effect and random effects models are calculated regardless of values choosen for arguments common and random. Accordingly, the estimate for the random effects model can be extracted from component TE.random of an object of class "meta" even if argument random = FALSE. However, all functions in R package **meta** will adequately consider the values for common and random. E.g. functions print.meta and forest.meta will not print results for the random effects model if random = FALSE.

A prediction interval will only be shown if prediction = TRUE.

Value

An object of class c("metacor", "meta") with corresponding generic functions (see meta-object).

Note

The function metagen is called internally to calculate individual and overall treatment estimates and standard errors.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Cooper H, Hedges LV, Valentine JC (2009): *The Handbook of Research Synthesis and Meta-Analysis*, 2nd Edition. New York: Russell Sage Foundation

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

See Also

meta-package, update.meta, metacont, metagen, print.meta

Examples

```
m1 <- metacor(c(0.85, 0.7, 0.95), c(20, 40, 10))

# Print correlations (back transformed from Fisher's z
# transformation)
#
summary(m1)

# Print Fisher's z transformed correlations
#
print(summary(m1), backtransf = FALSE)

# Forest plot with back transformed correlations
#
forest(m1)
```

```
# Forest plot with Fisher's z transformed correlations
#
forest(m1, backtransf = FALSE)

m2 <- update(m1, sm = "cor")
summary(m2)

## Not run:
# Identical forest plots (as back transformation is the identity
# transformation)
forest(m2)
forest(m2, backtransf = FALSE)

## End(Not run)
```

metacr *Meta-analysis of outcome data from Cochrane review*

Description

Wrapper function to perform meta-analysis for a single outcome of a Cochrane Intervention review.

Usage

```
metacr(
  x,
  comp.no = 1,
  outcome.no = 1,
  method,
  sm,
  level = gs("level"),
  common,
  random,
  prediction = gs("prediction") | !missing(method.predict),
  method.tau = "DL",
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.common = FALSE,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = "classic",
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  Q.Cochrane,
  swap.events,
  logscale,
  backtransf = gs("backtransf"),
  test.subgroup,
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  rob = NULL,
  tool = NULL,
  categories = NULL,
  col = NULL,
  symbols = NULL,
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title,
  complab,
  outclab,
  label.left,
  label.right,
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  keepdata = gs("keepdata"),
  warn = FALSE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

Arguments

`x` An object of class `rm5` or `cdir` created by R function `read.rm5` or `read.cdir` .

`comp.no` Comparison number.

`outcome.no` Outcome number.

`method` A character string indicating which method is to be used for pooling of studies. One of `"Inverse"`, `"MH"`, or `"Peto"`, can be abbreviated.

`sm` A character string indicating which summary measure (`"RR"`, `"OR"`, `"RD"`, `"ASD"`, `"HR"`, `"MD"`, or `"SMD"`, or `"ROM"`) is to be used for pooling of studies.

`level` The level used to calculate confidence intervals for individual studies.

`common` A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`prediction` A logical indicating whether a prediction interval should be printed.

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`tau.common` A logical indicating whether tau-squared should be the same across subgroups.

`method.I2` A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either `"Q"` or `"tau2"`, can be abbreviated (see `meta-package`).

`level.ma` The level used to calculate confidence intervals for meta-analysis estimates.

`method.random.ci`
A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

`adhoc.hakn.ci` A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see `meta-package`).

`level.predict` The level used to calculate prediction interval for a new study.

`method.predict` A character string indicating which method is used to calculate a prediction interval (see `meta-package`).

`adhoc.hakn.pi` A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see `meta-package`).

`seed.predict` A numeric value used as seed to calculate bootstrap prediction interval (see `meta-package`).

`Q.Cochrane` A logical indicating if the Mantel-Haenszel estimate is used in the calculation of the heterogeneity statistic Q which is implemented in RevMan 5.

`swap.events` A logical indicating whether events and non-events should be interchanged.

`logscale` A logical indicating whether effect estimates are entered on log-scale (ignored for `cdir` objects).

`backtransf` A logical indicating whether results should be back transformed in printouts and plots. If `backtransf=TRUE` (default), results for `sm="OR"` are printed as odds ratios rather than log odds ratios and results for `sm="ZCOR"` are printed as correlations rather than Fisher's z transformed correlations, for example.

`test.subgroup` A logical value indicating whether to print results of test for subgroup differences.

`prediction.subgroup`
A logical indicating whether prediction intervals should be printed for subgroups.

`seed.predict.subgroup`
A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

`rob` A logical indicating whether risk of bias (RoB) assessment should be considered in meta-analysis (only for `read.cdir` objects).

`tool` Risk of bias (RoB) tool (only for `read.cdir` objects).

`categories` Possible RoB categories (only for `read.cdir` objects).

`col` Colours for RoB categories (only for `read.cdir` objects).

`symbols` Corresponding symbols for RoB categories (only for `read.cdir` objects).

`text.common` A character string used in printouts and forest plot to label the pooled common effect estimate.

`text.random` A character string used in printouts and forest plot to label the pooled random effects estimate.

`text.predict` A character string used in printouts and forest plot to label the prediction interval.

`text.w.common` A character string used to label weights of common effect model.

`text.w.random` A character string used to label weights of random effects model.

`title` Title of meta-analysis / systematic review.

`complab` Comparison label.

`outclab` Outcome label.

`label.left` Graph label on left side of null effect in forest plot.

`label.right` Graph label on right side of null effect in forest plot.

`col.label.left` The colour of the graph label on the left side of the null effect.

`col.label.right`
The colour of the graph label on the right side of the null effect.

`keepdata` A logical indicating whether original data (set) should be kept in meta object.

`warn` A logical indicating whether warnings should be printed (e.g., if `incr` is added to studies with zero cell frequencies).

`warn.deprecated`
A logical indicating whether warnings should be printed if deprecated arguments are used.

`...` Additional arguments (to catch deprecated arguments).

Details

Cochrane intervention reviews are based on the comparison of two interventions. Each Cochrane intervention review can have a variable number of comparisons. For each comparison, a variable number of outcomes can be define. For each outcome, a separate meta-analysis is conducted. Review Manager 5 (RevMan 5) was the software used for preparing and maintaining Cochrane Reviews.

This wrapper function can be used to perform meta-analysis for a single outcome of a Cochrane intervention review. Internally, R functions `metabin`, `metacont`, and `metagen` are called - depending on the definition of the outcome in RevMan 5.

Information on the risk of bias RoB) assessment can be provided with arguments `tool`, `categories`, `col` and `symbols`. This is not useful if an overall RoB assessment has been done. In this case use `rob` to add the full flexible RoB information to a `metacr` object.

Note, it is recommended to choose the RevMan 5 settings before executing `metacr`, i.e., `settings.meta("revman5")`.

Value

An object of class `"meta"` and - depending on outcome type utilised in Cochrane intervention review for selected outcome - `"metabin"`, `"metacont"`, or `"metagen"` with corresponding generic functions (see `meta-object`).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Review Manager (RevMan) [Computer program]. Version 5.4. The Cochrane Collaboration, 2020

See Also

meta-package, rob, metabin, metacont, metagen, read.cdir, read.rm5, settings.meta

Examples

```
# Locate export data file "Fleiss1993_CR.csv"
# in sub-directory of package "meta"
#
filename <- system.file("extdata", "Fleiss1993_CR.csv", package = "meta")
#
Fleiss1993_CR <- read.rm5(filename)

# Choose RevMan 5 settings and store old settings
#
oldset <- settings.meta("revman5", quietly = FALSE)

# Same result as R command example(Fleiss1993bin)
#
metacr(Fleiss1993_CR)

# Same result as R command example(Fleiss1993cont)
#
metacr(Fleiss1993_CR, 1, 2)
forest(metacr(Fleiss1993_CR, 1, 2))

# Change summary measure to RR
#
m1 <- metacr(Fleiss1993_CR)
update(m1, sm="RR")

# Use old settings
#
settings.meta(oldset)
```

---

metacum.meta *Cumulative meta-analysis*

Description

Performs a cumulative meta-analysis.

Usage

```
## S3 method for class 'meta'
metacum(
  x,
  pooled,
  sortvar,
  prediction,
  overall = x$overall,
  text.pooled,
  no = 1,
  cid = NULL,
  cid.below.null = NULL,
  cid.above.null = NULL,
  small.values = "desirable",
  ...
)

metacum(x, ...)

## Default S3 method:
metacum(x, ...)
```

Arguments

`x`	An object of class `meta`.
`pooled`	A character string indicating whether a common effect or random effects model is used for pooling. Either missing (see Details), `"common"`, or `"random"`, can be abbreviated.
`sortvar`	An optional vector used to sort the individual studies (must be of same length as `x$TE`).
`prediction`	A logical indicating whether to report prediction intervals.
`overall`	A logical indicating whether overall results should be reported.
`text.pooled`	A character string used in printouts and forest plots to label the pooled effect estimate.
`no`	A numeric specifying which meta-analysis results to consider.
`cid`	A numeric value or vector specifying clinically important differences (CID) / decision thresholds used to calculate expected proportions of clinically important benefit or harm (see `cidprop`).
`cid.below.null`	A single numeric defining the decision threshold below the null effect to distinguish clinically important from not important effects (see `cidprop`).
`cid.above.null`	A single numeric defining the decision threshold above the null effect to distinguish clinically important from not important effects (see `cidprop`).
`small.values`	A character string specifying whether small treatment effects indicate a beneficial (`"desirable"`) or harmful (`"undesirable"`) effect, can be abbreviated (see `cidprop`).
`...`	Additional arguments (ignored).

Details

A cumulative meta-analysis is performed. Studies are included sequentially as defined by `sortvar`.

Information from object `x` is utilised if argument `pooled` is missing. A common effect model is assumed (`pooled = "common"`) if argument `x$common` is `TRUE`; a random effects model is assumed (`pooled = "random"`) if argument `x$random` is `TRUE` and `x$common` is `FALSE`.

Value

An object of class `"metacum"` with dedicated print and forest functions.

The following list elements provide results from meta-analyses, each adding one study at a time (see `meta-object` for more information on these list elements):

> studlab, TE, seTE, df.random, lower, upper, statistic, pval,
> lower.predict, upper.predict, df.predict, w (sum of weights),
> tau2, se.tau2, lower.tau2, upper.tau2, tau, lower.tau, upper.tau,
> I2, lower.I2, upper.I2, Rb, n.harmonic.mean, t.harmonic.mean,
> k, k.study, k.all, k.TE, k.MH.

The following list elements contain results of the original meta-analysis:

> TE.pooled, seTE.pooled, df.random.pooled,
> lower.pooled, upper.pooled, statistic.pooled, pval.pooled,
> lower.predict.pooled, upper.predict.pooled,
> df.predict.pooled, w.pooled,
> tau2.pooled, se.tau2.pooled, lower.tau2.pooled, upper.tau2.pooled,
> tau.pooled, lower.tau.pooled, upper.tau.pooled,
> I2.pooled, lower.I2.pooled, upper.I2.pooled, Rb.pooled,
> n.harmonic.mean.pooled, t.harmonic.mean.pooled,
> k.pooled, k.study.pooled, k.all.pooled, k.TE.pooled, k.MH.pooled.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

References

Cooper H & Hedges LV (1994): *The Handbook of Research Synthesis*. Newbury Park, CA: Russell Sage Foundation

See Also

`forest.metacum`, `print.metacum`, `cidprop`

Examples

```
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac,
  data = Fleiss1993bin, studlab = study, sm = "RR", method = "I")
m1
metacum(m1)
metacum(m1, pooled = "random")

forest(metacum(m1))
forest(metacum(m1, pooled = "random"))

metacum(m1, sortvar = study)
metacum(m1, sortvar = 7:1)

m2 <- update(m1, title = "Fleiss1993bin meta-analysis", backtransf = FALSE)
metacum(m2)

data(Fleiss1993cont)
m3 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD")
metacum(m3)
```

---

metagen *Generic inverse variance meta-analysis*

Description

Common effect and random effects meta-analysis based on estimates (e.g. log hazard ratios) and their standard errors. The inverse variance method is used for pooling.

Three-level random effects meta-analysis (Van den Noortgate et al., 2013) is available by internally calling rma.mv function from R package **metafor** (Viechtbauer, 2010).

Usage

```
metagen(
  TE,
  seTE,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  cluster = NULL,
  rho = 0,
  cycles = NULL,
  sm = "",
  method.ci = if (missing(df)) "z" else "t",
  level = gs("level"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  overall = common | random,
  overall.hetstat = if (is.null(gs("overall.hetstat"))) common | random else
    gs("overall.hetstat"),
  prediction = gs("prediction") | !missing(method.predict),
  method.tau = gs("method.tau"),
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.preset = NULL,
  TE.tau = NULL,
  tau.common = gs("tau.common"),
  detail.tau = NULL,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  null.effect = 0,
  method.bias = gs("method.bias"),
  n.e = NULL,
  n.c = NULL,
  pval,
  df,
  lower,
  upper,
  level.ci = 0.95,
  median,
  q1,
  q3,
  min,
  max,
  method.mean = "Luo",
  method.sd = "Shi",
  approx.TE,
  approx.seTE,
  transf = gs("transf") & missing(func.transf),
  backtransf = gs("backtransf") | !missing(func.backtransf),
  func.transf,
  func.backtransf,
  args.transf,
  args.backtransf,
  pscale = 1,
  irscale = 1,
  irunit = "person-years",
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title = gs("title"),
  complab = gs("complab"),
  outclab = "",
  label.e = gs("label.e"),
  label.c = gs("label.c"),
  label.left = gs("label.left"),
  label.right = gs("label.right"),
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  subgroup,
  subgroup.name = NULL,
  print.subgroup.name = gs("print.subgroup.name"),
  sep.subgroup = gs("sep.subgroup"),
  test.subgroup = gs("test.subgroup"),
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  byvar,
  id,
  adhoc.hakn,
  keepdata = gs("keepdata"),
  keeprma = gs("keeprma"),
  warn = gs("warn"),
  warn.deprecated = gs("warn.deprecated"),
  control = NULL,
  ...
)
```

Arguments

`TE` Estimate of treatment effect, e.g., log hazard ratio or risk difference or an R object created with `pairwise`.

`seTE` Standard error of treatment estimate or standard deviation of n-of-1 trials.

`studlab` An optional vector with study labels.

`data` An optional data frame containing the study information.

`subset` An optional vector specifying a subset of studies to be used (see Details).

`exclude` An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots (see Details).

`cluster` An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.

`rho` Assumed correlation of estimates within a cluster.

`cycles` A numeric vector with the number of cycles per patient / study in n-of-1 trials.

sm
A character string indicating underlying summary measure, e.g., "RD", "RR", "OR", "ASD", "HR", "MD", "SMD", or "ROM".

method.ci
A character string indicating which method is used to calculate confidence intervals for individual studies, see Details.

level
The level used to calculate confidence intervals for individual studies.

common
A logical indicating whether a common effect meta-analysis should be conducted.

random
A logical indicating whether a random effects meta-analysis should be conducted.

overall
A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

overall.hetstat
A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

prediction
A logical indicating whether a prediction interval should be printed.

method.tau
A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see meta-package).

method.tau.ci
A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see meta-package).

level.hetstat
The level used to calculate confidence intervals for heterogeneity statistics.

tau.preset
Prespecified value for the square root of the between-study variance τ^2.

TE.tau
Overall treatment effect used to estimate the between-study variance tau-squared.

tau.common
A logical indicating whether tau-squared should be the same across subgroups.

detail.tau
Detail on between-study variance estimate.

method.I2
A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either "Q" or "tau2", can be abbreviated (see meta-package).

level.ma
The level used to calculate confidence intervals for meta-analysis estimates.

method.random.ci
A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see meta-package).

adhoc.hakn.ci
A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see meta-package).

level.predict
The level used to calculate prediction interval for a new study.

method.predict
A character string indicating which method is used to calculate a prediction interval (see meta-package).

adhoc.hakn.pi
A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see meta-package).

seed.predict
A numeric value used as seed to calculate bootstrap prediction interval (see meta-package).

`null.effect` A numeric value specifying the effect under the null hypothesis.

`method.bias` A character string indicating which test is to be used. Either `"Begg"`, `"Egger"`, or `"Thompson"`, can be abbreviated. See function `metabias`.

`n.e` Number of observations in experimental group (or total sample size in study).

`n.c` Number of observations in control group.

`pval` P-value (used to estimate the standard error).

`df` Degrees of freedom (used in test or to construct confidence intervals).

`lower` Lower limit of confidence interval (used to estimate the standard error).

`upper` Upper limit of confidence interval (used to estimate the standard error).

`level.ci` Level of confidence interval.

`median` Median (used to estimate the treatment effect and standard error).

`q1` First quartile (used to estimate the treatment effect and standard error).

`q3` Third quartile (used to estimate the treatment effect and standard error).

`min` Minimum (used to estimate the treatment effect and standard error).

`max` Maximum (used to estimate the treatment effect and standard error).

`method.mean` A character string indicating which method to use to approximate the mean from the median and other statistics (see Details).

`method.sd` A character string indicating which method to use to approximate the standard deviation from sample size, median, interquartile range and range (see Details).

`approx.TE` Approximation method to estimate treatment estimate (see Details).

`approx.seTE` Approximation method to estimate standard error (see Details).

`transf` A logical indicating whether inputs for arguments `TE`, `lower` and `upper` are already appropriately transformed to conduct the meta-analysis or on the original scale. If `transf = TRUE` (default), inputs are expected to be log odds ratios instead of odds ratios for `sm = "OR"` and Fisher's z transformed correlations instead of correlations for `sm = "ZCOR"`, for example.

`backtransf` A logical indicating whether results should be back transformed in printouts and plots. If `backtransf = TRUE` (default), results for `sm = "OR"` are printed as odds ratios rather than log odds ratios and results for `sm = "ZCOR"` are printed as correlations rather than Fisher's z transformed correlations, for example.

`func.transf` A function used to transform inputs for arguments `TE`, `lower` and `upper`.

`func.backtransf`
A function used to back-transform results.

`args.transf` An optional list to provide additional arguments to `func.transf`.

`args.backtransf`
An optional list to provide additional arguments to `func.backtransf`.

`pscale` A numeric giving scaling factor for printing of single event probabilities or risk differences, i.e. if argument `sm` is equal to `"PLOGIT"`, `"PLN"`, `"PRAW"`, `"PAS"`, `"PFT"`, or `"RD"`.

`irscale` A numeric defining a scaling factor for printing of single incidence rates or incidence rate differences, i.e. if argument `sm` is equal to `"IR"`, `"IRLN"`, `"IRS"`, `"IRFT"`, or `"IRD"`.

`irunit` A character specifying the time unit used to calculate rates, e.g. person-years.

`text.common` A character string used in printouts and forest plot to label the pooled common effect estimate.

`text.random` A character string used in printouts and forest plot to label the pooled random effects estimate.

`text.predict` A character string used in printouts and forest plot to label the prediction interval.

`text.w.common` A character string used to label weights of common effect model.

`text.w.random` A character string used to label weights of random effects model.

`title` Title of meta-analysis / systematic review.

`complab` Comparison label.

`outclab` Outcome label.

`label.e` Label for experimental group.

`label.c` Label for control group.

`label.left` Graph label on left side of null effect in forest plot.

`label.right` Graph label on right side of null effect in forest plot.

`col.label.left` The colour of the graph label on the left side of the null effect.

`col.label.right` The colour of the graph label on the right side of the null effect.

`subgroup` An optional vector to conduct a meta-analysis with subgroups.

`subgroup.name` A character string with a name for the subgroup variable.

`print.subgroup.name` A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

`sep.subgroup` A character string defining the separator between name of subgroup variable and subgroup label.

`test.subgroup` A logical value indicating whether to print results of test for subgroup differences.

`prediction.subgroup` A logical indicating whether prediction intervals should be printed for subgroups.

`seed.predict.subgroup` A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

`byvar` Deprecated argument (replaced by 'subgroup').

`id` Deprecated argument (replaced by 'cluster').

`adhoc.hakn` Deprecated argument (replaced by 'adhoc.hakn.ci').

`keepdata` A logical indicating whether original data (set) should be kept in meta object.

`keeprma` A logical indicating whether `rma.mv` object from three-level meta-analysis should be stored.

`warn` A logical indicating whether warnings should be printed (e.g., if studies are excluded from meta-analysis due to zero standard errors).

`warn.deprecated`
A logical indicating whether warnings should be printed if deprecated arguments are used.

`control` An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to `rma.uni` or `rma.mv`.

`...` Additional arguments (to catch deprecated arguments).

Details

This function provides the *generic inverse variance method* for meta-analysis which requires treatment estimates and their standard errors (Borenstein et al., 2010). The method is useful, e.g., for pooling of survival data (using log hazard ratio and standard errors as input). Arguments `TE` and `seTE` can be used to provide treatment estimates and standard errors directly. However, it is possible to derive these quantities from other information.

Argument `cycles` can be used to conduct a meta-analysis of n-of-1 trials according to Senn (2024). In this case, argument `seTE` does not contain the standard error but standard deviation for individual trials / patients. Trial-specific standard errors are calculated from an average standard deviation multiplied by the number of cycles minus 1, i.e., the degrees of freedom. Details of the meta-analysis method are provided in Senn (2024). Note, arguments used in the approximation of means or standard errors, like `lower` and `upper`, or `df`, are ignored for the meta-analysis of n-of-1 trials.

A three-level random effects meta-analysis model (Van den Noortgate et al., 2013) is utilised if argument `cluster` is used and at least one cluster provides more than one estimate. Internally, `rma.mv` is called to conduct the analysis and `weights.rma.mv` with argument `type = "rowsum"` is used to calculate random effects weights.

Default settings are utilised for several arguments (assignments using `gs` function). These defaults can be changed for the current R session using the `settings.meta` function.

Furthermore, R function `update.meta` can be used to rerun a meta-analysis with different settings.

Approximate treatment estimates:

Missing treatment estimates can be derived from

1. confidence limits provided by arguments `lower` and `upper`;
2. median, interquartile range and range (arguments `median`, `q1`, `q3`, `min`, and `max`);
3. median and interquartile range (arguments `median`, `q1` and `q3`);
4. median and range (arguments `median`, `min` and `max`).

For confidence limits, the treatment estimate is defined as the center of the confidence interval (on the log scale for relative effect measures like the odds ratio or hazard ratio).

If the treatment effect is a mean it can be approximated from sample size, median, interquartile range and range.

By default, methods described in Luo et al. (2018) are utilised (argument `method.mean = "Luo"`):

- equation (7) if sample size, median and range are available,
- equation (11) if sample size, median and interquartile range are available,
- equation (15) if sample size, median, range and interquartile range are available.

Instead the methods described in Wan et al. (2014) are used if argument `method.mean = "Wan"`:

- equation (2) if sample size, median and range are available,
- equation (14) if sample size, median and interquartile range are available,
- equation (10) if sample size, median, range and interquartile range are available.

The following methods are also available to estimate means from quantiles or ranges if R package **estmeansd** is installed:

- Method for Unknown Non-Normal Distributions (MLN) approach (Cai et al. (2021), argument `method.mean = "Cai"`),
- Quantile Estimation (QE) method (McGrath et al. (2020), argument `method.mean = "QE-McGrath"`)),
- Box-Cox (BC) method (McGrath et al. (2020), argument `method.mean = "BC-McGrath"`)).

By default, missing treatment estimates are replaced successively using these method, i.e., confidence limits are utilised before interquartile ranges. Argument `approx.TE` can be used to overwrite this default for each individual study:

- Use treatment estimate directly (entry `""` in argument `approx.TE`);
- confidence limits (`"ci"` in argument `approx.TE`);
- median, interquartile range and range (`"iqr.range"`);
- median and interquartile range (`"iqr"`);
- median and range (`"range"`).

Approximate standard errors:

Missing standard errors can be derived from

1. p-value provided by arguments `pval` and (optional) `df`;
2. confidence limits (arguments `lower`, `upper`, and (optional) `df`);
3. sample size, median, interquartile range and range (arguments `n.e` and / or `n.c`, `median`, `q1`, `q3`, `min`, and `max`);
4. sample size, median and interquartile range (arguments `n.e` and / or `n.c`, `median`, `q1` and `q3`);
5. sample size, median and range (arguments `n.e` and / or `n.c`, `median`, `min` and `max`).

For p-values and confidence limits, calculations are either based on the standard normal or *t*-distribution if argument `df` is provided. Furthermore, argument `level.ci` can be used to provide the level of the confidence interval.

Wan et al. (2014) describe methods to estimate the standard deviation (and thus the standard error by deviding the standard deviation with the square root of the sample size) from the sample size, median and additional statistics. Shi et al. (2020) provide an improved estimate of the standard deviation if the interquartile range and range are available in addition to the sample size and median. Accordingly, equation (11) in Shi et al. (2020) is the default (argument `method.sd = "Shi"`), if the median, interquartile range and range are provided (arguments `median`, `q1`, `q3`, `min` and `max`). The method by Wan et al. (2014) is used if argument `method.sd = "Wan"` and, depending on the sample size, either equation (12) or (13) is used. If only the interquartile range or range is available, equations (15) / (16) and (7) / (9) in Wan et al. (2014) are used, respectively. The sample size of individual studies must be provided with arguments `n.e` and / or `n.c`. The total sample size is calculated as `n.e + n.c` if both arguments are provided.

The following methods are also available to estimate standard deviations from quantiles or ranges if R package **estmeansd** is installed:

- Method for Unknown Non-Normal Distributions (MLN) approach (Cai et al. (2021), argument `method.mean = "Cai"`),
- Quantile Estimation (QE) method (McGrath et al. (2020), argument `method.mean = "QE-McGrath"`)),
- Box-Cox (BC) method (McGrath et al. (2020), argument `method.mean = "BC-McGrath"`)).

By default, missing standard errors are replaced successively using these method, e.g., p-value before confidence limits before interquartile range and range. Argument `approx.seTE` can be used to overwrite this default for each individual study:

- Use standard error directly (entry `""` in argument `approx.seTE`);
- p-value (`"pval"` in argument `approx.seTE`);
- confidence limits (`"ci"`);
- median, interquartile range and range (`"iqr.range"`);
- median and interquartile range (`"iqr"`);
- median and range (`"range"`).

Confidence intervals for individual studies:

For the mean difference (argument `sm = "MD"`), the confidence interval for individual studies can be based on the

- standard normal distribution (`method.ci = "z"`), or
- *t*-distribution (`method.ci = "t"`).

By default, the first method is used if argument `df` is missing and the second method otherwise.

Note, this choice does not affect the results of the common effect and random effects meta-analysis.

Subgroup analysis:

Argument `subgroup` can be used to conduct subgroup analysis for a categorical covariate. The `metareg` function can be used instead for more than one categorical covariate or continuous covariates.

Specify the null hypothesis of test for an overall effect:

Argument `null.effect` can be used to specify the (treatment) effect under the null hypothesis in a test for an overall effect.

By default (`null.effect = 0`), the null hypothesis corresponds to "no difference" (which is obvious for absolute effect measures like the mean difference (`sm = "MD"`) or standardised mean difference (`sm = "SMD"`)). For relative effect measures, e.g., risk ratio (`sm = "RR"`) or odds ratio (`sm = "OR"`), the null effect is defined on the log scale, i.e., $log(\mathrm{RR}) = 0$ or $log(\mathrm{OR}) = 0$ which is equivalent to testing RR = 1 or OR = 1.

Use of argument `null.effect` is especially useful for summary measures without a "natural" null effect, i.e., in situations without a second (treatment) group. For example, an overall proportion of 50% could be tested in the meta-analysis of single proportions with argument `null.effect = 0.5`.

Note, all tests for an overall effect are two-sided with the alternative hypothesis that the effect is unequal to `null.effect`.

Exclusion of studies from meta-analysis:

Arguments `subset` and `exclude` can be used to exclude studies from the meta-analysis. Studies are removed completely from the meta-analysis using argument `subset`, while excluded studies are shown in printouts and forest plots using argument `exclude` (see Examples). Meta-analysis results are the same for both arguments.

Presentation of meta-analysis results:

Internally, both common effect and random effects models are calculated regardless of values choosen for arguments `common` and `random`. Accordingly, the estimate for the random effects model can be extracted from component `TE.random` of an object of class `"meta"` even if argument `random = FALSE`. However, all functions in R package **meta** will adequately consider the values for `common` and `random`. For example, functions `print.meta` and `forest.meta` will not show results for the random effects model if `random = FALSE`.

A prediction interval will only be shown if `prediction = TRUE`.

Argument `pscale` can be used to rescale single proportions or risk differences, e.g. `pscale = 1000` means that proportions are expressed as events per 1000 observations. This is useful in situations with (very) low event probabilities.

Argument `irscale` can be used to rescale single rates or rate differences, e.g. `irscale = 1000` means that rates are expressed as events per 1000 time units, e.g. person-years. This is useful in situations with (very) low rates. Argument `irunit` can be used to specify the time unit used in individual studies (default: "person-years"). This information is printed in summaries and forest plots if argument `irscale` is not equal to 1.

Default settings for `common`, `random`, `pscale`, `irscale`, `irunit` and several other arguments can be set for the whole R session using `settings.meta`.

Value

An object of class `c("metagen", "meta")` with corresponding generic functions (see `meta-object`).

Note

R function `rma.uni` from R package **metafor** (Viechtbauer 2010) is called internally to estimate the between-study variance τ^2.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

References

Borenstein M, Hedges LV, Higgins JP, Rothstein HR (2010): A basic introduction to fixed-effect and random-effects models for meta-analysis. *Research Synthesis Methods*, **1**, 97–111

Cai S, Zhou J, Pan J (2021): Estimating the sample mean and standard deviation from order statistics and sample size in meta-analysis. *Statistical Methods in Medical Research*, **30**, 2701–2719

Luo D, Wan X, Liu J, Tong T (2018): Optimally estimating the sample mean from the sample size, median, mid-range, and/or mid-quartile range. *Statistical Methods in Medical Research*, **27**, 1785–805

McGrath S, Zhao X, Steele R, et al. and the DEPRESsion Screening Data (DEPRESSD) Collaboration (2020): Estimating the sample mean and standard deviation from commonly reported quantiles in meta-analysis. *Statistical Methods in Medical Research*, **29**, 2520–2537

Senn S (2024): The analysis of continuous data from n-of-1 trials using paired cycles: a simple tutorial. *Trials*, **25**.

Shi J, Luo D, Weng H, Zeng X-T, Lin L, Chu H, et al. (2020): Optimally estimating the sample standard deviation from the five-number summary. *Research Synthesis Methods*.

Viechtbauer W (2010): Conducting Meta-Analyses in R with the metafor Package. *Journal of Statistical Software*, **36**, 1–48

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

Wan X, Wang W, Liu J, Tong T (2014): Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range. *BMC Medical Research Methodology*, **14**, 135

See Also

meta-package, update.meta, metabin, metacont, pairwise, print.meta, settings.meta

Examples

```
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac, study,
  data = Fleiss1993bin, sm = "RR", method = "I")
m1

# Identical results using the generic inverse variance method with
# log risk ratio and its standard error:
# Note, argument 'n.e' in metagen() is used to provide the total
# sample size which is calculated from the group sample sizes n.e
# and n.c in meta-analysis m1.
m1.gen <- metagen(TE, seTE, studlab, n.e = n.e + n.c, data = m1, sm = "RR")
m1.gen
forest(m1.gen, leftcols = c("studlab", "n.e", "TE", "seTE"))

# Meta-analysis with prespecified between-study variance
#
metagen(m1$TE, m1$seTE, sm = "RR", tau.preset = sqrt(0.1))

# Meta-analysis of survival data:
#
logHR <- log(c(0.95, 1.5))
selogHR <- c(0.25, 0.35)
metagen(logHR, selogHR, sm = "HR")

# Paule-Mandel method to estimate between-study variance for data
# from Paule & Mandel (1982)
#
average <- c(27.044, 26.022, 26.340, 26.787, 26.796)
variance <- c(0.003, 0.076, 0.464, 0.003, 0.014)
#
metagen(average, sqrt(variance), sm = "MD", method.tau = "PM")
```

```
# Conduct meta-analysis using hazard ratios and 95% confidence intervals
#
# Data from Steurer et al. (2006), Analysis 1.1 Overall survival
# https://doi.org/10.1002/14651858.CD004270.pub2
#
study <- c("FCG on CLL 1996", "Leporrier 2001", "Rai 2000", "Robak 2000")
HR <- c(0.55, 0.92, 0.79, 1.18)
lower.HR <- c(0.28, 0.79, 0.59, 0.64)
upper.HR <- c(1.09, 1.08, 1.05, 2.17)
#
# Hazard ratios and confidence intervals as input
#
summary(metagen(HR, lower = lower.HR, upper = upper.HR,
  studlab = study, sm = "HR", transf = FALSE))
#
# Same result with log hazard ratios as input
#
summary(metagen(log(HR), lower = log(lower.HR), upper = log(upper.HR),
  studlab = study, sm = "HR"))
#
# Again, same result using an unknown summary measure and
# arguments 'func.transf' and 'func.backtransf'
#
summary(metagen(HR, lower = lower.HR, upper = upper.HR,
  studlab = study, sm = "Hazard ratio",
  func.transf = log, func.backtransf = exp))
#
# Finally, same result only providing argument 'func.transf' as the
# back-transformation for the logarithm is known
#
summary(metagen(HR, lower = lower.HR, upper = upper.HR,
  studlab = study, sm = "Hazard ratio",
  func.transf = log))

# Exclude MRC-1 and MRC-2 studies from meta-analysis, however,
# show them in printouts and forest plots
#
metabin(d.asp, n.asp, d.plac, n.plac, study,
  data = Fleiss1993bin, sm = "RR", method = "I",
  exclude = study %in% c("MRC-1", "MRC-2"))
#
# Exclude MRC-1 and MRC-2 studies completely from meta-analysis
#
metabin(d.asp, n.asp, d.plac, n.plac, study,
  data = Fleiss1993bin, sm = "RR", method = "I",
  subset = !(study %in% c("MRC-1", "MRC-2")))

# Exclude studies with total sample size above 1500
#
metabin(d.asp, n.asp, d.plac, n.plac, study,
  data = Fleiss1993bin, sm = "RR", method = "I",
  exclude = (n.asp + n.plac) > 1500)
```

```
# Exclude studies containing "MRC" in study name
#
metabin(d.asp, n.asp, d.plac, n.plac, study,
  data = Fleiss1993bin, sm = "RR", method = "I",
  exclude = grep("MRC", study))

# Use both arguments 'subset' and 'exclude'
#
metabin(d.asp, n.asp, d.plac, n.plac, study,
  data = Fleiss1993bin, sm = "RR", method = "I",
  subset = (n.asp + n.plac) > 1500,
  exclude = grep("MRC", study))

## Not run:
# Three-level model: effects of modified school calendars on
# student achievement
data(dat.konstantopoulos2011, package = "metadat")
metagen(yi, sqrt(vi), studlab = study, data = dat.konstantopoulos2011,
  sm = "SMD",
  cluster = district, detail.tau = c("district", "district/school"))

## End(Not run)
```

metainc *Meta-analysis of incidence rates*

Description

Calculation of common effect and random effects estimates (incidence rate ratio or incidence rate difference) for meta-analyses with event counts. Mantel-Haenszel, Cochran, inverse variance method, and generalised linear mixed model (GLMM) are available for pooling. For GLMMs, the `rma.glmm` function from R package **metafor** (Viechtbauer 2010) is called internally.

Usage

```
metainc(
  event.e,
  time.e,
  event.c,
  time.c,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  cluster = NULL,
  rho = 0,
  method = if (sm == "IRSD") "Inverse" else "MH",
  sm = gs("sminc"),
  incr = gs("incr"),
  method.incr = gs("method.incr"),
  incr.e = if (length(incr) > 1) incr else NULL,
  incr.c = if (length(incr) > 1) incr else NULL,
  model.glmm = "UM.FS",
  level = gs("level"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  overall = common | random,
  overall.hetstat = if (is.null(gs("overall.hetstat"))) common | random else
    gs("overall.hetstat"),
  prediction = gs("prediction") | !missing(method.predict),
  method.tau = ifelse(!is.na(charmatch(tolower(method), "glmm", nomatch = NA)), "ML",
    gs("method.tau")),
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.preset = NULL,
  TE.tau = NULL,
  tau.common = gs("tau.common"),
  detail.tau = NULL,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  method.bias = gs("method.bias"),
  n.e = NULL,
  n.c = NULL,
  backtransf = if (sm == "IRSD") FALSE else gs("backtransf"),
  irscale = 1,
  irunit = "person-years",
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title = gs("title"),
  complab = gs("complab"),
  outclab = "",
  label.e = gs("label.e"),
  label.c = gs("label.c"),
  label.left = gs("label.left"),
  label.right = gs("label.right"),
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  subgroup,
  subgroup.name = NULL,
  print.subgroup.name = gs("print.subgroup.name"),
  sep.subgroup = gs("sep.subgroup"),
  test.subgroup = gs("test.subgroup"),
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  byvar,
  hakn,
  adhoc.hakn,
  keepdata = gs("keepdata"),
  warn = gs("warn"),
  warn.deprecated = gs("warn.deprecated"),
  control = NULL,
  ...
)
```

Arguments

event.e Number of events in experimental group or an R object created with pairwise.

time.e Person time at risk in experimental group.

event.c Number of events in control group.

time.c Person time at risk in control group.

studlab An optional vector with study labels.

data An optional data frame containing the study information, i.e., event.e, time.e, event.c, and time.c.

subset An optional vector specifying a subset of studies to be used.

exclude An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots.

cluster An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.

rho Assumed correlation of estimates within a cluster.

method A character string indicating which method is to be used for pooling of studies. One of "MH", "Inverse", "Cochran", or "GLMM" can be abbreviated.

sm A character string indicating which summary measure ("IRR", "IRD", "IRSD", or "VE") is to be used for pooling of studies, see Details.

incr A numerical value which is added to cell frequencies for studies with a zero cell count or a numeric vector with the continuity correction for each study, see Details.

method.incr A character string indicating which continuity correction method should be used ("only0", "if0all", "all", or "user"), see Details.

incr.e Continuity correction in experimental group, see Details.

incr.c Continuity correction in control group, see Details.

`model.glmm` A character string indicating which GLMM should be used. One of `"UM.FS"`, `"UM.RS"`, and `"CM.EL"`, see Details.

`level` The level used to calculate confidence intervals for individual studies.

`common` A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`overall` A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

`overall.hetstat`
A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`prediction` A logical indicating whether a prediction interval should be printed.

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`tau.preset` Prespecified value for the square root of the between-study variance τ^2.

`TE.tau` Overall treatment effect used to estimate the between-study variance tau-squared.

`tau.common` A logical indicating whether tau-squared should be the same across subgroups.

`detail.tau` Detail on between-study variance estimate.

`method.I2` A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either `"Q"` or `"tau2"`, can be abbreviated (see `meta-package`).

`level.ma` The level used to calculate confidence intervals for meta-analysis estimates.

`method.random.ci`
A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

`adhoc.hakn.ci` A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see `meta-package`).

`level.predict` The level used to calculate prediction interval for a new study.

`method.predict` A character string indicating which method is used to calculate a prediction interval (see `meta-package`).

`adhoc.hakn.pi` A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see `meta-package`).

`seed.predict` A numeric value used as seed to calculate bootstrap prediction interval (see `meta-package`).

`method.bias` A character string indicating which test is to be used. Either `"Begg"`, `"Egger"`, or `"Thompson"`, can be abbreviated. See function `metabias`.

`n.e` Number of observations in experimental group (optional).

`n.c` Number of observations in control group (optional).

`backtransf` A logical indicating whether results for incidence rate ratio (`sm = "IRR"`) and vaccine efficacy or vaccine effectiveness (`sm = "VE"`) should be back transformed in printouts and plots. If TRUE (default), results will be presented as incidence rate ratios or vaccine efficacy / effectiveness; otherwise log incidence rate ratios or log vaccine rate ratios will be shown.

`irscale` A numeric defining a scaling factor for printing of incidence rate differences.

`irunit` A character string specifying the time unit used to calculate rates, e.g. person-years.

`text.common` A character string used in printouts and forest plot to label the pooled common effect estimate.

`text.random` A character string used in printouts and forest plot to label the pooled random effects estimate.

`text.predict` A character string used in printouts and forest plot to label the prediction interval.

`text.w.common` A character string used to label weights of common effect model.

`text.w.random` A character string used to label weights of random effects model.

`title` Title of meta-analysis / systematic review.

`complab` Comparison label.

`outclab` Outcome label.

`label.e` Label for experimental group.

`label.c` Label for control group.

`label.left` Graph label on left side of null effect in forest plot.

`label.right` Graph label on right side of null effect in forest plot.

`col.label.left` The colour of the graph label on the left side of the null effect.

`col.label.right`
The colour of the graph label on the right side of the null effect.

`subgroup` An optional vector to conduct a meta-analysis with subgroups.

`subgroup.name` A character string with a name for the subgroup variable.

`print.subgroup.name`
A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

`sep.subgroup` A character string defining the separator between name of subgroup variable and subgroup label.

`test.subgroup` A logical value indicating whether to print results of test for subgroup differences.

`prediction.subgroup`
A logical indicating whether prediction intervals should be printed for subgroups.

`seed.predict.subgroup`
: A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

`byvar` Deprecated argument (replaced by 'subgroup').

`hakn` Deprecated argument (replaced by 'method.random.ci').

`adhoc.hakn` Deprecated argument (replaced by 'adhoc.hakn.ci').

`keepdata` A logical indicating whether original data (set) should be kept in meta object.

`warn` A logical indicating whether warnings should be printed (e.g., if `incr` is added to studies with zero cell frequencies or if estimation problems exist in fitting a GLMM).

`warn.deprecated`
: A logical indicating whether warnings should be printed if deprecated arguments are used.

`control` An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to `rma.uni` or `rma.glmm`, respectively.

`...` Additional arguments passed on to `rma.glmm` function and to catch deprecated arguments.

Details

Calculation of common and random effects estimates for meta-analyses comparing two incidence rates.

The following measures of treatment effect are available:

- Incidence Rate Ratio (`sm = "IRR"`)
- Incidence Rate Difference (`sm = "IRD"`)
- Square root transformed Incidence Rate Difference (`sm = "IRSD"`)
- Vaccine efficacy or vaccine effectiveness (`sm = "VE"`)

Note, log incidence rate ratio (logIRR) and log vaccine ratio (logVR) are mathematical identical, however, back-transformed results differ as vaccine efficacy or effectiveness is defined as `VE = 100 * (1 - IRR)`.

A three-level random effects meta-analysis model (Van den Noortgate et al., 2013) is utilised if argument `cluster` is used and at least one cluster provides more than one estimate. Internally, `rma.mv` is called to conduct the analysis and `weights.rma.mv` with argument `type = "rowsum"` is used to calculate random effects weights.

Default settings are utilised for several arguments (assignments using `gs` function). These defaults can be changed for the current R session using the `settings.meta` function.

Furthermore, R function `update.meta` can be used to rerun a meta-analysis with different settings.

Meta-analysis method:

By default, both common effect and random effects models are considered (see arguments `common` and `random`). If `method` is `"MH"` (default), the Mantel-Haenszel method is used to calculate the common effect estimate (Greenland & Robbins, 1985); if `method` is `"Inverse"`, inverse variance weighting is used for pooling; if `method` is `"Cochran"`, the Cochran method is used for pooling

(Bayne-Jones, 1964, Chapter 8). For these three methods, the random effects estimate is always based on the inverse variance method.

A distinctive and frequently overlooked advantage of incidence rates is that individual patient data (IPD) can be extracted from count data. Accordingly, statistical methods for IPD, i.e., generalised linear mixed models, can be utilised in a meta-analysis of incidence rate ratios (Stijnen et al., 2010). These methods are available (argument `method = "GLMM"`) for the common effect and random effects model by calling the `rma.glmm` function from R package **metafor** internally.

Three different GLMMs are available for meta-analysis of incidence rate ratios using argument `model.glmm` (which corresponds to argument `model` in the `rma.glmm` function):

1. Poisson regression model with fixed study effects (default)
   (`model.glmm = "UM.FS"`, i.e., **U**nconditional **M**odel - **F**ixed **S**tudy effects)
2. Mixed-effects Poisson regression model with random study effects
   (`model.glmm = "UM.RS"`, i.e., **U**nconditional **M**odel - **R**andom **S**tudy effects)
3. Generalised linear mixed model (conditional Poisson-Normal)
   (`model.glmm = "CM.EL"`, i.e., **C**onditional **M**odel - **E**xact **L**ikelihood)

Details on these three GLMMs as well as additional arguments which can be provided using argument '`...`' in `metainc` are described in `rma.glmm` where you can also find information on the iterative algorithms used for estimation. Note, regardless of which value is used for argument `model.glmm`, results for two different GLMMs are calculated: common effect model (with fixed treatment effect) and random effects model (with random treatment effects).

Continuity correction:

Four approaches are available to apply a continuity correction:

- Only studies with a zero cell count (`method.incr = "only0", default`)
- All studies if at least one study has a zero cell count (`method.incr = "if0all"`)
- All studies irrespective of zero cell counts (`method.incr = "all"`)
- Use values provided in arguments `incr.e` and `incr.c` (`method.incr = "user"`)

For studies with a zero cell count, by default, 0.5 is added to all cell frequencies of these studies (argument `incr`). This continuity correction is used both to calculate individual study results with confidence limits and to conduct meta-analysis based on the inverse variance method. For Mantel-Haenszel method, Cochran method, and GLMMs, nothing is added to zero cell counts. Accordingly, estimates for these methods are not defined if the number of events is zero in all studies either in the experimental or control group.

Subgroup analysis:

Argument `subgroup` can be used to conduct subgroup analysis for a categorical covariate. The `metareg` function can be used instead for more than one categorical covariate or continuous covariates.

Exclusion of studies from meta-analysis:

Arguments `subset` and `exclude` can be used to exclude studies from the meta-analysis. Studies are removed completely from the meta-analysis using argument `subset`, while excluded studies are shown in printouts and forest plots using argument `exclude` (see Examples in `metagen`). Meta-analysis results are the same for both arguments.

Presentation of meta-analysis results:

Internally, both common effect and random effects models are calculated regardless of values choosen for arguments common and random. Accordingly, the estimate for the random effects model can be extracted from component TE.random of an object of class "meta" even if argument random = FALSE. However, all functions in R package **meta** will adequately consider the values for common and random. E.g. function print.meta will not print results for the random effects model if random = FALSE.

A prediction interval will only be shown if prediction = TRUE.

Value

An object of class c("metainc", "meta") with corresponding generic functions (see meta-object).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Bayne-Jones S et al. (1964): Smoking and Health: Report of the Advisory Committee to the Surgeon General of the United States. U-23 Department of Health, Education, and Welfare. Public Health Service Publication No. 1103.

Greenland S & Robins JM (1985): Estimation of a common effect parameter from sparse follow-up data. *Biometrics*, **41**, 55–68

Stijnen T, Hamza TH, Ozdemir P (2010): Random effects meta-analysis of event outcome in the framework of the generalized linear mixed model with applications in sparse data. *Statistics in Medicine*, **29**, 3046–67

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

Viechtbauer W (2010): Conducting Meta-Analyses in R with the Metafor Package. *Journal of Statistical Software*, **36**, 1–48

See Also

meta-package, metabin, update.meta, print.meta

Examples

```
data(smoking)
m1 <- metainc(d.smokers, py.smokers, d.nonsmokers, py.nonsmokers,
  data = smoking, studlab = study)
print(m1, digits = 2)

m2 <- update(m1, method = "Cochran")
print(m2, digits = 2)

data(lungcancer)
m3 <- metainc(d.smokers, py.smokers, d.nonsmokers, py.nonsmokers,
  data = lungcancer, studlab = study)
```

```
print(m3, digits = 2)

# Redo Cochran meta-analysis with inflated standard errors
#
# All cause mortality
#
TEa <- log((smoking$d.smokers/smoking$py.smokers) /
  (smoking$d.nonsmokers/smoking$py.nonsmokers))
seTEa <- sqrt(1 / smoking$d.smokers + 1 / smoking$d.nonsmokers +
  2.5 / smoking$d.nonsmokers)
metagen(TEa, seTEa, sm = "IRR", studlab = smoking$study)

# Lung cancer mortality
#
TEl <- log((lungcancer$d.smokers/lungcancer$py.smokers) /
  (lungcancer$d.nonsmokers/lungcancer$py.nonsmokers))
seTEl <- sqrt(1 / lungcancer$d.smokers + 1 / lungcancer$d.nonsmokers +
  2.25 / lungcancer$d.nonsmokers)
metagen(TEl, seTEl, sm = "IRR", studlab = lungcancer$study)

## Not run:
# Meta-analysis using generalised linear mixed models
# (only if R packages 'metafor' and 'lme4' are available)

# Poisson regression model (fixed study effects)
#
m4 <- metainc(d.smokers, py.smokers, d.nonsmokers, py.nonsmokers,
  data = smoking, studlab = study, method = "GLMM")
m4

# Mixed-effects Poisson regression model (random study effects)
#
update(m4, model.glmm = "UM.RS", nAGQ = 1)
#
# Generalised linear mixed model (conditional Poisson-Normal)
#
update(m4, model.glmm = "CM.EL")

## End(Not run)
```

`metainf.meta` *Influence analysis in meta-analysis using leave-one-out method*

Description

Performs an influence analysis. Pooled estimates are calculated omitting one study at a time.

Usage

```
## S3 method for class 'meta'
metainf(
  x,
  pooled,
  sortvar,
  prediction,
  overall = x$overall,
  text.pooled,
  no = 1,
  cid = NULL,
  cid.below.null = NULL,
  cid.above.null = NULL,
  small.values = "desirable",
  ...
)

metainf(x, ...)

## Default S3 method:
metainf(x, ...)
```

Arguments

`x` An object of class `meta`.

`pooled` A character string indicating whether a common effect or random effects model is used for pooling. Either missing (see Details), `"common"`, or `"random"`, can be abbreviated.

`sortvar` An optional vector used to sort the individual studies (must be of same length as `x$TE`).

`prediction` A logical indicating whether to report prediction intervals.

`overall` A logical indicating whether overall results should be reported.

`text.pooled` A character string used in printouts and forest plots to label the pooled effect estimate.

`no` A numeric specifying which meta-analysis results to consider.

`cid` A numeric value or vector specifying clinically important differences (CID) / decision thresholds used to calculate expected proportions of clinically important benefit or harm (see `cidprop`).

`cid.below.null` A single numeric defining the decision threshold below the null effect to distinguish clinically important from not important effects (see `cidprop`).

`cid.above.null` A single numeric defining the decision threshold above the null effect to distinguish clinically important from not important effects (see `cidprop`).

`small.values` A character string specifying whether small treatment effects indicate a beneficial (`"desirable"`) or harmful (`"undesirable"`) effect, can be abbreviated (see `cidprop`).

`...` Additional arguments (ignored).

Details

Performs a influence analysis; pooled estimates are calculated omitting one study at a time. Studies are sorted according to `sortvar`.

Information from object `x` is utilised if argument `pooled` is missing. A common effect model is assumed (`pooled = "common"`) if argument `x$common` is `TRUE`; a random effects model is assumed (`pooled = "random"`) if argument `x$random` is `TRUE` and `x$common` is `FALSE`.

Value

An object of class `"metainf"` with dedicated print and forest functions.

The following list elements provide results from meta-analyses, each excluding one study at a time (see `meta-object` for more information on these list elements):

> studlab, TE, seTE, df.random, lower, upper, statistic, pval,
> lower.predict, upper.predict, df.predict, w (sum of weights),
> tau2, se.tau2, lower.tau2, upper.tau2, tau, lower.tau, upper.tau,
> I2, lower.I2, upper.I2, Rb, n.harmonic.mean, t.harmonic.mean,
> k, k.study, k.all, k.TE, k.MH.

The following list elements contain results of the original meta-analysis:

> TE.pooled, seTE.pooled, df.random.pooled,
> lower.pooled, upper.pooled, statistic.pooled, pval.pooled,
> lower.predict.pooled, upper.predict.pooled,
> df.predict.pooled, w.pooled,
> tau2.pooled, se.tau2.pooled, lower.tau2.pooled, upper.tau2.pooled,
> tau.pooled, lower.tau.pooled, upper.tau.pooled,
> I2.pooled, lower.I2.pooled, upper.I2.pooled, Rb.pooled,
> n.harmonic.mean.pooled, t.harmonic.mean.pooled,
> k.pooled, k.study.pooled, k.all.pooled, k.TE.pooled, k.MH.pooled.

Author(s)

Guido Schwarzer `<guido.schwarzer@uniklinik-freiburg.de>`

References

Cooper H & Hedges LV (1994): *The Handbook of Research Synthesis*. Newbury Park, CA: Russell Sage Foundation

See Also

`forest.metainf`, `print.metainf`, `cidprop`

Examples

```
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac,
  data = Fleiss1993bin, studlab = study, sm = "RR", method = "I")
m1
metainf(m1)
metainf(m1, pooled = "random")

forest(metainf(m1))
forest(metainf(m1, pooled = "random"))

metainf(m1, sortvar = study)
metainf(m1, sortvar = 7:1)

m2 <- update(m1, title = "Fleiss1993bin meta-analysis", backtransf = FALSE)
metainf(m2)

data(Fleiss1993cont)
m3 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD")
metainf(m3)
```

---

metamean *Meta-analysis of single means*

Description

Calculation of an overall mean from studies reporting a single mean using the inverse variance method for pooling; inverse variance weighting is used for pooling.

Usage

```
metamean(
  n,
  mean,
  sd,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  cluster = NULL,
  rho = 0,
  median,
  q1,
  q3,
  min,
  max,
  method.mean = "Luo",
  method.sd = "Shi",
  approx.mean,
  approx.sd,
  sm = gs("smmean"),
  method.ci = gs("method.ci.cont"),
  level = gs("level"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  overall = common | random,
  overall.hetstat = if (is.null(gs("overall.hetstat"))) common | random else
    gs("overall.hetstat"),
  prediction = gs("prediction") | !missing(method.predict),
  method.tau = gs("method.tau"),
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.preset = NULL,
  TE.tau = NULL,
  tau.common = gs("tau.common"),
  detail.tau = NULL,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  null.effect = NA,
  method.bias = gs("method.bias"),
  backtransf = gs("backtransf"),
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title = gs("title"),
  complab = gs("complab"),
  outclab = "",
  label.left = gs("label.left"),
  label.right = gs("label.right"),
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  subgroup,
  subgroup.name = NULL,
  print.subgroup.name = gs("print.subgroup.name"),
  sep.subgroup = gs("sep.subgroup"),
  test.subgroup = gs("test.subgroup"),
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  byvar,
  adhoc.hakn,
  keepdata = gs("keepdata"),
  warn = gs("warn"),
  warn.deprecated = gs("warn.deprecated"),
  control = NULL,
  ...
)
```

Arguments

n	Number of observations.
mean	Estimated mean.
sd	Standard deviation.
studlab	An optional vector with study labels.
data	An optional data frame containing the study information.
subset	An optional vector specifying a subset of studies to be used.
exclude	An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots.
cluster	An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.
rho	Assumed correlation of estimates within a cluster.
median	Median (used to estimate the mean and standard deviation).
q1	First quartile (used to estimate the mean and standard deviation).
q3	Third quartile (used to estimate the mean and standard deviation).
min	Minimum (used to estimate the mean and standard deviation).
max	Maximum (used to estimate the mean and standard deviation).
method.mean	A character string indicating which method to use to approximate the mean from the median and other statistics (see Details).
method.sd	A character string indicating which method to use to approximate the standard deviation from sample size, median, interquartile range and range (see Details).
approx.mean	Approximation method to estimate means (see Details).
approx.sd	Approximation method to estimate standard deviations (see Details).
sm	A character string indicating which summary measure ("MRAW" or "MLN") is to be used for pooling of studies.
method.ci	A character string indicating which method is used to calculate confidence intervals for individual studies, see Details.
level	The level used to calculate confidence intervals for individual studies.
common	A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`overall` A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

`overall.hetstat`
A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`prediction` A logical indicating whether a prediction interval should be printed.

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`tau.preset` Prespecified value for the square root of the between-study variance τ^2.

`TE.tau` Overall treatment effect used to estimate the between-study variance tau-squared.

`tau.common` A logical indicating whether tau-squared should be the same across subgroups.

`detail.tau` Detail on between-study variance estimate.

`method.I2` A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either `"Q"` or `"tau2"`, can be abbreviated (see `meta-package`).

`level.ma` The level used to calculate confidence intervals for meta-analysis estimates.

`method.random.ci`
A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

`adhoc.hakn.ci` A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see `meta-package`).

`level.predict` The level used to calculate prediction interval for a new study.

`method.predict` A character string indicating which method is used to calculate a prediction interval (see `meta-package`).

`adhoc.hakn.pi` A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see `meta-package`).

`seed.predict` A numeric value used as seed to calculate bootstrap prediction interval (see `meta-package`).

`null.effect` A numeric value specifying the effect under the null hypothesis.

`method.bias` A character string indicating which test is to be used. Either `"Begg"`, `"Egger"`, or `"Thompson"`, can be abbreviated. See function `metabias`.

`backtransf` A logical indicating whether results should be back transformed in printouts and plots for `sm = "MLN"`. If TRUE (default), results will be presented as means; otherwise logarithm of means will be shown.

`text.common` A character string used in printouts and forest plot to label the pooled common effect estimate.

`text.random` A character string used in printouts and forest plot to label the pooled random effects estimate.

`text.predict` A character string used in printouts and forest plot to label the prediction interval.

`text.w.common` A character string used to label weights of common effect model.

`text.w.random` A character string used to label weights of random effects model.

`title` Title of meta-analysis / systematic review.

`complab` Comparison label.

`outclab` Outcome label.

`label.left` Graph label on left side of null effect in forest plot.

`label.right` Graph label on right side of null effect in forest plot.

`col.label.left` The colour of the graph label on the left side of the null effect.

`col.label.right`
The colour of the graph label on the right side of the null effect.

`subgroup` An optional vector to conduct a meta-analysis with subgroups.

`subgroup.name` A character string with a name for the subgroup variable.

`print.subgroup.name`
A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

`sep.subgroup` A character string defining the separator between name of subgroup variable and subgroup label.

`test.subgroup` A logical value indicating whether to print results of test for subgroup differences.

`prediction.subgroup`
A logical indicating whether prediction intervals should be printed for subgroups.

`seed.predict.subgroup`
A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

`byvar` Deprecated argument (replaced by 'subgroup').

`adhoc.hakn` Deprecated argument (replaced by 'adhoc.hakn.ci').

`keepdata` A logical indicating whether original data (set) should be kept in meta object.

`warn` A logical indicating whether warnings should be printed (e.g., if studies are excluded from meta-analysis due to zero standard deviations).

`warn.deprecated`
A logical indicating whether warnings should be printed if deprecated arguments are used.

`control` An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to `rma.uni`.

`...` Additional arguments (to catch deprecated arguments).

Details

Common effect and random effects meta-analysis of single means to calculate an overall mean; inverse variance weighting is used for pooling. Note, you should use R function metacont to compare means of pairwise comparisons instead of using metamean for each treatment arm separately which will break randomisation in randomised controlled trials.

A three-level random effects meta-analysis model (Van den Noortgate et al., 2013) is utilised if argument cluster is used and at least one cluster provides more than one estimate. Internally, rma.mv is called to conduct the analysis and weights.rma.mv with argument type = "rowsum" is used to calculate random effects weights.

Default settings are utilised for several arguments (assignments using gs function). These defaults can be changed for the current R session using the settings.meta function.

Furthermore, R function update.meta can be used to rerun a meta-analysis with different settings.

The following transformations of means are implemented to calculate an overall mean:

- Raw, i.e. untransformed, means (sm = "MRAW", default)
- Log transformed means (sm = "MLN")

Calculations are conducted on the log scale if sm = "MLN". Accordingly, list elements TE, TE.common, and TE.random contain the logarithm of means. In printouts and plots these values are back transformed if argument backtransf = TRUE (default).

Approximate means from sample sizes, medians and other statistics:

Missing means can be derived from

1. sample size, median, interquartile range and range (arguments n, median, q1, q3, min, and max),
2. sample size, median and interquartile range (arguments n, median, q1, and q3), or
3. sample size, median and range (arguments n, median, min, and max).

By default, methods described in Luo et al. (2018) are utilised (argument method.mean = "Luo"):

- equation (15) if sample size, median, interquartile range and range are available,
- equation (11) if sample size, median and interquartile range are available,
- equation (7) if sample size, median and range are available.

Instead the methods described in Wan et al. (2014) are used if argument method.mean = "Wan":

- equation (10) if sample size, median, interquartile range and range are available,
- equation (14) if sample size, median and interquartile range are available,
- equation (2) if sample size, median and range are available.

The following methods are also available to estimate means from quantiles or ranges if R package **estmeansd** is installed:

- Method for Unknown Non-Normal Distributions (MLN) approach (Cai et al. (2021), argument method.mean = "Cai"),
- Quantile Estimation (QE) method (McGrath et al. (2020), argument method.mean = "QE-McGrath")),
- Box-Cox (BC) method (McGrath et al. (2020), argument method.mean = "BC-McGrath")).

By default, missing means are replaced successively using interquartile ranges and ranges (if available), interquartile ranges (if available) and finally ranges. Argument approx.mean can be used to overwrite this behaviour for each individual study and treatment arm:

- use means directly (entry "" in argument approx.mean);
- median, interquartile range and range ("iqr.range");
- median and interquartile range ("iqr");
- median and range ("range").

Approximate standard deviations from sample sizes, medians and other statistics:

Missing standard deviations can be derived from

1. sample size, median, interquartile range and range (arguments n, median, q1, q3, min, and max),
2. sample size, median and interquartile range (arguments n, median, q1 and q3), or
3. sample size, median and range (arguments n, median, min and max).

Wan et al. (2014) describe methods to estimate the standard deviation from the sample size, median and additional statistics. Shi et al. (2020) provide an improved estimate of the standard deviation if the interquartile range and range are available in addition to the sample size and median. Accordingly, equation (11) in Shi et al. (2020) is the default (argument method.sd = "Shi"), if the median, interquartile range and range are provided. The method by Wan et al. (2014) is used if argument method.sd = "Wan" and, depending on the sample size, either equation (12) or (13) is used. If only the interquartile range or range is available, equations (15) / (16) and (7) / (9) in Wan et al. (2014) are used, respectively.

The following methods are also available to estimate standard deviations from quantiles or ranges if R package **estmeansd** is installed:

- Method for Unknown Non-Normal Distributions (MLN) approach (Cai et al. (2021), argument method.mean = "Cai"),
- Quantile Estimation (QE) method (McGrath et al. (2020), argument method.mean = "QE-McGrath")),
- Box-Cox (BC) method (McGrath et al. (2020), argument method.mean = "BC-McGrath")).

By default, missing standard deviations are replaced successively using these method, i.e., interquartile ranges and ranges are used before interquartile ranges before ranges. Argument approx.sd can be used to overwrite this default for each individual study and treatment arms:

- sample size, median, interquartile range and range ("iqr.range");
- sample size, median and interquartile range ("iqr");
- sample size, median and range ("range").

Confidence intervals for individual studies:

For untransformed means (argument sm = "MRAW"), the confidence interval for individual studies can be based on the

- standard normal distribution (method.ci = "z", default), or
- t-distribution (method.ci = "t").

Subgroup analysis:

Argument subgroup can be used to conduct subgroup analysis for a categorical covariate. The metareg function can be used instead for more than one categorical covariate or continuous covariates.

Exclusion of studies from meta-analysis:

Arguments subset and exclude can be used to exclude studies from the meta-analysis. Studies are removed completely from the meta-analysis using argument subset, while excluded studies are shown in printouts and forest plots using argument exclude (see Examples in metagen). Meta-analysis results are the same for both arguments.

Presentation of meta-analysis results:

Internally, both common effect and random effects models are calculated regardless of values choosen for arguments common and random. Accordingly, the estimate for the random effects model can be extracted from component TE.random of an object of class "meta" even if argument random = FALSE. However, all functions in R package **meta** will adequately consider the values for common and random. E.g. functions print.meta and forest.meta will not print results for the random effects model if random = FALSE.

A prediction interval will only be shown if prediction = TRUE.

Value

An object of class c("metamean", "meta") with corresponding generic functions (see meta-object).

Note

The function metagen is called internally to calculate individual and overall treatment estimates and standard errors.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Cai S, Zhou J, Pan J (2021): Estimating the sample mean and standard deviation from order statistics and sample size in meta-analysis. *Statistical Methods in Medical Research*, **30**, 2701–2719

Luo D, Wan X, Liu J, Tong T (2018): Optimally estimating the sample mean from the sample size, median, mid-range, and/or mid-quartile range. *Statistical Methods in Medical Research*, **27**, 1785–805

McGrath S, Zhao X, Steele R, et al. and the DEPRESsion Screening Data (DEPRESSD) Collaboration (2020): Estimating the sample mean and standard deviation from commonly reported quantiles in meta-analysis. *Statistical Methods in Medical Research*, **29**, 2520–2537

Shi J, Luo D, Weng H, Zeng X-T, Lin L, Chu H, et al. (2020): Optimally estimating the sample standard deviation from the five-number summary. *Research Synthesis Methods*.

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

Wan X, Wang W, Liu J, Tong T (2014): Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range. *BMC Medical Research Methodology*, **14**, 135

See Also

meta-package, update.meta, metamean, metagen

Examples

```
m1 <- metamean(rep(100, 3), 1:3, rep(1, 3))
m1

m2 <- update(m1, sm = "MLN")
m2

# With test for overall mean equal to 2
#
update(m1, null.effect = 2)
update(m2, null.effect = 2)

# Print results without back-transformation
#
update(m1, backtransf = FALSE)
update(m2, backtransf = FALSE)
update(m1, null.effect = 2, backtransf = FALSE)
update(m2, null.effect = 2, backtransf = FALSE)
```

metamerge *Merge results of two meta-analyses on the same data set*

Description

This function can be used to merge results of two meta-analyses into a single meta-analysis object if they are based on the same data set. This is, for example, useful to produce a forest plot of a random-effects meta-analysis with different estimates of the between-study variance τ^2.

Usage

```
metamerge(
  meta1,
  meta2,
  common1 = meta1$common,
  random1 = meta1$random,
  prediction1 = meta1$prediction,
  common2 = meta2$common,
  random2 = meta2$random,
  prediction2 = meta2$prediction,
  label1 = NULL,
  label2 = NULL,
  label1.common = label1,
  label2.common = label2,
  label1.random = label1,
  label2.random = label2,
  label1.predict = label1,
  label2.predict = label2,
  label1.subgroup = label1,
  label2.subgroup = label2,
  hetlabel1 = label1.random,
  hetlabel2 = label2.random,
  taulabel1 = label1.random,
  taulabel2 = label2.random,
  text.pooled1 = NULL,
  text.pooled2 = NULL,
  text.w.pooled1 = NULL,
  text.w.pooled2 = NULL,
  text.common1 = text.pooled1,
  text.common2 = text.pooled2,
  text.random1 = text.pooled1,
  text.random2 = text.pooled2,
  text.predict1 = text.pooled1,
  text.predict2 = text.pooled2,
  text.w.common1 = text.w.pooled1,
  text.w.common2 = text.w.pooled2,
  text.w.random1 = text.w.pooled1,
  text.w.random2 = text.w.pooled2,
  keep = FALSE,
  keep.Q = keep,
  keep.I2 = keep.Q,
  keep.w = keep,
  common = common1 | common2,
  random = random1 | random2,
  overall = common | random,
  overall.hetstat = common | random,
  prediction = prediction1 | prediction2,
  backtransf,
  warn.deprecated = gs("warn.deprecated"),
  pooled1,
  pooled2
)
```

Arguments

`meta1` First meta-analysis object (see Details).

`meta2` Second meta-analysis object (see Details).

`common1` A logical indicating whether results of common effect model should be considered for first meta-analysis.

`random1` A logical indicating whether results of random effects model should be considered for first meta-analysis.

`prediction1` A logical indicating whether prediction interval should be considered for first meta-analysis.

`common2` A logical indicating whether results of common effect model should be considered for second meta-analysis.

`random2` A logical indicating whether results of random effects model should be considered for second meta-analysis.

`prediction2` A logical indicating whether prediction interval should be considered for second meta-analysis.

`label1` Default setting for arguments 'label1.common', 'label1.random', 'label1.predict' and 'label1.subgroup'.

`label2` Default setting for arguments 'label2.common', 'label2.random', 'label2.predict' and 'label2.subgroup'.

`label1.common` A character string to label the common effect estimate from the first meta-analysis.

`label2.common` A character string to label the common effect estimate from the second meta-analysis.

`label1.random` A character string to label the random effects estimate from the first meta-analysis (default label for arguments 'hetlabel1' and 'taulabel1').

`label2.random` A character string to label the random effects estimate from the second meta-analysis (default label for arguments 'hetlabel2' and 'taulabel2').

`label1.predict` A character string to label the prediction interval from the first meta-analysis.

`label2.predict` A character string to label the prediction interval from the second meta-analysis.

`label1.subgroup` A character string to label the subgroup results from the first meta-analysis.

`label2.subgroup` A character string to label the subgroup results from the second meta-analysis.

`hetlabel1` A character string used to label heterogeneity statistics of the first meta-analysis.

`hetlabel2` A character string used to label heterogeneity statistics of the second meta-analysis.

`taulabel1` A character string used to label estimate of between-study variance of the first meta-analysis.

`taulabel2` A character string used to label estimate of between-study variance of the second meta-analysis.

`text.pooled1` A character string used in printouts and forest plot to label the results from the first meta-analysis.

`text.pooled2` A character string used in printouts and forest plot to label the results from the second meta-analysis.

`text.w.pooled1` A character string used to label weights of the first meta-analysis; can be of same length as the number of pooled estimates requested in argument `pooled1`.

`text.w.pooled2` A character string used to label weights of the second meta-analysis; can be of same length as the number of pooled estimates requested in argument `pooled1`.

`text.common1` A character string used in printouts and forest plot to label results for common effect models from the first meta-analysis.

`text.common2` A character string used in printouts and forest plot to label results for common effect models from the second meta-analysis.

text.random1 A character string used in printouts and forest plot to label results for random effects models from the first meta-analysis.

text.random2 A character string used in printouts and forest plot to label results for random effects models from the second meta-analysis.

text.predict1 A character string used in printouts and forest plot to label prediction interval from the first meta-analysis.

text.predict2 A character string used in printouts and forest plot to label prediction interval from the second meta-analysis.

text.w.common1 A character string used to label common effect weights of the first meta-analysis; can be of same length as the number of common effect estimates.

text.w.common2 A character string used to label common effect weights of the second meta-analysis; can be of same length as the number of common effect estimates.

text.w.random1 A character string used to label random effects weights of the first meta-analysis; can be of same length as the number of random effects estimates.

text.w.random2 A character string used to label random effects weights of the second meta-analysis; can be of same length as the number of random effects estimates.

keep A logical indicating whether to keep additional information from second meta-analysis.

keep.Q A logical indicating whether heterogeneity statistic Q of second meta-analysis should be kept or ignored.

keep.I2 A logical indicating whether heterogeneity statistic I2 of second meta-analysis should be kept or ignored.

keep.w A logical indicating whether weights of the second meta-analysis should be kept or ignored.

common A logical indicating whether results of common effect meta-analyses should be reported.

random A logical indicating whether results of random effects meta-analyses should be reported.

overall A logical indicating whether overall summaries should be reported.

overall.hetstat

A logical value indicating whether to print heterogeneity measures for overall treatment comparisons.

prediction A logical indicating whether prediction intervals should be reported.

backtransf A logical indicating whether results should be back transformed in printouts and plots. If backtransf=TRUE (default), results for sm="OR" are printed as odds ratios rather than log odds ratios, for example.

warn.deprecated

A logical indicating whether warnings should be printed if deprecated arguments are used.

pooled1 Deprecated argument (replaced by 'common1', 'random1', 'prediction1'). A character string indicating whether results of common effect or random effects model should be considered for first meta-analysis. Either "both", "common" or "random", can be abbreviated.

`pooled2`	Deprecated argument (replaced by 'common2', 'random2', 'prediction2'). A character string indicating whether results of common effect or random effects model should be considered for second meta-analysis. Either `"both"`, `"common"` or `"random"`, can be abbreviated.

Details

In R package **meta**, objects of class `"meta"` contain results of both common effect and random effects meta-analyses. This function enables the user to merge the results of two meta-analysis object if they are based on the same data set.

Applications of this function include printing and plotting results of the common effect or random effects meta-analysis and the

- trim-and-fill method (`trimfill`),
- limit meta-analyis (`limitmeta` from R package **metasens**),
- Copas selection model (`copas` from R package **metasens**),
- robust variance meta-analysis model (`robu` from R package **robumeta**).

The first argument (`meta1`) must be an object created by a meta-analysis function (see `meta-object`). If an object created with `limitmeta` or `copas` is provided as the first argument, this object will be returned, i.e., argument `meta2` will be ignored.

The second meta-analysis could be an object created by a meta-analysis function or with `trimfill`, `limitmeta`, `copas`, or `robu`.

The created meta-analysis object only contains the study results, i.e., estimated effects and confidence intervals, from the first meta-analysis which are shown in printouts and forest plots. This only makes a difference for meta-analysis methods where individual study results differ, e.g., Mantel-Haenszel and Peto method for binary outcomes (see `metabin`).

R function `metaadd` can be used to add pooled results from any (external) meta-analysis.

R function `metabind` can be used to print and plot the results of several meta-analyses without the restriction that the same data set has to be used. Accordingly, individual study results are ignored.

Value

An object of class `"meta"` and `"metamerge"` with corresponding generic functions (see `meta-object`).

The following list elements have a different meaning:

`TE`, `seTE`, `studlab`	Treatment estimate, standard error, and study labels (first meta-analysis).
`lower`, `upper`	Lower and upper confidence interval limits for individual studies (first meta-analysis).
`statistic`, `pval`	Statistic and p-value for test of treatment effect for individual studies (first meta-analysis.
`w.common`	Vector or matrix with common effect weights.
`w.random`	Vector or matrix with random effects weights.
`k`	Vector with number of estimates (same length as number of common effect and random effects estimates).

k.study Vector with number of studies (same length as number of common effect and random effects estimates).

k.all Vector with total number of studies (same length as number of common effect and random effects estimates).

k.TE Vector with number of studies with estimable effects (same length as number of common effect and random effects estimates).

k.MH Vector with number of studies combined with Mantel-Haenszel method (same length as number of common effect and random effects estimates).

TE.common Vector with common effect estimates.

seTE.common Vector with standard errors of common effect estimates.

lower.common Vector with lower confidence limits (common effect model).

upper.common Vector with upper confidence limits (common effect model).

statistic.common Vector with test statistics for test of overall effect (common effect model).

pval.common Vector with p-value of test for overall effect (common effect model).

TE.random Vector with random effects estimates.

seTE.random Vector with standard errors of random effects estimates.

lower.random Vector with lower confidence limits (random effects model).

upper.random Vector with upper confidence limits (random effects model).

statistic.random Vector with test statistics for test of overall effect (random effects model).

pval.random Vector with p-value of test for overall effect (random effects model).

Furthermore, meta-analysis results of common effect or random effects model are taken from first meta-analysis if only random effects or common effects models are selected from both meta-analyses (arguments pooled1 and pooled2).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

metagen, metabind, metaadd

Examples

```
# Print results with more significant digits and do not show confidence
# intervals for tau^2 and tau
oldset <- settings.meta(digits = 6, digits.stat = 4, digits.pval = 6,
  digits.Q = 6, digits.I2 = 4, digits.H = 4,
  print.tau2.ci = FALSE, print.tau.ci = FALSE)
oldopts <- options(width = 120)

data(Fleiss1993bin)
```

```
# Mantel-Haenszel method
m1 <- metabin(d.asp, n.asp, d.plac, n.plac, data = Fleiss1993bin,
  studlab = paste(study, year), sm = "OR")
# Peto method
m2 <- update(m1, method = "Peto")
# Inverse variance method (only common effect model)
m3 <- update(m2, method = "Inverse", random = FALSE)

# Merge results from MH and Peto method
# - show individual results for MH method
#   (as this is the first meta-analysis)
# - keep all additional information from Peto meta-analysis (i.e.,
#   weights, Q statistic and I2 statistic)
m12 <- metamerge(m1, m2,
  label1 = "REML", label2 = "REML-Peto",
  label1.common = "MH", label2.common = "Peto",
  text.common1 = "Mantel-Haenszel method",
  text.common2 = "Peto method",
  text.w.random1 = "REML", text.w.random2 = "REML-Peto",
  hetlabel1 = "MH/IV", hetlabel2 = "Peto",
  keep = TRUE)

# Add common effect results from inverse variance method
# - keep weights from IV meta-analysis
# - Q and I2 statistic are identical for sm = "MH" and sm = "Inverse"
#   as inverse variance method is used for sm = "MH" under random
#   effects model
m123 <- metamerge(m12, m3,
  label2 = "IV",
  text.common2 = "Inverse variance method",
  keep.w = TRUE)
summary(m123)
## Not run:
forest(m123, digits = 6)

# Merge results (show individual results for Peto method)
m21 <- metamerge(m2, m1,
  label1 = "REML-Peto", label2 = "REML",
  label1.common = "Peto", label2.common = "MH",
  hetlabel1 = "Peto", hetlabel2 = "MH/IV",
  text.common1 = "Peto method",
  text.common2 = "Mantel-Haenszel method",
  keep = TRUE)

# Add results from inverse variance method
# - keep weights from IV meta-analysis
# - Q and I2 statistic are identical for sm = "MH" and sm = "Inverse"
#   as inverse variance method is used for sm = "MH" under random
#   effects model
m213 <- metamerge(m21, m3,
  label2 = "IV",
  text.common2 = "Inverse variance method",
  keep.w = TRUE)
```

```
summary(m213)

# Random effects method using ML estimator for between-study variance tau2
m4 <- update(m1, common = FALSE, method.tau = "ML")

# Use DerSimonian-Laird estimator for tau2
m5 <- update(m4, method.tau = "DL")

# Use Paule-Mandel estimator for tau2
m6 <- update(m4, method.tau = "PM")

# Merge random effects results for ML and DL estimators
# - keep weights for DL estimator (which are different from ML)
m45 <- metamerge(m4, m5, label1 = "ML", label2 = "DL",
  text.w.random1 = "RE-ML", text.w.random2 = "RE-DL", keep.w = TRUE)
summary(m45)

# Add results for PM estimator
# - keep weights
m456 <- metamerge(m45, m6, label2 = "PM",
  text.w.random2 = "RE-PM", keep.w = TRUE)
summary(m456)

m123456 <- metamerge(m123, m456)
m123456

# Use Hartung-Knapp confidence intervals
# - do not keep information on Q, I2 and weights
m7 <- update(m4, method.random.ci = "HK",
  text.random = "Hartung-Knapp method")
m8 <- update(m5, method.random.ci = "HK",
  text.random = "Hartung-Knapp method")
m9 <- update(m6, method.random.ci = "HK",
  text.random = "Hartung-Knapp method")

# Merge results for Hartung-Knapp method (with REML and DL estimator)
# - RE weights for REML estimator are shown
m78 <- metamerge(m7, m8, label1 = "ML", label2 = "DL")
summary(m78)

m789 <- metamerge(m78, m9, label2 = "PM")
summary(m789)

# Merge everything
m1to9 <- metamerge(metamerge(m123, m456, keep.w = TRUE), m789)
summary(m1to9)

m10 <- update(m1, method = "GLMM")

m.all <- metamerge(m1to9, m10, keep.Q = TRUE,
  label2 = "GLMM", taulabel2 = "ML-GLMM")
summary(m.all)
```

```
forest(m.all, layout = "JAMA")
forest(m.all, details = TRUE)

## End(Not run)

settings.meta(oldset)
options(oldopts)
```

metaprop *Meta-analysis of single proportions*

Description

Calculation of an overall proportion from studies reporting a single proportion. Inverse variance method and generalised linear mixed model (GLMM) are available for pooling. For GLMMs, the rma.glmm function from R package **metafor** (Viechtbauer 2010) is called internally.

Usage

```
metaprop(
  event,
  n,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  cluster = NULL,
  rho = 0,
  method,
  sm = gs("smprop"),
  incr = gs("incr"),
  method.incr = gs("method.incr"),
  method.ci = gs("method.ci.prop"),
  level = gs("level"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  overall = common | random,
  overall.hetstat = if (is.null(gs("overall.hetstat"))) common | random else
    gs("overall.hetstat"),
  prediction = gs("prediction") | !missing(method.predict),
 method.tau = ifelse(!is.na(charmatch(tolower(method), "glmm", nomatch = NA)), "ML",
    gs("method.tau")),
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.preset = NULL,
  TE.tau = NULL,
  tau.common = gs("tau.common"),
  detail.tau = NULL,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  null.effect = NA,
  method.bias = gs("method.bias"),
  backtransf = gs("backtransf"),
  pscale = 1,
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title = gs("title"),
  complab = gs("complab"),
  outclab = "",
  label.left = gs("label.left"),
  label.right = gs("label.right"),
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  subgroup,
  subgroup.name = NULL,
  print.subgroup.name = gs("print.subgroup.name"),
  sep.subgroup = gs("sep.subgroup"),
  test.subgroup = gs("test.subgroup"),
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  byvar,
  hakn,
  adhoc.hakn,
  keepdata = gs("keepdata"),
  warn = gs("warn"),
  warn.deprecated = gs("warn.deprecated"),
  control = NULL,
  ...
)
```

Arguments

event	Number of events.
n	Number of observations.
studlab	An optional vector with study labels.
data	An optional data frame containing the study information, i.e., event and n.

`subset` An optional vector specifying a subset of studies to be used.

`exclude` An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots.

`cluster` An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.

`rho` Assumed correlation of estimates within a cluster.

`method` A character string indicating which method is to be used for pooling of studies. One of `"Inverse"` and `"GLMM"`, can be abbreviated.

`sm` A character string indicating which summary measure (`"PLOGIT"`, `"PAS"`, `"PFT"`, `"PLN"`, or `"PRAW"`) is to be used for pooling of studies, see Details.

`incr` A numeric which is added to event number and sample size of studies with zero or all events, i.e., studies with an event probability of either 0 or 1. Or a numeric vector with the continuity correction for each study.

`method.incr` A character string indicating which continuity correction method should be used (`"only0"`, `"if0all"`, or `"all"`), see Details.

`method.ci` A character string indicating which method is used to calculate confidence intervals for individual studies, see Details.

`level` The level used to calculate confidence intervals for individual studies.

`common` A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`overall` A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

`overall.hetstat`

A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`prediction` A logical indicating whether a prediction interval should be printed.

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`tau.preset` Prespecified value for the square root of the between-study variance τ^2.

`TE.tau` Overall treatment effect used to estimate the between-study variance tau-squared.

`tau.common` A logical indicating whether tau-squared should be the same across subgroups.

`detail.tau` Detail on between-study variance estimate.

`method.I2` A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either `"Q"` or `"tau2"`, can be abbreviated (see `meta-package`).

`level.ma` The level used to calculate confidence intervals for meta-analysis estimates.

`method.random.ci` A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

`adhoc.hakn.ci` A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see `meta-package`).

`level.predict` The level used to calculate prediction interval for a new study.

`method.predict` A character string indicating which method is used to calculate a prediction interval (see `meta-package`).

`adhoc.hakn.pi` A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see `meta-package`).

`seed.predict` A numeric value used as seed to calculate bootstrap prediction interval (see `meta-package`).

`null.effect` A numeric value specifying the effect under the null hypothesis.

`method.bias` A character string indicating which test is to be used. Either `"Begg"`, `"Egger"`, or `"Thompson"`, can be abbreviated. See function `metabias`.

`backtransf` A logical indicating whether results for transformed proportions (argument `sm != "PRAW"`) should be back transformed in printouts and plots. If TRUE (default), results will be presented as proportions; otherwise transformed proportions will be shown. See Details for presentation of confidence intervals.

`pscale` A numeric defining a scaling factor for printing of single event probabilities.

`text.common` A character string used in printouts and forest plot to label the pooled common effect estimate.

`text.random` A character string used in printouts and forest plot to label the pooled random effects estimate.

`text.predict` A character string used in printouts and forest plot to label the prediction interval.

`text.w.common` A character string used to label weights of common effect model.

`text.w.random` A character string used to label weights of random effects model.

`title` Title of meta-analysis / systematic review.

`complab` Comparison label.

`outclab` Outcome label.

`label.left` Graph label on left side of null effect in forest plot.

`label.right` Graph label on right side of null effect in forest plot.

`col.label.left` The colour of the graph label on the left side of the null effect.

`col.label.right` The colour of the graph label on the right side of the null effect.

`subgroup` An optional vector to conduct a meta-analysis with subgroups.

`subgroup.name` A character string with a name for the subgroup variable.

`print.subgroup.name`
: A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

`sep.subgroup`
: A character string defining the separator between name of subgroup variable and subgroup label.

`test.subgroup`
: A logical value indicating whether to print results of test for subgroup differences.

`prediction.subgroup`
: A logical indicating whether prediction intervals should be printed for subgroups.

`seed.predict.subgroup`
: A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

`byvar`
: Deprecated argument (replaced by 'subgroup').

`hakn`
: Deprecated argument (replaced by 'method.random.ci').

`adhoc.hakn`
: Deprecated argument (replaced by 'adhoc.hakn.ci').

`keepdata`
: A logical indicating whether original data (set) should be kept in meta object.

`warn`
: A logical indicating whether warnings should be printed (e.g., if estimation problems exist in fitting a GLMM).

`warn.deprecated`
: A logical indicating whether warnings should be printed if deprecated arguments are used.

`control`
: An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to `rma.uni` or `rma.glmm`, respectively.

`...`
: Additional arguments passed on to `rma.glmm` function and to catch deprecated arguments.

Details

This function provides methods for common effect and random effects meta-analysis of single proportions to calculate an overall proportion. Note, you should use R function `metabin` to compare proportions of pairwise comparisons instead of using `metaprop` for each treatment arm separately which will break randomisation in randomised controlled trials.

The following transformations of proportions are implemented to calculate an overall proportion:

- Logit transformation (`sm = "PLOGIT"`, default)
- Arcsine transformation (`sm = "PAS"`)
- Freeman-Tukey Double arcsine transformation (`sm = "PFT"`)
- Log transformation (`sm = "PLN"`)
- No transformation (`sm = "PRAW"`)

List elements `TE`, `TE.common`, `TE.random`, etc., contain the transformed proportions. In printouts and plots these values are back transformed if argument `backtransf = TRUE` (default).

A generalised linear mixed model (GLMM) - more specific, a random intercept logistic regression model - can be utilised for the meta-analysis of proportions (Stijnen et al., 2010). This is the default method for the logit transformation (argument `sm = "PLOGIT"`). Internally, the `rma.glmm` function from R package **metafor** is called to fit a GLMM.

Classic meta-analysis (Borenstein et al., 2010) utilising the (un)transformed proportions and corresponding standard errors in the inverse variance method is conducted by calling the `metagen` function internally. This is the only available method for all transformations but the logit transformation. The classic meta-analysis model with logit transformed proportions is used by setting argument `method = "Inverse"`.

A three-level random effects meta-analysis model (Van den Noortgate et al., 2013) is utilised if argument `cluster` is used and at least one cluster provides more than one estimate. Internally, `rma.mv` is called to conduct the analysis and `weights.rma.mv` with argument `type = "rowsum"` is used to calculate random effects weights.

Default settings are utilised for several arguments (assignments using `gs` function). These defaults can be changed for the current R session using the `settings.meta` function.

Furthermore, R function `update.meta` can be used to rerun a meta-analysis with different settings.

Choice of transformation / meta-analysis method:

Contradictory recommendations on the use of transformations of proportions have been published in the literature. For example, Barendregt et al. (2013) recommend the use of the Freeman-Tukey double arcsine transformation instead of the logit transformation whereas Warton & Hui (2011) strongly advise to use generalised linear mixed models with the logit transformation instead of the arcsine transformation.

Schwarzer et al. (2019) describe seriously misleading results in a meta-analysis with very different sample sizes due to problems with the back-transformation of the Freeman-Tukey transformation which requires a single sample size (Miller, 1978). Accordingly, Schwarzer et al. (2019) also recommend to use GLMMs for the meta-analysis of single proportions, however, admit that individual study weights are not available with this method. Meta-analysts which require individual study weights should consider the inverse variance method with the arcsine or logit transformation.

In order to prevent misleading conclusions for the Freeman-Tukey double arcsine transformation, sensitivity analyses using other transformations or using a range of sample sizes should be conducted (Schwarzer et al., 2019).

Continuity correction:

Three approaches are available to apply a continuity correction:

- Only studies with a zero cell count (`method.incr = "only0"`)
- All studies if at least one study has a zero cell count (`method.incr = "if0all"`)
- All studies irrespective of zero cell counts (`method.incr = "all"`)

If the summary measure is equal to "PLOGIT", "PLN", or "PRAW", the continuity correction is applied if a study has either zero or all events, i.e., an event probability of either 0 or 1.

By default, 0.5 is used as continuity correction (argument `incr`). This continuity correction is used both to calculate individual study results with confidence limits and to conduct meta-analysis based on the inverse variance method. For GLMMs no continuity correction is used.

Confidence intervals for individual studies:

Various methods are available to calculate confidence intervals for individual study results (see Agresti & Coull 1998 and Newcombe 1988):

- Clopper-Pearson interval also called 'exact' binomial interval (`method.ci = "CP"`, default)
- Wilson Score interval (`method.ci = "WS"`)
- Wilson Score interval with continuity correction (`method.ci = "WSCC"`)
- Agresti-Coull interval (`method.ci = "AC"`)
- Simple approximation interval (`method.ci = "SA"`)
- Simple approximation interval with continuity correction (`method.ci = "SACC"`)
- Normal approximation interval based on summary measure, i.e. defined by argument `sm` (`method.ci = "NAsm"`)

Note, with exception of the normal approximation based on the summary measure, i.e. `method.ci = "NAsm"`, the same confidence interval is calculated for individual studies for any summary measure (argument `sm`) as only number of events and observations are used in the calculation disregarding the chosen transformation.

Results will be presented for transformed proportions if argument `backtransf = FALSE`. In this case, argument `method.ci = "NAsm"` is used, i.e. confidence intervals based on the normal approximation based on the summary measure.

Subgroup analysis:

Argument `subgroup` can be used to conduct subgroup analysis for a categorical covariate. The `metareg` function can be used instead for more than one categorical covariate or continuous covariates.

Specify the null hypothesis of test for an overall proportion:

Argument `null.effect` can be used to specify the proportion used under the null hypothesis in a test for an overall effect.

By default (`null.effect = NA`), no hypothesis test is conducted as it is unclear which value is a sensible choice for the data at hand. An overall proportion of 50%, for example, could be tested by setting argument `null.effect = 0.5`.

Note, all tests for an overall effect are two-sided with the alternative hypothesis that the effect is unequal to `null.effect`.

Exclusion of studies from meta-analysis:

Arguments `subset` and `exclude` can be used to exclude studies from the meta-analysis. Studies are removed completely from the meta-analysis using argument `subset`, while excluded studies are shown in printouts and forest plots using argument `exclude` (see Examples in `metagen`). Meta-analysis results are the same for both arguments.

Presentation of meta-analysis results:

Internally, both common effect and random effects models are calculated regardless of values choosen for arguments `common` and `random`. Accordingly, the estimate for the random effects model can be extracted from component `TE.random` of an object of class `"meta"` even if argument `random = FALSE`. However, all functions in R package **meta** will adequately consider the values for `common` and `random`. E.g. function `print.meta` will not print results for the random effects model if `random = FALSE`.

Argument `pscale` can be used to rescale proportions, e.g. `pscale = 1000` means that proportions are expressed as events per 1000 observations. This is useful in situations with (very) low event probabilities.

A prediction interval will only be shown if `prediction = TRUE`.

Value

An object of class `c("metaprop", "meta")` with corresponding generic functions (see `meta-object`).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Agresti A & Coull BA (1998): Approximate is better than "exact" for interval estimation of binomial proportions. *The American Statistician*, **52**, 119–26

Barendregt JJ, Doi SA, Lee YY, Norman RE, Vos T (2013): Meta-analysis of prevalence. *Journal of Epidemiology and Community Health*, **67**, 974–8

Borenstein M, Hedges LV, Higgins JP, Rothstein HR (2010): A basic introduction to fixed-effect and random-effects models for meta-analysis. *Research Synthesis Methods*, **1**, 97–111

Freeman MF & Tukey JW (1950): Transformations related to the angular and the square root. *Annals of Mathematical Statistics*, **21**, 607–11

Miller JJ (1978): The inverse of the Freeman-Tukey double arcsine transformation. *The American Statistician*, **32**, 138

Newcombe RG (1998): Two-sided confidence intervals for the single proportion: comparison of seven methods. *Statistics in Medicine*, **17**, 857–72

Pettigrew HM, Gart JJ, Thomas DG (1986): The bias and higher cumulants of the logarithm of a binomial variate. *Biometrika*, **73**, 425–35

Schwarzer G, Chemaitelly H, Abu-Raddad LJ, Rücker G (2019): Seriously misleading results using inverse of Freeman-Tukey double arcsine transformation in meta-analysis of single proportions. *Research Synthesis Methods*, **10**, 476–83

Stijnen T, Hamza TH, Ozdemir P (2010): Random effects meta-analysis of event outcome in the framework of the generalized linear mixed model with applications in sparse data. *Statistics in Medicine*, **29**, 3046–67

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

Viechtbauer W (2010): Conducting meta-analyses in R with the metafor package. *Journal of Statistical Software*, **36**, 1–48

Warton DI, Hui FKC (2011): The arcsine is asinine: the analysis of proportions in ecology. *Ecology*, **92**, 3–10

See Also

`meta-package`, `update.meta`, `metacont`, `metagen`, `print.meta`, `forest.meta`

Examples

```
# Meta-analysis using generalised linear mixed model
#
metaprop(4:1, 10 * 1:4)
```

```
# Apply various classic meta-analysis methods to estimate
# proportions
#
m1 <- metaprop(4:1, 10 * 1:4, method = "Inverse")
m2 <- update(m1, sm = "PAS")
m3 <- update(m1, sm = "PRAW")
m4 <- update(m1, sm = "PLN")
m5 <- update(m1, sm = "PFT")
#
m1
m2
m3
m4
m5
#
forest(m1)
## Not run:
forest(m2)
forest(m3)
forest(m3, pscale = 100)
forest(m4)
forest(m5)

## End(Not run)

# Do not back transform results, e.g. print logit transformed
# proportions if sm = "PLOGIT" and store old settings
#
oldset <- settings.meta(backtransf = FALSE)
#
m6  <- metaprop(4:1, c(10, 20, 30, 40), method = "Inverse")
m7  <- update(m6, sm = "PAS")
m8  <- update(m6, sm = "PRAW")
m9  <- update(m6, sm = "PLN")
m10 <- update(m6, sm = "PFT")
#
forest(m6)
## Not run:
forest(m7)
forest(m8)
forest(m8, pscale = 100)
forest(m9)
forest(m10)

## End(Not run)

# Use old settings
#
settings.meta(oldset)

# Examples with zero events
#
m1 <- metaprop(c(0, 0, 10, 10), rep(100, 4), method = "Inverse")
```

```
m2 <- metaprop(c(0, 0, 10, 10), rep(100, 4), incr = 0.1, method = "Inverse")
#
m1
m2
#
## Not run:
forest(m1)
forest(m2)

## End(Not run)

# Example from Miller (1978):
#
death <- c(3, 6, 10, 1)
animals <- c(11, 17, 21, 6)
#
m3 <- metaprop(death, animals, sm = "PFT")
forest(m3)

# Data examples from Newcombe (1998)
# - apply various methods to estimate confidence intervals for
#   individual studies
#
event <- c(81, 15, 0, 1)
n <- c(263, 148, 20, 29)
#
m1 <- metaprop(event, n, method.ci = "SA", method = "Inverse")
m2 <- update(m1, method.ci = "SACC")
m3 <- update(m1, method.ci = "WS")
m4 <- update(m1, method.ci = "WSCC")
m5 <- update(m1, method.ci = "CP")
#
lower <- round(logit2p(rbind(NA, m1$lower, m2$lower, NA, m3$lower,
  m4$lower, NA, m5$lower)), 4)
upper <- round(logit2p(rbind(NA, m1$upper, m2$upper, NA, m3$upper,
  m4$upper, NA, m5$upper)), 4)
#
tab1 <- data.frame(
  scen1 = meta:::formatCI(lower[, 1], upper[, 1]),
  scen2 = meta:::formatCI(lower[, 2], upper[, 2]),
  scen3 = meta:::formatCI(lower[, 3], upper[, 3]),
  scen4 = meta:::formatCI(lower[, 4], upper[, 4])
  )
names(tab1) <- c("r=81, n=263", "r=15, n=148",
  "r=0, n=20", "r=1, n=29")
row.names(tab1) <- c("Simple", "- SA", "- SACC",
  "Score", "- WS", "- WSCC", "Binomial", "- CP")
tab1[is.na(tab1)] <- ""
# Newcombe (1998), Table I, methods 1-5:
tab1

# Same confidence interval, i.e. unaffected by choice of summary
# measure
```

```
#
print(metaprop(event, n, method.ci = "WS", method = "Inverse"), ma = FALSE)
print(metaprop(event, n, sm = "PLN", method.ci = "WS"), ma = FALSE)
print(metaprop(event, n, sm = "PFT", method.ci = "WS"), ma = FALSE)
print(metaprop(event, n, sm = "PAS", method.ci = "WS"), ma = FALSE)
print(metaprop(event, n, sm = "PRAW", method.ci = "WS"), ma = FALSE)

# Different confidence intervals as argument sm = "NAsm"
#
print(metaprop(event, n, method.ci = "NAsm", method = "Inverse"), ma = FALSE)
print(metaprop(event, n, sm = "PLN", method.ci = "NAsm"), ma = FALSE)
print(metaprop(event, n, sm = "PFT", method.ci = "NAsm"), ma = FALSE)
print(metaprop(event, n, sm = "PAS", method.ci = "NAsm"), ma = FALSE)
print(metaprop(event, n, sm = "PRAW", method.ci = "NAsm"), ma = FALSE)

# Different confidence intervals as argument backtransf = FALSE.
# Accordingly, method.ci = "NAsm" used internally.
#
print(metaprop(event, n, method.ci = "WS", method = "Inverse"),
  ma = FALSE, backtransf = FALSE)
print(metaprop(event, n, sm = "PLN", method.ci = "WS"),
  ma = FALSE, backtransf = FALSE)
print(metaprop(event, n, sm = "PFT", method.ci = "WS"),
  ma = FALSE, backtransf = FALSE)
print(metaprop(event, n, sm = "PAS", method.ci = "WS"),
  ma = FALSE, backtransf = FALSE)
print(metaprop(event, n, sm = "PRAW", method.ci = "WS"),
  ma = FALSE, backtransf = FALSE)

# Same results (printed on original and log scale, respectively)
#
print(metaprop(event, n, sm = "PLN", method.ci = "NAsm"), ma = FALSE)
print(metaprop(event, n, sm = "PLN"), ma = FALSE, backtransf = FALSE)
# Results for first study (on log scale)
round(log(c(0.3079848, 0.2569522, 0.3691529)), 4)

# Print results as events per 1000 observations
#
print(metaprop(6:8, c(100, 1200, 1000), method = "Inverse"),
  pscale = 1000, digits = 1)
```

---

metarate *Meta-analysis of single incidence rates*

Description

Calculation of an overall incidence rate from studies reporting a single incidence rate. Inverse variance method and generalised linear mixed model (GLMM) are available for pooling. For GLMMs, the `rma.glmm` function from R package **metafor** (Viechtbauer 2010) is called internally.

Usage

```
metarate(
  event,
  time,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  cluster = NULL,
  rho = 0,
  n = NULL,
  method = "Inverse",
  sm = gs("smrate"),
  incr = gs("incr"),
  method.incr = gs("method.incr"),
  method.ci = gs("method.ci.rate"),
  level = gs("level"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  overall = common | random,
  overall.hetstat = if (is.null(gs("overall.hetstat"))) common | random else
    gs("overall.hetstat"),
  prediction = gs("prediction") | !missing(method.predict),
  method.tau,
  method.tau.ci = gs("method.tau.ci"),
  level.hetstat = gs("level.hetstat"),
  tau.preset = NULL,
  TE.tau = NULL,
  tau.common = gs("tau.common"),
  detail.tau = NULL,
  method.I2 = gs("method.I2"),
  level.ma = gs("level.ma"),
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  level.predict = gs("level.predict"),
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  null.effect = NA,
  method.bias = gs("method.bias"),
  backtransf = gs("backtransf"),
  irscale = 1,
  irunit = "person-years",
  text.common = gs("text.common"),
  text.random = gs("text.random"),
  text.predict = gs("text.predict"),
  text.w.common = gs("text.w.common"),
  text.w.random = gs("text.w.random"),
  title = gs("title"),
  complab = gs("complab"),
  outclab = "",
  label.left = gs("label.left"),
  label.right = gs("label.right"),
  col.label.left = gs("col.label.left"),
  col.label.right = gs("col.label.right"),
  subgroup,
  subgroup.name = NULL,
  print.subgroup.name = gs("print.subgroup.name"),
  sep.subgroup = gs("sep.subgroup"),
  test.subgroup = gs("test.subgroup"),
  prediction.subgroup = gs("prediction.subgroup"),
  seed.predict.subgroup = NULL,
  byvar,
  hakn,
  adhoc.hakn,
  keepdata = gs("keepdata"),
  warn = gs("warn"),
  warn.deprecated = gs("warn.deprecated"),
  control = NULL,
  ...
)
```

Arguments

`event` Number of events.

`time` Person time at risk.

`studlab` An optional vector with study labels.

`data` An optional data frame containing the study information, i.e., event and time.

`subset` An optional vector specifying a subset of studies to be used.

`exclude` An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots.

`cluster` An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.

`rho` Assumed correlation of estimates within a cluster.

`n` Number of observations.

`method` A character string indicating which method is to be used for pooling of studies. One of `"Inverse"` and `"GLMM"`, can be abbreviated.

`sm` A character string indicating which summary measure (`"IR"`, `"IRLN"`, `"IRS"`, or `"IRFT"`) is to be used for pooling of studies, see Details.

`incr` A numeric which is added to the event number of studies with zero events, i.e., studies with an incidence rate of 0. Or a numeric vector with the continuity correction for each study.

`method.incr` A character string indicating which continuity correction method should be used (`"only0"`, `"if0all"`, or `"all"`), see Details.

`method.ci` A character string indicating whether to use approximate normal ("NAsm") or exact Poisson ("Poisson") confidence limits.

`level` The level used to calculate confidence intervals for individual studies.

`common` A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`overall` A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

`overall.hetstat` A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`prediction` A logical indicating whether a prediction interval should be printed.

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`tau.preset` Prespecified value for the square root of the between-study variance τ^2.

`TE.tau` Overall treatment effect used to estimate the between-study variance tau-squared.

`tau.common` A logical indicating whether tau-squared should be the same across subgroups.

`detail.tau` Detail on between-study variance estimate.

`method.I2` A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either `"Q"` or `"tau2"`, can be abbreviated (see `meta-package`).

`level.ma` The level used to calculate confidence intervals for meta-analysis estimates.

`method.random.ci` A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

`adhoc.hakn.ci` A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see `meta-package`).

`level.predict` The level used to calculate prediction interval for a new study.

`method.predict` A character string indicating which method is used to calculate a prediction interval (see `meta-package`).

`adhoc.hakn.pi` A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see `meta-package`).

`seed.predict` A numeric value used as seed to calculate bootstrap prediction interval (see `meta-package`).

`null.effect` A numeric value specifying the effect under the null hypothesis.

method.bias A character string indicating which test is to be used. Either "Begg", "Egger", or "Thompson", can be abbreviated. See function metabias.

backtransf A logical indicating whether results for transformed rates (argument sm != "IR") should be back transformed in printouts and plots. If TRUE (default), results will be presented as incidence rates; otherwise transformed rates will be shown.

irscale A numeric defining a scaling factor for printing of rates.

irunit A character string specifying the time unit used to calculate rates, e.g. person-years.

text.common A character string used in printouts and forest plot to label the pooled common effect estimate.

text.random A character string used in printouts and forest plot to label the pooled random effects estimate.

text.predict A character string used in printouts and forest plot to label the prediction interval.

text.w.common A character string used to label weights of common effect model.

text.w.random A character string used to label weights of random effects model.

title Title of meta-analysis / systematic review.

complab Comparison label.

outclab Outcome label.

label.left Graph label on left side of null effect in forest plot.

label.right Graph label on right side of null effect in forest plot.

col.label.left The colour of the graph label on the left side of the null effect.

col.label.right

The colour of the graph label on the right side of the null effect.

subgroup An optional vector to conduct a meta-analysis with subgroups.

subgroup.name A character string with a name for the subgroup variable.

print.subgroup.name

A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

sep.subgroup A character string defining the separator between name of subgroup variable and subgroup label.

test.subgroup A logical value indicating whether to print results of test for subgroup differences.

prediction.subgroup

A logical indicating whether prediction intervals should be printed for subgroups.

seed.predict.subgroup

A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

byvar Deprecated argument (replaced by 'subgroup').

hakn Deprecated argument (replaced by 'method.random.ci').

adhoc.hakn Deprecated argument (replaced by 'adhoc.hakn.ci').

keepdata A logical indicating whether original data (set) should be kept in meta object.

warn A logical indicating whether warnings should be printed (e.g., if incr is added to studies with zero cell frequencies or if estimation problems exist in fitting a GLMM).

warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments are used.

control An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to rma.uni or rma.glmm, respectively.

... Additional arguments passed on to rma.glmm function and to catch deprecated arguments.

Details

This function provides methods for common effect and random effects meta-analysis of single incidence rates to calculate an overall rate. Note, you should use R function metainc to compare incidence rates of pairwise comparisons instead of using metarate for each treatment arm separately which will break randomisation in randomised controlled trials.

The following transformations of incidence rates are implemented to calculate an overall rate:

- Log transformation (sm = "IRLN", default)
- Square root transformation (sm = "IRS")
- Freeman-Tukey Double arcsine transformation (sm = "IRFT")
- No transformation (sm = "IR")

List elements TE, TE.common, TE.random, etc., contain the transformed incidence rates. In printouts and plots these values are back transformed if argument backtransf = TRUE (default).

By default (argument method = "Inverse"), the inverse variance method (Borenstein et al., 2010) is used for pooling by calling metagen internally. A random intercept Poisson regression model (Stijnen et al., 2010) can be utilised instead with argument method = "GLMM" which calls the rma.glmm function from R package **metafor**.

A three-level random effects meta-analysis model (Van den Noortgate et al., 2013) is utilised if argument cluster is used and at least one cluster provides more than one estimate. Internally, rma.mv is called to conduct the analysis and weights.rma.mv with argument type = "rowsum" is used to calculate random effects weights.

Default settings are utilised for several arguments (assignments using gs function). These defaults can be changed for the current R session using the settings.meta function.

Furthermore, R function update.meta can be used to rerun a meta-analysis with different settings.

Continuity correction:

Three approaches are available to apply a continuity correction:

- Only studies with a zero cell count (method.incr = "only0")
- All studies if at least one study has a zero cell count (method.incr = "if0all")
- All studies irrespective of zero cell counts (method.incr = "all")

If the summary measure (argument sm) is equal to "IR" or "IRLN", the continuity correction is applied if a study has zero events, i.e., an incidence rate of 0.

By default, 0.5 is used as continuity correction (argument incr). This continuity correction is used both to calculate individual study results with confidence limits and to conduct meta-analysis based on the inverse variance method.

For the Freeman-Tukey (Freeman & Tukey, 1950) and square root transformation as well as GLMMs no continuity correction is used.

Subgroup analysis:

Argument subgroup can be used to conduct subgroup analysis for a categorical covariate. The metareg function can be used instead for more than one categorical covariate or continuous covariates.

Specify the null hypothesis of test for an overall effect:

Argument null.effect can be used to specify the rate used under the null hypothesis in a test for an overall effect.

By default (null.effect = NA), no hypothesis test is conducted as it is unclear which value is a sensible choice for the data at hand. An overall rate of 2, for example, could be tested by setting argument null.effect = 2.

Note, all tests for an overall effect are two-sided with the alternative hypothesis that the effect is unequal to null.effect.

Exclusion of studies from meta-analysis:

Arguments subset and exclude can be used to exclude studies from the meta-analysis. Studies are removed completely from the meta-analysis using argument subset, while excluded studies are shown in printouts and forest plots using argument exclude (see Examples in metagen). Meta-analysis results are the same for both arguments.

Presentation of meta-analysis results:

Internally, both common effect and random effects models are calculated regardless of values choosen for arguments common and random. Accordingly, the estimate for the random effects model can be extracted from component TE.random of an object of class "meta" even if argument random = FALSE. However, all functions in R package **meta** will adequately consider the values for common and random. E.g. function print.meta will not print results for the random effects model if random = FALSE.

Argument irscale can be used to rescale rates, e.g. irscale = 1000 means that rates are expressed as events per 1000 time units, e.g. person-years. This is useful in situations with (very) low rates. Argument irunit can be used to specify the time unit used in individual studies (default: "person-years"). This information is printed in summaries and forest plots if argument irscale is not equal to 1.

A prediction interval will only be shown if prediction = TRUE.

Value

An object of class c("metarate", "meta") with corresponding generic functions (see meta-object).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Borenstein M, Hedges LV, Higgins JP, Rothstein HR (2010): A basic introduction to fixed-effect and random-effects models for meta-analysis. *Research Synthesis Methods*, **1**, 97–111

Freeman MF & Tukey JW (1950): Transformations related to the angular and the square root. *Annals of Mathematical Statistics*, **21**, 607–11

Stijnen T, Hamza TH, Ozdemir P (2010): Random effects meta-analysis of event outcome in the framework of the generalized linear mixed model with applications in sparse data. *Statistics in Medicine*, **29**, 3046–67

Van den Noortgate W, López-López JA, Marín-Martínez F, Sánchez-Meca J (2013): Three-level meta-analysis of dependent effect sizes. *Behavior Research Methods*, **45**, 576–94

Viechtbauer W (2010): Conducting Meta-Analyses in R with the Metafor Package. *Journal of Statistical Software*, **36**, 1–48

See Also

`meta-package`, `update.meta`, `metacont`, `metagen`, `print.meta`

Examples

```
# Apply various meta-analysis methods to estimate incidence rates
#
m1 <- metarate(4:1, c(10, 20, 30, 40))
m2 <- update(m1, sm = "IR")
m3 <- update(m1, sm = "IRS")
m4 <- update(m1, sm = "IRFT")
#
m1
m2
m3
m4
#
forest(m1)
forest(m1, irscale = 100)
forest(m1, irscale = 100, irunit = "person-days")
forest(m1, backtransf = FALSE)
## Not run:
forest(m2)
forest(m3)
forest(m4)

## End(Not run)

m5 <- metarate(40:37, c(100, 200, 300, 400), sm = "IRFT")
m5
```

metareg.meta *Meta-regression*

Description

Meta-regression for objects of class meta. This is a wrapper function for the R function rma.uni in the R package **metafor** (Viechtbauer 2010).

Usage

```
## S3 method for class 'meta'
metareg(
  x,
  formula,
  method.tau = x$method.tau,
  hakn = x$method.random.ci == "HK",
  level.ma = x$level.ma,
  intercept = TRUE,
  ...
)

metareg(x, ...)

## Default S3 method:
metareg(x, ...)
```

Arguments

x	An object of class meta.
formula	Either a character string or a formula object.
method.tau	A character string indicating which method is used to estimate the between-study variance tau-squared. Either "FE", "DL", "REML", "ML", "HS", "SJ", "HE", or "EB", can be abbreviated.
hakn	A logical indicating whether the method by Hartung and Knapp should be used to adjust test statistics and confidence intervals.
level.ma	The level used to calculate confidence intervals for parameter estimates in the meta-regression model.
intercept	A logical indicating whether an intercept should be included in the meta-regression model.
...	Additional arguments passed to R function rma.uni.

Details

This R function is a wrapper function for R function rma.uni in the R package **metafor** (Viechtbauer 2010).

Note, results are not back-transformed in printouts of meta-analyses using summary measures with transformations, e.g., log risk ratios are printed instead of the risk ratio if argument `sm = "RR"` and logit transformed proportions are printed if argument `sm = "PLOGIT"`.

Argument '...' can be used to pass additional arguments to R function `rma.uni`. For example, argument `control` to provide a list of control values for the iterative estimation algorithm. See help page of R function `rma.uni` for more details.

Value

An object of class `c("metareg", "rma.uni", "rma")`. Please look at the help page of R function `rma.uni` for more details on the output from this function.

In addition, a list `.meta` is added to the output containing the following components:

`x`, `formula`, `method.tau`, `hakn`, `level.ma`, `intercept`
As defined above.

`dots`	Information provided in argument '...'.
`call`	Function call.
`version`	Version of R package **meta** used to create object.
`version.metafor`	Version of R package **metafor** used to create object.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Viechtbauer W (2010): Conducting Meta-Analyses in R with the Metafor Package. *Journal of Statistical Software*, **36**, 1–48

See Also

`bubble`, `summary.meta`, `metagen`

Examples

```
data(Fleiss1993cont)
# Add some (fictitious) grouping variables:
Fleiss1993cont$age <- c(55, 65, 55, 65, 55)
Fleiss1993cont$region <- c("Europe", "Europe", "Asia", "Asia", "Europe")

m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD")
## Not run:
# Error due to wrong ordering of arguments (order has changed in
# R package meta, version 3.0-0)
#
try(metareg(~ region, m1))
try(metareg(~ region, data = m1))
```

```
# Warning as no information on covariate is available
#
metareg(m1)

## End(Not run)

# Do meta-regression for covariate region
#
mu2 <- update(m1, subgroup = region, tau.common = TRUE, common = FALSE)
metareg(mu2)

# Same result for
# - tau-squared
# - test of heterogeneity
# - test for subgroup differences
# (as argument 'tau.common' was used to create mu2)
#
mu2
metareg(mu2, intercept = FALSE)
metareg(m1, region)

# Different result for
# - tau-squared
# - test of heterogeneity
# - test for subgroup differences
# (as argument 'tau.common' is - by default - FALSE)
#
mu1 <- update(m1, subgroup = region)
mu1

# Generate bubble plot
#
bubble(metareg(mu2))

# Do meta-regression with two covariates
#
metareg(mu1, region + age)

# Do same meta-regressions using formula notation
#
metareg(m1, ~ region)
metareg(mu1, ~ region + age)

# Do meta-regression using REML method and print intermediate
# results for iterative estimation algorithm; furthermore print
# results with three digits.
#
metareg(mu1, region, method.tau = "REML",
  control = list(verbose = TRUE), digits = 3)

# Use Hartung-Knapp method
#
```

```
mu3 <- update(mu2, method.random.ci = "HK")
mu3
metareg(mu3, intercept = FALSE)
```

nnt *Calculate the number needed to treat (NNT)*

Description

Calculate the number needed to treat (NNT) from estimated risk difference, risk ratio, odds ratio, or hazard ratio, and a baseline probability, i.e., control group event probability for binary outcomes or survival probability for hazard ratios.

Usage

```
nnt(x, ...)

## S3 method for class 'meta'
nnt(
  x,
  p.c,
  common = x$common,
  random = x$random,
  small.values = "desirable",
  ...
)

## Default S3 method:
nnt(x, p.c, sm, lower, upper, small.values = "desirable", transf = FALSE, ...)

## S3 method for class 'nnt.meta'
print(
  x,
  common = x$common,
  random = x$random,
  digits = gs("digits"),
  digits.prop = gs("digits.prop"),
  big.mark = gs("big.mark"),
  ...
)

## S3 method for class 'nnt.default'
print(
  x,
  digits = gs("digits"),
  digits.prop = gs("digits.prop"),
  big.mark = gs("big.mark"),
  ...
)
```

Arguments

`x` An object of class `meta`, or estimated treatment effect(s), i.e., risk difference(s), risk ratio(s), odds ratio(s), or hazard ratio(s).

`...` Additional arguments (to catch deprecated arguments).

`p.c` Baseline probability, i.e., control group event probability for binary outcomes or survival probability in control group for hazard ratios.

`common` A logical indicating whether NNTs should be calculated based on common effect estimate.

`random` A logical indicating whether NNTs should be calculated based on random effects estimate.

`small.values` A character string specifying whether small treatment effects indicate a beneficial (`"desirable"`) or harmful (`"undesirable"`) effect, can be abbreviated.

`sm` Summary measure.

`lower` Lower confidence interval limit.

`upper` Upper confidence interval limit.

`transf` A logical indicating whether treatment estimates and confidence limits are transformed or on the original scale. If `transf = TRUE`, inputs are expected to be log odds ratios instead of odds ratios for `sm = "OR"`, for example.

`digits` Minimal number of significant digits to print NNT and its confidence interval, see `print.default`.

`digits.prop` Minimal number of significant digits for proportions, see `print.default`.

`big.mark` A character used as thousands separator.

Details

The number needed to treat (NNT) is the estimated number of patients who need to be treated with a new treatment instead of a standard for one additional patient to benefit (Laupacis et al., 1988; Cook & Sackett, 1995). This definition of the NNT implies that the new treatment is more beneficial than the standard. If the new treatment is indeed less beneficial than the standard, the NNT gives the number of patients treated with the new treatment to observe an additional harmful event. Accordingly, the abbreviations NNTB and NNTH can be used to distinguish between beneficial and harmful NNTs (Altman, 1998).

NNTs can be easily computed from an estimated risk difference (RD), risk ratio (RR), or odds ratio (OR) and a given baseline probability (Higgins et al., 2023, section 15.4.4). It is also possible to calculate NNTs from hazard ratios (HR) (Altman & Andersen, 1999). Accordingly, NNTs can be calculated for meta-analyses generated with `metabin` or `metagen` if argument `sm` was equal to `"RD"`, `"RR"`, `"OR"`, or `"HR"`. It is also possible to provide only estimated treatment effects and baseline probabilities (see Examples).

The baseline probability can be specified using argument `p.c`. If this argument is missing, the minimum, mean, and maximum of the control event probabilities in the meta-analysis are used for metabin and control event probabilities of 0.1, 0.2, . . . , 0.9 are used for metagen. Note, the survival instead of mortality probability must be provided for hazard ratios.

Argument small.values can be used to specify whether small treatment effects indicate a beneficial ("desirable") or harmful ("undesirable") effect. For small.values = "desirable", odds, risk and hazard ratios below 1 and risk differences below 0 indicate that the new treatment is beneficial. For small.values = "undesirable", odds, risk and hazard ratios above 1 and risk differences above 0 indicate that the new treatment is beneficial.

Interpretation of (positive and negative) NNTs: A positive value for the estimated NNT indicates that the new treatment is beneficial, i.e., the NNT is actually an NNTB. On the other hand, a negative value for the estimated NNT indicates that the new treatment is harmful, i.e., the NNT is actually an NNTH.

The minimal value for the NNTB is 1. In this extreme case the new treatment is 100% effective and the standard treatment is 0% effective, i.e., only one patient has to be treated with the new treatment for one additional patient to benefit. The NNTB increases with decreasing difference between the two risks. If both risks are equal, the NNTB is infinite.

The other extreme is an NNT of -1 if the new treatment is 0% effective and the standard is 100% effective. Here, one additional harmful event is observed for each patient treated with the new treatment. The NNT approaches minus infinity if the difference between the two risks decreases to 0. Finally, an NNT of -1 translates to an NNTH of 1 with possible values from 1 to infinity.

Confidence interval for the NNT: Confidence limits for an NNT are derived from the lower and upper confidence limits of the summary measure using the same formulae as for the NNT (Higgins et al., 2023, section 15.4.4).

A peculiar problem arises if the confidence interval for the summary measure includes the null effect (i.e., RR = 1, OR = 1, HR = 1, or RD = 0). In this case the confidence interval for the NNT contains both NNTB and NNTH values and it seemingly does not include the estimated NNT.

As described above, a positive NNT value corresponds to an NNTB and the absolute value of a negative NNT is equal to an NNTH. Accordingly, a confidence interval for the NNT from 20 to -5 translates to NNTB values between 20 and infinity and NNTH values between 5 and infinity (Altman, 1998).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Altman DG (1998): Confidence intervals for the number needed to treat. *British Medical Journal*, **317**, 1309–12

Altman DG, Andersen PK (1999): Calculating the number needed to treat for trials where the outcome is time to an event. *British Medical Journal*, **319**, 1492–95

Cook RJ, Sackett DL (1995): The number needed to treat: a clinically useful measure of treatment effect. *British Medical Journal*, **310**, 452–54

Higgins JPT, Thomas J, Chandler J, Cumpston M, Li T, Page MJ, Welch VA (editors) (2023): *Cochrane Handbook for Systematic Reviews of Interventions Version 6.4 (updated August 2023)*. Available from https://training.cochrane.org/handbook/

Laupacis A, Sackett DL, Roberts RS (1988): An assessment of clinically useful measures of the consequences of treatment. *New England Journal of Medicine*, **318**, 1728–33

See Also

`metabin`, `metagen`

Examples

```
# Calculate NNTs for risk differences of -0.12 and -0.22
# (Cochrane Handbook, version 6.3, subsection 15.4.4.1)
nnt(c(-0.12, -0.22), sm = ”RD”)

# Calculate NNT for risk ratio of 0.92 and baseline risk of 0.3
# (Cochrane Handbook, version 6.3, subsection 15.4.4.2)
nnt(0.92, p.c = 0.3, sm = ”RR”)

# Calculate NNT for odds ratio of 0.73 and baseline risk of 0.3
# (Cochrane Handbook, version 6.3, subsection 15.4.4.3)
nnt(0.73, p.c = 0.3, sm = ”OR”)

# Calculate NNTs for Mantel-Haenszel odds ratio
data(Olkin1995)
m1 <-
  metabin(ev.exp, n.exp, ev.cont, n.cont, data = Olkin1995, random = FALSE)
nnt(m1)

# Calculate NNTs from hazard ratio at two and four years (example from
# Altman & Andersen, 1999). Note, argument 'p.c' must provide survival
# probabilities instead of mortality rates for the control group.
nnt(0.72, lower = 0.55, upper = 0.92, sm = ”HR”, p.c = 1 - c(0.33, 0.49))
```

`Olkin1995` *Thrombolytic Therapy after Acute Myocardial Infarction*

Description

Meta-analysis on Thrombolytic Therapy after Acute Myocardial Infarction

Format

A data frame with the following columns:

author	first author
year	year of publication
ev.exp	number of events in experimental group
n.exp	number of observations in experimental group
ev.cont	number of events in control group

n.cont number of observations in control group

Source

Olkin I (1995): Statistical and theoretical considerations in meta-analysis. *Journal of Clinical Epidemiology*, **48**, 133–46

Examples

```
data(Olkin1995)
metabin(ev.exp, n.exp, ev.cont, n.cont, data = Olkin1995)
```

---

or2smd *Conversion from log odds ratio to standardised mean difference*

Description

Conversion from log odds ratio to standardised mean difference using method by Hasselblad & Hedges (1995) or Cox (1970).

Usage

```
or2smd(
  lnOR,
  selnOR,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  method = "HH",
  ...
)
```

Arguments

`lnOR`	Log odds ratio(s) or meta-analysis object.
`selnOR`	Standard error(s) of log odds ratio(s) (ignored if argument `lnOR` is a meta-analysis object).
`studlab`	An optional vector with study labels (ignored if argument `lnOR` is a meta-analysis object).
`data`	An optional data frame containing the study information (ignored if argument `lnOR` is a meta-analysis object).
`subset`	An optional vector specifying a subset of studies to be used (ignored if argument `lnOR` is a meta-analysis object).
`exclude`	An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots (ignored if argument `lnOR` is a meta-analysis object).

method — A character string indicating which method is used to convert log odds ratios to standardised mean differences. Either "HH" or "CS", can be abbreviated.

... — Additional arguments passed on to metagen (ignored if argument lnOR is a meta-analysis object).

Details

This function implements the following methods for the conversion from log odds ratios to standardised mean difference:

- Hasselblad & Hedges (1995) assuming logistic distributions (method == "HH")
- Cox (1970) and Cox & Snell (1989) assuming normal distributions (method == "CS")

Internally, metagen is used to conduct a meta-analysis with the standardised mean difference as summary measure.

Argument lnOR can be either a vector of log odds ratios or a meta-analysis object created with metabin or metagen and the odds ratio as summary measure.

Argument selnOR is mandatory if argument lnOR is a vector and ignored otherwise. Additional arguments in ... are only passed on to metagen if argument lnOR is a vector.

Value

An object of class c("metagen", "meta") with corresponding generic functions (see meta-object).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Borenstein M, Hedges LV, Higgins JPT, Rothstein HR (2009): *Introduction to Meta-Analysis*. Chichester: Wiley

Cox DR (1970): *Analysis of Binary Data*. London: Chapman and Hall / CRC

Cox DR, Snell EJ (1989): *Analysis of Binary Data* (2nd edition). London: Chapman and Hall / CRC

Hasselblad V, Hedges LV (1995): Meta-analysis of screening and diagnostic tests. *Psychological Bulletin*, **117**, 167–78

See Also

smd2or, metabin, metagen, metacont

Examples

```
# Example from Borenstein et al. (2009), Chapter 7
#
mb <- or2smd(0.9069, sqrt(0.0676))
# TE = standardised mean difference (SMD); seTE = standard error of SMD
data.frame(SMD = round(mb$TE, 4), varSMD = round(mb$seTE^2, 4))
```

```
# Use dataset from Fleiss (1993)
#
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac,
  data = Fleiss1993bin, studlab = paste(study, year),
  sm = "OR", random = FALSE)
or2smd(m1)
```

Pagliaro1992 *Meta-analysis on Prevention of First Bleeding in Cirrhosis*

Description

Meta-analysis on Prevention of First Bleeding in Cirrhosis comparing beta-blocker or sclerotherapy with placebo.

Format

A data frame with the following columns:

id	study id
treat.exp	treatment in experimental group
logOR	log odds ratio
selogOR	standard error of log odds ratio
bleed.exp	number of bleedings in experimental group
n.cont	number of observations in experimental group
bleed.plac	number of bleedings in placebo group
n.plac	number of observations in placebo group

Source

Pagliaro L, D'Amico G et al. (1992): Prevention of first bleeding in cirrhosis. *Annals in Internal Medicine*, **117**, 59–70

Examples

```
data(Pagliaro1992)
sclero <- subset(Pagliaro1992, treat.exp == "Sclerotherapy")

m <- metagen(logOR, selogOR, data = sclero, sm = "OR")
m

# Thompson & Sharp (1999), Table IV, method (2)
metabias(m, method = "Egger")

# Thompson & Sharp (1999), Table IV, method (3a)
metabias(m, method = "Thompson")
```

```
# Thompson & Sharp (1999), Table IV, method (3b)
update(m, method.tau = "ML")
metabias(update(m, method.tau = "ML"), method = "Thompson")
```

pairwise *Transform meta-analysis data from two arm-based formats into contrast-based format*

Description

This function transforms data that are given in wide or long arm-based format (e.g. input format for WinBUGS) to a contrast-based format that is needed as input to R functions `metabin`, `metacont`, `metainc`, `metagen`, or `netmeta` from R package **netmeta**. The function can transform data with binary, continuous, or generic outcomes as well as incidence rates from arm-based to contrast-based format.

Usage

```
pairwise(
  treat,
  event,
  n,
  mean,
  sd,
  TE,
  seTE,
  time,
  agent,
  dose,
  data = NULL,
  studlab,
  method = "Inverse",
  sm = NULL,
  incr = gs("incr"),
  method.incr = gs("method.incr"),
  allstudies = gs("allstudies"),
  reference.group,
  keep.all.comparisons,
  sep.ag = "*",
  varnames = c("TE", "seTE"),
  append = !is.null(data),
  addincr = gs("addincr"),
  allincr = gs("allincr"),
  warn = FALSE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

Arguments

`treat` A list or vector with treatment information for individual treatment arms (see Details).

`event` A list or vector with information on number of events for individual treatment arms (see Details).

`n` A list or vector with information on number of observations for individual treatment arms (see Details).

`mean` A list or vector with estimated means for individual treatment arms (see Details).

`sd` A list or vector with information on the standard deviation for individual treatment arms (see Details).

`TE` A list or vector with estimated treatment effects for individual treatment arms (see Details).

`seTE` A list or vector with standard errors of estimated treatment effect for individual treatment arms (see Details).

`time` A list or vector with information on person time at risk for individual treatment arms (see Details).

`agent` A list or vector with agent information for individual treatment arms (see Details).

`dose` A list or vector with dose information for individual treatment arms (see Details).

`data` An optional data frame containing the study information.

`studlab` A vector with study labels (optional).

`method` A character string indicating which method is to be used to calculate treatment estimates (see Details).

`sm` A character string indicating which summary measure is to be used to calculate treatment estimates (see Details).

`incr` A numerical value which is added to cell frequencies for studies with a zero cell count, see Details.

`method.incr` A character string indicating which continuity correction method should be used (`"only0"`, `"if0all"`, or `"all"`), see `metabin`.

`allstudies` A logical indicating if studies with zero or all events in two treatment arms are to be included in the meta-analysis (applies only if `sm` is equal to `"RR"` or `"OR"`).

`reference.group`
Reference treatment (first treatment is used if argument is missing).

`keep.all.comparisons`
A logical indicating whether all pairwise comparisons or only comparisons with the study-specific reference group should be kept ('basic parameters').

`sep.ag` A character used as separator between agent in dose to create treatment labels.

`varnames` Character vector of length 2 with the variable names for the treatment estimate and its standard error; by default, "TE" and "seTE".

`append` Either a logical indicating whether variables from the dataset provided in argument `data` are appended to the dataset with pairwise comparisons or a character vector with variable names to append to the dataset.

`addincr`	Deprecated argument (replaced by 'method.incr'); see `metabin`.
`allincr`	Deprecated argument (replaced by 'method.incr'); see `metabin`.
`warn`	A logical indicating whether warnings should be printed (e.g., if studies are excluded due to only providing a single treatment arm).
`warn.deprecated`	A logical indicating whether warnings should be printed if deprecated arguments are used.
`...`	Additional arguments passed-through to the functions to calculate effects.

Details

The pairwise function transforms data given in (wide or long) arm-based format into the contrast-based format which consists of *pairwise* comparisons and which is needed as input to R functions `metabin`, `metacont`, `metainc`, `metagen`, or `netmeta` from R package **netmeta**.

The pairwise function can transform data with binary outcomes continuous outcomes (`metacont` function), incidence rates (`metainc` function), and generic outcomes (`metagen` function). Depending on the outcome, the following arguments are mandatory:

- treat, event, n (see `metabin`);
- treat, n, mean, sd (see `metacont`);
- treat, event, time (see `metainc`);
- treat, TE, seTE (see `metagen`).

(using the `metabin` function from R package meta),

Admissible values for arguments `method` and `sm` are outcome specific; see help pages of R functions `metabin`, `metacont`, `metainc`, and `metagen`.

Argument `treat` is mandatory to identify the individual treatments. The other arguments contain outcome specific data. These arguments must be either lists (wide arm-based format, i.e., one row per study) or vectors (long arm-based format, i.e., multiple rows per study) of the same length.

For the wide arm-based format, each list consists of as many vectors of the same length as the multi-arm study with the largest number of treatments. If a single multi-arm study has five arms, five vectors have to be provided for each lists. Two-arm studies have entries with `NA` for the third and subsequent vectors. Each list entry is a vector with information for each individual study; i.e., the length of this vector corresponds to the total number of studies incorporated in the network meta-analysis. Typically, list elements are part of a data frame (argument `data`, optional); see Examples. An optional vector with study labels can be provided which can be part of the data frame.

In the long arm-based format, argument `studlab` is mandatory to identify rows contributing to individual studies.

Additional arguments for meta-analysis functions can be provided using argument '...'; see help pages of R functions `metabin`, `metacont`, `metainc`, and `metagen`.

For standardised mean differences (argument `sm = "SMD"`), equations (4) and (5) in Crippa & Orsini (2016) are used to calculated SMDs and standard errors. These equations guarantee consistent SMDs and standard errors for multi-arm studies. Note, the summary measure is actually Cohen's d as Hedges' g is not consistent in multi-arm studies.

For binary outcomes, 0.5 is added to all cell frequencies (odds ratio) or only the number of events (risk ratio) for studies with a zero cell count. For odds ratio and risk ratio, treatment estimates and standard errors are only calculated for studies with zero or all events in both groups if `allstudies` is `TRUE`. This continuity correction is used both to calculate individual study results with confidence limits and to conduct meta-analysis based on the inverse variance method. For the risk difference, 0.5 is only added to all cell frequencies to calculate the standard error.

For incidence rates, 0.5 is added to all cell frequencies for the incidence rate ratio as summary measure. For the incidence risk difference, 0.5 is only added to all cell frequencies to calculate the standard error.

The value of pairwise is a data frame with as many rows as there are pairwise comparisons. For each study with *p* treatments, *p*(p-1) / 2* contrasts are generated. Each row contains the treatment effect (`TE`), its standard error (`seTE`), the treatments compared ((`treat1`), (`treat2`)) and the study label ((`studlab`)). Further columns are added according to type of data.

All variables from the original dataset are also part of the output dataset if argument `append = TRUE`. If data are provided in the long arm-based format, the value of a variable can differ between treatment arms; for example, the mean age or percentage of women in the treatment arm. In this situation, two variables instead of one variable will be included in the output dataset. The values "1" and "2" are added to the names for these variables, e.g. "mean.age1" and "mean.age2" for the mean age.

In general, any variable names in the original dataset that are identical to the main variable names (i.e., "TE", "seTE", ...) will be renamed to variable names with ending ".orig".

A reduced dataset with basic comparisons (Rücker & Schwarzer, 2014) can be generated using argument `keep.all.comparisons = FALSE`. Furthermore, the reference group for the basic comparisons can be specified with argument `reference.group`.

Use in network meta-analysis: R function `netmeta` expects data in a **contrast-based format**, where each row corresponds to a comparison of two treatments and contains a measure of the treatment effect comparing two treatments with standard error, labels for the two treatments and an optional study label. In contrast-based format, a three-arm study contributes three rows with treatment comparison and corresponding standard error for pairwise comparison *A* vs *B*, *A* vs *C*, and *B* vs *C* whereas a four-arm study contributes six rows / pairwise comparisons: *A* vs *B*, *A* vs *C*, . . . , *C* vs *D*.

Other programs for network meta-analysis in WinBUGS and Stata require data in an *arm-based* format, i.e. treatment estimate for each treatment arm instead of a difference of two treatments. A common **(wide) arm-based format** consists of one data row per study, containing treatment and other necessary information for all study arms. For example, a four-arm study contributes one row with four treatment estimates and corresponding standard errors for treatments *A*, *B*, *C*, and *D*. Another possible arm-based format is a long format where each row corresponds to a single study arm. Accordingly, in the **long arm-based format** a study contributes as many rows as treatments considered in the study.

Value

A data frame with the following columns:

`TE`	Treatment estimate comparing treatment 'treat1' and 'treat2'.
`seTE`	Standard error of treatment estimate.

studlab	Study labels.
treat1	First treatment in comparison.
treat2	Second treatment in comparison.
event1	Number of events for first treatment arm (for metabin and metainc).
event2	Number of events for second treatment arm (for metabin and metainc).
n1	Number of observations for first treatment arm (for metabin and metacont).
n2	Number of observations for second treatment arm (for metabin and metacont).
mean1	Estimated mean for first treatment arm (for metacont).
mean2	Estimated mean for second treatment arm (for metacont).
sd1	Standard deviation for first treatment arm (for metacont).
sd2	Standard deviation for second treatment arm (for metacont).
TE1	Estimated treatment effect for first treatment arm (for metagen).
TE2	Estimated treatment effect for second treatment arm (for metagen).
seTE1	Standard error of estimated treatment effect for first treatment arm (for metagen).
seTE2	Standard error of estimated treatment effect for second treatment arm (for metagen).
time1	Person time at risk for first treatment arm (for metainc).
time2	Person time at risk for second treatment arm (for metainc).
agent1	First agent in comparison.
agent2	Second agent in comparison.
dose1	Dose of first agent in comparison.
dose2	Dose of second agent in comparison.

All variables from the original dataset are also part of the output dataset; see Details.

Note

This function must not be confused with netpairwise which can be used to conduct pairwise meta-analyses for all comparisons with direct evidence in a network meta-analysis.

Author(s)

Gerta Rücker<gerta.ruecker@uniklinik-freiburg.de>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.c

References

Crippa A, Orsini N (2016): Dose-response meta-analysis of differences in means. *BMC Medical Research Methodology*, **16**:91.

See Also

longarm, metabin, metacont, metagen, metainc, netmeta, netgraph.netmeta, dat.senn2013, dat.franchini2012, dat.franchini2012

Examples

```
pw0 <- pairwise(studlab = study, treat = treatment,
  n = ni, mean = mi, sd = sdi, data = dat.senn2013,
  append = c("study", "comment"))
head(pw0)
# Meta-analysis of studies comparing metformin to placebo
metagen(pw0, subset = treat1 == "metformin" & treat2 == "placebo")

## Not run:
# Use pairwise() to run network meta-analyses
# (R package 'netmeta' must be available)
if (requireNamespace("netmeta", quietly = TRUE)) {
 # Example using continuous outcomes (internal call of function
 # metacont)
 #
 Franchini2012 <- dat.franchini2012
 # Transform data from arm-based format to contrast-based format
 pw1 <- pairwise(list(Treatment1, Treatment2, Treatment3),
   n = list(n1, n2, n3),
   mean = list(y1, y2, y3), sd = list(sd1, sd2, sd3),
   data = Franchini2012, studlab = Study)
 pw1

 # Conduct network meta-analysis
 library("netmeta")
 #
 net1 <- netmeta(pw1)
 net1

 # Draw network graphs
 #
 netgraph(net1, points = TRUE, cex.points = 3, cex = 1.5,
   thickness = "se.common")
 netgraph(net1, points = TRUE, cex.points = 3, cex = 1.5,
   plastic = TRUE, thickness = "se.common",
   iterate = TRUE)
 netgraph(net1, points = TRUE, cex.points = 3, cex = 1.5,
   plastic = TRUE, thickness = "se.common",
   iterate = TRUE, start = "eigen")

 # Example using generic outcomes (internal call of function
 # metagen)
 #
 # Calculate standard error for means y1, y2, y3
 Franchini2012$se1 <- with(Franchini2012, sqrt(sd1^2 / n1))
 Franchini2012$se2 <- with(Franchini2012, sqrt(sd2^2 / n2))
 Franchini2012$se3 <- with(Franchini2012, sqrt(sd3^2 / n3))
 # Transform data from arm-based format to contrast-based format
 # using means and standard errors (note, argument 'sm' has to be
 # used to specify that argument 'TE' is a mean difference)
 pw2 <- pairwise(list(Treatment1, Treatment2, Treatment3),
   TE = list(y1, y2, y3), seTE = list(se1, se2, se3),
   n = list(n1, n2, n3),
   data = Franchini2012, studlab = Study,
   sm = "MD")
 pw2

 # Compare pairwise objects pw1 (based on continuous outcomes) and pw2
 # (based on generic outcomes)
 #
 all.equal(
   pw1[, c("TE", "seTE", "studlab", "treat1", "treat2")],
   pw2[, c("TE", "seTE", "studlab", "treat1", "treat2")])

 # Same result as network meta-analysis based on continuous outcomes
 # (object net1)
 net2 <- netmeta(pw2)
 net2

 # Example with binary data
 #
 data(smokingcessation)
 # Transform data from arm-based format to contrast-based format
 # (internal call of metabin function). Argument 'sm' has to be used
 # for odds ratio as risk ratio (sm = "RR") is default of metabin
 # function.
 #
 pw3 <- pairwise(list(treat1, treat2, treat3),
   list(event1, event2, event3), list(n1, n2, n3),
   data = smokingcessation,
   sm = "OR")
 pw3

 # Conduct network meta-analysis
 #
 net3 <- netmeta(pw3)
 net3

 # Example with incidence rates
 #
 data(dietaryfat)

 # Transform data from arm-based format to contrast-based format
 #
 pw4 <- pairwise(list(treat1, treat2, treat3),
   list(d1, d2, d3), time = list(years1, years2, years3),
   studlab = ID,
   data = dietaryfat)
 pw4

 # Conduct network meta-analysis using incidence rate ratios (sm =
 # "IRR"). Note, the argument 'sm' is not necessary as this is the
 # default in R function metainc called internally.
 #
 net4 <- netmeta(pw4, sm = "IRR")
```

```
 summary(net4)

 # Example with long data format
 #
 # Transform data from long arm-based format to contrast-based
 # format Argument 'sm' has to be used for odds ratio as summary
 # measure; by default the risk ratio is used in the metabin
 # function called internally.
 #
 pw5 <- pairwise(treatment, event = r, n = N,
   studlab = author, data = dat.woods2010, sm = "OR")
 pw5

 # Conduct network meta-analysis
 net5 <- netmeta(pw5)
 net5
}

## End(Not run)
```

---

plot.cidprop *Plot density of prediction distribution highlighting areas of clinically important benefit or harm*

Description

Plot density of prediction distribution highlighting areas of clinically important benefit or harm

Usage

```
## S3 method for class 'cidprop'
plot(
  x,
  cid = NULL,
  cid.below.null = x$cid.below.null,
  cid.above.null = x$cid.above.null,
  label.cid = "",
  label.cid.below.null = x$label.cid.below.null,
  label.cid.above.null = x$label.cid.above.null,
  small.values = x$small.values,
  fill.cid.below.null = NULL,
  fill.cid.above.null = NULL,
  fill = "white",
  legend = FALSE,
  studies = TRUE,
  random = TRUE,
  col.diamond = gs("col.diamond"),
  col.diamond.lines = gs("col.diamond.lines"),
  prediction = TRUE,
  col.predict = gs("col.predict"),
  col.predict.lines = gs("col.predict.lines"),
  big.mark = gs("big.mark"),
  digits.cid = gs("digits.cid"),
  digits.percent = 1,
  digits.xaxis = gs("digits.forest"),
  xlab = NULL,
  ylab = NULL,
  xlim = NULL,
  ylim = NULL,
  labels.x = NULL,
  ...
)
```

Arguments

`x` An object of class `cidprop`.

`cid` A numeric value or vector specifying clinically important differences (CID) / decision thresholds used to calculate expected proportions of clinically important benefit or harm (see Details).

`cid.below.null` A numeric value or vector specifying CID limits below the null effect (see Details).

`cid.above.null` A numeric value or vector specifying CID limits above the null effect (see Details).

`label.cid` A character string or vector specifying labels for clinically important differences. Must be of same length as argument `cid`.

`label.cid.below.null` A character string or vector specifying labels for clinically important differences below the null effect. Must be of same length as argument `cid.below.null` (or `cid`).

`label.cid.above.null` A character string or vector specifying labels for clinically important differences above the null effect. Must be of same length as argument `cid.above.null` (or `cid`).

`small.values` A character string specifying whether small treatment effects indicate a beneficial (`"desirable"`) or harmful (`"undesirable"`), can be abbreviated.

`fill.cid.below.null` Background colour(s) for CID areas below null effect.

`fill.cid.above.null` Background colour(s) for CID areas above null effect.

`fill` Background colour for area between decision thresholds.

`legend` A logical indicating whether to print a legend with expected proportions of beneficial, harmful, or not important effects.

`studies` A logical indicating whether to print estimates of individual studies.

`random` A logical indicating whether to show diamond of the random effects meta-analysis.

`col.diamond` The colour of the diamond representing the results for the random effects model.

`col.diamond.lines` The colour of the outer lines of the diamond representing the results of the random effects model.

`prediction` A logical indicating whether to show the prediction interval.

`col.predict` The colour of the prediction interval.

`col.predict.lines` The colour of the outer lines of the prediction interval.

`big.mark` A character used as thousands separator.

`digits.cid` Minimal number of significant digits for decision thresholds, see `print.default`.

`digits.percent` Minimal number of significant digits for expected proportions, printed as percentages, see `print.default`.

`digits.xaxis` Minimal number of significant digits for labels on x-axis, see `print.default`.

`xlab` Label on x-axis.

`ylab` Label on y-axis.

`xlim` Limits for x-axis.

`ylim` Limits for y-axis.

`labels.x` Predefined labels for tick marks on x-axis.

`...` Additional arguments (ignored)

Details

Arguments `cid`, `cid.below.null`, `cid.above.null`, `label.cid`, `label.cid.below.null`, `label.cid.above.null`, and `small.values` are identical to the main arguments of R function `cidprop` which is called internally if any of these values has been provided by the user.

R packages **ggpubr** and **gridExtra** must be installed in order to add a legend to the plot with the CIDs, expected proportions of clinically benefit or harm, and the area colours (due to using R functions `ggarrange` and `tableGrob`). The data and colours shown in the legend are stored in the attribute 'data.cid' of the returned ggplot object (see Examples).

UTF-8 code for the less than or equal and greater than or equal signs are used in the legend. Accordingly, graphic devices with full UTF-8 support are required to save graphics, for example, `cairo_pdf` instead of `pdf` from R package **grDevices**.

Value

A ggplot object with additional class 'plot.cidprop'.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

See Also

cidprop

Examples

```
oldset <- settings.meta(digits.cid = 0)

m <- metagen(1:10 - 3, 1:10, sm = "MD")
#
pp1 <- cidprop(m, cid = 2)
pp1
#
pp2 <- cidprop(m, cid.below.null = 0.5, cid.above.null = 2)
pp2
#
pp3 <- cidprop(m, cid.below.null = 0.5, cid.above.null = 2,
  small.values = "u")
pp3

pp4 <- cidprop(m, cid = 1:2, label.cid = c("moderate", "large"))
pp4
#
pp5 <- cidprop(m, cid.below.null = -1.5, cid.above.null = 1:2,
  label.cid.below.null = "large",
  label.cid.above.null = c("moderate", "large"))
pp5

plot(pp1, xlim = c(-4, 4))
plot(pp2, xlim = c(-4, 4))
plot(pp3, xlim = c(-4, 4))
plot(pp4, xlim = c(-4, 4))
plpp5 <- plot(pp5, xlim = c(-4, 4))
plpp5
# Information on CIDs and colours
attr(plpp5, "data.cid")

## Not run:
# R packages 'ggpubr' and 'gridExtra' must be available
if (requireNamespace("ggpubr", quietly = TRUE) &
    requireNamespace("gridExtra", quietly = TRUE)) {
  #
  plot(pp1, xlim = c(-4, 4), legend = TRUE)
}

## End(Not run)

settings.meta(oldset)
```

print.meta *Print meta-analysis results*

Description

Print method for objects of class meta.

R function cilayout can be utilised to change the layout to print confidence intervals (both in printout from print.meta and print.summary.meta function as well as in forest plots). The default layout is "[lower; upper]". Another popular layout is "(lower - upper)" which is used throughout an R session by using R command `cilayout("(", " - ")`.

Argument `pscale` can be used to rescale single proportions or risk differences, e.g. `pscale = 1000` means that proportions are expressed as events per 1000 observations. This is useful in situations with (very) low event probabilities.

Argument `irscale` can be used to rescale single rates or rate differences, e.g. `irscale = 1000` means that rates are expressed as events per 1000 time units, e.g. person-years. This is useful in situations with (very) low rates. Argument `irunit` can be used to specify the time unit used in individual studies (default: "person-years"). This information is printed in summaries and forest plots if argument `irscale` is not equal to 1.

Usage

```
## S3 method for class 'meta'
print(
  x,
  common = x$common,
  random = x$random,
  prediction = x$prediction,
  overall = x$overall,
  overall.hetstat = x$overall.hetstat,
  test.subgroup = x$test.subgroup,
  test.subgroup.common = test.subgroup & common,
  test.subgroup.random = test.subgroup & random,
  prediction.subgroup = x$prediction.subgroup,
  backtransf = x$backtransf,
  pscale = x$pscale,
  irscale = x$irscale,
  irunit = x$irunit,
  subgroup.name = x$subgroup.name,
  print.subgroup.name = x$print.subgroup.name,
  sep.subgroup = x$sep.subgroup,
  nchar.subgroup = 35,
  sort.overall = NULL,
  sort.tau = NULL,
  sort.het = NULL,
  sort.Q = NULL,
  header = TRUE,
  print.CMH = x$print.CMH,
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = max(gs("digits.pval"), 2),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.Q = gs("digits.Q"),
  digits.df = gs("digits.df"),
  digits.pval.Q = max(gs("digits.pval.Q"), 2),
  digits.H = gs("digits.H"),
  digits.I2 = gs("digits.I2"),
  big.mark = gs("big.mark"),
  scientific.pval = gs("scientific.pval"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  print.tau2 = gs("print.tau2"),
  print.tau2.ci = gs("print.tau2.ci"),
  print.tau = gs("print.tau"),
  print.tau.ci = gs("print.tau.ci"),
  print.Q = gs("print.Q"),
  print.I2 = gs("print.I2"),
  print.I2.ci = gs("print.I2.ci"),
  print.H = gs("print.H"),
  print.Rb = gs("print.Rb"),
  text.tau2 = gs("text.tau2"),
  text.tau = gs("text.tau"),
  text.I2 = gs("text.I2"),
  text.Rb = gs("text.Rb"),
  details.methods = gs("details"),
  warn.backtransf = FALSE,
  func.backtransf = x$func.backtransf,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

cilayout(
  bracket = gs("CIbracket"),
  separator = gs("CIseparator"),
  lower.blank = gs("CIlower.blank"),
  upper.blank = gs("CIupper.blank")
)
```

Arguments

x	An object of class meta.
common	A logical indicating whether results for common effect meta-analysis should be printed.
random	A logical indicating whether results for random effects meta-analysis should be

printed.

`prediction` A logical indicating whether a prediction interval should be printed.

`overall` A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

`overall.hetstat` A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`test.subgroup` A logical value indicating whether to print results of test for subgroup differences.

`test.subgroup.common` A logical value indicating whether to print results of test for subgroup differences (based on common effect model).

`test.subgroup.random` A logical value indicating whether to print results of test for subgroup differences (based on random effects model).

`prediction.subgroup` A single logical or logical vector indicating whether / which prediction intervals should be printed for subgroups.

`backtransf` A logical indicating whether printed results should be back transformed. If `backtransf=TRUE`, results for `sm="OR"` are printed as odds ratios rather than log odds ratios and results for `sm="ZCOR"` are printed as correlations rather than Fisher's z transformed correlations, for example.

`pscale` A numeric giving scaling factor for printing of single event probabilities or risk differences, i.e. if argument `sm` is equal to `"PLOGIT"`, `"PLN"`, `"PRAW"`, `"PAS"`, `"PFT"`, or `"RD"`.

`irscale` A numeric defining a scaling factor for printing of single incidence rates or incidence rate differences, i.e. if argument `sm` is equal to `"IR"`, `"IRLN"`, `"IRS"`, `"IRFT"`, or `"IRD"`.

`irunit` A character specifying the time unit used to calculate rates, e.g. person-years.

`subgroup.name` A character string with a name for the grouping variable.

`print.subgroup.name` A logical indicating whether the name of the grouping variable should be printed in front of the group labels.

`sep.subgroup` A character string defining the separator between label and levels of grouping variable.

`nchar.subgroup` A numeric specifying the number of characters to print from subgroup labels.

`sort.overall` An optional vector used to sort meta-analysis results.

`sort.tau` An optional vector used to sort estimators of the between-study heterogeneity variance.

`sort.het` An optional vector used to sort heterogeneity statistics.

`sort.Q` An optional vector used to sort test of heterogeneity.

`header` A logical indicating whether information on title of meta-analysis, comparison and outcome should be printed at the beginning of the printout.

`print.CMH` A logical indicating whether result of the Cochran-Mantel-Haenszel test for overall effect should be printed.

`digits` Minimal number of significant digits, see `print.default`.

`digits.stat` Minimal number of significant digits for z- or t-value of test for overall effect, see `print.default`.

`digits.pval` Minimal number of significant digits for p-value of overall treatment effect, see `print.default`.

`digits.tau2` Minimal number of significant digits for between-study variance τ^2, see `print.default`.

`digits.tau` Minimal number of significant digits for τ, the square root of the between-study variance τ^2.

`digits.Q` Minimal number of significant digits for heterogeneity statistic Q, see `print.default`.

`digits.df` Minimal number of significant digits for degrees of freedom.

`digits.pval.Q` Minimal number of significant digits for p-value of heterogeneity test, see `print.default`.

`digits.H` Minimal number of significant digits for H statistic, see `print.default`.

`digits.I2` Minimal number of significant digits for I-squared and Rb statistic, see `print.default`.

`big.mark` A character used as thousands separator.

`scientific.pval`
A logical specifying whether p-values should be printed in scientific notation, e.g., 1.2345e-01 instead of 0.12345.

`zero.pval` A logical specifying whether p-values should be printed with a leading zero.

`JAMA.pval` A logical specifying whether p-values for test of overall effect should be printed according to JAMA reporting standards.

`print.tau2` A logical specifying whether between-study variance τ^2 should be printed.

`print.tau2.ci` A logical value indicating whether to print the confidence interval of τ^2.

`print.tau` A logical specifying whether τ, the square root of the between-study variance τ^2, should be printed.

`print.tau.ci` A logical value indicating whether to print the confidence interval of τ.

`print.Q` A logical value indicating whether to print the results of the test of heterogeneity.

`print.I2` A logical specifying whether heterogeneity statistic I^2 should be printed.

`print.I2.ci` A logical specifying whether confidence interval for heterogeneity statistic I^2 should be printed.

`print.H` A logical specifying whether heterogeneity statistic H should be printed.

`print.Rb` A logical specifying whether heterogeneity statistic R_b should be printed.

`text.tau2` Text printed to identify between-study variance τ^2.

`text.tau` Text printed to identify τ, the square root of the between-study variance τ^2.

`text.I2` Text printed to identify heterogeneity statistic I^2.

`text.Rb` Text printed to identify heterogeneity statistic R_b.

`details.methods`
A logical specifying whether details on statistical methods should be printed.

`warn.backtransf`
Deprecated argument (ignored).

`func.backtransf`
A function used to back-transform results.

`warn.deprecated`
A logical indicating whether warnings should be printed if deprecated arguments are used.

`...` Additional arguments (passed on to `prmatrix`).

`bracket` A character with bracket symbol to print lower confidence interval: "[", "(", "{", "".

`separator` A character string with information on separator between lower and upper confidence interval.

`lower.blank` A logical indicating whether blanks between left bracket and lower confidence limit should be printed.

`upper.blank` A logical indicating whether blanks between separator and upper confidence limit should be printed.

`print.metacum` *Print results of a cumulative meta-analysis*

Description

Print results of a cumulative meta-analysis

Usage

```
## S3 method for class 'metacum'
print(
  x,
  prediction = x$prediction,
  overall = x$overall,
  backtransf = x$backtransf,
  header = TRUE,
  lab.NA = ”.”,
  digits = gs(”digits”),
  digits.stat = gs(”digits.stat”),
  digits.pval = gs(”digits.pval”),
  digits.tau2 = gs(”digits.tau2”),
  digits.tau = gs(”digits.tau”),
  digits.I2 = gs(”digits.I2”),
  digits.cid = gs(”digits.cid”),
  digits.percent = 1,
  big.mark = gs(”big.mark”),
  scientific.pval = gs("scientific.pval"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  print.stat = FALSE,
  print.tau2 = TRUE,
  print.tau2.ci = FALSE,
  print.tau = TRUE,
  print.tau.ci = FALSE,
  print.I2 = TRUE,
  print.I2.ci = FALSE,
  print.prob = TRUE,
  text.tau2 = gs("text.tau2"),
  text.tau = gs("text.tau"),
  text.I2 = gs("text.I2"),
  details.methods = gs("details"),
  ...
)
```

Arguments

`x` An object of class `metacum`.

`prediction` A logical indicating whether prediction intervals should be printed.

`overall` A logical indicating whether overall results should be printed.

`backtransf` A logical indicating whether printed results should be back transformed. If `backtransf=TRUE`, results for `sm="OR"` are printed as odds ratios rather than log odds ratios, for example.

`header` A logical indicating whether information on title of meta-analysis, comparison and outcome should be printed at the beginning of the printout.

`lab.NA` A character string to label missing values.

`digits` Minimal number of significant digits, see `print.default`.

`digits.stat` Minimal number of significant digits for z- or t-value of test for overall effect, see `print.default`.

`digits.pval` Minimal number of significant digits for p-value of overall treatment effect, see `print.default`.

`digits.tau2` Minimal number of significant digits for between-study variance τ^2, see `print.default`.

`digits.tau` Minimal number of significant digits for τ, the square root of the between-study variance τ^2.

`digits.I2` Minimal number of significant digits for I-squared and Rb statistic, see `print.default`.

`digits.cid` Minimal number of significant digits for CID / decision thresholds, see `print.default`.

`digits.percent` Minimal number of significant digits for probabilities, printed as percentages, see `print.default`.

`big.mark` A character used as thousands separator.

`scientific.pval`

A logical specifying whether p-values should be printed in scientific notation, e.g., 1.2345e-01 instead of 0.12345.

`zero.pval` A logical specifying whether p-values should be printed with a leading zero.

`JAMA.pval` A logical specifying whether p-values for test of overall effect should be printed according to JAMA reporting standards.

`print.stat` A logical value indicating whether z- or t-value for test of treatment effect should be printed.

`print.tau2` A logical specifying whether between-study variance τ^2 should be printed.

`print.tau2.ci` A logical value indicating whether to print the confidence interval of τ^2.

`print.tau` A logical specifying whether τ, the square root of the between-study variance τ^2, should be printed.

`print.tau.ci` A logical value indicating whether to print the confidence interval of τ.

`print.I2` A logical specifying whether heterogeneity statistic I^2 should be printed.

`print.I2.ci` A logical specifying whether confidence interval for heterogeneity statistic I^2 should be printed.

`print.prob` A logical specifying whether to print probabilities of clinically important benefit or harm.

`text.tau2` Text printed to identify between-study variance τ^2.

`text.tau` Text printed to identify τ, the square root of the between-study variance τ^2.

`text.I2` Text printed to identify heterogeneity statistic I^2.

`details.methods`
A logical specifying whether details on statistical methods should be printed.

`...` Additional arguments (ignored).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

metacum, settings.meta

Examples

```
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac,
  data = Fleiss1993bin, studlab = study, sm = "RR", method = "I")
m1
metacum(m1)
metacum(m1, pooled = "random")
metacum(m1, pooled = "random", prediction = TRUE)
```

print.metainf *Print results of a leave-one-out meta-analysis*

Description

Print results of a leave-one-out meta-analysis

Usage

```
## S3 method for class 'metainf'
print(
  x,
  prediction = x$prediction,
  overall = x$overall,
  backtransf = x$backtransf,
  header = TRUE,
  lab.NA = ".",
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = gs("digits.pval"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.I2 = gs("digits.I2"),
  digits.cid = gs("digits.cid"),
  digits.percent = 1,
  big.mark = gs("big.mark"),
  scientific.pval = gs("scientific.pval"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  print.stat = FALSE,
  print.tau2 = TRUE,
  print.tau2.ci = FALSE,
  print.tau = TRUE,
  print.tau.ci = FALSE,
  print.I2 = TRUE,
  print.I2.ci = FALSE,
  print.prob = TRUE,
  text.tau2 = gs("text.tau2"),
  text.tau = gs("text.tau"),
  text.I2 = gs("text.I2"),
  details.methods = gs("details"),
  ...
)
```

Arguments

x An object of class metainf.

`prediction` A logical indicating whether prediction intervals should be printed.

`overall` A logical indicating whether overall results should be printed.

`backtransf` A logical indicating whether printed results should be back transformed. If `backtransf=TRUE`, results for `sm="OR"` are printed as odds ratios rather than log odds ratios, for example.

`header` A logical indicating whether information on title of meta-analysis, comparison and outcome should be printed at the beginning of the printout.

`lab.NA` A character string to label missing values.

`digits` Minimal number of significant digits, see `print.default`.

`digits.stat` Minimal number of significant digits for z- or t-value of test for overall effect, see `print.default`.

`digits.pval` Minimal number of significant digits for p-value of overall treatment effect, see `print.default`.

`digits.tau2` Minimal number of significant digits for between-study variance τ^2, see `print.default`.

`digits.tau` Minimal number of significant digits for τ, the square root of the between-study variance τ^2.

`digits.I2` Minimal number of significant digits for I-squared and Rb statistic, see `print.default`.

`digits.cid` Minimal number of significant digits for CID / decision thresholds, see `print.default`.

`digits.percent` Minimal number of significant digits for probabilities, printed as percentages, see `print.default`.

`big.mark` A character used as thousands separator.

`scientific.pval`
A logical specifying whether p-values should be printed in scientific notation, e.g., 1.2345e-01 instead of 0.12345.

`zero.pval` A logical specifying whether p-values should be printed with a leading zero.

`JAMA.pval` A logical specifying whether p-values for test of overall effect should be printed according to JAMA reporting standards.

`print.stat` A logical value indicating whether z- or t-value for test of treatment effect should be printed.

`print.tau2` A logical specifying whether between-study variance τ^2 should be printed.

`print.tau2.ci` A logical value indicating whether to print the confidence interval of τ^2.

`print.tau` A logical specifying whether τ, the square root of the between-study variance τ^2, should be printed.

`print.tau.ci` A logical value indicating whether to print the confidence interval of τ.

`print.I2` A logical specifying whether heterogeneity statistic I^2 should be printed.

`print.I2.ci` A logical specifying whether confidence interval for heterogeneity statistic I^2 should be printed.

`print.prob` A logical specifying whether to print probabilities of clinically important benefit or harm.

`text.tau2` Text printed to identify between-study variance τ^2.

`text.tau` Text printed to identify τ, the square root of the between-study variance τ^2.

`text.I2`	Text printed to identify heterogeneity statistic I^2.
`details.methods`	A logical specifying whether details on statistical methods should be printed.
`...`	Additional arguments (ignored).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

metainf, settings.meta

Examples

```
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac,
  data = Fleiss1993bin, studlab = study, sm = "RR", method = "I")
m1
metainf(m1)
metainf(m1, pooled = "random")
metainf(m1, pooled = "random", prediction = TRUE)
```

print.rm5 *Cochrane review: summary of meta-analyses*

Description

Calculate and print a summary of all meta-analyses in a Cochrane review.

Usage

```
## S3 method for class 'rm5'
print(x, comp.no, outcome.no, ...)
```

Arguments

`x`	An object of class `rm5`.
`comp.no`	Comparison number.
`outcome.no`	Outcome number.
`...`	Additional arguments (passed on to `metacr`).

Details

This function can be used to redo all or selected meta-analyses of a Cochrane Review of interventions (Higgins et al., 2023).

Review Manager 5 (RevMan 5) was the software used for preparing and maintaining Cochrane Reviews. In RevMan 5, subgroup analyses can be defined and data from a Cochrane review can be imported to R using the function `read.rm5`.

The R function `metacr` is called internally.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

References

Higgins JPT, Thomas J, Chandler J, Cumpston M, Li T, Page MJ, Welch VA (editors) (2023): *Cochrane Handbook for Systematic Reviews of Interventions Version 6.4 (updated August 2023)*. Available from `https://training.cochrane.org/handbook/`

See Also

`summary.meta`, `metacr`, `read.rm5`, `metabias.rm5`

Examples

```
# Locate export data file "Fleiss1993_CR.csv"
# in sub-directory of package "meta"
#
filename <- system.file("extdata", "Fleiss1993_CR.csv", package = "meta")
Fleiss1993_CR <- read.rm5(filename)

# Print results for all meta-analysis
#
Fleiss1993_CR

# Print results only for second outcome of first comparison
#
print(Fleiss1993_CR, comp.no = 1, outcome.no = 2)
```

`print.summary.meta` *Print detailed meta-analysis results*

Description

Print method for objects of class `summary.meta`.

Usage

```
## S3 method for class 'summary.meta'
print(
  x,
  sortvar,
  common = x$x$common,
  random = x$x$random,
  details = FALSE,
  ma = TRUE,
  overall = x$overall & ma,
  backtransf = x$backtransf,
  pscale = x$pscale,
  irscale = x$irscale,
  irunit = x$irunit,
  digits = gs("digits"),
  digits.se = gs("digits.se"),
  digits.stat = gs("digits.stat"),
  digits.pval = max(gs("digits.pval"), 2),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.Q = gs("digits.Q"),
  digits.df = gs("digits.df"),
  digits.pval.Q = max(gs("digits.pval.Q"), 2),
  digits.H = gs("digits.H"),
  digits.I2 = gs("digits.I2"),
  digits.prop = gs("digits.prop"),
  digits.weight = gs("digits.weight"),
  scientific.pval = gs("scientific.pval"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  big.mark = gs("big.mark"),
  print.tau2 = gs("print.tau2"),
  print.tau2.ci = gs("print.tau2.ci"),
  print.tau = gs("print.tau"),
  print.tau.ci = gs("print.tau.ci"),
  print.Q = gs("print.Q"),
  print.I2 = gs("print.I2"),
  print.I2.ci = gs("print.I2.ci"),
  print.H = gs("print.H"),
  print.Rb = gs("print.Rb"),
  text.tau2 = gs("text.tau2"),
  text.tau = gs("text.tau"),
  text.I2 = gs("text.I2"),
  text.Rb = gs("text.Rb"),
  truncate,
  text.truncate = "*** Output truncated ***",
  details.methods = TRUE,
  warn.backtransf = FALSE,
  ...
)
```

Arguments

x An object of class summary.meta

sortvar An optional vector used to sort the individual studies (must be of same length as x$TE).

common A logical indicating whether results of common effect meta-analysis should be printed.

random A logical indicating whether results of random effects meta-analysis should be printed.

details A logical indicating whether further details of individual studies should be printed.

ma A logical indicating whether the summary results of the meta-analysis should be printed.

overall A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

backtransf A logical indicating whether printed results should be back transformed. If backtransf = TRUE, results for sm = "OR" are printed as odds ratios rather than log odds ratios and results for sm = "ZCOR" are printed as correlations rather than Fisher's z transformed correlations, for example.

pscale A numeric giving scaling factor for printing of single event probabilities or risk differences, i.e. if argument sm is equal to "PLOGIT", "PLN", "PRAW", "PAS", "PFT", or "RD".

irscale A numeric defining a scaling factor for printing of single incidence rates or incidence rate differences, i.e. if argument sm is equal to "IR", "IRLN", "IRS", "IRFT", or "IRD".

irunit A character specifying the time unit used to calculate rates, e.g. person-years.

digits Minimal number of significant digits, see print.default.

digits.se Minimal number of significant digits for standard deviations and standard errors, see print.default.

digits.stat Minimal number of significant digits for z- or t-value of test for effect, see print.default.

digits.pval Minimal number of significant digits for p-value of test of treatment effect, see print.default.

digits.tau2 Minimal number of significant digits for between-study variance, see print.default.

digits.tau Minimal number of significant digits for square root of between-study variance, see print.default.

digits.Q Minimal number of significant digits for heterogeneity statistic Q, see print.default.

digits.df Minimal number of significant digits for degrees of freedom.

digits.pval.Q Minimal number of significant digits for p-value of heterogeneity test, see print.default.

digits.H Minimal number of significant digits for H statistic, see print.default.

`digits.I2` Minimal number of significant digits for I-squared and Rb statistic, see `print.default`.

`digits.prop` Minimal number of significant digits for proportions, see `print.default`.

`digits.weight` Minimal number of significant digits for weights, see `print.default`.

`scientific.pval`
A logical specifying whether p-values should be printed in scientific notation, e.g., 1.2345e-01 instead of 0.12345.

`zero.pval` A logical specifying whether p-values should be printed with a leading zero.

`JAMA.pval` A logical specifying whether p-values for test of overall effect should be printed according to JAMA reporting standards.

`big.mark` A character used as thousands separator.

`print.tau2` A logical specifying whether between-study variance τ^2 should be printed.

`print.tau2.ci` A logical value indicating whether to print the confidence interval of τ^2.

`print.tau` A logical specifying whether τ, the square root of the between-study variance τ^2, should be printed.

`print.tau.ci` A logical value indicating whether to print the confidence interval of τ.

`print.Q` A logical value indicating whether to print the results of the test of heterogeneity.

`print.I2` A logical specifying whether heterogeneity statistic I^2 should be printed.

`print.I2.ci` A logical specifying whether confidence interval for heterogeneity statistic I^2 should be printed.

`print.H` A logical specifying whether heterogeneity statistic H should be printed.

`print.Rb` A logical specifying whether heterogeneity statistic R$_b$ should be printed.

`text.tau2` Text printed to identify between-study variance τ^2.

`text.tau` Text printed to identify τ, the square root of the between-study variance τ^2.

`text.I2` Text printed to identify heterogeneity statistic I^2.

`text.Rb` Text printed to identify heterogeneity statistic R$_b$.

`truncate` An optional vector used to truncate the printout of results for individual studies (must be a logical vector of same length as `x$TE` or contain numerical values).

`text.truncate` A character string printed if study results were truncated from the printout.

`details.methods`
A logical specifying whether details on statistical methods should be printed.

`warn.backtransf`
Deprecated argument (ignored).

`...` Additional arguments (passed on to `print.meta` called internally).

Details

Print method for objects of class `summary.meta` giving detailed information on the meta-analysis.

Argument `pscale` can be used to rescale single proportions or risk differences, e.g. `pscale = 1000` means that proportions are expressed as events per 1000 observations. This is useful in situations with (very) low event probabilities.

Argument irscale can be used to rescale single rates or rate differences, e.g. irscale = 1000 means that rates are expressed as events per 1000 time units, e.g. person-years. This is useful in situations with (very) low rates. Argument irunit can be used to specify the time unit used in individual studies (default: "person-years"). This information is printed in summaries and forest plots if argument irscale is not equal to 1.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Cooper H & Hedges LV (1994), *The Handbook of Research Synthesis*. Newbury Park, CA: Russell Sage Foundation.

Crippa A, Khudyakov P, Wang M, Orsini N, Spiegelman D (2016), A new measure of between-studies heterogeneity in meta-analysis. *Statistics in Medicine*, **35**, 3661–75.

Higgins JPT & Thompson SG (2002), Quantifying heterogeneity in a meta-analysis. *Statistics in Medicine*, **21**, 1539–58.

See Also

summary.meta, update.meta, metabin, metacont, metagen

Examples

```
data(Fleiss1993cont)
m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD", studlab = paste(study, year))
sm1 <- summary(m1)
sm1

print(sm1, digits = 2)

## Not run:
# Use unicode characters to print tau^2, tau, and I^2
print(sm1,
  text.tau2 = "\u03c4\u00b2",
  text.tau = "\u03c4", text.I2 = "I\u00b2")

## End(Not run)
```

radial.meta *Radial plot*

Description

Draw a radial plot (also called Galbraith plot) which can be used to assess bias in meta-analysis.

Usage

```
## S3 method for class 'meta'
radial(
  x,
  xlim = NULL,
  ylim = NULL,
  xlab = "Inverse of standard error”,
  ylab = "Standardised treatment effect (z-score)”,
  common = TRUE,
  axes = TRUE,
  pch = 1,
  text = NULL,
  cex = 1,
  col = NULL,
  level = NULL,
  warn.deprecated = gs(”warn.deprecated”),
  fixed,
  ...
)

## Default S3 method:
radial(
  x,
  y,
  xlim = NULL,
  ylim = NULL,
  xlab = "Inverse of standard error”,
  ylab = "Standardised treatment effect (z-score)”,
  common = TRUE,
  axes = TRUE,
  pch = 1,
  text = NULL,
  cex = 1,
  col = NULL,
  level = NULL,
  ...
)
```

Arguments

`x` An object of class `meta`, or estimated treatment effect in individual studies.

`xlim` The x limits (min, max) of the plot.

`ylim` The y limits (min, max) of the plot.

`xlab` A label for the x-axis.

`ylab` A label for the y-axis.

`common` A logical indicating whether the pooled common effect estimate should be plotted.

axes	A logical indicating whether axes should be drawn on the plot.
pch	The plotting symbol used for individual studies.
text	A character vector specifying the text to be used instead of plotting symbol.
cex	The magnification to be used for plotting symbol.
col	A vector with colour of plotting symbols.
level	The confidence level utilised in the plot.
warn.deprecated	A logical indicating whether warnings should be printed if deprecated arguments are used.
fixed	Deprecated argument (replaced by 'common').
...	Graphical arguments as in par may also be passed as arguments.
y	Standard error of estimated treatment effect.

Details

A radial plot (Galbraith 1988a,b), also called Galbraith plot, is drawn in the active graphics window. If common is TRUE, the pooled estimate of the common effect model is plotted. If level is not NULL, the corresponding confidence limits are drawn.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Galbraith RF (1988a): Graphical display of estimates having differing standard errors. *Technometrics*, **30**, 271–81

Galbraith RF (1988b): A note on graphical presentation of estimated odds ratios from several clinical trials. *Statistics in Medicine*, **7**, 889–94

See Also

metabias, metabin, metagen, funnel

Examples

```
data(Olkin1995)
m1 <- metabin(ev.exp, n.exp, ev.cont, n.cont,
  data = Olkin1995, subset = c(41, 47, 51, 59),
  studlab = paste(author, year), sm = "RR", method = "I")

# Radial plot
#
radial(m1, level = 0.95)
```

read.cdir *Import data of Cochrane intervention review*

Description

Reads Cochrane data package (version 1) of a Cochrane intervention review and creates a data frame from it.

Usage

```
read.cdir(
  file,
  title = "Cochrane Review of Interventions",
  exdir = tempdir(),
  numbers.in.labels = TRUE,
 rob = !missing(tool) | !missing(categories) | !missing(col) | !missing(symbols),
  tool = NULL,
  categories = NULL,
  col = NULL,
  symbols = NULL,
  keep.orig = FALSE,
  ...
)

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

Arguments

`file`	The name of a file to read data values from.
`title`	Title of Cochrane review.
`exdir`	The directory to extract files to (the equivalent of 'unzip -d'). It will be created if necessary.
`numbers.in.labels`	A logical indicating whether comparison number and outcome number should be printed at the beginning of the comparison (argument `complab`) and outcome label (argument `outclab`); this is the default in RevMan Web.
`rob`	A logical indicating whether risk of bias (RoB) assessment should be considered in meta-analyses.
`tool`	Risk of bias (RoB) tool.
`categories`	Possible RoB categories.
`col`	Colours for RoB categories.
`symbols`	Corresponding symbols for RoB categories.
`keep.orig`	A logical indicating whether to return the original data files.
`...`	Additional arguments (passed on to `unzip`)
`x`	An object of class "cdir".

Details

RevMan Web is the current software used for preparing and maintaining Cochrane reviews. RevMan Web includes the ability to write systematic reviews of interventions or diagnostic test accuracy reviews.

This function provides the ability to read the Cochrane data package from a Cochrane intervention review created with RevMan Web. The ZIP-file is extracted with `unzip`.

Argument `title` can be used to overwrite the title of the Cochrane review.

Information on the risk of bias (RoB) assessment can be provided with arguments `tool`, `categories`, `col` and `symbols`. This is only useful if (i) all outcomes are based on the same RoB categories and (ii) an overall RoB assessment has not been done. If no overall RoB assessment was conducted, R function `metacr` can be used to provide the RoB information for a single outcome. R function `rob` is the most flexible way to add RoB information to a meta-analysis object.

Creation of Cochrane data package:

Two possible ways exist to create the ZIP-file.

In RevMan Web, press the "Export" button at the bottom of the *Default view* website. After a couple of seconds, the data package will be shown at the bottom of the *Default view* website under "Downloads".

In the Cochrane Library, press on "Download statistical data" in the Contents menu to download an rm5-file. This file can be converted to a data package in RevMan Web using *Help* - *Convert a RevMan 5 file*.

Value

A list consisting of a data frame 'data' with the study data and (if available) a data frame 'rob' with information on the risk of bias assessment. If `keep.orig = TRUE`, an additional list 'orig' is returned containing elements 'settings', 'datarows', 'subgroup' and 'rob' (if available).

The data frame 'data' contains the following variables:

`comp.no`	Comparison number.
`outcome.no`	Outcome number.
`group.no`	Group number.
`studlab`	Study label.
`year`	Year of publication.
`event.e`	Number of events in experimental group.
`n.e`	Number of observations in experimental group.
`event.c`	Number of events in control group.
`n.c`	Number of observations in control group.
`mean.e`	Estimated mean in experimental group.
`sd.e`	Standard deviation in experimental group.
`mean.c`	Estimated mean in control group.
`sd.c`	Standard deviation in control group.
`O.E`	Observed minus expected (IPD analysis).

`V` Variance of `O.E` (IPD analysis).

`TE`, `seTE` Estimated treatment effect and standard error of individual studies.

`lower`, `upper` Lower and upper limit of 95% confidence interval for treatment effect in individual studies.

`weight` Weight of individual studies (according to meta-analytical method used in respective meta-analysis - see details).

`order` Ordering of studies.

`grplab` Group label.

`type` Type of outcome. D = dichotomous, C = continuous, P = IPD.

`method` A character string indicating which method has been used for pooling of studies. One of `"Inverse"`, `"MH"`, or `"Peto"`.

`sm` A character string indicating which summary measure has been used for pooling of studies.

`model` A character string indicating which meta-analytical model has been used (either `"Fixed"` or `"Random"`).

`common` A logical indicating whether common effect meta-analysis has been used in respective meta-analysis (see details).

`random` A logical indicating whether random effects meta-analysis has been used in respective meta-analysis (see details).

`title` Title of Cochrane review.

`complab` Comparison label.

`outclab` Outcome label.

`label.e` Label for experimental group.

`label.c` Label for control group.

`label.left` Graph label on left side of forest plot.

`label.right` Graph label on right side of forest plot.

The data frame ’rob’ contains the following variables:

`studlab` Study label.

`D1`, `D2`, `...` Risk of bias domain 1, 2, ...

`D1.details`, `D2.details`, `...`
Details on risk of bias domain 1, 2, ...

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

References

https://documentation.cochrane.org/revman-kb/data-package-user-guide-243761660.html

https://documentation.cochrane.org/revman-kb/data-package-specification-249561249.html

See Also

metacr, rob, read.rm5

Examples

```
# Locate file "Fleiss1993.zip" with Cochrane data package in
# sub-directory of R package meta
#
filename <- system.file("extdata/Fleiss1993.zip", package = "meta")
Fleiss1993_CR <- read.cdir(filename)
Fleiss1993_CR

# Same result as R Command example(Fleiss1993bin):
#
metacr(Fleiss1993_CR)

# Same result as R Command example(Fleiss1993cont):
#
metacr(Fleiss1993_CR, 1, 2)
```

---

read.mtv *Import RevMan 4 data files (.mtv)*

Description

Reads a file created with RevMan 4 and creates a data frame from it.

Usage

```
read.mtv(file)
```

Arguments

`file`	The name of a file to read data values from.

Details

Reads a file created with RevMan 4 (Menu: "File" - "Export" - "Analysis data file...") and creates a data frame from it.

Value

A data frame containing the following components:

`comp.no`	Comparison number.
`outcome.no`	Outcome number.
`group.no`	Group number.

`studlab`	Study label.
`year`	Year of publication.
`event.e`	Number of events in experimental group.
`n.e`	Number of observations in experimental group.
`event.c`	Number of events in control group.
`n.c`	Number of observations in control group.
`mean.e`	Estimated mean in experimental group.
`sd.e`	Standard deviation in experimental group.
`mean.c`	Estimated mean in control group.
`sd.c`	Standard deviation in control group.
`O.E`	Observed minus expected (IPD analysis).
`V`	Variance of `O.E` (IPD analysis).
`order`	Ordering of studies.
`conceal`	Concealment of treatment allocation.
`grplab`	Group label.
`type`	Type of outcome. D = dichotomous, C = continuous, P = IPD.
`outclab`	Outcome label.
`graph.exp`	Graph label for experimental group.
`graph.cont`	Graph label for control group.
`label.exp`	Label for experimental group.
`label.cont`	Label for control group.
`complab`	Comparison label.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

References

Review Manager (RevMan) [Computer program]. Version 4.2. Copenhagen: The Nordic Cochrane Centre, The Cochrane Collaboration, 2003

See Also

`metabin`, `metacont`, `metagen`

Examples

```
# Locate MTV-data file "FLEISS1993.MTV" in sub-directory of R package
# meta
#
filename <- system.file("extdata/FLEISS1993.MTV", package = "meta")
fleiss1933.cc <- read.mtv(filename)

# Same result as R Command example(Fleiss1993bin):
#
metabin(event.e, n.e, event.c, n.c,
  data = fleiss1933.cc, subset = type == "D",
  studlab = paste(studlab, year))

# Same result: example(Fleiss1993cont)
#
metacont(n.e, mean.e, sd.e, n.c, mean.c, sd.c,
  data = fleiss1933.cc, subset = type == "C",
  studlab = paste(studlab, year))
```

---

read.rm5 *Import RevMan 5 analysis data*

Description

Reads analysis data from Cochrane intervention review created with RevMan 5 and creates a data frame from it.

Usage

```
read.rm5(
  file,
  sep = ",",
  quote = "\"",
  title,
  numbers.in.labels = TRUE,
  debug = 0
)
```

Arguments

file	The name of a file to read data values from.
sep	The field separator character (only considered for CSV-files). Values on each line of the file are separated by this character. The comma is the default field separator character in RevMan 5.
quote	The set of quoting characters (only considered for CSV-files). In RevMan 5 a "\"" is the default quoting character.

`title` Title of Cochrane review.

`numbers.in.labels`
A logical indicating whether comparison number and outcome number should be printed at the beginning of the comparison (argument `complab`) and outcome label (argument `outclab`); this is the default in RevMan 5.

`debug` An integer between 0 and 3 indicating whether to print debug messages (only considered for RM5-files).

Details

Review Manager 5 (RevMan 5) was the software used for preparing and maintaining Cochrane reviews. RevMan 5 includes the ability to write systematic reviews of interventions, diagnostic test accuracy reviews, methodology reviews and overviews of reviews.

This function provides the ability to read the analysis data from a Cochrane intervention review created with RevMan 5; a data frame is created from it. Cochrane intervention reviews are based on comparisons of two interventions.

By default in RevMan 5, the name of the exported CSV data file is the title of the Cochrane review. Furthermore, the title is part of the RM5-file. Argument `title` can be used to overwrite the title of the Cochrane review.

Import RM5-file:

A RM5-file (which is in a specific XML format) can be used directly to import the analysis dataset.

If the import fails, use argument `debug = 3` for more details.

Import CSV-file:

In the past, the following (rather complicated) procedure based on a CSV-file generated within RevMan 5 was necessary - which is only described here for backward compatibility.

In order to generate a data analysis file in RevMan 5 use the following Menu points: `"File"` - `"Export"` - `"Data and analyses"`. It is mandatory to include the following fields in the exported data file by selecting them with the mouse cursor in the Export Analysis Data Wizard: (i) Comparison Number, (ii) Outcome Number, (iii) Subgroup Number. When these fields are not selected a corresponding error message will be printed in R. It is recommended to include all fields in the exported data file except for the last field "Risk of bias tables". For example, in order to redo the meta-analysis in R for the RevMan 5 data type `"O-E and Variance"` the fields `"O-E"` and `"Variance"` have to be selected in the Export Analysis Data Wizard. If the last field "Risk of bias tables" is selected the import in R fails with an error message "line X did not have Y elements".

Value

A data frame containing the following components:

`comp.no`	Comparison number.
`outcome.no`	Outcome number.
`group.no`	Group number.
`studlab`	Study label.
`year`	Year of publication.
`event.e`	Number of events in experimental group.

`n.e` Number of observations in experimental group.

`event.c` Number of events in control group.

`n.c` Number of observations in control group.

`mean.e` Estimated mean in experimental group.

`sd.e` Standard deviation in experimental group.

`mean.c` Estimated mean in control group.

`sd.c` Standard deviation in control group.

`O.E` Observed minus expected (IPD analysis).

`V` Variance of `O.E` (IPD analysis).

`TE`, `seTE` Estimated treatment effect and standard error of individual studies.

`lower`, `upper` Lower and upper limit of 95% confidence interval for treatment effect in individual studies.

`weight` Weight of individual studies (according to meta-analytical method used in respective meta-analysis - see details).

`order` Ordering of studies.

`grplab` Group label.

`type` Type of outcome. D = dichotomous, C = continuous, P = IPD.

`method` A character string indicating which method has been used for pooling of studies. One of `"Inverse"`, `"MH"`, or `"Peto"`.

`sm` A character string indicating which summary measure has been used for pooling of studies.

`model` A character string indicating which meta-analytical model has been used (either `"Fixed"` or `"Random"`).

`common` A logical indicating whether common effect meta-analysis has been used in respective meta-analysis (see details).

`random` A logical indicating whether random effects meta-analysis has been used in respective meta-analysis (see details).

`outclab` Outcome label.

`k` Total number of studies combined in respective meta-analysis).

`event.e.pooled` Number of events in experimental group in respective meta-analysis (see details).

`n.e.pooled` Number of observations in experimental group in respective meta-analysis (see details).

`event.c.pooled` Number of events in control group in respective meta-analysis (see details).

`n.c.pooled` Number of observations in control group in respective meta-analysis (see details).

`TE.pooled` Estimated treatment effect in respective meta-analysis (see details).

`lower`, `upper` Lower and upper limit of 95% confidence interval for treatment effect in respective meta-analysis (see details).

`weight.pooled` Total weight in respective meta-analysis (see details).

`Z.pooled` Z-score for test of overall treatment effect in respective meta-analysis (see details).

`pval.pooled` P-value for test of overall treatment effect in respective meta-analysis (see details).

`Q` Heterogeneity statistic Q in respective meta-analysis (see details).

`pval.Q` P-value of heterogeneity statistic Q in respective meta-analysis (see details).

`I2` Heterogeneity statistic I^2 in respective meta-analysis (see details).

`tau2` Between-study variance (moment estimator of DerSimonian-Laird) in respective meta-analysis.

`Q.w` Heterogeneity statistic Q within groups in respective meta-analysis (see details).

`pval.Q.w` P-value of heterogeneity statistic Q within groups in respective meta-analysis (see details).

`I2.w` Heterogeneity statistic I^2 within groups in respective meta-analysis (see details).

`label.e` Label for experimental group.

`label.c` Label for control group.

`label.left` Graph label on left side of forest plot.

`label.right` Graph label on right side of forest plot.

`complab` Comparison label.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Review Manager (RevMan) [Computer program]. Version 5.4. The Cochrane Collaboration, 2020

See Also

summary.rm5, metabias.rm5, metabin, metacont, metagen, metacr, print.rm5

Examples

```
# Locate export data file "Fleiss1993_CR.csv"
# in sub-directory of package "meta"
#
filename <- system.file("extdata", "Fleiss1993_CR.csv", package = "meta")
Fleiss1993_CR <- read.rm5(filename)

# Same result as R command example(Fleiss1993bin):
#
metacr(Fleiss1993_CR)

# Same result as R command example(Fleiss1993cont):
#
metacr(Fleiss1993_CR, 1, 2)
```

```
## Not run:
# Locate file "Fleiss1993.rm5" in sub-directory of R package meta
#
filename <- system.file("extdata/Fleiss1993.rm5", package = "meta")
Fleiss1993_CR <- read.cdir(filename)
Fleiss1993_CR

# Same result as R Command example(Fleiss1993bin):
#
metacr(Fleiss1993_CR)

## End(Not run)
```

rob *Risk of bias assessment*

Description

Create table with risk of bias assessment or add table to existing meta-analysis

Usage

```
rob(
  item1,
  item2 = NULL,
  item3 = NULL,
  item4 = NULL,
  item5 = NULL,
  item6 = NULL,
  item7 = NULL,
  item8 = NULL,
  item9 = NULL,
  item10 = NULL,
  studlab = NULL,
  overall = NULL,
  weight = NULL,
  data = NULL,
  tool = gs("tool.rob"),
  domains = NULL,
  categories = NULL,
  cat1 = categories,
  cat2 = categories,
  cat3 = categories,
  cat4 = categories,
  cat5 = categories,
  cat6 = categories,
  cat7 = categories,
  cat8 = categories,
  cat9 = categories,
  cat10 = categories,
  cat.overall = categories,
  col = NULL,
  col1 = col,
  col2 = col,
  col3 = col,
  col4 = col,
  col5 = col,
  col6 = col,
  col7 = col,
  col8 = col,
  col9 = col,
  col10 = col,
  col.overall = col,
  symbols = NULL,
  symb1 = symbols,
  symb2 = symbols,
  symb3 = symbols,
  symb4 = symbols,
  symb5 = symbols,
  symb6 = symbols,
  symb7 = symbols,
  symb8 = symbols,
  symb9 = symbols,
  symb10 = symbols,
  symb.overall = symbols,
  legend = TRUE,
  overwrite = FALSE,
  warn = TRUE
)

## S3 method for class 'rob'
print(x, legend = attr(x, ”legend”), details = TRUE, ...)
```

Arguments

`item1` Risk of bias item 1 or a meta-analysis object of class `meta` with information on risk of bias assessment.

`item2` Risk of bias item 2.

`item3` Risk of bias item 3.

`item4` Risk of bias item 4.

`item5` Risk of bias item 5.

`item6` Risk of bias item 6.

`item7` Risk of bias item 7.

`item8`	Risk of bias item 8.
`item9`	Risk of bias item 9.
`item10`	Risk of bias item 10.
`studlab`	Study labels.
`overall`	Overall risk of bias assess.
`weight`	Weight for each study.
`data`	A data frame or a meta-analysis object of class `meta`.
`tool`	Risk of bias (RoB) tool.
`domains`	A character vector with names of RoB domains.
`categories`	Possible RoB categories.
`cat1`	Possible categories for RoB item 1.
`cat2`	Possible categories for RoB item 2.
`cat3`	Possible categories for RoB item 3.
`cat4`	Possible categories for RoB item 4.
`cat5`	Possible categories for RoB item 5.
`cat6`	Possible categories for RoB item 6.
`cat7`	Possible categories for RoB item 7.
`cat8`	Possible categories for RoB item 8.
`cat9`	Possible categories for RoB item 9.
`cat10`	Possible categories for RoB item 10.
`cat.overall`	Possible categories for overall RoB.
`col`	Colours for RoB categories.
`col1`	Colours for categories for RoB item 1.
`col2`	Colours for categories for RoB item 2.
`col3`	Colours for categories for RoB item 3.
`col4`	Colours for categories for RoB item 4.
`col5`	Colours for categories for RoB item 5.
`col6`	Colours for categories for RoB item 6.
`col7`	Colours for categories for RoB item 7.
`col8`	Colours for categories for RoB item 8.
`col9`	Colours for categories for RoB item 9.
`col10`	Colours for categories for RoB item 10.
`col.overall`	Colours for categories for overall RoB.
`symbols`	Corresponding symbols for RoB categories.
`symb1`	Corresponding symbols for RoB item 1.
`symb2`	Corresponding symbols for RoB item 2.
`symb3`	Corresponding symbols for RoB item 3.

`symb4` Corresponding symbols for RoB item 4.

`symb5` Corresponding symbols for RoB item 5.

`symb6` Corresponding symbols for RoB item 6.

`symb7` Corresponding symbols for RoB item 7.

`symb8` Corresponding symbols for RoB item 8.

`symb9` Corresponding symbols for RoB item 9.

`symb10` Corresponding symbols for RoB item 10.

`symb.overall` Corresponding symbols for overall RoB.

`legend` A logical specifying whether legend with RoB domains should be printed.

`overwrite` A logical indicating whether an existing risk of bias table in a meta-analysis object should be overwritten.

`warn` A logical indicating whether warnings should be printed.

`x` An object of class `rob`.

`details` A logical indicating whether to print details on categories and colours.

`...` Additional printing arguments.

Details

This function can be used to define a risk of bias (RoB) assessment for a meta-analysis which can be shown in a forest plot (`forest.meta`), summary weighted barplot (`barplot.rob`) or traffic light plot (`traffic_light`). It is also possible to extract the risk of bias assessment from a meta-analysis with RoB information.

The risk of bias table contains

- study labels;
- variables for individual RoB domains (with variable names A, B, . . .);
- an overall RoB assessment if argument `overall` is provided;
- weights for individual studies used in summary weighted barplots.

Note, an overall RoB assessment is mandatory to create a summary weighted barplot or a traffic light plot.

The RoB table is directly returned if argument `data` is a data frame or argument `item1` is a meta-analysis with risk of bias assessment. The RoB table is added as a new list element 'rob' to a meta-analysis object if argument `data` is a meta-analysis.

The user must either specify the categories and (optionally) domains of the RoB tool (using the eponymous arguments) or one of the following RoB tools.

Argument	**Risk of bias tool**
`tool = "RoB1"`	RoB 1 tool for randomized studies (Higgins et al., 2011)
`tool = "RoB2"`	RoB 2 tool for randomized studies (Sterne et al., 2019)
`tool = "RoB2-cluster"`	RoB 2 tool for cluster-randomized trials
`tool = "RoB2-crossover"`	RoB 2 tool for crossover trials
`tool = "ROBINS-I"`	Risk Of Bias In Non-randomized Studies - of Interventions

	(Sterne et al., 2016)
`tool = "ROBINS-E"`	Risk Of Bias In Non-randomized Studies - of Exposures (ROBINS-E Development Group, 2023)

These RoB tools are described on the website `https://www.riskofbias.info/`.

Risk of bias domains:

By default, i.e., if argument `domains` is not provided by the user, the following names are used for RoB domains.

- RoB 1 tool for randomized studies (RoB1):
  1. Random sequence generation (selection bias)
  2. Allocation concealment (selection bias)
  3. Blinding of participants and personnel (performance bias)
  4. Blinding of outcome assessment (detection bias)
  5. Incomplete outcome data (attrition bias)
  6. Selective reporting (reporting bias)
  7. Other bias
- RoB 2 tool for randomized studies (RoB2):
  1. Bias arising from the randomization process
  2. Bias due to deviations from intended intervention"
  3. Bias due to missing outcome data
  4. Bias in measurement of the outcome
  5. Bias in selection of the reported result
- RoB 2 tool for cluster-randomized trials (RoB2-cluster):
  1. Bias arising from the randomization process
  2. Bias arising from the identification or recruitment of participants into clusters
  3. Bias due to deviations from intended intervention
  4. Bias due to missing outcome data
  5. Bias in measurement of the outcome
  6. Bias in selection of the reported result
- RoB 2 tool for crossover trials (RoB2-crossover)
  1. Bias arising from the randomization process
  2. Bias arising from period and carryover effects
  3. Bias due to deviations from intended intervention
  4. Bias due to missing outcome data
  5. Bias in measurement of the outcome
  6. Bias in selection of the reported result
- Risk Of Bias In Non-randomized Studies - of Intervention (ROBINS-I):
  1. Risk of bias due to confounding
  2. Risk of bias in selection of participants into the study
  3. Risk of bias in classification of interventions
  4. Risk of bias due to deviations from intented interventions

5. Risk of bias due to missing outcome data
6. Risk of bias in measurement of the outcome
7. Risk of bias in the selection of the reported results

- Risk Of Bias In Non-randomized Studies - of Exposures (ROBINS-E):
  1. Risk of bias due to confounding
  2. Risk of bias arising from measurement of the exposure into the study (or into the analysis)
  3. Risk of bias due to post-exposure interventions
  4. Risk of bias due to deviations from intented interventions
  5. Risk of bias due to missing outcome data
  6. Risk of bias in measurement of the outcome
  7. Risk of bias in the selection of the reported results
- User-defined RoB assessment:
  1. First item
  2. Second item
  3. ...

It is possible to define additional bias domains for the available RoB tools. In this case, only the names for new RoB domains have to be provided in argument `domains`. If argument `domains` is not used to specify new domains, the names "Additional item 1" etc. will be used. It is also possible to modify the pre-defined domain names using argument `domains`.

The maximum number of bias domains / items is ten (see arguments `item1`, ..., `item10`).

Risk of bias categories, colours and symbols:

By default, the following settings are used.

RoB 1 tool:

Argument	**Values**
`categories`	"Low risk of bias", "Unclear risk of bias", "High risk of bias"
`col`	"green", "yellow", "red"
`symbols`	"+", "?", "-"

RoB 2 tools:

Argument	**Values**
`categories`	"Low risk of bias", "Some concerns", "High risk of bias"
`col`	"green", "yellow", "red"
`symbols`	"+", "?", "-"

ROBINS tools:

Argument	**Values**
`categories`	"Low risk", "Some concerns", "High risk", "Very high risk", "NI"
`col`	"green", "yellow", "red", "darkred", "darkgrey"
`symbols`	none

User-defined RoB tools:

Argument	Values
categories	Must be specified by the user
col	1, 2, ...
symbols	none

If colours (col) and symbols (symbols) are provided, they must be of the same length as the number of categories.

Value

A data frame with study labels and risk of bias items and additional class "rob".

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Higgins JPT, Altman DG, Gøtzsche PC, Jüni P, Moher D, Oxman AD et al. (2011): The Cochrane Collaboration's tool for assessing risk of bias in randomised trials. *British Medical Journal*, **343**: d5928

ROBINS-E Development Group (Higgins J, Morgan R, Rooney A et al.) (2023): Risk Of Bias In Non-randomized Studies - of Exposure (ROBINS-E) Available from: https://www.riskofbias.info/welcome/robins-e-tool.

Sterne JA, Hernán MA, Reeves BC, Savović J, Berkman ND, Viswanathan M, et al. (2016): ROBINS-I: a tool for assessing risk of bias in non-randomised studies of interventions. *British Medical Journal*, **355**: i4919

Sterne JAC, Savović J, Page MJ, Elbers RG, Blencowe NS, Boutron I, et al. (2019): RoB 2: a revised tool for assessing risk of bias in randomised trials. *British Medical Journal*, **366**: l4898.

See Also

forest.meta, barplot.rob, traffic_light

Examples

```
# Use RevMan 5 settings
oldset <- settings.meta("RevMan5", quietly = FALSE)

data(caffeine)

m1 <- metabin(h.caf, n.caf, h.decaf, n.decaf, sm = "OR",
  data = caffeine, studlab = paste(study, year))

# Add risk of bias assessment to meta-analysis
m2 <- rob(D1, D2, D3, D4, D5, overall = rob, data = m1, tool = "rob2")

# Print risk of bias assessment
rob(m2)
```

```
# Forest plot with risk of bias assessment
forest(m2)

# Use previous settings
settings.meta(oldset)
```

settings.meta *Print and change default meta-analysis settings in R package **meta***

Description

Print and change default settings to conduct and print or plot meta-analyses in R package **meta**. The following general settings are available: *Review Manager 5*, *Journal of the American Medical Association*.

Usage

```
settings.meta(..., quietly = TRUE)
```

Arguments

`...`	Arguments to change default settings.
`quietly`	A logical indicating whether information on settings should be printed.

Details

This function can be used to define defaults for several arguments (i.e., assignments using `gs`) of the following R functions: `metabin`, `metacont`, `metacor`, `metacr`, `metagen`, `metainc`, `metaprop`, `metarate`

Furthermore, some of these settings are considered to print meta-analysis results and to produce forest plots.

The function can be used to either change individual settings (see Examples) or use one of the following general settings:

- `settings.meta("RevMan5")`
- `settings.meta("BMJ")`
- `settings.meta("JAMA")`
- `settings.meta("IQWiG5")`
- `settings.meta("IQWiG6")`
- `settings.meta("geneexpr")`
- `settings.meta("meta4")`
- `settings.meta("meta7")`

The first command can be used to reproduce meta-analyses from Cochrane reviews conducted with *Review Manager 5* (RevMan 5) and specifies to use a RevMan 5 layout in forest plots.

The second command can be used to generate forest plots in BMJ layout.

The third command can be used to generate forest plots following instructions for authors of the *Journal of the American Medical Association*. Study labels according to JAMA guidelines can be generated using `labels.meta`.

The next commands implement the recommendations of the Institute for Quality and Efficiency in Health Care, Germany (IQWiG) accordinging to General Methods 5 and 6, respectively (https://www.iqwig.de/en/about-us/methods/methods-paper/).

The setting `"geneexpr"` can be used to print p-values in scientific notation and to suppress the calculation of confidence intervals for the between-study variance.

The last settings use the default settings of R package **meta**, version 4 and 7.0-0, respectively, or below.

RevMan 5 settings, in detail:

Argument	Value	Comment
`method.random.ci`	"classic"	only available method in RevMan 5
`method.tau`	"DL"	only available method in RevMan 5
`method.I2`	"Q"	only available method in RevMan 5
`tau.common`	FALSE	common between-study variance in subgroups
`MH.exact`	FALSE	exact Mantel-Haenszel method
`RR.Cochrane`	TRUE	calculation of risk ratios
`Q.Cochrane`	TRUE	calculation of heterogeneity statistic
`exact.smd`	FALSE	exact formulae for Hedges' g and Cohen's d
`layout`	"RevMan5"	layout for forest plots
`prediction`	FALSE	no prediction interval
`test.overall`	TRUE	print information on test of overall effect
`test.subgroup`	TRUE	print information on test for subgroup differences
`test.effect.subgroup`	TRUE	print information on test for effect in subgroups
`forest.I2`	TRUE	show heterogeneity statistic I2 in forest plots
`forest.tau2`	TRUE	show between-study heterogeneity variance in forest plots
`forest.tau`	FALSE	do not show between-study heterogeneity standard deviation in forest plots
`forest.Q`	TRUE	show heterogeneity statistic Q in forest plots
`forest.pval.Q`	TRUE	show p-value of test for heterogeneity in forest plots
`forest.Rb`	FALSE	do not show heterogeneity statistic Rb in forest plots
`digits.tau2`	3	number of digits for tau-squared
`digits.tau`	4	number of digits for square root of tau-squared
`digits.I2`	0	number of digits for I-squared measure
`CIbracket`,	"["	
`CIseparator`	", "	print confidence intervals as "`[., .]`"
`header.line`,	TRUE	print header line

BMJ settings:

Argument	**Value**	**Comment**
`layout`	"BMJ"	layout for forest plots
`test.overall`	TRUE	print information on test of overall effect
`test.subgroup`	FALSE	print information on test for subgroup differences
`test.effect.subgroup`	FALSE	print information on test for effect in subgroups
`forest.I2`	TRUE	show heterogeneity statistic I2 in forest plots
`forest.tau2`	TRUE	show between-study heterogeneity variance in forest plots
`forest.tau`	FALSE	do not show between-study heterogeneity standard deviation in forest plots
`forest.Q`	TRUE	show heterogeneity statistic Q in forest plots
`forest.pval.Q`	TRUE	show p-value of test for heterogeneity in forest plots
`forest.Rb`	FALSE	do not show heterogeneity statistic Rb in forest plots
`digits.I2`	0	number of digits for I-squared measure
`digits.pval`	2	number of digits for p-values
`CIbracket,`	"("	
`CIseparator`	" to "	print confidence intervals as "`(. to .)`"
`hetlab,`		"Test for heterogeneity: "
`header.line,`	TRUE	print header line

JAMA settings:

Argument	**Value**	**Comment**
`layout`	"JAMA"	layout for forest plots
`test.overall`	TRUE	print information on test of overall effect
`test.subgroup`	FALSE	print information on test for subgroup differences
`test.effect.subgroup`	FALSE	print information on test for effect in subgroups
`forest.I2`	TRUE	show heterogeneity statistic I2 in forest plots
`forest.tau2`	FALSE	do not show between-study heterogeneity variance in forest plots
`forest.tau`	FALSE	do not show between-study heterogeneity standard deviation in forest plots
`forest.Q`	TRUE	show heterogeneity statistic Q in forest plots
`forest.pval.Q`	TRUE	show p-value of test for heterogeneity in forest plots
`forest.Rb`	FALSE	do not show heterogeneity statistic Rb in forest plots
`digits.I2`	0	number of digits for I-squared measure
`digits.pval`	3	number of digits for p-values
`CIbracket,`	"("	
`CIseparator`	"-"	print confidence intervals as "`(.-.)`"
`zero.pval,`	FALSE	print p-values with leading zero
`JAMA.pval,`	TRUE	round p-values to three digits (for $0.001 < p \leq 0.01$) or two digits ($p > 0.01$)
`header.line,`	TRUE	print header line

IQWiG, General Methods 5 settings:

Argument	**Value**	**Comment**
`method.random.ci`	"HK"	Hartung-Knapp method

`prediction`	TRUE	Prediction interval

IQWiG, General Methods 6 settings:

Argument	**Value**	**Comment**
`method.random.ci`	"HK"	Hartung-Knapp method
`adhoc.hakn.ci`	"IQWiG6"	*ad hoc* variance correction
`method.tau`	"PM"	Paule-Mandel estimator for between-study variance
`prediction`	TRUE	Prediction interval

Settings for gene expression data:

Argument	**Value**	**Comment**
`scientific.pval`	TRUE	Scientific notation for p-values
`method.tau.ci`	FALSE	no confidence interval for between-study heterogeneity variance

Settings for **meta**, version 4 or below:

Argument	**Value**	**Comment**
`method.tau`	"DL"	DerSimonian-Laird estimator
`method.I2`	"Q"	Use Q to calculate I-squared
`method.predict`	"HTS"	prediction interval with *k-2* degrees of freedom
`exact.smd`	FALSE	Use exact formula for standardised mean difference (White and Thomas, 2005)
`text.common`	"Fixed effect model"	
`text.w.common`	"fixed"	
`warn.deprecated`	FALSE	Do not print warnings for deprecated arguments

Settings for **meta**, version 7.0-0 or below:

Argument	**Value**	**Comment**
`method.tau`	"REML"	REML estimator
`method.I2`	"Q"	Use Q to calculate I-squared
`method.predict`	"HTS"	prediction interval with *k-2* degrees of freedom
`exact.smd`	TRUE	Use exact formula for standardised mean difference (White and Thomas, 2005)
`text.common`	"Common effect model"	
`text.w.common`	"common"	
`warn.deprecated`	FALSE	Do not print warnings for deprecated arguments

A list of all arguments with current settings is printed using the command `settings.meta()`.

In order to reset all settings of R package **meta** the command `settings.meta(reset = TRUE)` can be used.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

White IR, Thomas J (2005): Standardized mean differences in individually-randomized and cluster-randomized trials, with applications to meta-analysis. *Clinical Trials*, **2**, 141–51

See Also

gs, forest.meta, print.meta, labels.meta

Examples

```
# Get listing of current settings
#
settings.meta()

# Meta-analyses using default settings
#
metabin(10, 20, 15, 20)
metaprop(4, 20)
metabin(10, 20, 15, 20, sm = "RD")
metaprop(4, 20, sm = "PLN")

# Change summary measure for R functions metabin and metaprop
# and store old settings
#
oldset <- settings.meta(smbin = "RD", smprop = "PLN")
#
metabin(10, 20, 15, 20)
metaprop(4, 20)

# Use old settings
#
settings.meta(oldset)

# Change level used to calculate confidence intervals
# (99%-CI for studies, 99.9%-CI for pooled effects)
#
metagen(1:3, 2:4 / 10, sm = "MD")
settings.meta(level = 0.99, level.ma = 0.999)
metagen(1:3, 2:4 / 10, sm = "MD")

# Always print a prediction interval
#
settings.meta(prediction = TRUE)
```

```
metagen(1:3, 2:4 / 10, sm = "MD")
metagen(4:6, 4:2 / 10, sm = "MD")

# Try to set unknown argument results in a warning
#
try(settings.meta(unknownarg = TRUE))

# Reset to default settings of R package meta
#
settings.meta("reset")
metabin(10, 20, 15, 20)
metaprop(4, 20)
metagen(1:3, 2:4 / 10, sm = "MD")

# Do not back transform results (e.g. print log odds ratios instead
# of odds ratios, print transformed correlations / proportions
# instead of correlations / proportions)
#
settings.meta(backtransf = FALSE)
metabin(10, 20, 15, 20)
metaprop(4, 20)
metacor(c(0.85, 0.7, 0.95), c(20, 40, 10))

# Forest plot using RevMan 5 style
#
settings.meta("RevMan5")
forest(metagen(1:3, 2:4 / 10, sm = "MD", common = FALSE),
  label.left = "Favours A", label.right = "Favours B",
  colgap.studlab = "2cm", colgap.forest.left = "0.2cm")

# Forest plot using JAMA style
#
settings.meta("JAMA")
forest(metagen(1:3, 2:4 / 10, sm = "MD", common = FALSE),
  label.left = "Favours A", label.right = "Favours B",
  colgap.studlab = "2cm", colgap.forest.left = "0.2cm")

# Use slightly different layout for confidence intervals
# (especially useful if upper confidence limit can be negative)
#
settings.meta(CIseparator = " - ")
forest(metagen(-(1:3), 2:4 / 10, sm = "MD", common = FALSE),
  label.left = "Favours A", label.right = "Favours B",
  colgap.studlab = "2cm", colgap.forest.left = "0.2cm")

# Use old settings
#
settings.meta(oldset)
```

smd2or *Conversion from standardised mean difference to log odds ratio*

Description

Conversion from standardised mean difference to log odds ratio using method by Hasselblad & Hedges (1995) or Cox (1970).

Usage

```
smd2or(
  smd,
  se.smd,
  studlab,
  data = NULL,
  subset = NULL,
  exclude = NULL,
  method = "HH",
  backtransf = gs("backtransf"),
  ...
)
```

Arguments

`smd` Standardised mean difference(s) (SMD) or meta-analysis object.

`se.smd` Standard error(s) of SMD (ignored if argument `smd` is a meta-analysis object).

`studlab` An optional vector with study labels (ignored if argument `smd` is a meta-analysis object).

`data` An optional data frame containing the study information (ignored if argument `smd` is a meta-analysis object).

`subset` An optional vector specifying a subset of studies to be used (ignored if argument `smd` is a meta-analysis object).

`exclude` An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots (ignored if argument `smd` is a meta-analysis object).

`method` A character string indicating which method is used to convert SMDs to log odds ratios. Either `"HH"` or `"CS"`, can be abbreviated.

`backtransf` A logical indicating whether odds ratios (if TRUE) or log odds ratios (if FALSE) should be shown in printouts and plots.

`...` Additional arguments passed on to `metagen` (ignored if argument `smd` is a meta-analysis object).

Details

This function implements the following methods for the conversion from standardised mean difference to log odds ratio:

- Hasselblad & Hedges (1995) assuming logistic distributions (`method == "HH"`)
- Cox (1970) and Cox & Snell (1989) assuming normal distributions (`method == "CS"`)

Internally, metagen is used to conduct a meta-analysis with the odds ratio as summary measure.

Argument smd can be either a vector of standardised mean differences or a meta-analysis object created with metacont or metagen and the standardised mean difference as summary measure.

Argument se.smd is mandatory if argument smd is a vector and ignored otherwise. Additional arguments in ... are only passed on to metagen if argument smd is a vector.

Value

An object of class c("metagen", "meta") with corresponding generic functions (see meta-object).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Borenstein M, Hedges LV, Higgins JPT, Rothstein HR (2009): *Introduction to Meta-Analysis*. Chichester: Wiley

Cox DR (1970): *Analysis of Binary Data*. London: Chapman and Hall / CRC

Cox DR, Snell EJ (1989): *Analysis of Binary Data* (2nd edition). London: Chapman and Hall / CRC

Hasselblad V, Hedges LV (1995): Meta-analysis of screening and diagnostic tests. *Psychological Bulletin*, **117**, 167–78

See Also

or2smd, metacont, metagen, metabin

Examples

```
# Example from Borenstein et al. (2009), Chapter 7
#
mb <- smd2or(0.5, sqrt(0.0205), backtransf = FALSE)
# TE = log odds ratio; seTE = standard error of log odds ratio
data.frame(lnOR = round(mb$TE, 4), varlnOR = round(mb$seTE^2, 4))

# Use dataset from Fleiss (1993)
#
data(Fleiss1993cont)
m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, sm = "SMD",
               studlab = paste(study, year))
smd2or(m1)
```

`smoking` *Smoking example*

Description

Meta-analyses on the effect of smoking on mortality risk.

Format

A data frame with the following columns:

study	study label
participants	total number of participants
d.smokers	number of deaths in smokers' group
py.smokers	person years at risk in smokers' group
d.nonsmokers	number of deaths in non-smokers' group
py.nonsmokers	person years at risk in non-smokers' group

Details

Data have been reconstructed based on the famous Smoking and Health Report to the Surgeon General (Bayne-Jones S et al., 1964). Data sets can be used to evaluate the risk of smoking on overall mortality (dataset `smoking`) and lung-cancer deaths (dataset `lungcancer`), respectively.

The person time is attributed such that the rate ratios are equal to the reported mortality ratios implicitly assuming that the data have arisen from a homogeneous age group; more detailed information by age is not available from the report. Note, the group of "non-smokers" actually consists of all participants except those who are smokers of cigarettes only. Information on real non-smokers is not available from the published Smoking and Health Report.

Source

Bayne-Jones S et al. (1964): Smoking and Health: Report of the Advisory Committee to the Surgeon General of the United States. U-23 Department of Health, Education, and Welfare. Public Health Service Publication No. 1103.

See Also

`metainc`

Examples

```
data(smoking)

m1 <- metainc(d.smokers, py.smokers, d.nonsmokers, py.nonsmokers,
  data = smoking, studlab = study)
print(m1, digits = 2)

data(lungcancer)
```

```
m2 <- metainc(d.smokers, py.smokers, d.nonsmokers, py.nonsmokers,
  data = lungcancer, studlab = study)
print(m2, digits = 2)
```

---

subset.longarm *Return subset of longarm object*

Description

The `subset` method returns a subset of a longarm object.

Usage

```
## S3 method for class 'longarm'
subset(x, subset, ...)
```

Arguments

`x`	An object of class `longarm`.
`subset`	A logical expression indicating elements or rows to keep: missing values are taken as false.
`...`	Additional arguments.

Value

A `longarm` object is returned.

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

See Also

`longarm`

Examples

```
# Transform data from wide arm-based format to contrast-based format
pw1 <- pairwise(list(Treatment1, Treatment2, Treatment3),
  n = list(n1, n2, n3),
  mean = list(y1, y2, y3), sd = list(sd1, sd2, sd3),
  data = dat.franchini2012, studlab = Study)

# Transform data from contrast-based to long arm-based format
# and only keep the main variables
la1 <- longarm(pw1, append = FALSE)
head(la1)
```

```
# Subset without Lieberman studies
subset(la1, grepl("Lieberman", studlab))
```

subset.pairwise *Return subset of pairwise object*

Description

The subset method returns a subset of a pairwise object.

Usage

```
## S3 method for class 'pairwise'
subset(x, subset, ...)
```

Arguments

x	An object of class pairwise.
subset	A logical expression indicating rows to keep; missing values are taken as false.
...	Additional arguments.

Value

A pairwise object is returned.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

pairwise, dat.franchini2012

Examples

```
# Transform data from arm-based format to contrast-based format
pw1 <- pairwise(list(Treatment1, Treatment2, Treatment3),
  n = list(n1, n2, n3),
  mean = list(y1, y2, y3), sd = list(sd1, sd2, sd3),
  data = dat.franchini2012, studlab = Study)
# Subset without Lieberman studies
subset(pw1, !grepl("Lieberman", studlab))[, 1:5]
```

summary.meta *Summary of meta-analysis results*

Description

Summary method for objects of class `meta`.

Usage

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

Arguments

`object`	An object of class `meta`.
`...`	Additional arguments (ignored).

Details

Summary method for objects of class `meta`.

Value

An object of classes `summary.meta` and `meta` (see `meta-object`.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Cooper H & Hedges LV (1994): *The Handbook of Research Synthesis*. Newbury Park, CA: Russell Sage Foundation

Crippa A, Khudyakov P, Wang M, Orsini N, Spiegelman D (2016): A new measure of between-studies heterogeneity in meta-analysis. *Statistics in Medicine*, **35**, 3661–75

Higgins JPT & Thompson SG (2002): Quantifying heterogeneity in a meta-analysis. *Statistics in Medicine*, **21**, 1539–58

See Also

`print.summary.meta`, `metabin`, `metacont`, `metagen`

Examples

```
data(Fleiss1993cont)
m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, studlab = paste(study, year), sm = "SMD")
summary(m1)

summary(update(m1, subgroup = c(1, 2, 1, 1, 2), subgroup.name = "group"))
forest(update(m1, subgroup = c(1, 2, 1, 1, 2), subgroup.name = "group"))

## Not run:
# Use unicode characters to print tau^2, tau, and I^2
print(summary(m1),
  text.tau2 = "\u03c4\u00b2", text.tau = "\u03c4", text.I2 = "I\u00b2")

## End(Not run)
```

---

summary.rm5 *Cochrane review: detailed summary of meta-analyses*

Description

Calculate and print a detailed summary of all meta-analyses in a Cochrane review.

Usage

```
## S3 method for class 'rm5'
summary(object, comp.no, outcome.no, ...)

## S3 method for class 'cdir'
summary(object, comp.no, outcome.no, ...)

## S3 method for class 'summary.rm5'
print(x, ...)

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

Arguments

`object`	An object of class `rm5` or `cdir`.
`comp.no`	Comparison number.
`outcome.no`	Outcome number.
`...`	Additional arguments (passed on to `metacr`).
`x`	An object of class `summary.rm5` or `summary.cdir`.

Details

This function can be used to rerun all or selected meta-analyses of a Cochrane Review of interventions (Higgins et al., 2023).

The R function metacr is called internally.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Higgins JPT, Thomas J, Chandler J, Cumpston M, Li T, Page MJ, Welch VA (editors) (2023): *Cochrane Handbook for Systematic Reviews of Interventions Version 6.4 (updated August 2023)*. Available from https://training.cochrane.org/handbook/

See Also

summary.meta, metacr, read.rm5, read.cdir, metabias.rm5, metabias.cdir

Examples

```
# Locate export data file "Fleiss1993_CR.csv"
# in sub-directory of package "meta"
#
filename <- system.file("extdata", "Fleiss1993_CR.csv", package = "meta")
Fleiss1993_CR <- read.rm5(filename)

# Print summary results for all meta-analysis
#
summary(Fleiss1993_CR)

# Print summary results only for second outcome of first comparison
#
summary(Fleiss1993_CR, comp.no = 1, outcome.no = 2)
```

---

traffic_light *Produce traffic light plot of risk of bias assessment*

Description

Produce traffic light plot of risk of bias assessment

Usage

```
traffic_light(object, colour = "cochrane", psize = 15, quiet = FALSE)
```

Arguments

object	An object of class rob.
colour	Specify colour scheme for the traffic light plot; see rob_summary.
psize	Size of the traffic lights.
quiet	A logical to suppress the display of the traffic light plot.

Details

This is a wrapper function for rob_traffic_light of R package **robvis** to produce a traffic light plot of risk of bias assessment.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

rob, barplot.rob, rob_traffic_light

Examples

```
# Use RevMan 5 settings
oldset <- settings.meta("RevMan5")

data(caffeine)

m1 <- metabin(h.caf, n.caf, h.decaf, n.decaf, sm = "OR",
  data = caffeine, studlab = paste(study, year))

# Add risk of bias assessment to meta-analysis
m2 <- rob(D1, D2, D3, D4, D5, overall = rob, data = m1, tool = "rob2")

# Print risk of bias assessment
rob(m2)

## Not run:
# Traffic light plot (R package 'robvis' must be available)
if (requireNamespace("robvis", quietly = TRUE))
 traffic_light(rob(m2))

## End(Not run)

# Use previous settings
settings.meta(oldset)
```

trimfill.meta *Trim-and-fill method to adjust for bias in meta-analysis*

Description

Trim-and-fill method for estimating and adjusting for the number and outcomes of missing studies in a meta-analysis.

Usage

```
## S3 method for class 'meta'
trimfill(
  x,
  left = NULL,
  ma.common = TRUE,
  type = "L",
  n.iter.max = 50,
  common = FALSE,
  random = TRUE,
  prediction = x$prediction,
  backtransf = x$backtransf,
  pscale = x$pscale,
  irscale = x$irscale,
  irunit = x$irunit,
  silent = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

## Default S3 method:
trimfill(
  x,
  seTE,
  left = NULL,
  ma.common = TRUE,
  type = "L",
  n.iter.max = 50,
  sm = "",
  studlab = NULL,
  level = 0.95,
  level.ma = level,
  common = FALSE,
  random = TRUE,
  method.random.ci = gs("method.random.ci"),
  adhoc.hakn.ci = gs("adhoc.hakn.ci"),
  method.tau = gs("method.tau"),
  method.tau.ci = if (method.tau == "DL") "J" else "QP",
  level.hetstat = gs("level.hetstat"),
  prediction = FALSE,
  level.predict = level,
  method.predict = gs("method.predict"),
  adhoc.hakn.pi = gs("adhoc.hakn.pi"),
  seed.predict = NULL,
  backtransf = TRUE,
  pscale = 1,
  irscale = 1,
  irunit = "person-years",
  silent = TRUE,
  ...
)
```

Arguments

`x` An object of class `meta`, or estimated treatment effect in individual studies.

`left` A logical indicating whether studies are supposed to be missing on the left or right side of the funnel plot. If NULL, the linear regression test for funnel plot symmetry (i.e., function `metabias(..., method="Egger")`) is used to determine whether studies are missing on the left or right side.

`ma.common` A logical indicating whether a common effect or random effects model is used to estimate the number of missing studies.

`type` A character indicating which method is used to estimate the number of missing studies. Either `"L"` or `"R"`.

`n.iter.max` Maximum number of iterations to estimate number of missing studies.

`common` A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`prediction` A logical indicating whether a prediction interval should be printed.

`backtransf` A logical indicating whether results should be back transformed in printouts and plots. If `backtransf=TRUE`, results for `sm="OR"` are printed as odds ratios rather than log odds ratios and results for `sm="ZCOR"` are printed as correlations rather than Fisher's z transformed correlations, for example.

`pscale` A numeric giving scaling factor for printing of single event probabilities or risk differences, i.e. if argument `sm` is equal to `"PLOGIT"`, `"PLN"`, `"PRAW"`, `"PAS"`, `"PFT"`, or `"RD"`.

`irscale` A numeric defining a scaling factor for printing of single incidence rates or incidence rate differences, i.e. if argument `sm` is equal to `"IR"`, `"IRLN"`, `"IRS"`, `"IRFT"`, or `"IRD"`.

`irunit` A character specifying the time unit used to calculate rates, e.g. person-years.

`silent` A logical indicating whether basic information on iterations shown.

`warn.deprecated`
A logical indicating whether warnings should be printed if deprecated arguments are used.

`...` Additional arguments (to catch deprecated arguments).

`seTE` Standard error of estimated treatment effect.

`sm` An optional character string indicating underlying summary measure, e.g., `"RD"`, `"RR"`, `"OR"`, `"ASD"`, `"HR"`, `"MD"`, `"SMD"`, or `"ROM"`; ignored if `x` is of class `meta`.

`studlab` An optional vector with study labels; ignored if `x` is of class `meta`.

`level` The level used to calculate confidence intervals for individual studies. If existing, `x$level` is used as value for `level`; otherwise 0.95 is used.

`level.ma` The level used to calculate confidence interval for the pooled estimate. If existing, `x$level.ma` is used as value for `level.ma`; otherwise 0.95 is used.

`method.random.ci`
A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

`adhoc.hakn.ci` A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see `meta-package`).

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`level.predict` The level used to calculate prediction interval for a new study.

`method.predict` A character string indicating which method is used to calculate a prediction interval (see `meta-package`).

`adhoc.hakn.pi` A character string indicating whether an *ad hoc* variance correction should be applied for the prediction interval (see `meta-package`).

`seed.predict` A numeric value used as seed to calculate bootstrap prediction interval (see `meta-package`).

Details

The trim-and-fill method (Duval, Tweedie 2000a, 2000b) can be used for estimating and adjusting for the number and outcomes of missing studies in a meta-analysis. The method relies on scrutiny of one side of a funnel plot for asymmetry assumed due to publication bias.

Three different methods have been proposed originally to estimate the number of missing studies. Two of these methods (L- and R-estimator) have been shown to perform better in simulations, and are available in this R function (argument `type`).

A common effect or random effects model can be used to estimate the number of missing studies (argument `ma.common`). Furthermore, a common effect and/or random effects model can be used to summaries study results (arguments `common` and `random`). Simulation results (Peters et al. 2007) indicate that the common-random model, i.e. using a common effect model to estimate the number of missing studies and a random effects model to summaries results, (i) performs better than the common-common model, and (ii) performs no worse than and marginally better in certain situations than the random-random model. Accordingly, the common-random model is the default.

An empirical comparison of the trim-and-fill method and the Copas selection model (Schwarzer et al. 2010) indicates that the trim-and-fill method leads to excessively conservative inference in practice. The Copas selection model is available in R package **metasens**.

The function metagen is called internally.

Value

An object of class c("trimfill", "metagen", "meta") with corresponding generic functions (see meta-object).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Duval S & Tweedie R (2000a): A nonparametric "Trim and Fill" method of accounting for publication bias in meta-analysis. *Journal of the American Statistical Association*, **95**, 89–98

Duval S & Tweedie R (2000b): Trim and Fill: A simple funnel-plot-based method of testing and adjusting for publication bias in meta-analysis. *Biometrics*, **56**, 455–63

Peters JL, Sutton AJ, Jones DR, Abrams KR, Rushton L (2007): Performance of the trim and fill method in the presence of publication bias and between-study heterogeneity. *Statisics in Medicine*, **10**, 4544–62

Schwarzer G, Carpenter J, Rücker G (2010): Empirical evaluation suggests Copas selection model preferable to trim-and-fill method for selection bias in meta-analysis *Journal of Clinical Epidemiology*, **63**, 282–8

See Also

metagen, metabias, funnel

Examples

```
data(Fleiss1993bin)
m1 <- metabin(d.asp, n.asp, d.plac, n.plac, data = Fleiss1993bin, sm = "OR")
tf1 <- trimfill(m1)
tf1
funnel(tf1)
funnel(tf1, pch = ifelse(tf1$trimfill, 1, 16), level = 0.9, random = FALSE)
#
# Use log odds ratios on x-axis
#
funnel(tf1, backtransf = FALSE)
funnel(tf1, pch = ifelse(tf1$trimfill, 1, 16), level = 0.9, random = FALSE,
  backtransf = FALSE)

trimfill(m1$TE, m1$seTE, sm = m1$sm)
```

trimfill.rm5 *Cochrane review: trim-and-fill method*

Description

Conduct trim-and-fill analysis for all meta-analyses in a Cochrane review.

Usage

```
## S3 method for class 'rm5'
trimfill(x, comp.no, outcome.no, ...)

## S3 method for class 'cdir'
trimfill(x, comp.no, outcome.no, ...)

## S3 method for class 'trimfill.rm5'
print(x, ...)

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

Arguments

`x`	An object of class `rm5`, `trimfill.rm5`, `cdir` or `trimfill.cdir`.
`comp.no`	Comparison number.
`outcome.no`	Outcome number.
`...`	Additional arguments (passed on to `metacr`).

Details

This function can be used to conduct a trim-and-fill analysis for all or selected meta-analyses in a Cochrane review of intervention studies (Higgins et al., 2023).

The R function `metacr` is called internally.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

Higgins JPT, Thomas J, Chandler J, Cumpston M, Li T, Page MJ, Welch VA (editors) (2023): *Cochrane Handbook for Systematic Reviews of Interventions Version 6.4 (updated August 2023)*. Available from https://training.cochrane.org/handbook/

See Also

`summary.meta`, `metacr`, `read.rm5`, `read.cdir`, `metabias.rm5`, `metabias.cdir`

Examples

```
# Locate export data file "Fleiss1993_CR.csv"
# in sub-directory of package "meta"
#
filename <- system.file("extdata", "Fleiss1993_CR.csv", package = "meta")
Fleiss1993_CR <- read.rm5(filename)

# Conduct trim-and-fill analysis
#
trimfill(Fleiss1993_CR)

# Conduct trim-and-fill analysis only for second outcome of first
# comparison
#
trimfill(Fleiss1993_CR, comp.no = 1, outcome.no = 2)
```

update.meta *Update a meta-analysis object*

Description

Update an existing meta-analysis object.

Usage

```
## S3 method for class 'meta'
update(
  object,
  data = object$data,
  subset,
  studlab,
  exclude,
  cluster,
  rho = object$rho,
  cycles,
  method,
  sm = object$sm,
  incr = object$incr,
  method.incr = object$method.incr,
  allstudies = object$allstudies,
  incr.e,
  incr.c,
  incr.event,
  MH.exact = object$MH.exact,
  RR.Cochrane = object$RR.Cochrane,
  Q.Cochrane = object$Q.Cochrane,
  model.glmm = object$model.glmm,
  level = object$level,
  level.ma = object$level.ma,
  common = object$common,
  random = object$random,
  overall = object$overall,
  overall.hetstat = object$overall.hetstat,
  method.random.ci = object$method.random.ci,
  adhoc.hakn.ci = object$adhoc.hakn.ci,
  method.predict = object$method.predict,
  adhoc.hakn.pi = object$adhoc.hakn.pi,
  seed.predict = object$seed.predict,
  method.tau = object$method.tau,
  method.tau.ci = object$method.tau.ci,
  level.hetstat = object$level.hetstat,
  tau.preset = object$tau.preset,
  TE.tau = object$TE.tau,
  tau.common = object$tau.common,
  detail.tau = object$detail.tau,
  method.I2 = object$method.I2,
  prediction = object$prediction,
  level.predict = object$level.predict,
  null.effect = object$null.effect,
  method.bias = object$method.bias,
  backtransf = object$backtransf,
  func.backtransf = object$func.backtransf,
  args.backtransf = object$args.backtransf,
  pscale = object$pscale,
  irscale = object$irscale,
  irunit = object$irunit,
  text.common = object$text.common,
  text.random = object$text.random,
  text.predict = object$text.predict,
  text.w.common = object$text.w.common,
  text.w.random = object$text.w.random,
  title = object$title,
  complab = object$complab,
  outclab = object$outclab,
  label.e = object$label.e,
  label.c = object$label.c,
  label.left = object$label.left,
  label.right = object$label.right,
  col.label.left = object$col.label.left,
  col.label.right = object$col.label.right,
  n.e = object$n.e,
  n.c = object$n.c,
  method.mean = object$method.mean,
  method.sd = object$method.sd,
  approx.mean.e = object$approx.mean.e,
  approx.mean.c = object$approx.mean.c,
  approx.sd.e = object$approx.sd.e,
  approx.sd.c = object$approx.sd.c,
  approx.mean = object$approx.mean,
  approx.sd = object$approx.sd,
  approx.TE = object$approx.TE,
  approx.seTE = object$approx.seTE,
  pooledvar = object$pooledvar,
  method.smd = object$method.smd,
  sd.glass = object$sd.glass,
  exact.smd = object$exact.smd,
  method.ci = object$method.ci,
  subgroup,
  subgroup.name = object$subgroup.name,
  print.subgroup.name = object$print.subgroup.name,
  sep.subgroup = object$sep.subgroup,
  test.subgroup = object$test.subgroup,
  prediction.subgroup = object$prediction.subgroup,
  seed.predict.subgroup = object$seed.predict.subgroup,
  byvar,
  id,
  print.CMH = object$print.CMH,
  keepdata = TRUE,
  left = object$left,
  ma.common = object$ma.common,
  type = object$type,
  n.iter.max = object$n.iter.max,
  warn = FALSE,
  warn.deprecated = gs("warn.deprecated"),
  verbose = FALSE,
  control = object$control,
  ...
)
```

Arguments

`object` An object of class `meta`.

`data` Dataset.

`subset` Subset.

`studlab` Study label.

`exclude` An optional vector specifying studies to exclude from meta-analysis, however, to include in printouts and forest plots.

`cluster` An optional vector specifying which estimates come from the same cluster resulting in the use of a three-level meta-analysis model.

`rho` Assumed correlation of estimates within a cluster.

`cycles` A numeric vector with the number of cycles per patient / study in n-of-1 trials (see `metagen`).

`method` A character string indicating which method is to be used for pooling of studies (see `metabin`, `metainc`, `metaprop` and `metarate`).

`sm` A character string indicating which summary measure is used for pooling.

`incr` Information on increment added to cell frequencies of studies with zero cell counts (see `metabin`, `metainc`, `metaprop` and `metarate`).

`method.incr` A character string indicating which continuity correction method should be used (see `metabin`, `metainc`, `metaprop` and `metarate`).

`allstudies` A logical indicating if studies with zero or all events in both groups are to be included in the meta-analysis (applies only to `metabin` object with `sm` equal to `"RR"` or `"OR"`).

`incr.e` Continuity correction in experimental group (see `metabin` and `metainc`).

`incr.c` Continuity correction in control group (see `metabin` and `metainc`).

`incr.event` Continuity correction (see `metaprop` and `metarate`).

`MH.exact` A logical indicating if `incr` is not to be added to all cell frequencies for studies with a zero cell count to calculate the pooled estimate based on the Mantel-Haenszel method (applies only to `metabin` object).

`RR.Cochrane` A logical indicating which method to use as continuity correction for the risk ratio (see `metabin`).

`Q.Cochrane` A logical indicating which method to use to calculate the heterogeneity statistic Q (see `metabin`).

`model.glmm` A character string indicating which GLMM model should be used (see `metabin` and `metainc`).

`level` The level used to calculate confidence intervals for individual studies.

`level.ma` The level used to calculate confidence intervals for meta-analysis estimates.

`common` A logical indicating whether a common effect meta-analysis should be conducted.

`random` A logical indicating whether a random effects meta-analysis should be conducted.

`overall` A logical indicating whether overall summaries should be reported. This argument is useful in a meta-analysis with subgroups if overall results should not be reported.

`overall.hetstat`
A logical value indicating whether to print heterogeneity measures for overall treatment comparisons. This argument is useful in a meta-analysis with subgroups if heterogeneity statistics should only be printed on subgroup level.

`method.random.ci`
A character string indicating which method is used to calculate confidence interval and test statistic for random effects estimate (see `meta-package`).

`adhoc.hakn.ci` A character string indicating whether an *ad hoc* variance correction should be applied in the case of an arbitrarily small Hartung-Knapp variance estimate (see `meta-package`).

`method.predict` A character string indicating which method is used to calculate a prediction interval (see `meta-package`).

`adhoc.hakn.pi` A character string indicating whether an *ad hoc* variance correction should be applied for prediction interval (see `meta-package`).

`seed.predict` A numeric value used as seed to calculate bootstrap prediction interval (see `meta-package`).

`method.tau` A character string indicating which method is used to estimate the between-study variance τ^2 and its square root τ (see `meta-package`).

`method.tau.ci` A character string indicating which method is used to estimate the confidence interval of τ^2 and τ (see `meta-package`).

`level.hetstat` The level used to calculate confidence intervals for heterogeneity statistics.

`tau.preset` Prespecified value for the square root of the between-study variance τ^2.

`TE.tau` Overall treatment effect used to estimate the between-study variance tau-squared.

`tau.common` A logical indicating whether tau-squared should be the same across subgroups.

`detail.tau` Detail on between-study variance estimate.

`method.I2` A character string indicating which method is used to estimate the heterogeneity statistic I^2. Either `"Q"` or `"tau2"`, can be abbreviated (see `meta-package`).

`prediction` A logical indicating whether a prediction interval should be printed.

`level.predict` The level used to calculate prediction interval for a new study.

`null.effect` A numeric value specifying the effect under the null hypothesis.

`method.bias` A character string indicating which test for funnel plot asymmetry is to be used, can be abbreviated. See function `metabias`.

`backtransf` A logical indicating whether results should be back transformed in printouts and plots. If `backtransf = TRUE`, results for `sm = "OR"` are printed as odds ratios rather than log odds ratios and results for `sm = "ZCOR"` are printed as correlations rather than Fisher's z transformed correlations, for example.

`func.backtransf`
A function used to back-transform results (see `metagen`).

`args.backtransf`
An optional list to provide additional arguments to `func.backtransf` (see `metagen`).

`pscale` A numeric giving scaling factor for printing of single event probabilities or risk differences, i.e. if argument `sm` is equal to `"PLOGIT"`, `"PLN"`, `"PRAW"`, `"PAS"`, `"PFT"`, or `"RD"`.

`irscale` A numeric defining a scaling factor for printing of single incidence rates or incidence rate differences, i.e. if argument `sm` is equal to `"IR"`, `"IRLN"`, `"IRS"`, `"IRFT"`, or `"IRD"`.

`irunit` A character specifying the time unit used to calculate rates, e.g. person-years.

`text.common` A character string used in printouts and forest plot to label the pooled common effect estimate.

`text.random` A character string used in printouts and forest plot to label the pooled random effects estimate.

`text.predict` A character string used in printouts and forest plot to label the prediction interval.

`text.w.common` A character string used to label weights of common effect model.

`text.w.random`	A character string used to label weights of random effects model.
`title`	Title of meta-analysis / systematic review.
`complab`	Comparison label.
`outclab`	Outcome label.
`label.e`	Label for experimental group.
`label.c`	Label for control group.
`label.left`	Graph label on left side of null effect in forest plot.
`label.right`	Graph label on right side of null effect in forest plot.
`col.label.left`	The colour of the graph label on the left side of the null effect.
`col.label.right`	The colour of the graph label on the right side of the null effect.
`n.e`	Number of observations in experimental group (only for `metagen` or `metainc` object).
`n.c`	Number of observations in control group (only for `metagen` or `metainc` object).
`method.mean`	A character string indicating which method to use to approximate the mean from the median and other statistics (see `metacont` and `metamean`).
`method.sd`	A character string indicating which method to use to approximate the standard deviation from sample size, median, interquartile range and range (see `metacont` and `metamean`).
`approx.mean.e`	Approximation method to estimate means in experimental group (see `metacont`).
`approx.mean.c`	Approximation method to estimate means in control group (see `metacont`).
`approx.sd.e`	Approximation method to estimate standard deviations in experimental group (see `metacont`).
`approx.sd.c`	Approximation method to estimate standard deviations in control group (see `metacont`).
`approx.mean`	Approximation method to estimate means (see `metamean`).
`approx.sd`	Approximation method to estimate standard deviations (see `metamean`).
`approx.TE`	Approximation method to estimate treatment estimate (see `metagen`).
`approx.seTE`	Approximation method to estimate standard error (see `metagen`).
`pooledvar`	A logical indicating if a pooled variance should be used for the mean difference or ratio of means (see `metacont`).
`method.smd`	A character string indicating which method is used to estimate the standardised mean difference (see `metacont`).
`sd.glass`	A character string indicating which standard deviation is used in the denominator for Glass' method to estimate the standardised mean difference (only for metacont object with `sm = "SMD"`). Either `"control"` using the standard deviation in the control group (`sd.c`) or `"experimental"` using the standard deviation in the experimental group (`sd.e`), can be abbreviated.
`exact.smd`	A logical indicating whether exact formulae should be used in estimation of the standardised mean difference and its standard error.

`method.ci` A character string indicating which method is used to calculate confidence intervals for individual studies. Either `"z"`, `"t"`, `"WS"`, `"WSCC"`, `"AC"`, `"SA"`, `"SACC"`, `"NAsm"`, or `"Poisson"`, can be abbreviated. See functions `metacont`, `metaprop` and `metarate`.

`subgroup` An optional vector to conduct a meta-analysis with subgroups.

`subgroup.name` A character string with a name for the subgroup variable.

`print.subgroup.name`
A logical indicating whether the name of the subgroup variable should be printed in front of the group labels.

`sep.subgroup` A character string defining the separator between name of subgroup variable and subgroup label.

`test.subgroup` A logical value indicating whether to print results of test for subgroup differences.

`prediction.subgroup`
A logical indicating whether prediction intervals should be printed for subgroups.

`seed.predict.subgroup`
A numeric vector providing seeds to calculate bootstrap prediction intervals within subgroups. Must be of same length as the number of subgroups.

`byvar` Deprecated argument (replaced by ’subgroup’).

`id` Deprecated argument (replaced by ’cluster’).

`print.CMH` A logical indicating whether result of the Cochran-Mantel-Haenszel test for overall effect should be printed.

`keepdata` A logical indicating whether original data (set) should be kept in meta object.

`left` A logical indicating whether studies are supposed to be missing on the left or right side of the funnel plot. If NULL, the linear regression test for funnel plot symmetry (i.e., function `metabias(..., method = "linreg")`) is used to determine whether studies are missing on the left or right side.

`ma.common` A logical indicating whether a common effect or random effects model is used to estimate the number of missing studies.

`type` A character indicating which method is used to estimate the number of missing studies. Either `"L"` or `"R"`.

`n.iter.max` Maximum number of iterations to estimate number of missing studies.

`warn` A logical indicating whether warnings should be printed (e.g., if `incr` is added to studies with zero cell frequencies).

`warn.deprecated`
A logical indicating whether warnings should be printed if deprecated arguments are used.

`verbose` A logical indicating whether to print information on updates of older meta versions.

`control` An optional list to control the iterative process to estimate the between-study variance τ^2. This argument is passed on to `rma.uni` or `rma.glmm`, respectively.

`...` Additional arguments (ignored at the moment).

Details

Wrapper function to update an existing meta-analysis object which was created with R function metabin, metacont, metacor, metagen, metainc, metamean, metaprop, or metarate. More details on function arguments are available in help files of respective R functions.

This function can also be used for objects of class 'trimfill', 'metacum', and 'metainf'.

Value

An object of class "meta" and "metabin", "metacont", "metacor", "metainc", "metagen", "metamean", "metaprop", or "metarate" (see meta-object).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

metabin, metacont, metacor, metagen, metainc, metamean, metaprop, metarate

Examples

```
data(Fleiss1993cont)
m1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, studlab = paste(study, year), sm = "SMD")
m1

# Change summary measure (from 'SMD' to 'MD')
#
update(m1, sm = "MD")

# Restrict analysis to subset of studies
#
update(m1, subset = 1:2)

# Use different levels for confidence intervals
#
m2 <- update(m1, level = 0.66, level.ma = 0.99)
print(m2, digits = 2)
forest(m2)
```

---

weights.meta *Calculate absolute and percentage weights for meta-analysis*

Description

This function returns a data frame containing information on absolute and percentage weights of individual studies contributing to common effect and random effects meta-analysis.

Usage

```
## S3 method for class 'meta'
weights(
  object,
  common = object$common,
  random = object$random,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

Arguments

object
: An object of class meta.

common
: A logical indicating whether absolute and percentage weights from the common effect model should be calculated.

random
: A logical indicating whether absolute and percentage weights from the random effects model should be calculated.

warn.deprecated
: A logical indicating whether warnings should be printed if deprecated arguments are used.

...
: Additional arguments (to catch deprecated arguments).

Value

A data frame with the following variables is returned:

Variable	Definition	Condition
w.common	absolute weights in common effect model	(if common = TRUE)
p.common	percentage weights in common effect model	(if common = TRUE)
w.random	absolute weights in random effects model	(if random = TRUE)
p.random	percentage weights in random effects model	(if random = TRUE)

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

metabin, metacont, metagen

Examples

```
data(Fleiss1993cont)
# Do meta-analysis (common effect and random effects model)
#
meta1 <- metacont(n.psyc, mean.psyc, sd.psyc, n.cont, mean.cont, sd.cont,
  data = Fleiss1993cont, studlab = paste(study, year), sm = "SMD")

# Print weights for common effect and random effects meta-analysis
```

```
#
weights(meta1)

# Do meta-analysis (only random effects model)
#
meta2 <- update(meta1, common = FALSE)

# Print weights for random effects meta-analysis
#
weights(meta2)

# Print weights for common effect and random effects meta-analysis
#
weights(meta2, common = TRUE)
```

woodyplants *Elevated CO_2 and total biomass of woody plants*

Description

Meta-analysis on effects of elevated CO_2 on total biomass of woody plants

This dataset has been used as an example in Hedges et al. (1999) to describe methods for the meta-analysis of response ratios. The complete dataset with 102 observations and 26 variables is available online as a supplement. Here only a subset of 10 variables is provided and used in the examples.

Format

A data frame with the following columns:

obsno	observation number
papno	database paper number
treat	treatment code
level	treatment level
n.elev	number of observations in experimental group (elevated CO_2-level)
mean.elev	estimated mean in experimental group
sd.elev	standard deviation in experimental group
n.amb	number of observations in control group (ambient CO_2-level)
mean.amb	estimated mean in control group
sd.amb	standard deviation in control group

References

Hedges LV, Gurevitch J, Curtis PS (1999): The meta-analysis of response ratios in experimental ecology. *Ecology*, **80**, 1150–6

Examples

```
data(woodyplants)

# Meta-analysis of response ratios (Hedges et al., 1999)
#
m1 <- metacont(n.elev, mean.elev, sd.elev, n.amb, mean.amb, sd.amb,
  data = woodyplants, sm = "ROM", studlab = paste(obsno, papno, sep = " / "))
print(m1, prediction = TRUE)

# Meta-analysis for plants grown with low soil fertility treatment
#
m2 <- update(m1, subset = (treat == "fert" & level == "low"))
print(m2, prediction = TRUE)

# Meta-analysis for plants grown under low light conditions
#
m3 <- update(m1, subset = (treat == "light" & level == "low"))
print(m3, prediction = TRUE)
```

[.longarm *Extract parts of longarm object*

Description

Auxiliary function to extract parts of longarm object.

Usage

```
## S3 method for class 'longarm'
x[...]
```

Arguments

x	An object of class longarm.
...	Additional arguments (passed on to [.data.frame).

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

longarm, subset.longarm, dat.franchini2012

Examples

```
# Transform data from wide arm-based format to contrast-based format
pw1 <- pairwise(list(Treatment1, Treatment2, Treatment3),
  n = list(n1, n2, n3),
  mean = list(y1, y2, y3), sd = list(sd1, sd2, sd3),
  data = dat.franchini2012, studlab = Study)

# Transform data from contrast-based to long arm-based format
# and only keep the main variables
la1 <- longarm(pw1, append = FALSE)
head(la1)

la1[la1$studlab == "Lieberman 1998", ]
```

`[.pairwise` *Extract parts of pairwise object*

Description

Auxiliary function to extract parts of `pairwise` object.

Usage

```
## S3 method for class 'pairwise'
x[...]
```

Arguments

`x`	An object of class `pairwise`.
`...`	Additional arguments (passed on to [.data.frame).

Author(s)

Guido Schwarzer <`guido.schwarzer@uniklinik-freiburg.de`>

See Also

`pairwise`, `subset.pairwise`, `dat.franchini2012`

Examples

```
# Transform data from arm-based format to contrast-based format
pw1 <- pairwise(list(Treatment1, Treatment2, Treatment3),
  n = list(n1, n2, n3),
  mean = list(y1, y2, y3), sd = list(sd1, sd2, sd3),
  data = dat.franchini2012, studlab = Study)

pw1[, 1:5]
```

```
pw1[!grepl("Lieberman", pw1$studlab), 1:5]
```

Index