Package ‘netmeta’

May 6, 2023

Title Network Meta-Analysis using Frequentist Methods

Version 2.8-2

Date 2023-05-06

Depends R (>= 4.0.0), meta (>= 6.2-0)

Imports magic, MASS, ggplot2 (>= 3.0.0), metafor

Suggests colorspace, rgl, hasseDiagram (>= 0.1.3), grid, mvtnorm, gridExtra, igraph (>= 1.0.1), tictoc, writexl, R.rsp

URL https://github.com/guido-s/netmeta

https://link.springer.com/book/10.1007/978-3-319-21416-0

Description A comprehensive set of functions providing frequentist methods for network meta-analysis (Balduzzi et al., 2023) <doi:10.18637/jss.v106.i02> and supporting Schwarzer et al. (2015) <doi:10.1007/978-3-319-21416-0>, Chapter 8 ''Network Meta-Analysis'':
- additive network meta-analysis for combinations of treatments (Rücker et al., 2020) <doi:10.1002/bimj.201800167>;
- network meta-analysis of binary data using the Mantel-Haenszel or non-central hypergeometric distribution method (Efthimiou et al., 2019) <doi:10.1002/sim.8158>;
- rankograms and ranking of treatments by the Surface under the cumulative ranking curve (SUCRA) (Salanti et al., 2013) <doi:10.1016/j.jclinepi.2010.03.016>;
- league table with network meta-analysis results;
- 'comparison-adjusted' funnel plot (Chaimani & Salanti, 2012) <doi:10.1002/jrsm.57>;
- measures characterizing the flow of evidence between two treatments by König et al. (2013) <doi:10.1002/sim.6001>;

License GPL (>= 2)
Encoding UTF-8
VignetteBuilder R.rsp
RoxygenNote 7.2.3
NeedsCompilation no

Author Gerta Rücker [aut] (<https://orcid.org/0000-0002-2192-2560>),
    Ulrike Krahn [aut],
    Jochem König [aut] (<https://orcid.org/0000-0003-4683-0360>),
    Orestis Efthimiou [aut] (<https://orcid.org/0000-0002-0955-7572>),
    Annabel Davies [aut] (<https://orcid.org/0000-0003-2320-7701>),
    Theodoros Papakonstantinou [aut] (<https://orcid.org/0000-0002-6630-6817>),
    Guido Schwarzer [aut, cre] (<https://orcid.org/0000-0001-6214-9087>)

Maintainer Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>
Repository CRAN
Date/Publication 2023-05-06 10:10:05 UTC

R topics documented:

netmeta-package ................................................................. 4
as.data.frame.netconnection ............................................. 7
as.data.frame.netmeta ...................................................... 8
Baker2009 ................................................................. 9
decomp.design ........................................................... 10
dietaryfat ............................................................... 13
discomb ............................................................... 14
Dogliotti2014 ........................................................... 22
Dong2013 ............................................................... 23
forest.netbind ........................................................... 24
forest.netcomb ........................................................... 26
forest.netcomparison .................................................. 29
forest.netcomplex ...................................................... 32
forest.netmeta ........................................................... 34
forest.netsplit ........................................................... 38
Franchini2012 ........................................................... 42
funnel.netmeta ........................................................... 43
Gurusamy2011 ........................................................... 48
hasse ................................................................. 49
hatmatrix ............................................................ 51
heatplot ............................................................. 54
heatplot.netmeta ..................................................... 55
R topics documented:

invmat .................................................. 56
Linde2015 ............................................. 58
Linde2016 ............................................. 59
metabias.netmeta ....................................... 60
netbind ................................................. 63
netcomb ............................................... 65
netcomparison ......................................... 73
netcomplex ........................................... 77
netconnection.default ............................... 81
netcontrib ............................................. 84
netdistance ........................................... 87
netgraph .............................................. 88
netgraph.discomb .................................... 90
netgraph.netcomb ................................... 91
netgraph.netconnection ......................... 92
netgraph.netimpact .................................. 94
netgraph.netmeta ................................... 96
netheat ................................................ 104
netimpact ............................................ 107
netleague ............................................. 108
netmatrix ............................................. 113
netmeasures ......................................... 115
netmeta ................................................ 118
netmetabin ........................................... 129
netpairwise ........................................... 137
netposet ............................................... 141
netrank .............................................. 146
netsplit ............................................... 149
nettable .............................................. 155
pairwise ............................................... 158
plot.netposet ........................................ 165
plot.netrank .......................................... 169
plot.rankogram ....................................... 173
print.decomp.design ................................. 175
print.netbind ........................................ 176
print.netcomb ....................................... 178
print.netimpact ...................................... 180
print.summary.netcomb ............................. 181
print.summary.netmeta .............................. 183
radial.netmeta ....................................... 187
rankogram ............................................ 190
Senn2013 ............................................... 193
smokingcessation .................................... 195
Stowe2010 ............................................ 196
subset.pairwise ...................................... 197
summary.netcomb .................................... 198
summary.netmeta .................................... 200
treats .................................................. 203
R package netmeta (Balduzzi et al., 2023) provides frequentist methods for network meta-analysis and supports Schwarzer et al. (2015), Chapter 8 on network meta-analysis https://link.springer.com/book/10.1007/978-3-319-21416-0.

Details

R package netmeta is an add-on package for meta providing the following network meta-analysis models:

- frequentist network meta-analysis (function netmeta) based on Rücker (2012) and Rücker & Schwarzer (2014);
- additive network meta-analysis for combinations of treatments (netcomb for connected networks, discomb for disconnected networks) (Rücker et al., 2020a);
- network meta-analysis of binary data (netmetabin) using the Mantel-Haenszel or non-central hypergeometric distribution method (Efthimiou et al., 2019).

The following methods are available to present results of a network meta-analysis:

- network graphs (netgraph) described in Rücker & Schwarzer (2016);
- forest plots (forest.netmeta, forest.netcomb);
- league tables with network meta-analysis results (netleague);
- tables with network, direct and indirect estimates (nettable) looking similar to the statistical part of a GRADE table for a network meta-analysis (Puhan et al., 2014).

The following methods are implemented to rank treatments:

- rankograms (rankogram) (Salanti et al., 2011);
- ranking of treatments (netrank) based on P-scores (Rücker & Schwarzer, 2015) or the Surface Under the Cumulative RAnking curve (SUCRA) (Salanti et al., 2011);
- partial order of treatment rankings (netposet, plot.netposet) and Hasse diagram (hasse) according to Carlsen & Bruggemann (2014) and Rücker & Schwarzer (2017).

Available functions to evaluate network inconsistency:

- split direct and indirect evidence (netsplit) to check for consistency (Dias et al., 2010; Efthimiou et al., 2019);
- net heat plot (netheat) and design-based decomposition of Cochran’s Q (decomp.design) described in Krahn et al. (2013).
Additional methods and functions:

- information on network connectivity (netconnection);
- contribution of direct comparisons to network estimates (netcontrib) (Papakonstantinou et al., 2018; Davies et al., 2022);
- importance of individual studies measured by reduction of precision if removed from network (netimpact) (Rücker et al., 2020b);
- ‘comparison-adjusted’ funnel plot (funnel.netmeta) to assess funnel plot asymmetry in network meta-analysis (Chaimani & Salanti, 2012);
- conduct pairwise meta-analyses for all comparisons with direct evidence in a network meta-analysis (netpairwise);
- results of several network meta-analyses can be combined with netbind to show these results in a forest plot (forest.netbind);
- measures characterizing the flow of evidence between two treatments (netmeasures) described in König et al. (2013);
- calculate comparison effects of two arbitrary complex interventions in component network meta-analysis (netcomparison);
- calculate effect of arbitrary complex interventions in component network meta-analysis (netcomplex).

Functions and datasets from netmeta are utilised in Schwarzer et al. (2015), Chapter 8 "Network Meta-Analysis", https://link.springer.com/book/10.1007/978-3-319-21416-0. Type help(package = "netmeta") for a listing of all R functions available in netmeta. Type citation("netmeta") on how to cite netmeta in publications.

To report problems and bugs

- type bug.report(package = "netmeta") 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 netmeta is available on GitHub https://github.com/guido-s/netmeta.

Author(s)

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

References


Create a data frame from an object of class netconnection

Description

The as.data.frame method returns a data frame containing information on membership of studies / pairwise comparisons to a (sub)network.

Usage

## S3 method for class 'netconnection'
as.data.frame(x, ...)

Arguments

x  An object of class netconnection.
...
Additional arguments (ignored).

Value

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

Author(s)

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

See Also

netconnection

Examples

# Artificial example with two subnetworks
#
t1 <- c("G", "B", "B", "D", "A", "F")
t2 <- c("B", "C", "E", "E", "H", "A")
#
nc2 <- netconnection(t1, t2)
print(nc2, details = TRUE)

as.data.frame(nc2)
as.data.frame.netmeta  
Create a data frame from an object of class netmeta

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

Arguments
- **x**: An object of class `netmeta`.
- **row.names**: NULL or a character vector giving the row names for the data frame.
- **optional**: A logical. If TRUE, setting row names and converting column names (to syntactic names) is optional.
- **details**: A logical. If TRUE, additional variables of less interest are included in data frame.
- **...**: Additional arguments.

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

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also
`netmeta`

Examples
```r
data(smokingcessation)

# Transform data from arm-based format to contrast-based format#
pl <- pairwise(list(treat1, treat2, treat3),
        event = list(event1, event2, event3), n = list(n1, n2, n3),
        data = smokingcessation, sm = "OR")

# Conduct random effects network meta-analysis and show data frame#
ett1 <- netmeta(pl, common = FALSE)
```
as.data.frame(net1)

## Not run:
data(Senn2013)

# Conduct network meta-analysis
#
net2 <- netmeta(TE, seTE, treat1, treat2, studlab,
    data = Senn2013, sm = "MD")

as.data.frame(net2)
as.data.frame(net2, details = TRUE)

## End(Not run)

Network meta-analysis of pharmacologic treatments for chronic obstructive pulmonary disease

Description

This data set comes from a systematic review of randomized controlled trials on pharmacologic treatments for chronic obstructive pulmonary disease (COPD) (Baker et al., 2009).

The primary outcome, occurrence of one or more episodes of COPD exacerbation, is binary (yes / no). For this outcome, five drug treatments (fluticasone, budesonide, salmeterol, formoterol, tiotropium) and two combinations (fluticasone + salmeterol, budesonide + formoterol) were compared to placebo. The authors considered the two combinations as separate treatments instead of evaluating the individual components.

Format

A data frame with the following columns:

- **study**: study label
- **year**: year of publication
- **id**: study ID
- **treatment**: treatment
- **exac**: one or more episodes of COPD exacerbation
- **total**: number of individuals in treatment arm

Source

See Also

pairwise, metabin, netmetabin

Examples

data(Baker2009)
Baker2009

## Not run:
# 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.
#
# pl <- pairwise(treatment, exac, total, studlab = paste(study, year),
data = Baker2009, sm = "OR")

# Conduct network meta-analysis
#
net1 <- netmeta(pl, ref = "plac")

# Conduct component network meta-analysis
#
cnet1 <- netcomb(net1)
cnet1

## End(Not run)

decomp.design  Design-based decomposition of Cochran’s Q in network meta-analysis

Description

This function performs a design-based decomposition of Cochran’s Q for assessing the homogeneity in the whole network, the homogeneity within designs, and the homogeneity/consistency between designs. It allows also an assessment of the consistency assumption after detaching the effect of single designs.

Usage

decomp.design(
  x,
  tau.preset = x$tau.preset,
  warn = TRUE,
  nchar.trts = x$nchar.trts
)
Arguments

- **x**: An object of class `netmeta`.
- **tau.preset**: An optional value for the square-root of the between-study variance \( \tau^2 \) (see Details).
- **warn**: A logical indicating whether warnings should be printed.
- **nchar.trts**: A numeric defining the minimum number of characters used to create unique treatment names.

Details

In the context of network meta-analysis and the assessment of the homogeneity and consistency assumption, a generalized Cochran's Q statistic for multivariate meta-analysis can be used as shown in Krahn et al. (2013). This Q statistic can be decomposed in a sum of within-design Q statistics and one between-designs Q statistic that incorporates the concept of design inconsistency, see Higgins et al. (2012).

For assessing the inconsistency in a random effects model, the between-designs Q statistic can be calculated based on a full design-by-treatment interaction random effects model (see Higgins et al., 2012). This Q statistic will be automatically given in the output (\( \tau^2 \) estimated by the method of moments (see Jackson et al., 2012). Alternatively, the square-root of the between-study variance can be prespecified by argument `tau.preset` to obtain a between-designs Q statistic (in `Q.inc.random`, its design-specific contributions `Q.inc.design.random.preset`) as well as residuals after detaching of single designs (`residuals.inc.detach.random.preset`).

Since an inconsistent treatment effect of one design can simultaneously inflate several residuals, Krahn et al. (2013) suggest for locating the inconsistency in a network to fit a set of extended models allowing for example for a deviating effect of each study design in turn. The recalculated between-designs Q statistics are given in list component `Q.inc.detach`. The change of the inconsistency contribution of single designs can be investigated in more detail by a net heat plot (see function `netheat`). Designs where only one treatment is involved in other designs of the network or where the removal of corresponding studies would lead to a splitting of the network do not contribute to the inconsistency assessment. These designs are not included in `Q.inc.detach`.

Value

Network meta-analysis with a single design: `NULL`. Otherwise, a list containing the following components:

- **Q.decomp**: Data frame with Q statistics (variable `Q`) based on the common effects model to assess the homogeneity/consistency in the whole network, within designs, and between designs. Corresponding degrees of freedom (df) and p-values (p.val) are also given.

- **Q.het.design**: Data frame with design-specific decomposition of the within-designs Q statistic (Q) of the common effects model, corresponding degrees of freedom (df) and p-values (p.val) are given.

- **Q.inc.detach**: Data frame with between-designs Q statistics (Q) of the common effects model after detaching of single designs, corresponding degrees of freedom (df) and p-values (p.val) are given.
Q.inc.design  A named vector with contributions of single designs to the between design Q statistic given in Q.decomp.

Q.inc.random  Data frame with between-designs Q statistic (Q) based on a random effects model with square-root of between-study variance tau.within estimated embedded in a full design-by-treatment interaction model, corresponding degrees of freedom (df) and p-value (p.val).

Q.inc.random.preset  Data frame with between-designs Q statistic (Q) based on a random effects model with prespecified square-root of between-study variance tau.preset in the case if argument tau.preset is not NULL, corresponding degrees of freedom (df) and p-value (p.val).

Q.inc.design.random.preset  A named vector with contributions of single designs to the between design Q statistic based on a random effects model with prespecified square-root of between-study variance tau.preset in the case if argument tau.preset is given.

residuals.inc.detach  Matrix with residuals, i.e. design-specific direct estimates minus the corresponding network estimates after detaching the design of the column.

residuals.inc.detach.random.preset  Matrix with residuals analogous to residuals.inc.detach but based on a random effects model with prespecified square-root of between-study variance tau.preset in the case if argument tau.preset is not NULL.

call  Function call.

version  Version of R package netmeta used to create object.

Author(s)
Ulrike Krahn <ulrike.krahn@bayer.com>, Jochem König <koenigjo@uni-mainz.de>

References


See Also
netmeta, netheat
dietaryfat

Examples

data(Senn2013)

# Only consider first five studies (to reduce runtime of example)
# studies <- unique(Senn2013$studlab)
Senn2013.5 <- subset(Senn2013, studlab %in% studies[1:5])

# Conduct network meta-analysis with placebo as reference treatment
# net1 <- netmeta(TE, seTE, treat1, treat2, studlab,
# data = Senn2013.5, sm = "MD", reference = "plac")

# Decomposition of Cochran's Q
# decomp.design(net1)

---

dietaryfat  

Network meta-analysis of dietary fat

Description

Network meta-analysis comparing the effects of two diets to control on mortality.

The data are rates, given as the number of deaths and person-years. These data are used as an example in the supplemental material of Dias et al. (2013).

Format

A data frame with the following columns:

- `treat1`: treatment 1
- `treat2`: treatment 2
- `treat3`: treatment 3
- `years1`: person years arm 1
- `years2`: person years arm 2
- `years3`: person years arm 3
- `d1`: events (deaths) arm 1
- `d2`: events (deaths) arm 2
- `d3`: events (deaths) arm 3
- `ID`: study ID

Source

discomb

Additive network meta-analysis for combinations of treatments (disconnected networks)

Description

Some treatments in a network meta-analysis may be combinations of other treatments or have common components. The influence of individual components can be evaluated in an additive network meta-analysis model assuming that the effect of treatment combinations is the sum of the effects of its components. This function implements this additive model in a frequentist way and is particularly intended for disconnected networks.
Usage

```r
discomb(
  TE,
  sETE,
  treat1,
  treat2,
  studlab,
  data = NULL,
  subset = NULL,
  inactive = NULL,
  sep.comps = "+",
  C.matrix,
  sm,
  level = gs("level"),
  level.ma = gs("level.ma"),
  common = gs("common"),
  random = gs("random") | !is.null(tau.preset),
  reference.group,
  baseline.reference = TRUE,
  seq = NULL,
  tau.preset = NULL,
  tol.multiarm = 0.001,
  tol.multiarm.se = NULL,
  details.chkmultiarm = FALSE,
  details.chkident = FALSE,
  sep.trts = ":",
  nchar.comps = 666,
  func.inverse = invmat,
  backtransf = gs("backtransf"),
  title = "",
  warn = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  nchar.trts = nchar.comps,
  ...
)
```

Arguments

- **TE** Estimate of treatment effect, i.e. difference between first and second treatment (e.g. log odds ratio, mean difference, or log hazard ratio). Or an R object created with `pairwise`.
- **sETE** Standard error of treatment estimate.
- **treat1** Label/Number for first treatment.
- **treat2** Label/Number for second treatment.
- **studlab** An optional - but important! - vector with study labels (see `netmeta`).
- **data** An optional data frame containing the study information.
- **subset** An optional vector specifying a subset of studies to be used.
inactive A character string defining the inactive treatment component (see Details).
sep.comps A single character to define separator between treatment components.
C.matrix C matrix (see Details).
sm A character string indicating underlying summary measure, e.g., "RD", "RR", "OR", "ASD", "HR", "MD", "SMD", or "ROM".
level The level used to calculate confidence intervals for individual comparisons.
level.ma The level used to calculate confidence intervals for network estimates.
common A logical indicating whether a common effects / common effects network meta-analysis should be conducted.
random A logical indicating whether a random effects network meta-analysis should be conducted.
reference.group Reference treatment (first treatment is used if argument is missing).
baseline.reference A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa. This argument is only considered if reference.group has been specified.
seq A character or numerical vector specifying the sequence of treatments in printouts.
tau.preset An optional value for the square-root of the between-study variance $\tau^2$.
tol.multiarm A numeric for the tolerance for consistency of treatment estimates in multi-arm studies which are consistent by design.
tol.multiarm.se A numeric for the tolerance for consistency of standard errors in multi-arm studies which are consistent by design. This check is not conducted if the argument is NULL.
details.chkmultiarm A logical indicating whether treatment estimates and / or variances of multi-arm studies with inconsistent results or negative multi-arm variances should be printed.
details.chkident A logical indicating whether details on unidentifiable components should be printed.
sep.trts A character used in comparison names as separator between treatment labels.
nchar.comps A numeric defining the minimum number of characters used to create unique names for components (see Details).
func.inverse R function used to calculate the pseudoinverse of the Laplacian matrix L (see netmeta).
backtransf A logical indicating whether results should be back transformed in printouts and forest plots. If backtransf = TRUE, results for sm = "OR" are presented as odds ratios rather than log odds ratios, for example.
title Title of meta-analysis / systematic review.
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.

nchar.trts Deprecated argument (replaced by nchar.comps).

Details

Treatments in network meta-analysis (NMA) can be complex interventions. Some treatments may be combinations of others or have common components. The standard analysis provided by `netmeta` is a NMA where all existing (single or combined) treatments are considered as different nodes in the network. Exploiting the fact that some treatments are combinations of common components, an additive component network meta-analysis (CNMA) model can be used to evaluate the influence of individual components. This model assumes that the effect of a treatment combination is the sum of the effects of its components which implies that common components cancel out in comparisons.

This R function can be used for disconnected networks. Use `netmeta` and `netcomb` for connected networks.

The additive CNMA model has been implemented using Bayesian methods (Mills et al., 2012; Welton et al., 2013). This function implements the additive model in a frequentist way (Rücker et al., 2020).

The underlying multivariate model is given by

$$\delta = B \theta, \theta = C \beta$$

with

- $\delta$ vector of true treatment effects (differences) from individual studies,
- $B$ design matrix describing the structure of the network,
- $\theta$ parameter vector that represents the existing combined treatments,
- $C$ matrix describing how the treatments are composed,
- $\beta$ parameter vector representing the treatment components.

All parameters are estimated using weighted least squares regression.

Argument `inactive` can be used to specify a single component that does not have any therapeutic value. Accordingly, it is assumed that the treatment effect of the combination of this component with an additional treatment component is equal to the treatment effect of the additional component alone.

Argument `sep.comps` can be used to specify the separator between individual components. By default, the matrix $C$ is calculated internally from treatment names. However, it is possible to specify a different matrix using argument `C.matrix`.

By default, component names are not abbreviated in printouts. However, in order to get more concise printouts, argument `nchar.comps` can be used to define the minimum number of characters for abbreviated component names (see `abbreviate`, argument `minlength`). R function `treats` is utilised internally to create abbreviated component names.
Value

An object of classes discomb and netcomb with corresponding print, summary, and forest functions. The object is a list containing the following components:

- `studlab`: Study labels.
- `treat1`: Label/Number for first treatment.
- `treat2`: Label/Number for second treatment.
- `TE`: Estimate of treatment effect, i.e. difference between first and second treatment.
- `seTE`: Standard error of treatment estimate.
- `seTE.adj.common`, `seTE.adj.random`: Standard error of treatment estimate, adjusted for multi-arm studies.
- `event1`: Number of events in first treatment group.
- `event2`: Number of events in second treatment group.
- `n1`: Number of observations in first treatment group.
- `n2`: Number of observations in second treatment group.
- `k`: Total number of studies.
- `m`: Total number of pairwise comparisons.
- `n`: Total number of treatments.
- `d`: Total number of designs (corresponding to the unique set of treatments compared within studies).
- `c`: Total number of components.
- `trts`: Treatments included in network meta-analysis.
- `comps`: Unique list of components present in the network.
- `TE.cnma.common`, `TE.cnma.random`: A vector of length \( m \) of consistent treatment effects estimated by the additive (common and random effects) model.
- `seTE.cnma.common`, `seTE.cnma.random`: A vector of length \( m \) with standard errors estimated by the additive (common and random effects) model.
- `lower.cnma.common`, `lower.cnma.random`: A vector of length \( m \) of lower confidence interval limits for consistent treatment effects estimated by the additive (common and random effects) model.
- `upper.cnma.common`, `upper.cnma.random`: A vector of length \( m \) of upper confidence interval limits for consistent treatment effects estimated by the additive (common and random effects) model.
- `statistic.cnma.common`, `statistic.cnma.random`: A vector of length \( m \) of z-values for the test of an overall effect estimated by the additive (common and random effects) model.
- `pval.cnma.common`, `pval.cnma.random`: A vector of length \( m \) of p-values for the test of an overall effect estimated by the additive (common and random effects) model.
TE.common, TE.random

\(n \times n\) matrix with overall treatment effects estimated by the additive (common and random effects) model.

seTE.common, seTE.random

\(n \times n\) matrix with standard errors estimated by the additive (common and random effects) model.

lower.common, upper.common, lower.random, upper.random

\(n \times n\) matrices with lower and upper confidence interval limits estimated by the additive (common and random effects) model.

statistic.common, pval.common, statistic.random, pval.random

\(n \times n\) matrices with z-values and p-values for test of overall effect estimated by the additive (common and random effects) model.

Comp.common, Comp.random

A vector of component effects (common and random effects model).

seComp.common, seComp.random

A vector with corresponding standard errors (common and random effects model).

lower.Comp.common, lower.Comp.random

A vector with lower confidence limits for components (common and random effects model).

upper.Comp.common, upper.Comp.random

A vector with upper confidence limits for components (common and random effects model).

statistic.Comp.common, statistic.Comp.random

A vector with z-values for the overall effect of components (common and random effects model).

pval.Comp.common, pval.Comp.random

A vector with p-values for the overall effect of components (common and random effects model).

Comb.common, Comb.random

A vector of combination effects (common and random effects model).

seComb.common, seComb.random

A vector with corresponding standard errors (common and random effects model).

lower.Comb.common, lower.Comb.random

A vector with lower confidence limits for combinations (common and random effects model).

upper.Comb.common, upper.Comb.random

A vector with upper confidence limits for combinations (common and random effects model).

statistic.Comb.common, statistic.Comb.random

A vector with z-values for the overall effect of combinations (common and random effects model).

pval.Comb.common, pval.Comb.random

A vector with p-values for the overall effect of combinations (common and random effects model).

Q.additive

Overall heterogeneity / inconsistency statistic (additive model).

df.Q.additive

Degrees of freedom for test of heterogeneity / inconsistency (additive model).
pval.Q.additive
P-value for test of heterogeneity / inconsistency (additive model).

tau
Square-root of between-study variance (additive model).

I2
I-squared (additive model).

Q.standard
Overall heterogeneity / inconsistency statistic (standard model).
df.Q.standard
Degrees of freedom for test of heterogeneity / inconsistency (standard model).
pval.Q.standard
P-value for test of heterogeneity / inconsistency (standard model).

Q.diff
Test statistic for difference in goodness of fit between standard and additive model.
df.Q.diff
Degrees of freedom for difference in goodness of fit between standard and additive model.
pval.Q.diff
P-value for difference in goodness of fit between standard and additive model.

X.matrix
Design matrix ($m \times n$).

B.matrix
Edge-vertex incidence matrix ($m \times n$).

C.matrix
As defined above.

sm
Summary measure.

level.ma
Level for confidence intervals.

common, random, tau.preset
As defined above.

sep.trts
A character used in comparison names as separator between treatment labels.

nchar.comps
A numeric defining the minimum number of characters used to create unique component names.

inactive, sep.comps
As defined above.

backtransf
A logical indicating whether results should be back transformed in printouts and forest plots.

title
Title of meta-analysis / systematic review.

x
As defined above.

call
Function call.

version
Version of R package netmeta used to create object.

Note
This function calculates effects for individual components and complex interventions present in the network.

R function netcomplex can be used to calculate the effect for arbitrary complex interventions in a component network meta-analysis. Furthermore, R function netcomparison can be used to calculate the effect for comparisons of two arbitrary complex intervention in a component network meta-analysis.

Author(s)
Gerta Rücker <gerta.ruecker@uniklinik-freiburg.de>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>
References


See Also

`netcomb`, `forest.netcomb`, `summary.netcomb`, `netmeta`, `netconnection`, `netcomplex`, `netcomparison`

Examples

```r
# Artificial dataset
#
t2 <- c("C", "B", "B + C", "A + D", "E", "F", "F")

# mean <- c(4.1, 2.05, 0, 0, 0.1, 0.1, 0.05)
# se.mean <- rep(0.1, 7)
# study <- paste("study", c(1:4, 5, 5, 5))
# dat <- data.frame(mean, se.mean, t1, t2, study,
#        stringsAsFactors = FALSE)
#        "B", "B + C", "C", "D", "E", "F")
# comps <- LETTERS[1:6]

# Use netconnection() to display network information
# netconnection(t1, t2, study)

dc1 <- discomb(mean, se.mean, t1, t2, study, seq = trts)
dc1

forest(dc1, ref = "F")

# Define C matrix manually (which will produce the same results)
#
# C <- rbind(c(1, 0, 0, 0, 0, 0), # A
#     c(1, 1, 0, 0, 0, 0), # A + B
#     c(1, 0, 1, 0, 0, 0), # A + C
```
c(1, 0, 0, 1, 0, 0), # A + D
c(0, 1, 0, 0, 0, 0), # B
c(0, 1, 0, 0, 0, 0), # B + C
c(0, 0, 1, 0, 0, 0), # C
c(0, 0, 1, 0, 0, 0), # D
\[c(0, 0, 0, 1, 0, 0), # E
\[c(0, 0, 0, 0, 1, 0), # F
\]

#
\[\text{colnames(C) <- comps}
\]
\[\text{rownames(C) <- trts}
\]
\[\text{#}
\]
\[\text{dc2 <- discomb(mean, se.mean, t1, t2, study, seq = trts,}
\]
\[\text{C.matrix = C)}
\]
\[\text{#}
\]
\[\text{# Compare C matrices}
\]
\[\text{#}
\]
\[\text{all.equal(dc1$C.matrix, dc2$C.matrix)}
\]

---

**Dogliotti2014**

*Network meta-analysis of antithrombotic treatments in patients with non-valvular atrial fibrillation*

---

**Description**

This data set comes from a systematic review aiming to determine the effects of eight antithrombotic treatments in reducing the incidence of major thrombotic events in patients with non-valvular atrial fibrillation (Dogliotti et al., 2014). The review included 20 studies (79,808 participants), four of which were three-arm studies. The primary outcome is stroke reduction.

**Format**

A data frame with the following columns:

- **study**: study label
- **id**: study ID
- **treatment**: treatment
- **stroke**: number of strokes
- **total**: number of individuals in treatment arm

**Source**


**See Also**

pairwise, metabin, netmetabin
Examples

```r
data(Dogliotti2014)
Dogliotti2014

## Not run:
# 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.
#
p1 <- pairwise(treatment, stroke, total, studlab = study,
               data = Dogliotti2014, sm = "OR")
#
# Conduct Mantel-Haenszel network meta-analysis
#
netmetabin(p1, ref = "plac")

## End(Not run)
```

Description

Network meta-analysis comparing inhaled medications in patients with chronic obstructive pulmonary disease.

Format

A data frame with the following columns:

- `id`: study ID
- `treatment`: treatment
- `death`: mortality
- `randomized`: number of individuals in treatment arm

Source


See Also

`pairwise`, `metabin`, `netmetabin`

Examples

```r
data(Dong2013)
```
# Only consider first ten studies (to reduce runtime of example)
first10 <- subset(Dong2013, id <= 10)

# 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.

p1 <- pairwise(treatment, death, randomized, studlab = id,
data = first10, sm = "OR")

# Conduct Mantel-Haenszel network meta-analysis
# netmetabin(p1, ref = "plac")

## Not run:
# Conduct Mantel-Haenszel network meta-analysis for the whole
# dataset

p2 <- pairwise(treatment, death, randomized, studlab = id,
data = Dong2013, sm = "OR")
netmetabin(p2, ref = "plac")

## End(Not run)

---

### forest.netbind

**Forest plot showing results of two or more network meta-analyses**

#### Description

Forest plot to show network estimates of two or more network meta-analyses.

#### Usage

```r
## S3 method for class 'netbind'
forest(
x,
pooled = ifelse(x$x$x$random, "random", "common"),
equal.size = TRUE,
leftcols = "studlab",
leftlabs = "Treatment",
rightcols = c("effect", "ci"),
rightlabs = NULL,
subset.treatments,
digits = gs("digits.forest"),
digits.prop = max(gs("digits.pval") - 2, 2),
backtransf = x$x$backtransf,
```

---
forest.netbind

lab.NA = "",
smlab,
...
)

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

Arguments

x An object of class netbind.
pooled A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.
equal.size A logical indicating whether all squares should be of equal size. Otherwise, the square size is proportional to the precision of estimates.
leftcols A character vector specifying columns to be plotted on the left side of the forest plot (see Details).
leftlabs A character vector specifying labels for columns on left side of the forest plot.
rightcols A character vector specifying columns to be plotted on the right side of the forest plot (see Details).
rightlabs A character vector specifying labels for columns on right side of the forest plot.
subset.treatments A character vector specifying treatments to show in forest plot as comparators to the reference.
digits Minimal number of significant digits for treatment effects and confidence intervals, see print.default.
digits.prop Minimal number of significant digits for the direct evidence proportion.
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.
lab.NA A character string to label missing values.
smlab A label printed at top of figure. By default, text indicating either common or random effects model is printed.
... Additional arguments for forest.meta function.

Details

A forest plot, also called confidence interval plot, is drawn in the active graphics window.
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. If argument rightcols is FALSE, no columns will be plotted on the right side.
For more information see help page of forest.meta function.

Author(s)

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

netbind, netcomb, forest.meta

Examples

data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce
# runtime of example)
#
face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Standard random effects NMA model (with placebo as reference
# treatment)
#
net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
  data = face, reference.group = "placebo",
  sm = "OR", common = FALSE)

# Additive CNMA model with placebo as inactive component and
# reference
#
cmp1 <- netcomb(net1, inactive = "placebo")

# Combine results of standard NMA and CNMA
#
nb1 <- netbind(nc1, net1,
  name = c("Additive CNMA", "Standard NMA"),
  col.study = c("red", "black"),
  col.square = c("red", "black"))

forest(nb1,
  col.by = "black", addrow.subgroups = FALSE,
  fontsize = 10, spacing = 0.7, squaresize = 0.9,
  label.left = "Favours Placebo",
  label.right = "Favours other")

---

Forest plot for additive network meta-analysis

Description

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

Usage

## S3 method for class 'netcomb'
forest(
x,

---

forest.netcomb

Forest plot for additive network meta-analysis
pooled = ifelse(x$random, "random", "common"),
reference.group = x$reference.group,
baseline.reference = x$baseline.reference,
leftcols = "studlab",
leftlabs = "Treatment",
rightcols = c("effect", "ci"),
rightlabs = NULL,
digits = gs("digits.forest"),
smlab = NULL,
sortvar = x$seq,
backtransf = x$backtransf,
lab.NA = ".",
add.data,
drop.reference.group = FALSE,
weight.study = "same",
...
)

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

**Arguments**

- **x**
  - An object of class netcomb.

- **pooled**
  - A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.

- **reference.group**
  - Reference treatment(s).

- **baseline.reference**
  - A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa.

- **leftcols**
  - A character vector specifying (additional) columns to be plotted on the left side of the forest plot or a logical value (see forest.meta help page for details).

- **leftlabs**
  - A character vector specifying labels for (additional) columns on left side of the forest plot (see forest.meta help page for details).

- **rightcols**
  - A character vector specifying (additional) columns to be plotted on the right side of the forest plot or a logical value (see forest.meta help page for details).

- **rightlabs**
  - A character vector specifying labels for (additional) columns on right side of the forest plot (see forest.meta help page for details).

- **digits**
  - Minimal number of significant digits for treatment effects and confidence intervals, see print.default.

- **smlab**
  - A label printed at top of figure. By default, text indicating either common or random effects model is printed.

- **sortvar**
  - An optional vector used to sort the individual studies (must be of same length as the total number of treatments).
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.

lab.NA A character string to label missing values.

add.data An optional data frame with additional columns to print in forest plot (see Details).

drop.reference.group A logical indicating whether the reference group should be printed in the forest plot.

weight.study A character string indicating weighting used to determine size of squares or diamonds.

... Additional arguments for forest.meta function.

Details
A forest plot, also called confidence interval plot, is drawn in the active graphics window. Argument sortvar can be either a numeric or character vector with length of number of treatments. If sortvar is numeric the order function is utilised internally to determine the order of values. If sortvar is character it must be a permutation of the treatment names. It is also possible to sort by treatment comparisons (sortvar = TE, etc.), standard error (sortvar = seTE), and number of studies with direct treatment comparisons (sortvar = k).

Argument add.data can be used to add additional columns to the forest plot. This argument must be a data frame with the same row names as the treatment effects matrices in R object x, i.e., x$TE.common or x$TE.random.

For more information see help page of forest.meta function.

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also
netcomb, discomb, forest.meta

Examples
data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce # runtime of example)
# face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
# net1 <- netmeta(lnOR, selnOR, treat1, treat2, id, data = face, ref = "placebo", sm = "OR", common = FALSE)
# Additive model for treatment components (with placebo as inactive treatment)
#
# ncl <- netcomb(net1, inactive = "placebo")
#
# forest(nc1)

## Not run:
# Specify, order of treatments
#
trts <- c("TCA", "SSRI", "SNRI", "NRI", "Low-dose SARI", "NaSSa", "rMAO-A", "Ind drug", "Hypericum", "Face-to-face CBT", "Face-to-face PST", "Face-to-face interpsy", "Face-to-face psychodyn", "Other face-to-face", "Remote CBT", "Self-help CBT", "No contact CBT", "Face-to-face CBT + SSRI", "Face-to-face interpsy + SSRI", "Face-to-face PST + SSRI", "UC", "Placebo")
#
# Note, three treatments are actually combinations of 'SSRI' with other components:
# "Face-to-face CBT + SSRI",
# "Face-to-face interpsy + SSRI",
# "Face-to-face PST + SSRI"

# Conduct random effects network meta-analysis
#
net2 <- netmeta(lnOR, selnOR, treat1, treat2, id, data = Linde2016, ref = "placebo", seq = trts, sm = "OR", common = FALSE)
#
net2

# Additive model for treatment components (with placebo as inactive treatment)
#
nc2 <- netcomb(net2, inactive = "placebo")
#
forest(nc2)

## End(Not run)

forest.netcomparison

Forest plot for complex interventions in component network meta-analysis

Description

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

```r
## S3 method for class 'netcomparison'
forest(
x,
pooled = ifelse(x$random, "random", "common"),
leftcols = c("studlab", "treat1", "treat2"),
leftlabs = c("Comparison", "Trt 1", "Trt 2"),
rightcols = c("effect", "ci", "statistic", "pval"),
rightlabs = c(NA, NA, "z", "p-value"),
nchar.comps = x$nchar.trts,
digits = gs("digits.forest"),
digits.stat = gs("digits.stat"),
digits.pval = gs("digits.pval"),
smlab = NULL,
backtransf = x$backtransf,
lab.NA = ".",
weight.study = "same",
...)
```

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

Arguments

- **x**: An object of class netcomparison.
- **pooled**: A character string indicating whether results for the common (“common”) or random effects model (“random”) should be plotted. Can be abbreviated.
- **leftcols**: A character vector specifying (additional) columns to be plotted on the left side of the forest plot or a logical value (see forest.meta help page for details).
- **leftlabs**: A character vector specifying labels for (additional) columns on left side of the forest plot (see forest.meta help page for details).
- **rightcols**: A character vector specifying (additional) columns to be plotted on the right side of the forest plot or a logical value (see forest.meta help page for details).
- **rightlabs**: A character vector specifying labels for (additional) columns on right side of the forest plot (see forest.meta help page for details).
- **nchar.comps**: A numeric defining the minimum number of characters used to create unique names for components.
- **digits**: Minimal number of significant digits for treatment effects and confidence intervals, see print.default.
- **digits.stat**: Minimal number of significant digits for tests of overall effect, see print.default.
- **digits.pval**: Minimal number of significant digits for p-value of overall effects, see print.default.
- **smlab**: A label printed at top of figure. By default, text indicating either common or random effects model is printed.
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.

lab.NA  A character string to label missing values.

weight.study  A character string indicating weighting used to determine size of squares or diamonds.

...  Additional arguments for forest.meta function.

Details

A forest plot, also called confidence interval plot, is drawn in the active graphics window. For more information see help page of forest.meta function.

Author(s)

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

See Also

netcomparison, netcomb, discomb, forest.meta

Examples

data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce # runtime of example)
# face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
# net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
# data = face, ref = "placebo", sm = "OR", common = FALSE)

# Additive model for treatment components (with placebo as inactive # treatment)
# ncl1 <- netcomb(net1, inactive = "placebo")

# Some comparisons
# t1 <- c("F + TCA", "F + Plac", "SSRI + Plac + TCA")
t2 <- c("UC", "Plac", "UC")
# netcomparison(ncl1, t1, t2)
# forest(netcomparison(ncl1, t1, t2))
forest(netcomparison(ncl1, c("F", "TCA"), "UC"), nchar.comps = 4)
Forest plot for complex interventions in component network meta-analysis

Description

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

Usage

```r
## S3 method for class 'netcomplex'
forest(
x,
pooled = ifelse(x$random, "random", "common"),
leftcols = "studlab",
leftlabs = NULL,
rightcols = c("effect", "ci", "statistic", "pval"),
rightlabs = c(NA, NA, "z", "p-value"),
nchar.comps = x$nchar.trts,
digits = gs("digits.forest"),
digits.stat = gs("digits.stat"),
digits.pval = gs("digits.pval"),
smlab = NULL,
backtransf = x$backtransf,
lab.NA = ".",
weight.study = "same",
...
)

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

Arguments

- `x` An object of class `netcomplex`.
- `pooled` A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.
- `leftcols` A character vector specifying (additional) columns to be plotted on the left side of the forest plot or a logical value (see `forest.meta` help page for details).
- `leftlabs` A character vector specifying labels for (additional) columns on left side of the forest plot (see `forest.meta` help page for details).
- `rightcols` A character vector specifying (additional) columns to be plotted on the right side of the forest plot or a logical value (see `forest.meta` help page for details).
- `rightlabs` A character vector specifying labels for (additional) columns on right side of the forest plot (see `forest.meta` help page for details).
A numeric defining the minimum number of characters used to create unique names for components.

digits

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

digits.stat

Minimal number of significant digits for tests of overall effect, see `print.default`.

digits.pval

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

smlab

A label printed at top of figure. By default, text indicating either common or random effects model is printed.

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.

lab.NA

A character string to label missing values.

weight.study

A character string indicating weighting used to determine size of squares or diamonds.

... Additional arguments for `forest.meta` function.

**Details**

A forest plot, also called confidence interval plot, is drawn in the active graphics window. For more information see help page of `forest.meta` function.

**Author(s)**

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

**See Also**

`netcomplex, netcomb, discomb, forest.meta`

**Examples**

data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce # runtime of example)
# face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
# net1 <- netmeta(lnOR, selnOR, treat1, treat2, id, #
# data = face, ref = "placebo", sm = "OR", common = FALSE)

# Additive model for treatment components (with placebo as inactive # treatment)
# ncl <- netcomb(net1, inactive = "placebo")
# Some complex interventions

ints <- c("F + TCA", "F + Plac", "SSRI + Plac + TCA")
netcomplex(nc1, ints)

# Components effects

forest(netcomplex(nc1, nc1$comps))
## S3 method for class 'netmeta'
plot(x, ...)

Arguments

x  An object of class netmeta.

pooled  A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.

reference.group  Reference treatment(s).

baseline.reference  A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa.

labels  An optional vector with treatment labels.

equal.size  A logical indicating whether all squares should be of equal size. Otherwise, the square size is proportional to the precision of estimates.

leftcols  A character vector specifying 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 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).

digits  Minimal number of significant digits for treatment effects and confidence intervals, see print.default.

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

nsim  Number of simulations to calculate SUCRAs.

digits.prop  Minimal number of significant digits for P-scores, SUCRAs and direct evidence proportions, see print.default and netrank.

smlab  A label printed at top of figure. By default, text indicating either common or random effects model is printed.

sortvar  An optional vector used to sort treatments (must be of same length as the total number of treatments).

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.

lab.NA  A character string to label missing values.

add.data  An optional data frame with additional columns to print in forest plot (see Details).
drop.reference.group
A logical indicating whether the reference group should be printed in the forest plot.

col.by
The colour to print information on subgroups.

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

... Additional arguments for forest.meta function.

Details
A forest plot, also called confidence interval plot, is drawn in the active graphics window.

Argument sortvar can be either a numeric or character vector with length of number of treatments. If sortvar is numeric the order function is utilised internally to determine the order of values. If sortvar is character it must be a permutation of the treatment names. It is also possible to provide either sortvar = Pscore, sortvar = "Pscore", sortvar = -Pscore, or sortvar = "-Pscore" in order to sort treatments according to the ranking generated by netrank which is called internally. It is also possible to use "SUCRA" instead of "Pscore". Similar expressions are possible to sort by treatment comparisons (sortvar = TE, etc.), standard error (sortvar = seTE), number of studies with direct treatment comparisons (sortvar = k), and direct evidence proportion (sortvar = prop.direct, see also netmeasures).

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 following columns are available:

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;studlab&quot;</td>
<td>Treatments</td>
</tr>
<tr>
<td>&quot;TE&quot;</td>
<td>Network estimates (either from common or random effects model)</td>
</tr>
<tr>
<td>&quot;seTE&quot;</td>
<td>Corresponding standard errors</td>
</tr>
<tr>
<td>&quot;Pscore&quot;</td>
<td>P-scores (see netrank)</td>
</tr>
<tr>
<td>&quot;SUCRA&quot;</td>
<td>SUCRAs (see netrank)</td>
</tr>
<tr>
<td>&quot;n.trts&quot;</td>
<td>Number of participants per treatment arm</td>
</tr>
<tr>
<td>&quot;k&quot;</td>
<td>Number of studies in pairwise comparisons</td>
</tr>
<tr>
<td>&quot;prop.direct&quot;</td>
<td>Direct evidence proportions (see netmeasures)</td>
</tr>
<tr>
<td>&quot;effect&quot;</td>
<td>(Back-transformed) network estimates</td>
</tr>
<tr>
<td>&quot;ci&quot;</td>
<td>Confidence intervals</td>
</tr>
<tr>
<td>&quot;effect.ci&quot;</td>
<td>(Back-transformed) network estimates and confidence intervals</td>
</tr>
</tbody>
</table>

As a sidenote, the rather odd column name "studlab" to describe the treatment comparisons comes from internally calling forest.meta which uses study labels as the essential information.

Argument add.data can be used to add additional columns to the forest plot. This argument must be a data frame with row names equal to the treatment names in R object x, i.e., x$trts.

See help page of forest.meta for more information on the generation of forest plots and additional arguments.

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>
See Also

forest.meta

Examples

data(smokingcessation)

# Transform data from arm-based format to contrast-based format
#
p1 <- pairwise(list(treat1, treat2, treat3),
  event = list(event1, event2, event3), n = list(n1, n2, n3),
  data = smokingcessation, sm = "OR")

# Conduct random effects network meta-analysis
#
net1 <- netmeta(p1, common = FALSE)

forest(net1)

## Not run:
data(Senn2013)

# Conduct network meta-analysis
#
net2 <- netmeta(TE, seTE, treat1, treat2, studlab,
  data = Senn2013, sm = "MD")

forest(net2, ref = "plac")
forest(net2, xlim = c(-1.5, 1), ref = "plac",
  xlab = "HbA1c difference", rightcols = FALSE)

# Random effects effect model
#
net3 <- netmeta(TE, seTE, treat1, treat2, studlab,
  data = Senn2013, sm = "MD", common = FALSE)

forest(net3, xlim = c(-1.5, 1), ref = "plac",
  xlab = "HbA1c difference")

# Add column with P-Scores on right side of forest plot
#
forest(net3, xlim = c(-1.5, 1), ref = "plac",
  xlab = "HbA1c difference",
  rightcols = c("effect", "ci", "Pscore"),
  just.addcols = "right")

# Add column with P-Scores on left side of forest plot
#
forest(net3, xlim = c(-1.5, 1), ref = "plac",
  xlab = "HbA1c difference",
  leftcols = c("studlab", "Pscore"),
  just.addcols = "right")
# Sort forest plot by descending P-Score

```r
forest(net3, xlim = c(-1.5, 1), ref = "plac",
xlab = "HbA1c difference",
rightcols = c("effect", "ci", "Pscore"),
just.addcols = "right",
sortvar = -Pscore)
```

# Drop reference group and sort by and print number of studies with
# direct treatment comparisons

```r
forest(net3, xlim = c(-1.5, 1), ref = "plac",
xlab = "HbA1c difference",
leftcols = c("studlab", "k"),
leftlabs = c("Contrast\nto Placebo", "Direct\nComparisons"),
sortvar = -k,
drop = TRUE,
smlab = "Random Effects Model")
```

## End(Not run)

---

**forest.netsplit Forest plot for direct and indirect evidence**

### Description

Forest plot to show direct and indirect evidence in network meta-analysis. Furthermore, estimates from network meta-analysis as well as prediction intervals can be printed.

### Usage

```r
## S3 method for class 'netsplit'
forest(
x, pooled = ifelse(x$random, "random", "common"),
show = "both",
subgroup = "comparison",
overall = TRUE,
direct = TRUE,
indirect = TRUE,
prediction = x$prediction,
only.reference = FALSE,
sortvar = NULL,
text.overall = "Network estimate",
text.direct = "Direct estimate",
text.indirect = "Indirect estimate",
text.predict = "Prediction interval",
```

[Go back to the top of the page](#)
Arguments

`x` An object of class `netsplit`.

`pooled` A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.

`show` A character string indicating which comparisons should be printed (see Details).

`subgroup` A character string indicating which layout should be used in forest plot: subgroups by comparisons ("comparison") or subgroups by estimates ("estimate"). Can be abbreviated.

`overall` A logical indicating whether network meta-analysis estimates should be printed.

`direct` A logical indicating whether direct estimates should be printed.

`indirect` A logical indicating whether indirect estimates should be printed.

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

`only.reference` A logical indicating whether only comparisons with the reference group should be printed.

`sortvar` An optional vector used to sort comparisons (must be of same length as the total number of comparisons).

`text.overall` A character string used in the plot to label the network estimates.

`text.direct` A character string used in the plot to label the direct estimates.
text.indirect  A character string used in the plot to label the indirect estimates.
text.predict   A character string used in the plot to label the prediction interval.
type.overall  A character string specifying how to plot treatment effects and confidence intervals for the overall network evidence.
type.direct   A character string specifying how to plot treatment effects and confidence intervals for the direct evidence.
type.indirect A character string specifying how to plot treatment effects and confidence intervals for the indirect evidence.
col.square    The colour for squares.
col.square.lines The colour for the outer lines of squares.
col.inside    The colour for results and confidence limits if confidence limits are completely within squares squares.
col.diamond   The colour of diamonds.
col.diamond.lines The colour of the outer lines of diamonds.
col.predict   Background colour of prediction intervals.
col.predict.lines Colour of outer lines of prediction intervals.
equal.size    A logical indicating whether all squares should be of equal size. Otherwise, the square size is proportional to the precision of estimates.
leftcols      A character vector specifying columns to be plotted on the left side of the forest plot (see Details).
leftlabs      A character vector specifying labels for columns on left side of the forest plot.
rightcols     A character vector specifying columns to be plotted on the right side of the forest plot (see Details).
rightlabs     A character vector specifying labels for columns on right side of the forest plot.
digits        Minimal number of significant digits for treatment effects and confidence intervals, see print.default.
digits.prop   Minimal number of significant digits for the direct evidence proportion.
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.
lab.NA         A character string to label missing values.
smlab         A label printed at top of figure. By default, text indicating either common or random effects model is printed.
... Additional arguments for forest.meta function.
Details

A forest plot, also called confidence interval plot, is drawn in the active graphics window. 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. If argument `rightcols` is `FALSE`, no columns will be plotted on the right side.

If direct estimates are included in the forest plot (direct = `TRUE`, default), the following columns will be printed on the left side of the forest plot: the comparisons (column "studlab" in `forest.meta`), number of pairwise comparisons ("k"), direct evidence proportion ("k"), and I\(^2\) from pairwise comparison ("I2").

If direct estimates are not included in the forest plot (direct = `FALSE`), only the comparisons ("studlab") are printed on the left side of the forest plot.

For more information see help page of `forest.meta` function.

Argument `show` determines which comparisons are printed:

- "all" All comparisons
- "both" Only comparisons contributing both direct and indirect evidence
- "with.direct" Comparisons providing direct evidence
- "direct.only" Comparisons providing only direct evidence
- "indirect.only" Comparisons providing only indirect evidence

Author(s)

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

See Also

`forest.meta`

Examples

data(Senn2013)

# Only consider first five studies (to reduce runtime of example)
# studies <- unique(Senn2013$studlab)
Senn2013.5 <- subset(Senn2013, studlab %in% studies[1:5])

net1 <- netmeta(TE, seTE, treat1.long, treat2.long,
    studlab, data = Senn2013.5, common = FALSE, reference = "plac")

# ns1 <- netsplit(net1)

# Forest plot showing comparisons contributing both direct and
# indirect evidence
# forest(ns1, fontsize = 6, spacing = 0.5, addrow.subgroups = FALSE)

## Not run:
# Forest plot showing comparisons contributing direct evidence
#
forest(ns1, fontsize = 6, spacing = 0.5, addrow.subgroups = FALSE,
    show = "with.direct")

# Forest plot only showing network estimates compared to reference
# group and prediction intervals
#
forest(ns1, fontsize = 8, spacing = 0.75, show = "all",
    only.reference = TRUE, prediction = TRUE,
    direct = FALSE, indirect = FALSE)
## End(Not run)

---

**Network meta-analysis of treatments for Parkinson’s disease**

**Description**

Network meta-analysis comparing the effects of a number of treatments for Parkinson’s disease. The data are the mean lost work-time reduction in patients given dopamine agonists as adjunct therapy in Parkinson’s disease (Franchini et al. 2012). The data are given as sample size, mean and standard deviation in each trial arm. Treatments are placebo and four active drugs. These data are used as an example in the supplemental material of Dias et al. (2013) where placebo is coded as 1 and the four active drugs as 2 to 5.

**Format**

A data frame with the following columns:

<table>
<thead>
<tr>
<th>Study</th>
<th>Treatment1</th>
<th>Treatment2</th>
<th>Treatment3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>study label</td>
<td>treatment 1</td>
<td>treatment 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>treatment effect arm 1</td>
<td>treatment effect arm 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Standard deviation arm 1</td>
<td>Standard deviation arm 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sample size arm 1</td>
<td>Sample size arm 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sd1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sd2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sd3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
funnel.netmeta

Source


See Also

pairwise, metacont, netmeta, netgraph.netmeta

Examples

data(Franchini2012)

    # Transform data from arm-based format to contrast-based format
    #
    p1 <- pairwise(list(Treatment1, Treatment2, Treatment3),
        n = list(n1, n2, n3),
        mean = list(y1, y2, y3),
        sd = list(sd1, sd2, sd3),
        data = Franchini2012, studlab = Study)
    p1

    # Conduct network meta-analysis
    net1 <- netmeta(p1)
    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,
        thickness = "se.common",
        iterate = TRUE, plastic = TRUE)
    netgraph(net1, points = TRUE, cex.points = 3, cex = 1.5,
        thickness = "se.common",
        iterate = TRUE, plastic = TRUE, start = "eigen")

Description

Draw a ‘comparison-adjusted’ funnel plot to assess funnel plot asymmetry in network meta-analysis.
funnel.netmeta

Usage

```r
## S3 method for class 'netmeta'
funnel(
  x,
  order,
  pooled = ifelse(x$random, "random", "common"),
  xlab = NULL,
  level = x$level,
  pch,
  col = "black",
  legend = TRUE,
  pos.legend = "topright",
  pos.tests = "topleft",
  lump.comparator = FALSE,
  text.comparator = "comparator",
  method.bias,
  text.linreg = "(Egger)",
  text.rank = "(Begg-Mazumdar)",
  text.mm = "(Thompson-Sharp)",
  sep.trts = x$sep.trts,
  nchar.trts = x$nchar.trts,
  backtransf = x$backtransf,
  digits.pval = gs("digits.pval"),
  warn.deprecated = gs("warn.deprecated"),
  linreg = FALSE,
  rank = FALSE,
  mm = FALSE,
  ...
)
```

Arguments

- `x` An object of class `netmeta`.
- `order` A mandatory character or numerical vector specifying the order of treatments or list of comparators (see Details).
- `pooled` A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.
- `xlab` A label for the x-axis.
- `level` The confidence level utilised in the plot. For the funnel plot, confidence limits are not drawn if `yaxis = "size"`.
- `pch` The plotting symbol(s) used for individual studies within direct comparisons.
- `col` The colour(s) used for individual studies within direct comparisons.
- `legend` A logical indicating whether a legend with information on direct comparisons should be added to the plot.
- `pos.legend` The position of the legend describing plotting symbols and colours for direct comparisons.
funnel.netmeta

pos.tests The position of results for test(s) of funnel plot asymmetry.
lump.comparator A logical indicating whether comparators should be lumped, e.g., to specify inactive treatments. Information on direct comparisons should be added to the plot.
text.comparator A character string used in the plot to label the comparator if lump.comparator is TRUE.
method.bias A character vector indicating which test(s) for funnel plot asymmetry to use. Admissible values are "Begg", "Egger", and "Thompson", can be abbreviated. See function metabias.
text.linreg A character string used in the plot to label the Egger test for funnel plot asymmetry.
text.rank A character string used in the plot to label the Begg test for funnel plot asymmetry.
text.mm A character string used in the plot to label the Thompson-Sharp test for funnel plot asymmetry.
sep.trts A character used in comparison names as separator between treatment labels.
nchar.trts A numeric defining the minimum number of characters used to create unique treatment names (see netmeta).
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.
digits.pval Minimal number of significant digits for p-value of test(s) for funnel plot asymmetry.
warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments are used.
linreg Deprecated argument (replaced by method.bias).
rank Deprecated argument (replaced by method.bias).
mm Deprecated argument (replaced by method.bias).
... Additional graphical arguments passed as arguments to funnel.meta.

Details

A 'comparison-adjusted' funnel plot (Chaimani & Salanti, 2012) is drawn in the active graphics window.

Argument order is mandatory to determine the order of treatments (Chaimani et al., 2013):

"Before using this plot, investigators should order the treatments in a meaningful way and make assumptions about how small studies differ from large ones. For example, if they anticipate that newer treatments are favored in small trials, then they could name the treatments from oldest to newest so that all comparisons refer to 'old versus new intervention'. Other possibilities include
defining the comparisons so that all refer to an active treatment versus placebo or sponsored versus non-sponsored intervention."

Alternatively, it is possible to only provide a single or few treatment name(s) in argument order to define the comparator(s). In this case only comparisons with this / these treatment(s) will be considered. If argument lump.comparator is TRUE, all comparators will be lumped into a single group. The text for this group can be specified with argument text.comparator.

In the funnel plot, if yaxis is "se", the standard error of the treatment estimates is plotted on the y-axis which is likely to be the best choice (Sterne & Egger, 2001). Other possible choices for yaxis are "invvar" (inverse of the variance), "invse" (inverse of the standard error), and "size" (study size).

Value

A data frame with the following columns:

- studlab: Study label.
- treat1: Label/Number for first treatment.
- treat2: Label/Number for second treatment.
- comparison: Treatment comparison.
- TE: Estimate of treatment effect, e.g., log odds ratio.
- TE.direct: Pooled estimate from direct evidence.
- TE.adj: ‘Comparison-adjusted’ treatment effect (TE - TE.direct).
- seTE: Standard error of treatment estimate.
- pch: Plotting symbol(s).
- col: Colour of plotting symbol(s).

Author(s)

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

References


See Also

netmeta, funnel.meta, metabias
funnel.netmeta

Examples

## Not run:
data(Senn2013)

net1 <- netmeta(TE, seTE, treat1, treat2, studlab,
data = Senn2013, sm = "MD")

# 'Comparison-adjusted' funnel plot not created as argument 'order'
# is missing
# try(funnel(net1))

# Only show comparisons with placebo
# funnel(net1, order = "pl")

# Add result for Egger test of funnel plot asymmetry
# funnel(net1, order = "pl", method.bias = "Egger",
digits.pval = 2)

# (Non-sensical) alphabetic order of treatments with placebo as
# last treatment
# ord <- c("a", "b", "me", "mi", "pi", "r", "si", "su", "v", "pl")
funnel(net1, order = ord)

# Add results for tests of funnel plot asymmetry and use different
# plotting symbols and colours
# funnel(net1, order = ord,
pch = rep(c(15:18, 1), 3), col = 1:3,
method.bias = c("Egger", "Begg", "Thompson"), digits.pval = 2)

# Same results for tests of funnel plot asymmetry using reversed
# order of treatments
# funnel(net1, order = rev(ord),
pch = rep(c(15:18, 1), 3), col = 1:3,
method.bias = c("Egger", "Begg", "Thompson"), digits.pval = 2)

# Calculate tests for funnel plot asymmetry
# f1 <- funnel(net1, order = ord)
# metabias(metagen(TE.adj, seTE, data = f1))
metabias(metagen(TE.adj, seTE, data = f1), method = "Begg")
metabias(metagen(TE.adj, seTE, data = f1), method = "Thompson")

## End(Not run)
Network meta-analysis comparing the effects of a number of interventions for decreasing blood loss and blood transfusion requirements during liver transplantation.

Format

A data frame with the following columns:

- **study**: study information (first author, year)
- **treatment**: treatment
- **death**: mortality at 60 days post-transplantation
- **n**: number of individuals in treatment arm

Source


See Also

- pairwise
- metabin
- netmetabin

Examples

data(Gurusamy2011)

# Only consider three studies (to reduce runtime of example)
#
three <- subset(Gurusamy2011, study %in% studies)

# 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.
#
# p1 <- pairwise(treatment, death, n, studlab = study,
# data = three, sm = "OR")

# Conduct Mantel-Haenszel network meta-analysis
#
netmetabin(p1, ref = "cont")

## Not run:
p2 <- pairwise(treatment, death, n, studlab = study, data = Gurusamy2011, sm = "OR")

# Conduct Mantel-Haenszel network meta-analysis
netmetabin(p2, ref = "cont")

## End(Not run)

---

**Description**

This function generates a Hasse diagram for a partial order of treatment ranks in a network meta-analysis.

**Usage**

```r
dx <- ifelse(dx$random, "random", "common")
dxnewpage = TRUE

hasse(x, pooled = dx, dxnewpage = TRUE)
```

**Arguments**

- `x`: An object of class `netposet` (mandatory).
- `pooled`: A character string indicating whether Hasse diagram should be drawn for common ("common") or random effects model ("random"). Can be abbreviated.
- `newpage`: A logical value indicating whether a new figure should be printed in an existing graphics window. Otherwise, the Hasse diagram is added to the existing figure.

**Details**

Generate a Hasse diagram (Carlsen & Bruggemann, 2014) for a partial order of treatment ranks in a network meta-analysis (Rücker & Schwarzer, 2017).

This R function is a wrapper function for R function `hasse` in R package `hasseDiagram` (Krzysztof Ciomek, https://github.com/kciomek/hasseDiagram), i.e., function `hasse` can only be used if R package `hasseDiagram` is installed.

**Author(s)**

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

**References**


See Also

netmeta, netposet, netrank, plot.netrank

Examples

## Not run:
# Use depression dataset
#
data(Linde2015)

# Define order of treatments
#
trts <- c("TCA", "SSRI", "SNRI", "NRI",
  "Low-dose SARI", "NaSSa", "rMAO-A", "Hypericum", "Placebo")

# Outcome labels
#
outcomes <- c("Early response", "Early remission")

# (1) Early response
#
p1 <- pairwise(treat = list(treatment1, treatment2, treatment3),
  event = list(resp1, resp2, resp3),
  n = list(n1, n2, n3),
  studlab = id, data = Linde2015, sm = "OR")
#
net1 <- netmeta(p1, common = FALSE,
  seq = trts, ref = "Placebo", small.values = "undesirable")

# (2) Early remission
#
p2 <- pairwise(treat = list(treatment1, treatment2, treatment3),
  event = list(remi1, remi2, remi3),
  n = list(n1, n2, n3),
  studlab = id, data = Linde2015, sm = "OR")
#
net2 <- netmeta(p2, common = FALSE,
  seq = trts, ref = "Placebo", small.values = "undesirable")

# Partial order of treatment rankings
#
po <- netposet(netrank(net1), netrank(net2), outcomes = outcomes)

# Hasse diagram
#
hasse(po)

## End(Not run)
hatmatrix

Derive hat matrix from network meta-analysis

Description

Auxiliary function to derive hat matrix from network meta-analysis

Usage

hatmatrix(
  x,
  method = "Ruecker",
  type,
  common = x$common,
  random = x$random,
  nchar.trts = x$nchar.trts,
  nchar.studlab = x$nchar.studlab
)

## S3 method for class 'hatmatrix'
print(
  x,
  common = x$x$common,
  random = x$x$random,
  nchar.trts = x$x$nchar.trts,
  nchar.studlab = x$x$nchar.studlab,
  digits = gs("digits"),
  legend = TRUE,
  legend.studlab = TRUE,
  ...
)

Arguments

x A netmeta object.

method A character string indicating which method is used to derive the hat matrix. Either "Ruecker", "Krahn" or "Davies" (can be abbreviated, see Details).

type A character string indicating which specific hat matrix should be derived (can be abbreviated, see Details).

common A logical indicating whether a hat matrix should be printed for the common effects network meta-analysis.

random A logical indicating whether a hat matrix should be printed for the random effects network meta-analysis.

nchar.trts A numeric defining the minimum number of characters used to create unique treatment names.
nchar.studlab  A numeric defining the minimum number of characters used to create unique study labels.
digits      Minimal number of significant digits, see `print.default`.
legend      A logical indicating whether a legend should be printed.
legend.studlab  A logical indicating whether a legend should be printed for abbreviated study labels.
...         Additional arguments (ignored).

Details

This auxiliary function can be used to derive various hat matrices from a network meta-analysis object.

**Hat matrix by Rücker (2012):**

This hat matrix is estimated if `method = "Ruecker"`.  
Let $n$ be the number of different treatments (nodes, vertices) in a network and let $m$ be the number of existing comparisons (edges) between the treatments. If there are only two-arm studies, $m$ is equal to the number of studies, $k$. Let `seTE.adj.common` and `seTE.adj.random` be the vectors of adjusted standard errors under the common and random effects model (see `netmeta`). Let $W$ be the $m \times m$ diagonal matrix that contains the inverse variance $1 / seTE.adj.common^2$ or $1 / seTE.adj.random^2$.

The given comparisons define the network structure. Therefrom an $m \times n$ design matrix $X$ (edge-vertex incidence matrix) is formed; for more precise information, see Rücker (2012). Moreover, the $n \times n$ Laplacian matrix $L$ and its Moore-Penrose pseudoinverse $L^+$ are calculated (both matrices play an important role in graph theory and electrical network theory). Using these matrices, the variances based on both direct and indirect comparisons can be estimated. The hat matrix $H$ is estimated by $H = XL^+X^TW = X(X^TWX)^+X^TW$.

**Hat matrices by Krahn et al. (2013):**

One of the following hat matrices is estimated if `method = "Krahn"`.  
Use of `type = "design"` (default) results in a hat matrix of dimension $n(n-1)/2 \times s$, where $s$ is the sum of the number of independent comparisons from each design.  
Use of `type = "studies"` results in a hat matrix of dimension $n(n-1)/2 \times l$, where $l$ is the number of independent pairwise comparisons, i.e., a three-arm study contributes two pairwise comparisons.

**Hat matrices by Davies et al. (2022):**

One of three hat matrices is estimated if `method = "Davies"`.  
Here, we focus on the hat matrix of the aggregate (two-step) version of the graph theoretical NMA model. In the first step, a pairwise meta-analysis is performed across each edge using the adjusted weights (these account for correlations due to multi-arm trials). From this we obtain direct treatment effect estimates (and corresponding aggregate weights) associated with each edge. In step two, we combine these direct estimates in a network meta analysis to obtain the network estimates. This is done using weighted least squares regression. The hat matrix associated with this second step is called the aggregate hat matrix.

All three versions of the aggregate hat matrix contain the same information: the second two can be derived directly from the first. They differ in their dimensionality.
Each row of the hat matrix that represents a treatment comparison \((ij)\) describes the flow of evidence through each edge for that comparison. This defines a directed acyclic 'flow graph' from node \(i\) to node \(j\).

1. Use of \texttt{type = "short"} (default) results in a hat matrix of dimension \(e \times e\), where \(e\) is the number of (unique) edges (direct comparisons) in the network. This is the aggregate hat matrix described in Davies et al. (2022). Each row and column represents a pair of treatments for which there is at least one direct comparison.

2. Use of \texttt{type = "long"} results in a hat matrix of dimension \(n(n-1)/2 \times e\). There is a row for every possible pair of treatments in the network - regardless of whether there is direct evidence for this comparison. Each column represents a pair of treatments for which there is at least one direct comparison. The extra rows can be calculated from the short hat matrix using consistency equations.

3. Use of \texttt{type = "full"} results in a hat matrix of dimension \(n(n-1)/2 \times n(n-1)/2\). In comparison to the long hat matrix, columns of zeroes are added for comparisons that do not have any direct evidence. Therefore, there is a row and column for every pair of treatments in the network. This hat matrix is used to calculate the transition matrices for the random walk in \texttt{netcontrib}.

**Value**

A list with two hat matrices: \texttt{common} (common effects model) and \texttt{random} (random effects model).

**Author(s)**

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

**References**


**See Also**

\texttt{netmeta, netcontrib, netheat}

**Examples**

```r
data(Dong2013)
# Only consider first ten studies for concise output
first10 <- subset(Dong2013, id <= 10)
p1 <- pairwise(treatment, death, randomized, studlab = id,
               data = first10, sm = "OR")
net1 <- netmeta(p1, common = FALSE)
hatmatrix(net1)
hatmatrix(net1, method = "k")
```
heatplot

Generic function for heat plots

Description
Generic function for heat plots

Usage
heatplot(x, ...)

Arguments
x
An R object.

...
Additional arguments.

Details
For more details, look at the following function to generate heat plots:

- heatplot.netmeta
- heatplot.crossnma (if installed)

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

Examples
data(Senn2013)

# Only consider first five studies (to reduce runtime of example)
# studies <- unique(Senn2013$studlab)
Senn2013.5 <- subset(Senn2013, studlab %in% studies[1:5])

# Conduct random effects network meta-analysis with
# placebo as reference treatment
# net1 <- netmeta(TE, seTE, treat1.long, treat2.long, studlab,
# data = Senn2013.5, sm = "MD", common = FALSE, reference = "plac")

# Generate a heat plot (with abbreviated treatment labels)
heatplot.netmeta

#
heatplot(net1, nchar.trts = 4)

heatplot.netmeta  Heat Plot

Description

Produces a heat plot containing treatment estimates with confidence intervals for all possible pair-wise comparisons.

Usage

## S3 method for class 'netmeta'
heatplot(
  x,
  pooled = ifelse(x$random, "random", "common"),
  seq = x$seq,
  nchar.trts = x$nchar.trts,
  low.colour = "red",
  mid.colour = "white",
  high.colour = "springgreen4",
  size = 6,
  size.trt = 16,
  size.axis = 12,
  digits = gs("digits.forest"),
  backtransf = x$backtransf,
  ...
)

Arguments

x  An object of class netmeta.
pooled  A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.
seq  A character or numerical vector specifying the sequence of treatments in rows and columns of the heat plot.
nchar.trts  A numeric defining the minimum number of characters used to create unique treatment names.
low.colour  A string indicating the colour of low relative treatment effects for the heat plot (e.g odds ratio of ~0.5)
mid.colour  A string indicating the colour of null relative treatment effects for the heat plot (e.g odds ratio of ~1.0).
high.colour  A string indicating the colour of high relative treatment effects for the heat plot (e.g odds ratio of ~2.0).
The size of cell entries with the relative treatment effect and confidence intervals.

size.trt

The size of treatment names placed on the top and left of the plot.

size.axis

The size of labels on the top and left of the plot

digits

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

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.

Additional arguments.

Value

League heat plot, where a color scale is used to represent the values of relative treatment effects.

Examples

data(Senn2013)

# Only consider first five studies (to reduce runtime of example)
# studies <- unique(Senn2013$studlab)
Senn2013.5 <- subset(Senn2013, studlab %in% studies[1:5])

# Conduct random effects network meta-analysis with
# placebo as reference treatment
# net1 <- netmeta(TE, seTE, treat1.long, treat2.long, studlab,
# data = Senn2013.5, sm = "MD", common = FALSE, reference = "plac")

# Generate a heat plot (with abbreviated treatment labels)
# heatmap(net1, nchar.trts = 4)

---

**invmat**

*Moore-Penrose Pseudoinverse of a Matrix*

Description

Calculates the Moore-Penrose pseudoinverse of a square matrix X.

Usage

`invmat(X)`

Arguments

X

A square matrix.
Details

This function is used by default in R package netmeta to calculate the Moore-Penrose pseudoinverse \( L^+ \) of the Laplacian matrix \( L \) (Rücker, 2012):
\[
L^+ = (X - J / n)^{-1} + J / n
\]
with identity matrix \( J \) of dimension \( nxn \).

Value

The Moore-Penrose pseudoinverse for matrix \( X \).

Author(s)

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

References


See Also

netmeta, solve

Examples

data(smokingcessation)

pl <- pairwise(list(treat1, treat2, treat3),
  event = list(event1, event2, event3), n = list(n1, n2, n3),
  data = smokingcessation, sm = "OR")
net1 <- netmeta(pl)

invmat(net1$L.matrix.common)

## Not run:
data(Senn2013)

net2 <- netmeta(TE, seTE, treat1.long, treat2.long, studlab,
  data = Senn2013)
L1 <- net2$L.matrix.common
L2 <- invmat(net2$Lplus.matrix.common)
all.equal(round(L1, 10), round(L2, 10))

## End(Not run)
Description

Network meta-analysis of nine classes of antidepressants including placebo for the primary care setting; partly shown in Linde et al. (2015), supplementary Table 2.

Format

A data frame with the following columns:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Study ID</td>
</tr>
<tr>
<td>author</td>
<td>First author</td>
</tr>
<tr>
<td>year</td>
<td>Publication year</td>
</tr>
<tr>
<td>treatment1</td>
<td>First treatment</td>
</tr>
<tr>
<td>treatment2</td>
<td>Second treatment</td>
</tr>
<tr>
<td>treatment3</td>
<td>Third treatment</td>
</tr>
<tr>
<td>n1</td>
<td>Number of patients receiving first treatment</td>
</tr>
<tr>
<td>resp1</td>
<td>Number of early responder (treatment 1)</td>
</tr>
<tr>
<td>remi1</td>
<td>Number of early remissions (treatment 1)</td>
</tr>
<tr>
<td>loss1</td>
<td>Number of patients loss to follow-up (treatment 1)</td>
</tr>
<tr>
<td>loss.ae1</td>
<td>Number of patients loss to follow-up due to adverse events (treatment 1)</td>
</tr>
<tr>
<td>ae1</td>
<td>Number of patients with adverse events (treatment 1)</td>
</tr>
<tr>
<td>n2</td>
<td>Number of patients receiving second treatment</td>
</tr>
<tr>
<td>resp2</td>
<td>Number of early responder (treatment 2)</td>
</tr>
<tr>
<td>remi2</td>
<td>Number of early remissions (treatment 2)</td>
</tr>
<tr>
<td>loss2</td>
<td>Number of patients loss to follow-up (treatment 2)</td>
</tr>
<tr>
<td>loss.ae2</td>
<td>Number of patients loss to follow-up due to adverse events (treatment 2)</td>
</tr>
<tr>
<td>ae2</td>
<td>Number of patients with adverse events (treatment 2)</td>
</tr>
<tr>
<td>n3</td>
<td>Number of patients receiving third treatment</td>
</tr>
<tr>
<td>resp3</td>
<td>Number of early responder (treatment 3)</td>
</tr>
<tr>
<td>remi3</td>
<td>Number of early remissions (treatment 3)</td>
</tr>
<tr>
<td>loss3</td>
<td>Number of patients loss to follow-up (treatment 3)</td>
</tr>
<tr>
<td>loss.ae3</td>
<td>Number of patients loss to follow-up due to adverse events (treatment 3)</td>
</tr>
<tr>
<td>ae3</td>
<td>Number of patients with adverse events (treatment 3)</td>
</tr>
</tbody>
</table>

Source


See Also

pairwise, metabin, netmeta, netposet
Examples

data(Linde2015)

# Transform data from arm-based format to contrast-based format
# Outcome: early response
p1 <- pairwise(list(treatment1, treatment2, treatment3),
               event = list(resp1, resp2, resp3),
               n = list(n1, n2, n3),
               studlab = id, data = Linde2015, sm = "OR")
p1

## Not run:
# Define order of treatments
trts <- c("TCA", "SSRI", "SNRI", "NRI", "Low-dose SARI",
          "NaSSa", "rMAO-A", "Hypericum", "Placebo")

# Conduct random effects network meta-analysis
net1 <- netmeta(p1, common = FALSE, reference = "Placebo", seq = trts)
print(net1, digits = 2)

## End(Not run)

---

Network meta-analysis of primary care depression treatments

Description

Network meta-analysis of 22 treatments (including placebo and usual care) for the primary care of depression.

Format

A data frame with the following columns:

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Study ID</td>
</tr>
<tr>
<td>author</td>
<td>First author</td>
</tr>
<tr>
<td>year</td>
<td>Year of publication</td>
</tr>
<tr>
<td>treat1</td>
<td>First treatment (abbreviated)</td>
</tr>
<tr>
<td>treat2</td>
<td>Second treatment (abbreviated)</td>
</tr>
<tr>
<td>treat1.long</td>
<td>First treatment</td>
</tr>
<tr>
<td>treat2.long</td>
<td>Second treatment</td>
</tr>
<tr>
<td>lnOR</td>
<td>Response after treatment (log odds ratio)</td>
</tr>
<tr>
<td>selnOR</td>
<td>Standard error of log odds ratio</td>
</tr>
<tr>
<td>resp1</td>
<td>Responder (first treatment)</td>
</tr>
<tr>
<td>n1</td>
<td>Sample size (first treatment)</td>
</tr>
<tr>
<td>resp2</td>
<td>Responder (second treatment)</td>
</tr>
<tr>
<td>n2</td>
<td>Sample size (second treatment)</td>
</tr>
</tbody>
</table>
Source


See Also

netmeta, netcomb

Examples

data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce
# runtime of example)
#
face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
#
net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
  data = face, reference.group = "placebo",
  sm = "OR", common = FALSE, nchar.trts = 6)
#
net1

## Not run:
# Conduct random effects network meta-analysis
#
net2 <- netmeta(lnOR, selnOR, treat1, treat2, id,
  data = Linde2016, reference.group = "placebo",
  sm = "OR", common = FALSE, nchar.trts = 6)
#
net2

## End(Not run)

metabias.netmeta  Test of funnel plot asymmetry in network meta-analysis

Description

Test of funnel plot asymmetry in network meta-analysis

Usage

## S3 method for class 'netmeta'
metabias(
  x,
  

---
order, 
pooled = ifelse(x$random, "random", "common"), 
method.bias = "Egger", 
lump.comparator = FALSE, 
... 
)

Arguments

x
An object of class netmeta.

order
A mandatory character or numerical vector specifying the order of treatments or list of comparators (see Details).

pooled
A character string indicating whether results for the common ("common") or random effects model ("random") should be used in test of funnel plot asymmetry. Can be abbreviated.

method.bias
A character vector indicating which test(s) for funnel plot asymmetry to use. Admissible values are "Begg", "Egger", and "Thompson", can be abbreviated. See function metabias.meta.

lump.comparator
A logical indicating whether comparators should be lumped, e.g., to specify inactive treatments. Information on direct comparisons should be added to the plot.

... Additional arguments (passed on to metabias.meta).

Details

Test of funnel plot asymmetry in network meta-analysis

Argument order is mandatory to determine the order of treatments (Chaimani et al., 2013):

"[...] investigators should order the treatments in a meaningful way and make assumptions about how small studies differ from large ones. For example, if they anticipate that newer treatments are favored in small trials, then they could name the treatments from oldest to newest so that all comparisons refer to ‘old versus new intervention’. Other possibilities include defining the comparisons so that all refer to an active treatment versus placebo or sponsored versus non-sponsored intervention."

Alternatively, it is possible to only provide a single or few treatment name(s) in argument order to define the comparator(s). In this case only comparisons with this / these treatment(s) will be considered. If argument lump.comparator is TRUE, all comparators will be lumped into a single group.

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 statistic.
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 Network meta-analysis object.
version Version of R package meta used to create object.
version.netmeta Version of R package netmeta 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 comparisons.
k.min Minimum number of comparisons to perform test for funnel plot asymmetry.
version Version of R package meta used to create object.
version.netmeta Version of R package netmeta used to create object.

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References

See Also
netmeta, funnel.netmeta, metabias

Examples
## Not run:
data(Senn2013)
net1 <- netmeta(TE, seTE, treat1, treat2, studlab,
data = Senn2013, sm = "MD")

# Test for asymmetry in 'comparison-adjusted' funnel plot not
# conducted as argument 'order' is missing
netbind

# Test for funnel plot asymmetry comparing active treatments with # placebo
metabias(net1, order = "pl")

# Rank test
# metabias(net1, order = "pl", method.bias = "Begg")

ord <- c("a", "b", "me", "mi", "pi", "r", "si", "su", "v", "pl")
metabias(net1, order = ord)

## End(Not run)

---

**netbind**

*Combine network meta-analysis objects*

**Description**

This function can be used to combine network meta-analysis objects which is especially useful to generate a forest plot with results of several network meta-analyses.

**Usage**

```r
netbind(
    ..., 
    name, 
    common, 
    random, 
    col.study = "black", 
    col.inside = "white", 
    col.square = "gray", 
    col.square.lines = col.square, 
    backtransf, 
    reference.group, 
    baseline.reference, 
    warn.deprecated = gs("warn.deprecated"), 
    fixed, 
    comb.fixed, 
    comb.random
)
```
Arguments

... Any number of network meta-analysis objects or a single list with network meta-analyses.

name An optional character vector providing descriptive names for network meta-analysis objects.

common A logical indicating whether results for the common effects model should be reported.

random A logical indicating whether results for the random effects model should be reported.

col.study The colour for network estimates and confidence limits.

col.inside The colour for network estimates and confidence limits if confidence limits are completely within squares.

col.square The colour for squares.

col.square.lines The colour for the outer lines of squares.

backtransf A logical indicating whether results should be back transformed. If backtransf = TRUE (default), results for sm = "OR" are printed as odds ratios rather than log odds ratios, for example.


baseline.reference A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa. This argument is only considered if reference.group has been specified.

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

fixed Deprecated argument; replaced by common.

comb.fixed Deprecated argument; replaced by common.

comb.random Deprecated argument; replaced by random.

Value

An object of class "netbind" with corresponding forest function. The object is a list containing the following components:

common A data frame with results for the common effects model.

random A data frame with results for the random effects model.

sm Summary measure used in network meta-analyses.

level.ma Level for confidence intervals.

reference.group, baseline.reference As defined above.
netcomb

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also
netmeta, netcomb, discomb, forest.netbind

Examples

data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce
# runtime of example)
face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Standard random effects NMA model (with placebo as reference
# treatment)
net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
data = face, reference.group = "placebo",
sm = "OR", common = FALSE)

# Additive CNMA model with placebo as inactive component and
# reference
nc1 <- netcomb(net1, inactive = "placebo")

# Combine results of standard NMA and CNMA
nb1 <- netbind(nc1, net1,
name = c("Additive CNMA", "Standard NMA"),
col.study = c("red", "black"), col.square = c("red", "black"))
forest(nb1,
col.by = "black", addrow.subgroups = FALSE,
fontsize = 10, spacing = 0.7, squaresize = 0.9,
label.left = "Favours Placebo",
label.right = "Favours other")

Description

Some treatments in a network meta-analysis may be combinations of other treatments or have common components. The influence of individual components can be evaluated in an additive network meta-analysis model assuming that the effect of treatment combinations is the sum of the effects of its components. This function implements this additive model in a frequentist way.
Usage

netcomb(
  x,
  inactive = NULL,
  sep.comps = "*",
  C.matrix,
  common = x$common,
  random = x$random | !is.null(tau.preset),
  tau.preset = NULL,
  details.chkident = FALSE,
  nchar.comps = x$nchar.trts,
  func.inverse = invmat,
  backtransf = x$backtransf,
  na.unident = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

Arguments

x An object of class netmeta.

inactive A character string defining the inactive treatment component (see Details).

sep.comps A single character to define separator between treatment components.

C.matrix C matrix (see Details).

common A logical indicating whether a common effects network meta-analysis should be conducted.

random A logical indicating whether a random effects network meta-analysis should be conducted.

tau.preset An optional value for the square-root of the between-study variance $\tau^2$.

details.chkident A logical indicating whether details on unidentifiable components should be printed.

nchar.comps A numeric defining the minimum number of characters used to create unique names for components (see Details).

func.inverse R function used to calculate the pseudoinverse of the Laplacian matrix L (see netmeta).

backtransf A logical indicating whether results should be back transformed in printouts and forest plots.

na.unident A logical indicating whether unidentifiable components and combinations should be set to missing values.

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

... Additional arguments (to catch deprecated arguments).
Details

Treatments in network meta-analysis (NMA) can be complex interventions. Some treatments may be combinations of others or have common components. The standard analysis provided by netmeta is a NMA where all existing (single or combined) treatments are considered as different nodes in the network. Exploiting the fact that some treatments are combinations of common components, an additive component network meta-analysis (CNMA) model can be used to evaluate the influence of individual components. This model assumes that the effect of a treatment combination is the sum of the effects of its components which implies that common components cancel out in comparisons.

The additive CNMA model has been implemented using Bayesian methods (Mills et al., 2012; Welton et al., 2013). This function implements the additive model in a frequentist way (Rücker et al., 2020).

The underlying multivariate model is given by

\[ \delta = B \theta, \theta = C \beta \]

with

- \( \delta \) vector of true treatment effects (differences) from individual studies,
- \( B \) design matrix describing the structure of the network,
- \( \theta \) parameter vector that represents the existing combined treatments,
- \( C \) matrix describing how the treatments are composed,
- \( \beta \) parameter vector representing the treatment components.

All parameters are estimated using weighted least squares regression.

Argument `inactive` can be used to specify a single component that does not have any therapeutic value. Accordingly, it is assumed that the treatment effect of the combination of this component with an additional treatment component is equal to the treatment effect of the additional component alone.

Argument `sep.comps` can be used to specify the separator between individual components. By default, the matrix \( C \) is calculated internally from treatment names. However, it is possible to specify a different matrix using argument `C.matrix`.

By default, component names are not abbreviated in printouts. However, in order to get more concise printouts, argument `nchar.comps` can be used to define the minimum number of characters for abbreviated component names (see `abbreviate`, argument `minlength`). R function `treats` is utilised internally to create abbreviated component names.

Value

An object of class `netcomb` with corresponding `print`, `summary`, and `forest` functions. The object is a list containing the following components:

- `studlab` Study labels.
- `treat1` Label/Number for first treatment.
- `treat2` Label/Number for second treatment.
- `TE` Estimate of treatment effect, i.e. difference between first and second treatment.
seTE \text{Standard error of treatment estimate.}
seTE\.adj\.common, seTE\.adj\.random
\text{Standard error of treatment estimate, adjusted for multi-arm studies.}
design \text{Design of study providing pairwise comparison.}
event1 \text{Number of events in first treatment group.}
event2 \text{Number of events in second treatment group.}
n1 \text{Number of observations in first treatment group.}
n2 \text{Number of observations in second treatment group.}
k \text{Total number of studies.}
m \text{Total number of pairwise comparisons.}
n \text{Total number of treatments.}
d \text{Total number of designs (corresponding to the unique set of treatments compared within studies).}
c \text{Total number of components.}
trts \text{Treatments included in network meta-analysis.}
k\.trts \text{Number of studies evaluating a treatment.}
n\.trts \text{Number of observations receiving a treatment (if arguments n1 and n2 are provided).}
events\.trts \text{Number of events observed for a treatment (if arguments event1 and event2 are provided).}
studies \text{Study labels coerced into a factor with its levels sorted alphabetically.}
narms \text{Number of arms for each study.}
designs \text{Unique list of designs present in the network. A design corresponds to the set of treatments compared within a study.}
comps \text{Unique list of components present in the network.}
TE\.nma\.common, TE\.nma\.random
\text{A vector of length } m \text{ of consistent treatment effects estimated by network meta-analysis (nma) (common and random effects model).}
seTE\.nma\.common, seTE\.nma\.random
\text{A vector of length } m \text{ of effective standard errors estimated by network meta-analysis (common and random effects model).}
lower\.nma\.common, lower\.nma\.random
\text{A vector of length } m \text{ of lower confidence interval limits for consistent treatment effects estimated by network meta-analysis (common and random effects model).}
upper\.nma\.common, upper\.nma\.random
\text{A vector of length } m \text{ of upper confidence interval limits for the consistent treatment effects estimated by network meta-analysis (common and random effects model).}
statistic\.nma\.common, statistic\.nma\.random
\text{A vector of length } m \text{ of z-values for test of treatment effect for individual comparisons (common and random effects model).}
pval.nma.common, pval.nma.random
A vector of length $m$ of p-values for test of treatment effect for individual comparisons (common and random effects model).

TE.cnma.common, TE.cnma.random
A vector of length $m$ of consistent treatment effects estimated by the additive (common and random effects) model.

seTE.cnma.common, seTE.cnma.random
A vector of length $m$ with standard errors estimated by the additive (common and random effects) model.

lower.cnma.common, lower.cnma.random
A vector of length $m$ of lower confidence interval limits for consistent treatment effects estimated by the additive (common and random effects) model.

upper.cnma.common, upper.cnma.random
A vector of length $m$ of upper confidence interval limits for consistent treatment effects estimated by the additive (common and random effects) model.

statistic.cnma.common, statistic.cnma.random
A vector of length $m$ of $z$-values for the test of an overall effect estimated by the additive (common and random effects) model.

pval.cnma.common, pval.cnma.random
A vector of length $m$ of p-values for the test of an overall effect estimated by the additive (common and random effects) model.

TE.common, TE.random
$n \times n$ matrix with overall treatment effects estimated by the additive (common and random effects) model.

seTE.common, seTE.random
$n \times n$ matrix with standard errors estimated by the additive (common and random effects) model.

lower.common, upper.common, lower.random, upper.random
$n \times n$ matrices with lower and upper confidence interval limits estimated by the additive (common and random effects) model.

statistic.common, pval.common, statistic.random, pval.random
$n \times n$ matrices with $z$-values and p-values for test of overall effect estimated by the additive (common and random effects) model.

Comp.common, Comp.random
A vector of component effects (common and random effects model).

seComp.common, seComp.random
A vector with corresponding standard errors (common and random effects model).

lower.Comp.common, lower.Comp.random
A vector with lower confidence limits for components (common and random effects model).

upper.Comp.common, upper.Comp.random
A vector with upper confidence limits for components (common and random effects model).

statistic.Comp.common, statistic.Comp.random
A vector with $z$-values for the overall effect of components (common and random effects model).
A vector with p-values for the overall effect of components (common and random effects model).

Comb.common, Comb.random
A vector of combination effects (common and random effects model).

seComb.common, seComb.random
A vector with corresponding standard errors (common and random effects model).

lower.Comb.common, lower.Comb.random
A vector with lower confidence limits for combinations (common and random effects model).

upper.Comb.common, upper.Comb.random
A vector with upper confidence limits for combinations (common and random effects model).

statistic.Comb.common, statistic.Comb.random
A vector with z-values for the overall effect of combinations (common and random effects model).

pval.Comb.common, pval.Comb.random
A vector with p-values for the overall effect of combinations (common and random effects model).

Q.additive
Overall heterogeneity / inconsistency statistic (additive model).

df.Q.additive
Degrees of freedom for test of heterogeneity / inconsistency (additive model).

pval.Q.additive
P-value for test of heterogeneity / inconsistency (additive model).

I2, lower.I2, upper.I2
I-squared, lower and upper confidence limits.

Q.standard
Overall heterogeneity / inconsistency statistic (standard model).

df.Q.standard
Degrees of freedom for test of heterogeneity / inconsistency (standard model).

pval.Q.standard
P-value for test of heterogeneity / inconsistency (standard model).

Q.diff
Test statistic for difference in goodness of fit between standard and additive model.

df.Q.diff
Degrees of freedom for difference in goodness of fit between standard and additive model.

pval.Q.diff
P-value for difference in goodness of fit between standard and additive model.

A.matrix
Adjacency matrix ($n \times n$).

B.matrix
Edge-vertex incidence matrix ($m \times n$).

C.matrix
As defined above.

sm
Summary measure.

level.ma
Level for confidence intervals.

common, random, tau.preset
As defined above.

sep.trts
A character used in comparison names as separator between treatment labels.
**Note**

This function calculates effects for individual components and complex interventions present in the network.

R function `netcomplex` can be used to calculate the effect for arbitrary complex interventions in a component network meta-analysis. Furthermore, R function `netcomparison` can be used to calculate the effect for comparisons of two arbitrary complex intervention in a component network meta-analysis.

**Author(s)**

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

**References**


**See Also**

`discomb`, `netmeta`, `forest.netcomb`, `print.netcomb`, `netcomplex`, `netcomparison`

**Examples**

```r
data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce
# runtime of example)
```
# face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
#
# net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
#     data = face, ref = "placebo", sm = "OR", common = FALSE)
net1
forest(net1, xlim = c(0.2, 50))

# Additive model for treatment components (with placebo as inactive
# treatment)
#
# ncl <- netcomb(net1, inactive = "placebo")
nc1
forest(nc1, xlim = c(0.2, 50))

## Not run:

# Specify, order of treatments
#
# trts <- c("TCA", "SSRI", "SNRI", "NRI", "Low-dose SARI", "NaSSa",
#     "rMAO-A", "Ind drug", "Hypericum", "Face-to-face CBT",
#     "Face-to-face PST", "Face-to-face interpsy", "Face-to-face psychodyn",
#     "Other face-to-face", "Remote CBT", "Self-help CBT", "No contact CBT",
#     "Face-to-face CBT + SSRI", "Face-to-face interpsy + SSRI",
#     "Face-to-face PST + SSRI", "UC", "Placebo")
#
# Note, three treatments are actually combinations of 'SSRI' with
# other components:
# "Face-to-face CBT + SSRI",
# "Face-to-face interpsy + SSRI",
# "Face-to-face PST + SSRI"

# Conduct random effects network meta-analysis
#
# net2 <- netmeta(lnOR, selnOR, treat1, treat2, id,
#     data = Linde2016, ref = "placebo",
#     seq = trts, sm = "OR", common = FALSE)
net2
forest(net2, xlim = c(0.2, 50))

# Additive model for treatment components (with placebo as inactive
# treatment)
#
# nc2 <- netcomb(net2, inactive = "placebo")
nc2
forest(nc2, xlim = c(0.2, 50))

## End(Not run)
netcomparison

Calculate comparison effects of two arbitrary complex interventions in component network meta-analysis

Description

Calculate comparison effects of two arbitrary complex interventions (i.e., combinations of several components) in component network meta-analysis.

Usage

netcomparison(
  x,
  treat1,
  treat2,
  common = x$common,
  random = x$random,
  level = x$level.ma,
  nchar.comps = x$nchar.comps,
  backtransf = x$backtransf,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

## S3 method for class 'netcomparison'
print(
  x,
  common = x$common,
  random = x$random,
  backtransf = x$backtransf,
  nchar.comps = x$nchar.comps,
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = gs("digits.pval"),
  scientific.pval = gs("scientific.pval"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  big.mark = gs("big.mark"),
  legend = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

Arguments

x An object of class netcomb or netcomparison (print function).

treat1 A character vector defining the first complex intervention(s).
treat2 A character vector defining the second complex intervention(s).
common A logical indicating whether results for common effects model should be conducted.
random A logical indicating whether results for random effects model should be conducted.
level The level used to calculate confidence intervals for combinations of components.
nchar.comps A numeric defining the minimum number of characters used to create unique names for components (see Details).
backtransf A logical indicating whether printed results should be back transformed. If \texttt{backtransf=TRUE}, results for \texttt{sm=\"OR\"} are printed as odds ratios rather than log odds ratios.
warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments are used.
... Additional arguments (to catch deprecated arguments).
digits Minimal number of significant digits, see \texttt{print.default}.
digits.stat Minimal number of significant digits for z-value of test for overall effect, see \texttt{print.default}.
digits.pval Minimal number of significant digits for p-values, see \texttt{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 combination effect should be printed according to JAMA reporting standards.
big.mark A character used as thousands separator.
legend A logical indicating whether a legend should be printed.

Details

R functions \texttt{netcomb} and \texttt{discomb} calculate effects for individual components and complex interventions present in the component network meta-analysis (CNMA). This function can be used to calculate the effect for comparisons of two arbitrary complex interventions defined by arguments \texttt{treat1} and \texttt{treat2}.

All complex interventions occurring in the network are considered for the first complex intervention if argument \texttt{treat1} is missing. The reference group defined in the (C)NMA is used as second complex intervention if argument \texttt{treat2} is missing. The first complex intervention in the (C)NMA is used if the reference group is not defined.

The following matrices are needed to calculate comparison effects of arbitrary complex interventions, (Rücker et al., 2020, Section 3.2):

- B matrix describing how comparisons are composed by complex interventions,
- C matrix describing how the complex interventions are composed by the components.
Internally, both matrices are constructed based on arguments \(x, \text{treat1}\) and \(\text{treat2}\).

By default, component names are not abbreviated in printouts. However, in order to get more concise printouts, argument \text{nchar.comps}\ can be used to define the minimum number of characters for abbreviated component names (see \text{abbreviate}, argument \text{minlength}). R function \text{treats} is utilised internally to create abbreviated component names.

**Value**

A list is returned by the function \text{netcomparison} with the following elements:

- \text{comparison} \quad \text{Comparison.}
- \text{TE.common, TE.random} \quad \text{A vector of comparison effects (common and random effects model).}
- \text{seTE.common, seTE.random} \quad \text{A vector with corresponding standard errors (common and random effects model).}
- \text{lower.common, lower.random} \quad \text{A vector with lower confidence limits for comparisons (common and random effects model).}
- \text{upper.common, upper.random} \quad \text{A vector with upper confidence limits for comparisons (common and random effects model).}
- \text{statistic.common, statistic.random} \quad \text{A vector with z-values for the overall effect of comparisons (common and random effects model).}
- \text{pval.common, pval.random} \quad \text{A vector with p-values for the overall effect of comparisons (common and random effects model).}
- \text{trts} \quad \text{Treatments included in comparisons.}
- \text{comps} \quad \text{Components included in comparisons.}
- \text{treat1, treat2} \quad \text{A defined above.}
- \text{common, random} \quad \text{A defined above.}
- \text{level, nchar.comps, backtransf, x} \quad \text{A defined above.}
- \text{B.matrix} \quad \text{B matrix.}
- \text{C.matrix} \quad \text{C matrix.}

**Note**

R function \text{netcomplex} can be used to calculate the effect for arbitrary complex interventions in a component network meta-analysis.

**Author(s)**

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

See Also
netcomb, discomb, netcomplex

Examples
data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce
# runtime of example)
# face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
# net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
# data = face, ref = "placebo", sm = "OR", common = FALSE)

# Additive model for treatment components (with placebo as inactive
# treatment)
# nc1 <- netcomb(net1, inactive = "placebo")

# Result for comparison Face-to-face PST vs TCA
netcomparison(nc1, "Face-to-face PST", "TCA", nchar.comps = 4)
netcomparison(nc1, "F", "T", nchar.comps = 4)

# Result for comparison Face-to-face PST vs TCA + Placebo
netcomparison(nc1, "Face-to-face PST", "TCA + Plac", nchar.comps = 4)

## Not run:
# Artificial example
nc2 <- <net2, TE, seTE, t1, t2, random = FALSE)

# Result for comparison A vs B + D
etcomparison(nc2, "A", "B + D")

# Same results
netcomparison(nc2, "A", "B+D")
netcomparison(nc2, "A", "D+B")
netcomparison(nc2, "a", "d+b")

# Generated B matrix
netcomplex

Calculate effect of arbitrary complex interventions in component network meta-analysis

Description

Calculate effect of arbitrary complex interventions (i.e., combinations of several components) in component network meta-analysis.

Usage

netcomplex(
  x,
  complex,
  common = x$common,
  random = x$random,
  level = x$level.ma,
  nchar.comps = x$nchar.trts,
  backtransf = x$backtransf,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

## S3 method for class 'netcomplex'
print(
  x,
  common = x$common,
  random = x$random,
  backtransf = x$backtransf,
  nchar.comps = x$nchar.comps,
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = gs("digits.pval"),
  scientific.pval = gs("scientific.pval"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  big.mark = gs("big.mark"),
  legend = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
Arguments

- **x**: An object of class `netcomb` or `netcomplex` (print function).
- **complex**: A matrix, vector or single numeric defining the complex intervention(s) (see Details).
- **common**: A logical indicating whether results for common effects model should be conducted.
- **random**: A logical indicating whether results for random effects model should be conducted.
- **level**: The level used to calculate confidence intervals for combinations of components.
- **nchar.comps**: A numeric defining the minimum number of characters used to create unique names for components (see Details).
- **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.
- **warn.deprecated**: A logical indicating whether warnings should be printed if deprecated arguments are used.
- **...**: Additional arguments (to catch deprecated arguments).
- **digits**: Minimal number of significant digits, see `print.default`.
- **digits.stat**: Minimal number of significant digits for z-value of test for overall effect, see `print.default`.
- **digits.pval**: Minimal number of significant digits for p-values, 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 combination effect should be printed according to JAMA reporting standards.
- **big.mark**: A character used as thousands separator.
- **legend**: A logical indicating whether a legend should be printed.

Details

R functions `netcomb` and `discomb` only report results for complex interventions present in the network. This function can be used to calculate the effect for arbitrary complex interventions.

Complex interventions can be specified using argument **complex**:

- a character vector with definition of complex interventions,
- a single numeric defining the number of components to combine in a complex intervention,
- a dedicated C matrix.
In order to calculate effects of arbitrary complex interventions, a C matrix is needed which describes how the complex interventions are composed by the components (Rücker et al., 2020, Section 3.2). The C matrix is constructed internally if not provided by argument complex. All complex interventions occurring in the network are considered if argument complex is missing.

By default, component names are not abbreviated in printouts. However, in order to get more concise printouts, argument nchar.comps can be used to define the minimum number of characters for abbreviated component names (see abbreviate, argument minlength). R function treats is utilised internally to create abbreviated component names.

Value

A list is returned by the function netcomplex with the following elements:

- **complex**  
  Complex intervention(s).

- **Comb.common**, **Comb.random**
  A vector of combination effects (common and random effects model).

- **seComb.common**, **seComb.random**
  A vector with corresponding standard errors (common and random effects model).

- **lower.Comb.common**, **lower.Comb.random**
  A vector with lower confidence limits for combinations (common and random effects model).

- **upper.Comb.common**, **upper.Comb.random**
  A vector with upper confidence limits for combinations (common and random effects model).

- **statistic.Comb.common**, **statistic.Comb.random**
  A vector with z-values for the overall effect of combinations (common and random effects model).

- **pval.Comb.common**, **pval.Comb.random**
  A vector with p-values for the overall effect of combinations (common and random effects model).

- **common**, **random**
  A defined above.

- **level**, **nchar.comps**, **backtransf**, **x**
  A defined above.

- **C.matrix**
  C matrix.

Note

R function netcomparison can be used to calculate the effect for comparisons of two arbitrary complex intervention in a component network meta-analysis.

Author(s)

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

References

See Also

`netcomb, discomb, netcomparison`

Examples

data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce # runtime of example)
# face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
# net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
#   data = face, ref = "placebo", sm = "OR", common = FALSE)

# Additive model for treatment components (with placebo as inactive # treatment)
# nc1 <- netcomb(net1, inactive = "placebo")

# Result for combination Face-to-face PST + SSRI
netcomplex(nc1, "Face-to-face PST + SSRI", nchar.comps = 4)
netcomplex(nc1, "F + S", nchar.comps = 4)

# Result for combination Face-to-face PST + SSRI + Placebo
netcomplex(nc1, "Face-to-face PST + SSRI + Plac", nchar.comps = 4)

## Not run:  
# Artificial example  
t1 <- rep("A", 3)  
t2 <- c("B+C", "A+C", "C+D")  
TE <- c(0, 1, 0)  
seTE <- rep(1, 3)  
# Conduct (C)NMA  
net2 <- netmeta(TE, seTE, t1, t2, random = FALSE)
nc2 <- netcomb(net2)

# Result for combination A + B + C  
netcomplex(nc2, "A + B + C")  
# Same results  
netcomplex(nc2, "A+B+C")  
netcomplex(nc2, "B+C+A")  
netcomplex(nc2, "C+B+A")  
netcomplex(nc2, "c+b+a")

# Generated C matrix  
netcomplex(nc2, c(LETTERS[1:4], "A+B+C"))$C.matrix

# Results for all possible combinations of two components
netcomplex(nc2, 2)
# Results for all possible combinations of three components
netcomplex(nc2, 3)

## End(Not run)

---

**netconnection.default**  
*Get information on network connectivity (number of subnetworks, distance matrix)*

### Description

To determine the network structure and to test whether a given network is fully connected. Network information is provided as a triple of vectors `treat1`, `treat2`, and `studlab` where each row corresponds to an existing pairwise treatment comparison `(treat1, treat2)` in a study `(studlab)`. The function calculates the number of subnetworks (connectivity components; value of 1 corresponds to a fully connected network) and the distance matrix (in block-diagonal form in the case of subnetworks). If some treatments are combinations of other treatments or have common components, an analysis based on the additive network meta-analysis model might be possible, see `discomb` function.

### Usage

```r
## Default S3 method:
netconnection(
  data = NULL,
  treat1,
  treat2,
  studlab = NULL,
  subset = NULL,
  sep.trts = ":",
  nchar.trts = 666,
  title = "",
  details.disconnected = FALSE,
  warn = FALSE,
  ...
)
```

```r
## S3 method for class 'pairwise'
netconnection(
  data,
  sep.trts = ":",
  nchar.trts = 666,
  title = "",
  details.disconnected = FALSE,
  warn = FALSE,
  ...
)
```
## S3 method for class 'netconnection'
print(
  x,
  digits = max(4, .Options$digits - 3),
  nchar.trts = x$nchar.trts,
  details = FALSE,
  details.disconnected = x$details.disconnected,
  ...
)

netconnection(data, ...)

### Arguments

data
An optional data frame containing the study information.
treat1
Label / number for first treatment or a data frame created with `pairwise`
treat2
Label / number for second treatment.
studlab
An optional - but important! - vector with study labels (see Details).
subset
An optional vector specifying a subset of studies to be used.
sep.trts
A character used in comparison names as separator between treatment labels.
nchar.trts
A numeric defining the minimum number of characters used to create unique
treatment names.
title
Title of meta-analysis / systematic review.
details.disconnected
A logical indicating whether to print more details for disconnected networks.
warn
A logical indicating whether warnings should be printed.
...
Additional arguments (ignored at the moment)
x
An object of class `netconnection`.
digits
Minimal number of significant digits, see `print.default`.
details
A logical indicating whether to print the distance matrix.

### Value

An object of class `netconnection` with corresponding print function. The object is a list contain-
ing the following components:

treat1, treat2, studlab, title, warn, nchar.trts
  As defined above.
k
  Total number of studies.
m
  Total number of pairwise comparisons.
n
  Total number of treatments.
n.subnets
  Number of subnetworks; equal to 1 for a fully connected network.
D.matrix       Distance matrix.
A.matrix       Adjacency matrix.
L.matrix       Laplace matrix.
call           Function call.
version        Version of R package netmeta used to create object.

Author(s)
Gerta Rücker <gerta.ruecker@uniklinik-freiburg.de>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also
netmeta, netdistance, discomb

Examples

```r
data(Senn2013)
c1 <- netconnection(treat1, treat2, studlab, data = Senn2013)
c1
# Extract number of (sub)networks
#
c1$n.subnets

# Extract distance matrix
#
c1$D.matrix

## Not run:
# Conduct network meta-analysis (results not shown)
#
net1 <- netmeta(TE, seTE, treat1, treat2, studlab, data = Senn2013)

# Artificial example with two subnetworks
#
t1 <- c("G", "B", "B", "D", "A", "F")
t2 <- c("B", "C", "E", "E", "H", "A")
#
c2 <- netconnection(t1, t2)
print(nc2, details = TRUE)

# Number of subnetworks
#
c2$n.subnets

# Extract distance matrix
#
c2$D.matrix

# Conduct network meta-analysis (results in an error message due to
```
netcontrib

Contribution matrix in network meta-analysis

Description
This function generates the contribution of direct comparisons to every network treatment comparison. The output is a matrix where rows represent network treatment effects and columns represent the contribution of direct treatment effects.

Usage
netcontrib(
  x,
  method = "shortestpath",
  hatmatrix.F1000 = FALSE,
  common = x$common,
  random = x$random,
  nchar.trts = x$nchar.trts,
  warn.deprecated = gs("warn.deprecated"),
  verbose = FALSE,
  ...
)

## S3 method for class 'netcontrib'
print(
  x,
  common = x$x$common,
  random = x$x$random,
  digits = 4,
  nchar.trts = x$nchar.trts,
  legend = TRUE
)
warn.deprecated = gs("warn.deprecated"),
...
)

Arguments

x  An object of class netmeta or netcontrib.
method  A character string indicating which method is to calculate the contribution
matrix. Either "randomwalk" or "shortestpath", can be abbreviated.
hatmatrix.F1000  A logical indicating whether hat matrix given in F1000 article should be used
for method = "shortestpath".
common  A logical indicating whether a contribution matrix should be printed for the
common effects network meta-analysis.
random  A logical indicating whether a contribution matrix should be printed for the
random effects network meta-analysis.
nchar.trts  A numeric defining the minimum number of characters used to create unique

treatment names (see Details).
warn.deprecated  A logical indicating whether warnings should be printed if deprecated arguments
are used.
verbose  A logical indicating whether progress information should be printed.
...  Additional arguments.
digits  Minimal number of significant digits, see print.default.
legend  A logical indicating whether a legend should be printed.

Details

In network meta-analysis (NMA), it is important to assess the influence of limitations or other
characteristics of individual studies on the estimates obtained from the network. To this end, the
contribution matrix shows how much each direct treatment effect contributes to each treatment
effect estimate from network meta-analysis.

We use ideas from graph theory to derive the proportion that is contributed by each direct treatment
effect. We start with the 'projection' matrix in a two-step network meta-analysis model, called the H
matrix, which is analogous to the hat matrix in a linear regression model. H entries are translated to
proportion contributions based on the observation that the rows of H can be interpreted as flow net-
works. A stream is defined as the composition of a path and its associated flow (Papakonstantinou
et al., 2018).

To account for multi-arm trials, we use the H matrix from a two-step (aggregate) version of the graph
theoretical NMA model (Davies et al., 2022). This H matrix can be obtained from hatmatrix with
argument method = "davies".

Two methods are implemented to estimate the streams and as a result, the proportion contributions:
(1) If argument method = "randomwalk", an analytical random-walk (RW) approach is used (Davies
et al., 2022). Here, the "full" version of the aggregate H matrix (hatmatrix with arguments method
= "davies" and type = "full") is used to define RW transition matrices. For each pair of treatments (ij) in the network, the elements in the corresponding row of H-full define a transition matrix from node i to node j. We use the igraph package to find every (directed) path from node i to node j. The flow through each path is then equal to the probability that a walker takes that path. This is simply the product of the transition probabilities associated with each edge along the path.

(2) If argument method = "shortestpath", an iterative algorithm is used (Papakonstantinou et al., 2018). Broadly speaking, each iteration of the algorithm consists of the following steps: (i) A path in the evidence flow network is selected. (ii) The minimum flow through the edges making up the path is identified. This is assigned as the flow associated with the path. (iii) The flow of the path is subtracted from the values of flow in the edges that make up that path. This means that the edge corresponding to the minimum flow in that path is removed from the graph. (iv) A new path is then selected from the remaining graph. The process repeats until all the evidence flow in the edges has been assigned to a path.

In the original F1000 paper (Papakonstantinou et al., 2018), the hat matrix used did not account for correlations due to multi-arm trials. For reproducibility the result of this version can be obtained by specifying hatmatrix.F1000 = TRUE for method = "shortestpath". For other purposes, this method is not recommended.

Once the streams have been identified (either by method (1) or (2)), the proportion contribution of each direct comparison is equal to the sum over the flow of evidence in each path containing that edge divided by the number of edges that make up that path.

By default, treatment names are not abbreviated in printouts. However, in order to get more concise printouts, argument nchar.trts can be used to define the minimum number of characters for abbreviated treatment names (see abbreviate, argument minlength). R function treats is utilised internally to create abbreviated treatment names.

Calculation of network contributions can be compute-intensive for the random-walk approach in large networks. Crude information on the computation progress is printed if argument verbose is TRUE. In addition, computation times are printed if R package tictoc is installed.

**Value**

An object of class netcontrib with corresponding print function. The object is a list containing the following components:

- **common**
  - Numeric matrix of percentage contributions of direct comparisons for each network comparison for the common effects model.

- **random**
  - Numeric matrix of percentage contributions of direct comparisons for each network comparison for the random effects model.

- **x**
  - As defined above.

- **tictoc.common**
  - Computation times under common effects model (if R package tictoc is installed).

- **tictoc.random**
  - Computation times under random effects model (if R package tictoc is installed).

with the contribution matrices for common and random NMA. Each matrix has the percentage contributions of each direct comparison as columns for each network comparison, direct or indirect as rows.
**netdistance**

Calculate distance matrix for an adjacency matrix

**Description**

Calculate distance matrix for an adjacency matrix based on distance algorithm by Müller et al. (1987).

**Usage**

```r
gertadistance(x)
```

**Arguments**

- `x` Either a netmeta object or an adjacency matrix.

**Author(s)**

Gerta Rücker <gerta.ruecker@uniklinik-freiburg.de>
netgraph

Generic function for network graphs

Description

Generic function for network graphs

Usage

netgraph(x, ...)

Arguments

x An R object.

... Additional arguments.
Details

For more details, look at the following functions to generate network graphs:

- `netgraph.netmeta`
- `netgraph.netimpact`
- `netgraph.netconnection`
- `netgraph.netcomb`
- `netgraph.discomb`

Author(s)

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

Examples

data(smokingcessation)

# Transform data from arm-based format to contrast-based format
#
p1 <- pairwise(list(treat1, treat2, treat3),
  event = list(event1, event2, event3), n = list(n1, n2, n3),
  data = smokingcessation, sm = "OR")

# Conduct random effects network meta-analysis
#
net1 <- netmeta(p1, common = FALSE)

# Network graph with default settings
#
netgraph(net1)

## Not run:
data(Senn2013)

# Generation of an object of class 'netmeta' with reference
treatment 'plac'
#
net2 <- netmeta(TE, seTE, treat1, treat2, studlab,
  data = Senn2013, sm = "MD", reference = "plac")

# Network graph with default settings
#
netgraph(net2)

data(Woods2010)
p3 <- pairwise(treatment, event = r, n = N,
  studlab = author, data = Woods2010, sm = "OR")
net3 <- netmeta(p3)

# Network graph with default settings
#
netgraph(net3)

# Network graph with
# - number of studies for each pairwise comparison and
# - number of participants for each treatment arm
#
netgraph(net3, number.of.studies = TRUE,
       labels = paste0(trts, " (n=", n.trts, ")"))

## End(Not run)

---

netgraph.discomb  Network graph for objects of class discomb

**Description**

This function generates a graph of the evidence network.

**Usage**

```r
## S3 method for class 'discomb'
netgraph(x, plastic = FALSE, ...)
```

**Arguments**

- `x` An object of class `discomb`.
- `plastic` A logical indicating whether the appearance of the comparisons should be in '3D look'.
- `...` Additional arguments passed on to `netgraph.netmeta` (see Details).

**Details**

The arguments `seq` and `iterate` are used internally and cannot be specified by the user.

**Author(s)**

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

**See Also**

`discomb`, `netgraph.netmeta`
Examples

# Artificial dataset
#
t2 <- c("C" , "B" , "B + C", "A + D", "E", "F", "F")
#
mean <- c(4.1, 2.05, 0, 0, 0.1, 0.1, 0.05)
se.mean <- rep(0.1, 7)
#
study <- paste("study", c(1:4, 5, 5, 5))
#
dat <- data.frame(mean, se.mean, t1, t2, study,
    stringsAsFactors = FALSE)
#
    "B", "B + C", "C", "D", "E", "F")
#
comps <- LETTERS[1:6]
#
# Use netconnection() to display network information
#
netconnection(t1, t2, study)

dc1 <- discomb(mean, se.mean, t1, t2, study, seq = trts)

netgraph(dc1)

---

netgraph.netcomb  
Network graph for objects of class netcomb

Description

This function generates a graph of the evidence network.

Usage

## S3 method for class 'netcomb'
netgraph(x, ...)

Arguments

x  
An object of class netcomb.

...  
Additional arguments passed on to netgraph.netmeta.

Author(s)

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

`netcomb`, `netgraph.netmeta`

Examples

data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce
# runtime of example)
#
face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
#
net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
  data = face, ref = "placebo", sm = "OR", common = FALSE)

# Additive model for treatment components (with placebo as inactive
# treatment)
#
nc1 <- netcomb(net1, inactive = "placebo")

netgraph(nc1)

Description

This function generates a graph of the evidence network.

Usage

```r
## S3 method for class 'netconnection'
netgraph(
  x,
  seq,
  col = seq_len(x$n.subnets),
  reference.group = NULL,
  plastic = FALSE,
  ...,
)
```
Arguments

- **x**: An object of class `netconnection`.
- **seq**: A character or numerical vector specifying the sequence of treatments arrangement (anticlockwise if `start.layout = "circle"`).
- **col**: A single color (or vector of colors) for lines connecting treatments (edges) if argument `plastic = FALSE` (see Details).
- **reference.group**: Reference treatment (only relevant for disconnected networks).
- **plastic**: A logical indicating whether the appearance of the comparisons should be in '3D look'.
- **...**: Additional arguments passed on to `netgraph.netmeta` (see Details).

Details

Argument `col` can be a single color for all edges, a vector of length equal to the number of edges, or a vector of length equal to the number of subnetworks. Argument `reference.group` is only considered in disconnected networks, i.e., if more than one (sub)network exists, and if argument `col` provides colors for subnetworks. In this case, the first color provided in argument `col` defines the color for the subnetwork with the reference treatment.

Author(s)

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

See Also

- `netconnection`
- `netgraph.netmeta`

Examples

```r
# Artificial example with two subnetworks

#
t1 <- c("G", "B", "B", "D", "A", "F")
t2 <- c("B", "C", "E", "E", "H", "A")
#
nc1 <- netconnection(t1, t2)
print(nc1, details = TRUE)

netgraph(nc1, points = TRUE, adj = 0.5, bg.points = "lightgray")
netgraph(nc1, points = TRUE, adj = 0.5, bg.points = "lightgray",
        plastic = TRUE)
```
Description

This function generates a graph of the evidence network.

Usage

```r
## S3 method for class 'netimpact'
netgraph(
  x,
  col.ignore = "red",
  number.of.studies = TRUE,
  main,
  sub,
  multiarm = FALSE,
  col.multiarm = NULL,
  alpha.transparency = 0.5,
  col.ignore.multiarm = "transparent",
  col = "black",
  plastic = FALSE,
  ...
)
```

Arguments

- **x**: An object of class `netimpact`.
- **col.ignore**: A character string indicating color for comparisons removed from network, either "transparent" or any color defined in `colours`.
- **number.of.studies**: A logical indicating whether number of studies should be added to network graph.
- **main**: Main title.
- **sub**: Subtitle.
- **multiarm**: A logical indicating whether multi-arm studies should be marked in plot.
- **col.multiarm**: Either a function from R package colorspace or grDevice to define colors for multi-arm studies or a character vector with colors to highlight multi-arm studies.
- **alpha.transparency**: The alpha transparency of colors used to highlight multi-arm studies (0 means transparent and 1 means opaque).
- **col.ignore.multiarm**: A character string indicating color to mark multi-arm studies removed from network, either "transparent" or any color defined in `colours`.
col

A single color (or vector of colors) for lines connecting treatments (edges) if argument plastic = FALSE. Length of the vector must be equal to the number of edges.

plastic

A logical indicating whether the appearance of the comparisons should be in '3D look'.

... Additional arguments passed on to netgraph.netmeta.

Author(s)

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

See Also

netimpact, netgraph.netmeta

Examples

data(Franchini2012)

# Only consider first two studies (to reduce runtime of example)
# studies <- unique(Franchini2012$Study)
p1 <- pairwise(list(Treatment1, Treatment2, Treatment3),
  n = list(n1, n2, n3),
  mean = list(y1, y2, y3),
  sd = list(sd1, sd2, sd3),
  data = subset(Franchini2012, Study %in% studies[1:2]),
  studlab = Study)

net1 <- netmeta(p1)
ni1 <- netimpact(net1, verbose = TRUE)
netgraph(ni1)
netgraph(ni1, plastic = TRUE)

## Not run:
p2 <- pairwise(list(Treatment1, Treatment2, Treatment3),
  n = list(n1, n2, n3),
  mean = list(y1, y2, y3),
  sd = list(sd1, sd2, sd3),
  data = Franchini2012,
  studlab = Study)

net2 <- netmeta(p2)
ni2 <- netimpact(net2, verbose = TRUE)
netgraph(ni2)
netgraph(ni2, plastic = TRUE)

## End(Not run)
Description

This function generates a graph of the evidence network.

Usage

```r
## S3 method for class 'netmeta'
netgraph(
x,
seq = x$seq,
labels = x$trts,
cex = 1,
adj = NULL,
srt.labels = 0,
offset = if (!is.null(adj) & all(unique(adj) == 0.5)) 0 else 0.0175,
scale = 1.1,
col = if (iterate) "slateblue" else "black",
plastic = !(iterate & allfigures),
thickness = "number.of.studies",
lwd = 5,
lwd.min = lwd/2.5,
lwd.max,
rescale.thickness,
dim = "2d",
rotate = 0,
highlight = NULL,
col.highlight = "red2",
scale.highlight = 1,
multiarm = FALSE,
col.multiarm = NULL,
alpha.transparency = 0.5,
points = !missing(cex.points),
cex.points = 1,
pch.points = 20,
col.points = if (length(pch.points) == 1 & pch.points == 21) "black" else "red",
bg.points = "red",
points.min,
points.max,
rescale.pointsiz;
number.of.studies = FALSE,
cex.number.of.studies = cex,
col.number.of.studies = "white",
bg.number.of.studies = "black",
pos.number.of.studies = 0.5,
```
\begin{verbatim}
start.layout = ifelse(dim == "2d", "circle", "eigen"),
eig1 = 2,
eig2 = 3,
eig3 = 4,
iterate = FALSE,
tol = 1e-04,
maxit = 500,
allfigures = FALSE,
A.matrix = x$A.matrix,
N.matrix = sign(A.matrix),
D.matrix = netdistance(N.matrix),
xpos = NULL,
ypos = NULL,
zpos = NULL,
figure = TRUE,
...
)
\end{verbatim}

**Arguments**

- **x**  
  An object of class netmeta (mandatory).

- **seq**  
  A character or numerical vector specifying the sequence of treatments arrangement (anticlockwise if `start.layout = "circle"`).

- **labels**  
  An optional vector with treatment labels.

- **cex**  
  The magnification to be used for treatment labels.

- **adj**  
  One, two, or three values in \([0, 1]\) (or a vector / matrix with length / number of rows equal to the number of treatments) specifying the x (and optionally y and z) adjustment for treatment labels.

- **srt.labels**  
  The character string "orthogonal" (can be abbreviated), a single numeric or numerical vector with value(s) between -180 and 180 specifying the angle to rotate treatment labels (see Details).

- **offset**  
  Distance between edges (i.e. treatments) in graph and treatment labels for 2-D plots (value of 0.0175 corresponds to a difference of 1.75% of the range on x- and y-axis).

- **scale**  
  Additional space added outside of edges (i.e. treatments). Increase this value for larger treatment labels (value of 1.10 corresponds to an additional space of 10% around the network graph).

- **col**  
  A single color (or vector of colors) for lines connecting treatments (edges) if argument `plastic = FALSE`. Length of the vector must be equal to the number of edges (see list element 'comparisons' in `netmeta`).

- **plastic**  
  A logical indicating whether the appearance of the comparisons should be in '3D look' (not to be confused with argument `dim`).

- **thickness**  
  Either a character variable to determine the method to plot line widths (see Details) or a matrix of the same dimension and row and column names as argument `A.matrix` with information on line width.

- **lwd**  
  A numeric for scaling the line width of comparisons.
lwd.min Minimum line width in network graph. All connections with line widths below this values will be set to lwd.min.

lwd.max Maximum line width in network graph. The connection with the largest value according to argument thickness will be set to this value.

rescale.thickness A logical value or R function to scale the thickness of lines (see Details).

dim A character string indicating whether a 2- or 3-dimensional plot should be produced, either "2d" or "3d".

rotate A single numeric with value between -180 and 180 specifying the angle to rotate nodes in a circular network.

highlight A character vector identifying comparisons that should be marked in the network graph, e.g. highlight = "treat1:treat2".

col.highlight Color(s) to highlight the comparisons given by highlight.

scale.highlight Scaling factor(s) for the line width(s) to highlight the comparisons given by highlight.

multiarm A logical indicating whether multi-arm studies should be marked in plot.

col.multiarm Either a function from R package colorspace or grDevice to define colors for multi-arm studies or a character vector with colors to highlight multi-arm studies.

alpha.transparency The alpha transparency of colors used to highlight multi-arm studies (0 means transparent and 1 means opaque).

points A logical indicating whether points should be printed at nodes (i.e. treatments) of the network graph.

cex.points, pch.points, col.points, bg.points Corresponding size, type, color, and background color for points. Can be a vector with length equal to the number of treatments.

points.min Minimum point size. All points with size below this values will be set to points.min.

points.max Maximum point size in network graph. The node with the largest value according to argument cex.points will be set to this value.

rescale.points A logical value or R function to scale the point size (see Details).

number.of.studies A logical indicating whether number of studies should be added to network graph.

cex.number.of.studies The magnification to be used for number of studies.

col.number.of.studies Color for number of studies.

bg.number.of.studies Color for shadow around number of studies.
pos.number.of.studies
A single value (or vector of values) in [0, 1] specifying the position of the number of studies on the lines connecting treatments (edges). Length of the vector must be equal to the number of edges.

start.layout
A character string indicating which starting layout is used if iterate = TRUE. If "circle" (default), the iteration starts with a circular ordering of the vertices; if "eigen", eigenvectors of the Laplacian matrix are used, calculated via generic function eigen (spectral decomposition); if "prcomp", eigenvectors of the Laplacian matrix are calculated via generic function prcomp (principal component analysis); if "random", a random layout is used, drawn from a bivariate normal.

eig1
A numeric indicating which eigenvector is used as x coordinate if start = "eigen" or "prcomp" and iterate = TRUE. Default is 2, the eigenvector to the second-smallest eigenvalue of the Laplacian matrix.

eig2
A numeric indicating which eigenvector is used as y-coordinate if start = "eigen" or "prcomp" and iterate = TRUE. Default is 3, the eigenvector to the third-smallest eigenvalue of the Laplacian matrix.

eig3
A numeric indicating which eigenvector is used as z-coordinate if start = "eigen" or "prcomp" and iterate = TRUE. Default is 4, the eigenvector to the fourth-smallest eigenvalue of the Laplacian matrix.

iterate
A logical indicating whether the stress majorization algorithm is carried out for optimization of the layout.

tol
A numeric for the tolerance for convergence if iterate = TRUE.

maxit
An integer defining the maximum number of iteration steps if iterate = TRUE.

allfigures
A logical indicating whether all iteration steps are shown if iterate = TRUE. May slow down computations if set to TRUE (especially if plastic = TRUE).

A.matrix
Adjacency matrix (n x n) characterizing the structure of the network graph. Row and column names must be the same set of values as provided by argument seq.

N.matrix
Neighborhood matrix (n x n) replacing A.matrix if neighborhood is to be specified differently from node adjacency in the network graph, for example content-based. Row and column names must be the same set of values as provided by argument seq.

D.matrix
Distance matrix (n x n) replacing A.matrix and N.matrix if distances should be provided directly. Row and column names must be the same set of values as provided by argument seq.

xpos
Vector (n) of x coordinates.

ypos
Vector (n) of y coordinates.

zpos
Vector (n) of z coordinates.

figure
A logical indicating whether network graph should be shown.

... Additional graphical arguments.

Details
This function generates a network graph for an R object created with netmeta.
Layout of network graph: The network is laid out in the plane, where the nodes in the graph layout correspond to the treatments and edges display the observed treatment comparisons. For the default setting, nodes are placed on a circle. Other starting layouts are "eigen", "prcomp", and "random" (Rücker & Schwarzer 2015). If `iterate = TRUE`, the layout is further optimized using the stress majorization algorithm. This algorithm specifies an 'ideal' distance (e.g., the graph distance) between two nodes in the plane. In the optimal layout, these distances are best approximated in the sense of least squares. Starting from an initial layout, the optimum is approximated in an iterative process called stress majorization (Kamada and Kawai 1989, Michailidis and de Leeuw 2001, Hu 2012). The starting layout can be chosen as a circle or coming from eigenvectors of the Laplacian matrix (corresponding to Hall's algorithm, Hall 1970), calculated in different ways, or random. Moreover, it can be chosen whether the iteration steps are shown (argument `allfigures = TRUE`).

An optimized circular presentation which typically has a reduced (sometimes minimal) number of crossings can be achieved by using argument `seq = "optimal"` in combination with argument `start.layout`. Note, is not possible of prespecify the best value for argument `start.layout` for any situation as the result depends on the network structure.

Definition of line widths: Argument `thickness` providing the line width of edges (comparisons) can be a matrix of the same dimension as argument `A.matrix` or any of the following character strings (which can be abbreviated):

- Proportional to number of studies comparing two treatments (`thickness = "number.of.studies", default`)
- Proportional to inverse standard error of common effects model comparing two treatments (`thickness = "se.common"`)  
- Proportional to inverse standard error of random effects model comparing two treatments (`thickness = "se.random"`)  
- Weight from common effects model comparing two treatments (`thickness = "w.common"`)  
- Weight from random effects model comparing two treatments (`thickness = "w.random"`)  
- Same line width for all comparisons (`thickness = "equal"`)  

Only evidence from direct treatment comparisons is considered to determine the line width if argument `thickness` is equal to any but the last method.

Line widths are determined by argument `lwd` if all lines have the same width. This is possible if either argument `thickness = "equal"`, all pairwise comparisons have the same number of studies for `thickness = "number.of.studies"` or all direct comparisons are equally precise.

Otherwise, the line width of the thickest line is equal to the value of argument `lwd.max` and all lines with a thickness below the value of argument `lwd.min` are set to this value. Default for argument `lwd.max` is $4 \times lwd$.

Argument `rescale.thickness` can be used to provide a function to specify the relative line width of edges (comparisons). By default, the square root function `sqrt` is used in order to lessen differences in line widths. Argument `rescale.thickness = FALSE` or `rescale.thickness = I`, i.e., the identity function `I`, can be used to not rescale line widths.

Definition of point sizes: Points are printed at nodes (treatments) if argument `points = TRUE` or argument `cex.points` is provided.

Point sizes are equal to the value of argument `cex.points` if all points are of equal size.
Otherwise, the point size of the largest point is equal to the value of argument `points.max` and all points smaller than the value of argument `points.min` are set to this value. The default for argument `points.max` is equal to the largest value provided in argument `cex.points` if this largest value is below or equal to 25. Otherwise the default is `points.max = 8`.

Argument `rescale.pointsize` can be used to provide a function to specify relative point sizes. Point sizes are not rescaled at all if they are all equal or the largest `cex.points` value is below or equal to 25. Otherwise, the square root function `sqrt` is used in order to lessen the differences in point sizes. Argument `rescale.pointsize = FALSE` or `rescale.pointsize = I`, i.e., the identity function I, can be used to not rescale point sizes.

**Other settings:** Argument `srt.labels` can be used to specific the rotation (in degrees) of the treatment labels. If `srt.labels` is equal to "orthogonal", treatment labels are orthogonal to the circle. If `srt.labels` is a single numeric, all labels are rotated by this degree. If `srt.labels` is a numeric vector, it must be of the same length as the number of treatments and labels are rotated counter-clockwise starting on the right side. Finally, if `srt.labels` is a named numeric vector, it must be of the same length as the number of treatments and the names must be equal to the treatment names (and treatment labels are rotated according to the specified values).

Further, a couple of graphical parameters can be specified, such as color and appearance of the edges (treatments) and the nodes (comparisons), whether special comparisons should be highlighted and whether multi-arm studies should be indicated as colored polygons. By default, if R package colorspace is available the `sequential_hcl` function is used to highlight multi-arm studies; otherwise the `rainbow` is used.

In order to generate 3-D plots (argument `dim = "3d"`), R package `rgl` is necessary. Note, under macOS the X.Org X Window System must be available (see https://www.xquartz.org).

**Value**

A list containing two data frames with information on nodes and edges.

**List element ’nodes’**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>trts</code></td>
<td>Treatment names.</td>
</tr>
<tr>
<td><code>labels</code></td>
<td>Treatment labels.</td>
</tr>
<tr>
<td><code>seq</code></td>
<td>Sequence of treatment labels.</td>
</tr>
<tr>
<td><code>srt</code></td>
<td>String rotation.</td>
</tr>
<tr>
<td><code>xpos</code></td>
<td>Position of treatment / edge on x-axis.</td>
</tr>
<tr>
<td><code>ypos</code></td>
<td>Position of treatment / edge on y-axis.</td>
</tr>
<tr>
<td><code>zpos</code></td>
<td>Position of treatment / edge on z-axis (for 3-D plots).</td>
</tr>
<tr>
<td><code>xpos.labels</code></td>
<td>Position of treatment labels on x-axis (for 2-D plots).</td>
</tr>
<tr>
<td><code>ypos.labels</code></td>
<td>Position of treatment labels on y-axis (for 2-D plots).</td>
</tr>
<tr>
<td><code>offset.x</code></td>
<td>Offset of treatment labels on x-axis (for 2-D plots).</td>
</tr>
<tr>
<td><code>offset.y</code></td>
<td>Offset of treatment labels on y-axis (for 2-D plots).</td>
</tr>
<tr>
<td><code>cex</code></td>
<td>Point size of treatments / edges.</td>
</tr>
<tr>
<td><code>col</code></td>
<td>Color for points.</td>
</tr>
<tr>
<td><code>pch</code></td>
<td>Point type.</td>
</tr>
</tbody>
</table>
bg  Background color for points.
adj.x  Adjustment for treatment label on x-axis.
adj.y  Adjustment for treatment label on y-axis.
adj.z  Adjustment for treatment label on z-axis (for 3-D plots).

**List element 'edges'**

- treat1  Name of first treatment.
- treat2  Name of second treatment.
- n.stud  Number of studies directly comparing treatments.
- xpos  Position of number of studies on x-axis.
- ypos  Position of number of studies on y-axis.
- adj  Adjustment of number of studies.
- pos.number.of.studies  Position of number of studies on edge.
- col  Color for edges.

**Author(s)**

Gerta Rücker <gerta.ruecker@uniklinik-freiburg.de>, Ulrike Krahn <ulrike.krahn@bayer.com>,
Jochem König <koenigjo@uni-mainz.de>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

**References**


**See Also**

*netmeta*
Examples

data(smokingcessation)

# Transform data from arm-based format to contrast-based format
#
p1 <- pairwise(list(treat1, treat2, treat3),
    event = list(event1, event2, event3), n = list(n1, n2, n3),
    data = smokingcessation, sm = "OR")

# Conduct random effects network meta-analysis
#
net1 <- netmeta(p1, common = FALSE)

# Network graph with default settings
#
netgraph(net1)

## Not run:
data(Senn2013)

# Generation of an object of class 'netmeta' with reference treatment 'plac'
#
net2 <- netmeta(TE, seTE, treat1, treat2, studlab,
    data = Senn2013, sm = "MD", reference = "plac")

# Network graph with default settings
#
netgraph(net2)

# Network graph with specified order of the treatments and one highlighted comparison
#
trts <- c("plac", "benf", "migl", "acar", "sulf",
    "metf", "rosi", "piog", "sita", "vild")
netgraph(net2, highlight = "rosi:plac", seq = trts)

# Same network graph using argument 'seq' in netmeta function
#
net3 <- netmeta(TE, seTE, treat1, treat2, studlab,
    data = Senn2013, sm = "MD", reference = "plac", seq = trts)
netgraph(net3, highlight = "rosi:plac")

# Network graph optimized, starting from a circle, with multi-arm study colored
#
netgraph(net2, start = "circle", iterate = TRUE,
    multiarm = TRUE, col.multiarm = "purple")

# Network graph optimized, starting from a circle, with multi-arm
# study colored and all intermediate iteration steps visible
#
netgraph(net2, start = "circle", iterate = TRUE, 
multiarm = TRUE, col.multiarm = "purple", 
allfigures = TRUE)

# Network graph optimized, starting from Laplacian eigenvectors, 
# with multi-arm study colored 
# netgraph(net2, start = "eigen", 
# multiarm = TRUE, col.multiarm = "purple")

# Network graph optimized, starting from different Laplacian 
# eigenvectors, with multi-arm study colored 
# netgraph(net2, start = "prcomp", 
# multiarm = TRUE, col.multiarm = "purple")

# Network graph optimized, starting from random initial layout, 
# with multi-arm study colored 
# netgraph(net2, start = "random", 
# multiarm = TRUE, col.multiarm = "purple")

# Network graph without plastic look and one highlighted comparison 
# netgraph(net2, plastic = FALSE, highlight = "rosi:plac")

# Network graph with same thickness for all comparisons 
# netgraph(net2, thickness = "equal")

# Network graph with changed labels and specified order of the 
# treatments 
# netgraph(net2, seq = c(1, 3, 5, 2, 9, 4, 7, 6, 8, 10), 
# labels = LETTERS[1:10])

# Rotate treatment labels (orthogonal to circle) 
# netgraph(net2, srt.labels = "o")

# Network graph in 3-D (opens a new device, where you may rotate and 
# zoom the plot using the mouse / the mouse wheel). 
# The rgl package must be installed for 3-D plots. 
# netgraph(net2, dim = "3d")

## End(Not run)
Description

This function creates a net heat plot, a graphical tool for locating inconsistency in network meta-analyses.

Usage

```
netheat(
  x,
  random = FALSE,
  tau.preset = NULL,
  showall = TRUE,
  nchar.trts = x$nchar.trts,
  ...
)
```

Arguments

- `x`: An object of class `netmeta`.
- `random`: A logical indicating whether the net heat plot should be based on a random effects model.
- `tau.preset`: An optional value for the square-root of the between-study variance $\tau^2$ for a random effects model on which the net heat plot will be based.
- `showall`: A logical indicating whether results should be shown for all designs or only a sensible subset (see Details).
- `nchar.trts`: A numeric defining the minimum number of characters used to create unique treatment names.
- `...`: Additional arguments.

Details

The net heat plot is a matrix visualization proposed by Krahn et al. (2013) that highlights hot spots of inconsistency between specific direct evidence in the whole network and renders transparent possible drivers.

In this plot, the area of a gray square displays the contribution of the direct estimate of one design in the column to a network estimate in a row. In combination, the colors show the detailed change in inconsistency when relaxing the assumption of consistency for the effects of single designs. The colors on the diagonal represent the inconsistency contribution of the corresponding design. The colors on the off-diagonal are associated with the change in inconsistency between direct and indirect evidence in a network estimate in the row after relaxing the consistency assumption for the effect of one design in the column. Cool colors indicate an increase and warm colors a decrease: the stronger the intensity of the color, the greater the difference between the inconsistency before and after the detachment. So, a blue colored element indicates that the evidence of the design in the column supports the evidence in the row. A clustering procedure is applied to the heat matrix in order to find warm colored hot spots of inconsistency. In the case that the colors of a column corresponding to design $d$ are identical to the colors on the diagonal, the detaching of the effect of design $d$ dissolves the total inconsistency in the network.
The pairwise contrasts corresponding to designs of three- or multi-arm studies are marked by ‘_’ following the treatments of the design.

Designs where only one treatment is involved in other designs of the network or where the removal of corresponding studies would lead to a splitting of the network do not contribute to the inconsistency assessment. By default, (showall = TRUE), these designs are not incorporated into the net heat plot. If showall = FALSE, additional designs with minimal contribution to the inconsistency Q statistic are not incorporated (i.e., designs with abs(Q.inc.design) <= .Machine$double.eps^0.5).).

In the case of random = TRUE, the net heat plot is based on a random effects model generalised for multivariate meta-analysis in which the between-study variance \( \tau^2 \) is estimated by the method of moments (see Jackson et al., 2012) and embedded in a full design-by-treatment interaction model (see Higgins et al., 2012).

Author(s)

Ulrike Krahn <ulrike.krahn@bayer.com>

References


See Also

*netmeta*

Examples

data(Senn2013)

# Only consider first five studies (to reduce runtime of example)
# studies <- unique(Senn2013$studlab)
Senn2013.5 <- subset(Senn2013, studlab %in% studies[1:5])

# Conduct network meta-analysis with placebo as reference treatment
# net1 <- netmeta(TE, seTE, treat1, treat2, studlab,
# data = Senn2013.5, sm = "MD", reference = "plac")

# Generate a net heat plot based on a common effects model
# netheat(net1)

# Not run:
# Generate a net heat plot based on a random effects model
netimpact  

Determine the importance of individual studies in network meta-analysis

Description

This function measures the importance of individual studies in network meta-analysis by the reduction of the precision if the study is removed / ignored from the network (Rücker et al., 2020).

Usage

netimpact(
  x,
  seTE.ignore = 100 * max(x$seTE, na.rm = TRUE),
  event.ignore = 0.01,
  verbose = FALSE
)

Arguments

x  An object of class netmeta.
seTE.ignore  Assumed (large) standard error in order to mimicking the removal of individual studies from the network meta-analysis (ignored for netmetabin objects).
event.ignore  Assumed event number mimicking the removal of individual studies from the network meta-analysis (considered for netmetabin objects).
verbose  A logical indicating whether information on the estimation progress should be printed.

Value

An object of class "netimpact" with corresponding netgraph and print function. The object is a list containing the following components:

impact.common  A matrix with contributions of individual studies (columns) to comparisons (rows) under the common effects model.
impact.random  A matrix with contributions of individual studies (columns) to comparisons (rows) under the random effects model.
ignored.comparisons  List with comparisons of ignored study.
seTE.ignore, event.ignore, x
  As defined above.

nets  List of all network meta-analyses (removing a single study).
version  Version of R package netmeta used to create object.
Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>, Gerta Rücker <gerta.ruecker@uniklinik-freiburg.de>

References

See Also
netmeta, netmetabin, netgraph.netimpact, print.netimpact

Examples
data(Franchini2012)

# Only consider first two studies (to reduce runtime of example)
# studies <- unique(Franchini2012$Study)
pl <- pairwise(list(Treatment1, Treatment2, Treatment3),
n = list(n1, n2, n3),
mean = list(y1, y2, y3), sd = list(sd1, sd2, sd3),
data = subset(Franchini2012, Study %in% studies[1:2]),
studlab = Study)

net1 <- netmeta(pl)
ni1 <- netimpact(net1, verbose = TRUE)
ni1

netgraph(ni1)
seq = x$seq,
ci = TRUE,
backtransf = TRUE,
direct = FALSE,
digits = gs("digits"),
big.mark = gs("big.mark"),
text.NA = ".",
bracket = gs("CIbracket"),
separator = gs("CIseparator"),
lower.blank = gs("CILower.blank"),
upper.blank = gs("CIUpper.blank"),
writeXl = !missing(path),
path = "leaguetable.xlsx",
overwrite = FALSE,
warn.deprecated = gs("warn.deprecated"),
...
)

## S3 method for class 'netleague'

print(
  x,
  common = x$x$common,
  random = x$x$random,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

Arguments

x An object of class netmeta or netleague (mandatory).
y An object of class netmeta (optional).
common A logical indicating whether a league table should be printed for the common
effects network meta-analysis.
random A logical indicating whether a league table should be printed for the random
effects network meta-analysis.
seq A character or numerical vector specifying the sequence of treatments in rows
and columns of a league table.
ci A logical indicating whether confidence intervals should be shown.
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.
direct A logical indicating whether league table with network estimates (default) or es-
timates from direct comparisons should be generated if argument y is not missing.
digits Minimal number of significant digits, see print.default.
big.mark A character used as thousands separator.
text.NA A character string to label missing values.
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.
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.
warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments are used.
... Additional arguments (passed on to write_xlsx to create Excel file).

Details

A league table is a square matrix showing all pairwise comparisons in a network meta-analysis (Hutton et al., 2015). Typically, both treatment estimates and confidence intervals are shown.

If argument y is not provided, the league table contains the network estimates from network meta-analysis object x in the lower triangle and the direct treatment estimates from pairwise comparisons in the upper triangle. Note, for the random-effects model, the direct treatment estimates are based on the common between-study variance $\tau^2$ from the network meta-analysis, i.e. the square of list element x$\tau$.

If argument y is provided, the league table contains information on treatment comparisons from network meta-analysis object x in the lower triangle and from network meta-analysis object y in the upper triangle. This is, for example, useful to print information on efficacy and safety in the same league table.

By default, an R object with the league tables is generated. Alternatively, an Excel file is created if argument writexl = TRUE.

This implementation reports pairwise comparisons of the treatment in the column versus the treatment in the row in the lower triangle and row versus column in the upper triangle. This is a common presentation for network meta-analyses which allows to easily compare direction and magnitude of treatment effects. For example, given treatments A, B, and C, the results reported in the first row and second column as well as second row and first column are from the pairwise comparison A versus B. Note, this presentation is different from the printout of a network meta-analysis object which reports opposite pairwise comparisons in the lower and upper triangle, e.g., A versus B in the first row and second column and B versus A in the second row and first column.

If the same network meta-analysis object is used for arguments x and y, reciprocal treatment estimates will be shown in the upper triangle (see examples), e.g., the comparison B versus A.

R function netrank can be used to change the order of rows and columns in the league table (see examples).
Value

An object of class `netleague` with corresponding print function if `writexl = FALSE`. The object is a list containing the league tables in list elements 'common' and 'random'. An Excel file is created if `writexl = TRUE`. In this case, NULL is returned in R.

Author(s)

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

References


See Also

`netmeta`, `netposet`, `netrank`

Examples

```r
# Network meta-analysis of count mortality statistics
#
# data(Woods2010)

p0 <- pairwise(treatment, event = r, n = N,
  studlab = author, data = Woods2010, sm = "OR")
net0 <- netmeta(p0)

oldopts <- options(width = 100)

# League table for common and random effects model with
# - network estimates in lower triangle
# - direct estimates in upper triangle
#
# netleague(net0, digits = 2, bracket = "(", separator = " - ")

# League table for common effects model
#
# netleague(net0, random = FALSE, digits = 2)

# Change order of treatments according to treatment ranking (random
# effects model)
#
# netleague(net0, common = FALSE, digits = 2, seq = netrank(net0))
 #
# print(netrank(net0), common = FALSE)

## Not run:
# Create a CSV file with league table for random effects model
#
```
league0 <- netleague(net0, digits = 2, bracket = "(", separator = " to ")
# write.table(league0$random, file = "league0-random.csv",
# row.names = FALSE, col.names = FALSE, sep = ",")
# # Create Excel files with league tables
# # (if R package writexl is available)
# # netleague(net0, digits = 2, bracket = "(", separator = " to ",
# path = tempfile(fileext = ".xlsx"))

## End(Not run)

# Use depression dataset
#
data(Linde2015)

# Define order of treatments
#
trts <- c("TCA", "SSRI", "SNRI", "NRI",
  "Low-dose SARI", "NaSSa", "rMAO-A", "Hypericum", "Placebo")

# Outcome labels
#
outcomes <- c("Early response", "Early remission")

# (1) Early response
#
p1 <- pairwise(treat = list(treatment1, treatment2, treatment3),
  event = list(resp1, resp2, resp3), n = list(n1, n2, n3),
  studlab = id, data = Linde2015, sm = "OR")
# net1 <- netmeta(p1, common = FALSE,
#  seq = trts, ref = "Placebo", small = "undesirable")

# (2) Early remission
#
p2 <- pairwise(treat = list(treatment1, treatment2, treatment3),
  event = list(remi1, remi2, remi3), n = list(n1, n2, n3),
  studlab = id, data = Linde2015, sm = "OR")
# net2 <- netmeta(p2, common = FALSE,
#  seq = trts, ref = "Placebo", small = "undesirable")

options(width = 200)
netleague(net1, digits = 2)

netleague(net1, digits = 2, ci = FALSE)
netleague(net2, digits = 2, ci = FALSE)

# League table for two outcomes with
# - network estimates of first outcome in lower triangle
# - network estimates of second outcome in upper triangle
# netleague(net1, net2, digits = 2, ci = FALSE)

netleague(net1, net2, seq = netrank(net1), ci = FALSE)
netleague(net1, net2, seq = netrank(net2), ci = FALSE)

print(netrank(net1))
print(netrank(net2))

# Report results for network meta-analysis twice
# netleague(net1, net1, seq = netrank(net1), ci = FALSE, backtransf = FALSE)
netleague(net1, net1, seq = netrank(net1), ci = FALSE, backtransf = FALSE, direct = TRUE)

options(oldopts)

## Not run:
# Generate a partial order of treatment rankings
# np <- netposet(net1, net2, outcomes = outcomes)

# Requires R package 'hasse'
# hasse(np)
plot(np)

## End(Not run)

---

**netmatrix**

Create a matrix with additional information for pairwise comparisons

**Description**

Auxiliary function to create a matrix with additional information for pairwise comparisons

**Usage**

```r
netmatrix(  
  x,  
  var,  
  levels,  
  labels = levels,  
  func = "mode",  
  ties.method = "random"
)
```
Arguments

- **x**: A `netmeta` object.
- **var**: Variable with additional information.
- **levels**: An optional vector of the values that `var` might have taken (see `factor`).
- **labels**: An optional vector with labels for `var` (see `factor`).
- **func**: A character string with the function name to summarize values within pairwise comparisons; see Details.
- **ties.method**: A character string describing how ties are handled if `func = "mode"`; see Details.

Details

For each pairwise comparison, unique values will be calculated for the variable `var` based on the argument `func`: "mode" (most common value), "min" (minimum value), "max", "mean", "median", and "sum". In order to determine the most common value, the argument `ties.method` can be used in the case of ties with "first" meaning that the first / smallest value will be selected; similar for "last" (last / largest value) and "random" (random selection).

Value

A matrix with the same row and column names as the adjacency matrix `x$A.matrix`.

Author(s)

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

See Also

`netmeta`, `netgraph.netmeta`

Examples

data(smokingcessation)
# Add variable with (fictitious) risk of bias values
# with 1 = "low risk" and 2 = "high risk"
# smokingcessation$rob <- rep(1:2, 12)

pl <- pairwise(list(treat1, treat2, treat3),
    event = list(event1, event2, event3), n = list(n1, n2, n3),
    data = smokingcessation, sm = "OR")
net1 <- netmeta(pl, common = FALSE, ref = "A")

# Generate network graph with information on risk of bias
# col.rob <- netmatrix(net1, rob, ties.method = "last",
#     levels = 1:2, labels = c("green", "yellow"))
# netgraph(net1, plastic = FALSE, col = col.rob,
#     cex.points = 5, bg.points = "gray", adj = 0.5)
netmeasures

Measures for characterizing a network meta-analysis

Description

This function provides measures for quantifying the direct evidence proportion, the mean path
length and the minimal parallelism (the latter on aggregated and study level) of mixed treatment
comparisons (network estimates) as well as the evidence flow per design, see König et al. (2013).
These measures support the critical evaluation of the network meta-analysis results by rendering
transparent the process of data pooling.

Usage

netmeasures(
  x,
  random = x$random | !missing(tau.preset),
  tau.preset = x$tau.preset,
  warn = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

Arguments

x An object of class netmeta.
random A logical indicating whether random effects model should be used to calculate
  network measures.
tau.preset An optional value for the square-root of the between-study variance $\tau^2$.
warn A logical indicating whether warnings should be printed.
warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments
  are used.
... Additional arguments (to catch deprecated arguments).
Details

The direct evidence proportion gives the absolute contribution of direct effect estimates combined for two-arm and multi-arm studies to one network estimate.

Concerning indirectness, comparisons with a mean path length beyond two should be interpreted with particular caution, as more than two direct comparisons have to be combined serially on average.

Large indices of parallelism, either on study-level or on aggregated level, can be considered as supporting the validity of a network meta-analysis if there is only a small amount of heterogeneity.

The network estimates for two treatments are linear combinations of direct effect estimates comparing these or other treatments. The linear coefficients can be seen as the generalization of weights known from classical meta-analysis. These coefficients are given in the projection matrix $H$ of the underlying model. For multi-arm studies, the coefficients depend on the choice of the study-specific baseline treatment, but the absolute flow of evidence can be made explicit for each design as shown in König et al. (2013) and is given in $H.\tilde{t}$.

All measures are calculated based on the common effects meta-analysis by default. In the case that in function `netmeta` the argument `random = TRUE`, all measures are calculated for a random effects model. The value of the square-root of the between-study variance $\tau^2$ can also be prespecified by argument `tau.preset` in function `netmeta`.

Value

A list containing the following components:

- `random`, `tau.preset`
  - As defined above.
- `proportion`  
  - A named vector of the direct evidence proportion of each network estimate.
- `meanpath`  
  - A named vector of the mean path length of each network estimate.
- `minpar`  
  - A named vector of the minimal parallelism on aggregated level of each network estimate.
- `minpar.study`  
  - A named vector of the minimal parallelism on study level of each network estimate.
- `H.\tilde{t}`  
  - Design-based hat matrix with information on absolute evidence flow per design. The number of rows is equal to the number of possible pairwise treatment comparisons and the number of columns is equal to the number of designs.

Author(s)

Ulrike Krahn <ulrike.krahn@bayer.com>, Jochem König <koenigjo@uni-mainz.de>

References


See Also

`netmeta`
Examples

data(smokingcessation)

# Transform data from arm-based format to contrast-based format
#
p1 <- pairwise(list(treat1, treat2, treat3),
   event = list(event1, event2, event3), n = list(n1, n2, n3),
   data = smokingcessation, sm = "OR")

# Conduct network meta-analysis
#
net1 <- netmeta(p1)

# Calculate measures based on a common effects model
#
nm1 <- netmeasures(net1)

# Plot of minimal parallelism versus mean path length
#
plot(nm1$meanpath, nm1$minpar, type = "n",
   xlab = "Mean path length", ylab = "Minimal parallelism")
text(nm1$meanpath, nm1$minpar, names(nm1$meanpath), cex = 0.8)

## Not run:
data(Senn2013)

# Conduct common effects network meta-analysis with reference
treatment 'plac', i.e. placebo
#
net2 <- netmeta(TE, seTE, treat1, treat2, studlab,
   data = Senn2013, sm = "MD", reference = "plac", random = FALSE)

# Calculate measures based on a common effects model
#
nm2 <- netmeasures(net2)

# Plot of minimal parallelism versus mean path length
#
plot(nm2$meanpath, nm2$minpar, type = "n",
   xlab = "Mean path length", ylab = "Minimal parallelism")
text(nm2$meanpath, nm2$minpar, names(nm2$meanpath), cex = 0.8)

# Conduct random effects network meta-analysis with reference
treatment 'plac', i.e. placebo
#
net3 <- netmeta(TE, seTE, treat1, treat2, studlab,
   data = Senn2013, sm = "MD", reference = "plac", common = FALSE)

# Calculate measures based on a random effects model
#
nm3 <- netmeasures(net3)
Description

Network meta-analysis is a generalisation of pairwise meta-analysis that compares all pairs of treatments within a number of treatments for the same condition. The graph-theoretical approach for network meta-analysis uses methods that were originally developed in electrical network theory. It has been found to be equivalent to the frequentist approach to network meta-analysis which is based on weighted least squares regression (Rücker, 2012).

Print method for objects of class netmeta.

Usage

```r
netmeta(
  TE, 
  seTE, 
  treat1, 
  treat2, 
  studlab, 
  data = NULL, 
  subset = NULL, 
  sm, 
  level = gs("level"), 
  level.ma = gs("level.ma"), 
  common = gs("common"), 
  random = gs("random") | !is.null(tau.preset), 
  prediction = FALSE, 
  level.predict = gs("level.predict"), 
  reference.group, 
  baseline.reference = TRUE, 
  small.values = "desirable", 
  all.treatments = NULL, 
  seq = NULL, 
  method.tau = "DL", 
  tau.preset = NULL, 
  tol.multiarm = 0.001, 
  tol.multiarm.se = NULL, 
  details.chkmultiarm = FALSE, 
  sep.trts = ",", 
  nchar.trts = 666, 
  nchar.studlab = 666, 
  func.inverse = invmat, 
  n1 = NULL,
)```
n2 = NULL,
event1 = NULL,
event2 = NULL,
incr = NULL,
sd1 = NULL,
sd2 = NULL,
time1 = NULL,
time2 = NULL,
backtransf = gs("backtransf"),
title = "",
keepdata = gs("keepdata"),
control = NULL,
warn = TRUE,
warn.deprecated = gs("warn.deprecated"),
nchar = nchar.trts,
...
)

## S3 method for class 'netmeta'
print(
  x,
  common = x$common,
  random = x$random,
  prediction = x$prediction,
  reference.group = x$reference.group,
  baseline.reference = x$baseline.reference,
  all.treatments = x$all.treatments,
  backtransf = x$backtransf,
  nchar.trts = x$nchar.trts,
  header = TRUE,
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = max(gs("digits.pval"), 2),
  digits.pval.Q = max(gs("digits.pval.Q"), 2),
  digits.Q = gs("digits.Q"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.I2 = gs("digits.I2"),
  scientific.pval = gs("scientific.pval"),
  big.mark = gs("big.mark"),
  text.tau2 = gs("text.tau2"),
  text.tau = gs("text.tau"),
  text.I2 = gs("text.I2"),
  legend = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
Arguments

**TE**  
Estimate of treatment effect, i.e. difference between first and second treatment (e.g. log odds ratio, mean difference, or log hazard ratio). Or an R object created with `pairwise`.

**seTE**  
Standard error of treatment estimate.

**treat1**  
Label/Number for first treatment.

**treat2**  
Label/Number for second treatment.

**studlab**  
An optional - but important! - vector with study labels (see Details).

**data**  
An optional data frame containing the study information.

**subset**  
An optional vector specifying a subset of studies to be used.

**sm**  
A character string indicating underlying summary measure, e.g., "RD", "RR", "OR", "ASD", "HR", "MD", "SMD", or "ROM".

**level**  
The level used to calculate confidence intervals for individual comparisons.

**level.ma**  
The level used to calculate confidence intervals for network estimates.

**common**  
A logical indicating whether results for the common effects model should be printed.

**random**  
A logical indicating whether results for the random effects model should be printed.

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

**level.predict**  
The level used to calculate prediction intervals for a new study.

**reference.group**  
Reference treatment.

**baseline.reference**  
A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa. This argument is only considered if `reference.group` has been specified.

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

**all.treatments**  
A logical or "NULL". If TRUE, matrices with all treatment effects, and confidence limits will be printed.

**seq**  
A character or numerical vector specifying the sequence of treatments in printouts.

**method.tau**  
A character string indicating which method is used to estimate the between-study variance $\tau^2$ and its square root $\tau$. Either "DL", "REML", or "ML", can be abbreviated.

**tau.preset**  
An optional value for manually setting the square-root of the between-study variance $\tau^2$.

**tol.multiarm**  
A numeric for the tolerance for consistency of treatment estimates in multi-arm studies which are consistent by design.
tol.multiarm.se
A numeric for the tolerance for consistency of standard errors in multi-arm studies which are consistent by design. This check is not conducted if the argument is NULL.

details.chkmultiarm
A logical indicating whether treatment estimates and/or variances of multi-arm studies with inconsistent results or negative multi-arm variances should be printed.

deprecated
A character used in comparison names as separator between treatment labels.

nchar.trts
A numeric defining the minimum number of characters used to create unique treatment names.

nchar.studlab
A numeric defining the minimum number of characters used to create unique study labels.

func.inverse
R function used to calculate the pseudoinverse of the Laplacian matrix L (see Details).

n1
Number of observations in first treatment group.

n2
Number of observations in second treatment group.

event1
Number of events in first treatment group.

event2
Number of events in second treatment group.

incr
Numerical value added to cell frequencies (for details, see pairwise).

sd1
Standard deviation in first treatment group.

sd2
Standard deviation in second treatment group.

time1
Person time at risk in first treatment group.

time2
Person time at risk in second treatment group.

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

title
Title of meta-analysis/systematic review.

keepdata
A logical indicating whether original data (set) should be kept in netmeta object.

control
An optional list to control the iterative process to estimate the between-study variance \( \tau^2 \). This argument is passed on to rma.mv.

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.

deprecated
A numeric for the tolerance for consistency of standard errors in multi-arm studies which are consistent by design. This check is not conducted if the argument is NULL.

nchar
Deprecated argument (replaced by nchar.trts).

...
Additional arguments.

x
An object of class netmeta.

header
A logical indicating whether information on title of meta-analysis, comparison and outcome should be printed at the beginning of the printout.
**Details**

Network meta-analysis using R package **netmeta** is described in detail in Schwarzer et al. (2015), Chapter 8.

Let $n$ be the number of different treatments (nodes, vertices) in a network and let $m$ be the number of existing comparisons (edges) between the treatments. If there are only two-arm studies, $m$ is the number of studies. Let $\text{TE}$ and $\text{seTE}$ be the vectors of observed effects and their standard errors. Let $W$ be the $m \times m$ diagonal matrix that contains the inverse variance $1 / \text{seTE}^2$. The given comparisons define the network structure. Therefrom an $m \times n$ design matrix $X$ (edge-vertex incidence matrix) is formed; for more precise information, see Rücker (2012). Moreover, the $n \times n$ Laplacian matrix $L$ and its Moore-Penrose pseudoinverse $L^+$ are calculated (both matrices play an important role in graph theory and electrical network theory). Using these matrices, the variances based on both direct and indirect comparisons can be estimated. Moreover, the hat matrix $H$ can be estimated by $H = XL + X^tW = X(X^tW X)^+X^tW$ and finally consistent treatment effects can be estimated by applying the hat matrix to the observed (potentially inconsistent) effects. $H$ is a projection matrix which maps the observed effects onto the consistent $(n-1)$-dimensional subspace. This is the Aitken estimator (Senn et al., 2013). As in pairwise meta-analysis, the $Q$ statistic measures the deviation from consistency. $Q$ can be separated into parts for each pairwise meta-analysis and a part for remaining inconsistency between comparisons.

Often multi-arm studies are included in a network meta-analysis. In multi-arm studies, the treatment effects on different comparisons are not independent, but correlated. This is accounted for by reweighting all comparisons of each multi-arm study. The method is described in Rücker (2012) and Rücker and Schwarzer (2014).

Comparisons belonging to multi-arm studies are identified by identical study labels (argument `studlab`). It is therefore important to use identical study labels for all comparisons belonging to the

**Parameters**

- **digits**: Minimal number of significant digits, see `print.default`.
- **digits.stat**: Minimal number of significant digits for tests of overall effect, see `print.default`.
- **digits.pval**: Minimal number of significant digits for p-value of overall effects, see `print.default`.
- **digits.pval.Q**: Minimal number of significant digits for p-value of heterogeneity tests, see `print.default`.
- **digits.Q**: Minimal number of significant digits for heterogeneity statistics, 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.
- **text.tau2**: Text printed to identify between-study variance $\tau^2$.
- **text.tau**: Text printed to identify $\tau$, the square root of the between-study variance $\tau^2$.
- **text.I2**: Text printed to identify heterogeneity statistic $I^2$.
- **legend**: A logical indicating whether a legend should be printed.

### Examples

```r
```

This example sets the number of significant digits to 3, prints p-values in scientific notation, uses a comma as thousands separator, and prints the text labels for various statistics. A legend is also included.

### Notes

- The `netmeta` package provides functions for conducting network meta-analysis, including estimation of treatment effects, testing of consistency and heterogeneity, and visualization of results.
- It supports various methods of meta-analysis, such as fixed-effect and random-effects models, and allows for the inclusion of multiple treatment comparisons.
- The package is designed to handle both direct and indirect comparisons, making it suitable for complex network meta-analyses.
- Users can customize the presentation of results by adjusting parameters such as the number of significant digits, scientific notation, and text labels.

---

**References**

same multi-arm study, e.g., study label "Willms1999" for the three-arm study in the data example (Senn et al., 2013). The function netmeta then automatically accounts for within-study correlation by reweighting all comparisons of each multi-arm study.

Data entry for this function is in contrast-based format, that is, data are given as contrasts (differences) between two treatments (argument \( \text{TE} \)) with standard error (argument \( \text{seTE} \)). In principle, meta-analysis functions from R package meta, e.g. \text{metabin} for binary outcomes or \text{metacont} for continuous outcomes, can be used to calculate treatment effects separately for each treatment comparison which is a rather tedious enterprise. If data are provided in arm-based format, that is, data are given for each treatment arm separately (e.g. number of events and participants for binary outcomes), a much more convenient way to transform data into contrast-based form is available. Function \text{pairwise} can automatically transform data with binary outcomes (using the \text{metabin} function from R package \text{meta}), continuous outcomes (\text{metacont} function), incidence rates (\text{metainc} function), and generic outcomes (\text{metagen} function). Additional arguments of these functions can be provided (see help page of function \text{pairwise}).

Note, all pairwise comparisons must be provided for a multi-arm study. Consider a multi-arm study of \( p \) treatments with known variances. For this study, treatment effects and standard errors must be provided for each of \( p(p - 1) / 2 \) possible comparisons. For instance, a three-arm study contributes three pairwise comparisons, a four-arm study even six pairwise comparisons. Function \text{pairwise} automatically calculates all pairwise comparisons for multi-arm studies.

A simple random effects model assuming that a constant heterogeneity variance is added to each comparison of the network can be defined via a generalised methods of moments estimate of the between-studies variance \( \tau^2 \) (Jackson et al., 2012). This is added to the observed sampling variance \( \text{seTE}^2 \) of each comparison in the network (before appropriate adjustment for multi-arm studies). Then, as in standard pairwise meta-analysis, the procedure is repeated with the resulting enlarged standard errors.

For the random-effects model, the direct treatment estimates are based on the common between-study variance \( \tau^2 \) from the network meta-analysis.

Internally, both common and random effects models are calculated regardless of values choosen for arguments \text{common} and \text{random}. Accordingly, the network estimates for the random effects model can be extracted from component \text{TE.random} of an object of class "netmeta" even if argument \text{random} = FALSE. However, all functions in R package netmeta will adequately consider the values for common and random. E.g. function \text{print.summary.netmeta} will not print results for the random effects model if \text{random} = FALSE.

By default, treatment names are not abbreviated in printouts. However, in order to get more concise printouts, argument \text{nchar.trts} can be used to define the minimum number of characters for abbreviated treatment names (see \text{abbreviate}, argument \text{minlength}). R function \text{treats} is utilised internally to create abbreviated treatment names.

Names of treatment comparisons are created by concatenating treatment labels of pairwise comparisons using \text{sep.trts} as separator (see \text{paste}). These comparison names are used in the covariance matrices \text{Cov.common} and \text{Cov.random} and in some R functions, e.g, \text{decomp.design}. By default, a colon is used as the separator. If any treatment label contains a colon the following characters are used as separator (in consecutive order): ":", ",", "/", "+", ",", ":", ",", and ":". If all of these characters are used in treatment labels, a corresponding error message is printed asking the user to specify a different separator.
Value

An object of class netmeta with corresponding print, summary, forest, and netrank functions. The object is a list containing the following components:

- `studlab, treat1, treat2, TE, seTE`
  - As defined above.
- `seTE.adj.common, seTE.adj.random`
  - Standard error of treatment estimate, adjusted for multi-arm studies.
- `design`
  - Design of study providing pairwise comparison.
- `n1, n2, event1, event2, incr`
  - As defined above.
- `mean1, mean2, sd1, sd2, time1, time2`
  - As defined above.
- `sd1, sd2, time1, time2`
  - As defined above.
- `k`
  - Total number of studies.
- `m`
  - Total number of pairwise comparisons.
- `n`
  - Total number of treatments.
- `d`
  - Total number of designs (corresponding to the unique set of treatments compared within studies).
- `trts`
  - Treatments included in network meta-analysis.
- `k.trts`
  - Number of studies evaluating a treatment.
- `n.trts`
  - Number of observations receiving a treatment (if arguments `n1` and `n2` are provided).
- `events.trts`
  - Number of events observed for a treatment (if arguments `event1` and `event2` are provided).
- `multiarm`
  - Logical vector to identify pairwise comparisons from multi-arm studies.
- `n.arms`
  - Number of treatment arms in study providing pairwise comparison.
- `studies`
  - Vector with unique study labels.
- `narms`
  - Number of arms for each study.
- `designs`
  - Vector with unique designs present in the network. A design corresponds to the set of treatments compared within a study.
- `TE.nma.common, TE.nma.random`
  - A vector of length `m` of consistent treatment effects estimated by network meta-analysis (nma) (common / random effects model).
- `seTE.nma.common, seTE.nma.random`
  - A vector of length `m` of effective standard errors estimated by network meta-analysis (common / random effects model).
- `lower.nma.common, lower.nma.random`
  - A vector of length `m` of lower confidence interval limits for consistent treatment effects estimated by network meta-analysis (common effects / random effects model).
upper.nma.common, upper.nma.random

A vector of length $m$ of upper confidence interval limits for the consistent treatment effects estimated by network meta-analysis (common effects / random effects model).

statistic.nma.common, statistic.nma.random

A vector of length $m$ of z-values for test of treatment effect for individual comparisons (common / random effects model).

pval.nma.common, pval.nma.random

A vector of length $m$ of p-values for test of treatment effect for individual comparisons (common / random effects model).

leverage.common

A vector of length $m$ of leverages, interpretable as factors by which variances are reduced using information from the whole network.

w.common, w.random

A vector of length $m$ of weights of individual studies (common / random effects model).

Q.common

A vector of length $m$ of contributions to total heterogeneity / inconsistency statistic.

TE.common, TE.random

$n \times n$ matrix with estimated overall treatment effects (common / random effects model).

seTE.common, seTE.random

$n \times n$ matrix with standard errors (common / random effects model).

lower.common, upper.common, lower.random, upper.random

$n \times n$ matrices with lower and upper confidence interval limits (common / random effects model).

statistic.common, pval.common, statistic.random, pval.random

$n \times n$ matrices with z-value and p-value for test of overall treatment effect (common / random effects model).

seTE.predict

$n \times n$ matrix with standard errors for prediction intervals.

lower.predict, upper.predict

$n \times n$ matrices with lower and upper prediction interval limits.

prop.direct.common, prop.direct.random

A named vector of the direct evidence proportion of each network estimate. (common effects / random effects model).

TE.direct.common, TE.direct.random

$n \times n$ matrix with estimated treatment effects from direct evidence (common effects / random effects model).

seTE.direct.common, seTE.direct.random

$n \times n$ matrix with estimated standard errors from direct evidence (common effects / random effects model).

lower.direct.common, upper.direct.common, lower.direct.random,

$n \times n$ matrices with lower and upper confidence interval limits from direct evidence (common / random effects model).

upper.direct.random

$n \times n$ matrices with lower and upper confidence interval limits from direct evidence (common effects / random effects model).
statistic.direct.common, pval.direct.common, statistic.direct.random,
  n x n matrices with z-value and p-value for test of overall treatment effect from
direct evidence (common / random effects model).

pval.direct.random
  n x n matrices with z-value and p-value for test of overall treatment effect from
direct evidence (common / random effects model).

TE.indirect.common, TE.indirect.random
  n x n matrix with estimated treatment effects from indirect evidence (common /random effects model).

seTE.indirect.common, seTE.indirect.random
  n x n matrix with estimated standard errors from indirect evidence (common /random effects model).

lower.indirect.common, upper.indirect.common, lower.indirect.random,
  n x n matrices with lower and upper confidence interval limits from indirect evi-
dence (common / random effects model).

upper.indirect.random
  n x n matrices with lower and upper confidence interval limits from indirect evi-
dence (common / random effects model).

statistic.indirect.common, pval.indirect.common, statistic.indirect.random,
  n x n matrices with z-value and p-value for test of overall treatment effect from
indirect evidence (common / random effects model).

pval.indirect.random
  n x n matrices with z-value and p-value for test of overall treatment effect from
indirect evidence (common / random effects model).

Q
  Overall heterogeneity / inconsistency statistic.

df.Q
  Degrees of freedom for test of heterogeneity / inconsistency.

pval.Q
  P-value for test of heterogeneity / inconsistency.

I2, lower.I2, upper.I2
  I-squared, lower and upper confidence limits.

tau
  Square-root of between-study variance.

Q.heterogeneity
  Overall heterogeneity statistic.

df.Q.heterogeneity
  Degrees of freedom for test of overall heterogeneity.

pval.Q.heterogeneity
  P-value for test of overall heterogeneity.

Q.inconsistency
  Overall inconsistency statistic.

df.Q.inconsistency
  Degrees of freedom for test of overall inconsistency.

pval.Q.inconsistency
  P-value for test of overall inconsistency.

Q.decomp
  Data frame with columns 'treat1', 'treat2', 'Q', 'df' and 'pval.Q', providing heterogeneity statistics for each pairwise meta-analysis of direct comparisons.
A. matrix Adjacency matrix \((nxn)\).
X. matrix Design matrix \((mxn)\).
B. matrix Edge-vertex incidence matrix \((mxn)\).

L. matrix.common, L. matrix.random
Laplacian matrix \((nxn)\).
Lplus. matrix.common, Lplus. matrix.random
Moore-Penrose pseudoinverse of the Laplacian matrix \((nxn)\).
Q. matrix Matrix of heterogeneity statistics for pairwise meta-analyses, where direct comparisons exist \((nxn)\).
G. matrix Matrix with variances and covariances of comparisons \((mxm)\). G is defined as \(BL+B^t\).
H. matrix.common, H. matrix.random
Hat matrix \((mxm)\), defined as \(H = GW = BL+B^tW\).
n. matrix \(nxn\) matrix with number of observations in direct comparisons (if arguments \(n_1\) and \(n_2\) are provided).
events. matrix \(nxn\) matrix with number of events in direct comparisons (if arguments \(event_1\) and \(event_2\) are provided).
P. common, P. random \(nxn\) matrix with direct evidence proportions (common / random effects model).
Cov. common Variance-covariance matrix (common effects model)
Cov. random Variance-covariance matrix (random effects model)
sm, level, level.ma
As defined above.
common, random As defined above.
prediction, level.predict
As defined above.
reference.group, baseline.reference, small.values, all.treatments
As defined above.
seq, tau.preset, tol.multiarm, tol.multiarm.se
As defined above.
details.chkmultiarm, sep.trts, nchar.trts
As defined above.
backtransf, title, warn, warn.deprecated
As defined above.
call Function call.
version Version of R package netmeta used to create object.

Note
R function rma.mv from R package metafor (Viechtbauer 2010) is called internally to estimate the between-study variance \(\tau^2\) for the (restricted) maximum likelihood method. For binary outcomes, incidence rates, and the mean difference, the variance-covariance matrix is calculated if arguments \(event_1\), \(event_2\), \(n_1\), and \(n_2\) (binary outcomes); \(event_1\), \(event_2\), \(time_1\), and \(time_2\) (incidence rates); \(n_1\), \(n_2\), \(sd_1\), and \(sd_2\) (mean difference) are provided. For data sets preprocessed with pairwise the respective variables are selected automatically.
Author(s)

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

References

Rücker G, Schwarzer G (2014): Reduce dimension or reduce weights? Comparing two approaches to multi-arm studies in network meta-analysis. Statistics in Medicine, 33, 4353–69

See Also

pairwise, forest.netmeta, netrank, metagen

Examples

data(smokingcessation)

# Transform data from arm-based format to contrast-based format
#
p1 <- pairwise(list(treat1, treat2, treat3),
                event = list(event1, event2, event3),
                n = list(n1, n2, n3),
                data = smokingcessation, sm = "OR")

# Conduct random effects network meta-analysis
#
net1 <- netmeta(p1, common = FALSE)
net1

## Not run:
data(Senn2013)

# Conduct common effects network meta-analysis
#
net2 <- netmeta(TE, seTE, treat1, treat2, studlab,
                data = Senn2013, sm = "MD", random = FALSE)
net2
net2$Q.decomp

# Comparison with reference group
#
print(net2, reference = "plac")

# Conduct random effects network meta-analysis
#
net3 <- netmeta(TE, seTE, treat1, treat2, studlab,
data = Senn2013, sm = "MD", common = FALSE)
net3

# Change printing order of treatments with placebo last and use
# long treatment names
#
trts <- c("acar", "benf", "metf", "migl", "pio2",
          "rosi", "sita", "sulf", "vild", "plac")
net4 <- netmeta(TE, seTE, treat1.long, treat2.long, studlab,
data = Senn2013, sm = "MD", common = FALSE,
      seq = trts, reference = "Placebo")
print(net4, digits = 2)

## End(Not run)

netmetabin

Network meta-analysis of binary outcome data

Description

Provides three models for the network meta-analysis of binary data (Mantel-Haenszel method, based on the non-central hypergeometric distribution, and the inverse variance method).

Usage

netmetabin(
  event1,
  n1,
  event2,
  n2,
  treat1,
  treat2,
  studlab,
data = NULL,
  subset = NULL,
  sm,
  method = "MH",
  cc.pooled = FALSE,
  incr,
  allincr,
  addincr,
  allstudies,
  level = gs("level"),
)
level.ma = gs("level.ma"),
common = gs("common"),
random = method == "Inverse" & (gs("random") | !is.null(tau.preset)),
prediction = FALSE,
level.predict = gs("level.predict"),
reference.group = "",
baseline.reference = TRUE,
all.treatments = NULL,
seq = NULL,
tau.preset = NULL,
tol.multiarm = 0.001,
tol.multiarm.se = NULL,
details.chkmultiarm = FALSE,
details.chkdata = TRUE,
sep.trts = ":",
nchar.trts = 666,
func.inverse = invmat,
backtransf = gs("backtransf"),
title = "",
keepdata = gs("keepdata"),
warn = TRUE,
warn.deprecated = gs("warn.deprecated"),
...
)

Arguments

event1 Number of events (first treatment).
n1 Number of observations (first treatment).
event2 Number of events (second treatment).
n2 Number of observations (second treatment)
treat1 Label/Number for first treatment.
treat2 Label/Number for second treatment.
studlab An optional - but important! - vector with study labels (see Details).
data An optional data frame containing the study information.
subset An optional vector specifying a subset of studies to be used.
sm A character string indicating underlying summary measure, i.e., "RD", "RR", "OR", "ASD".
method A character string indicating which method is to be used for pooling of studies.
One of "Inverse", "MH", or "NCH", can be abbreviated.
cc.pooled A logical indicating whether incr should be used as a continuity correction,
when calculating the network meta-analysis estimates.
incr A numerical value which is added to each cell count, i.e., to the numbers of
events and non-events, of all treatment arms in studies with zero events or non-
events in any of the treatment arms ("continuity correction").
allincr A logical indicating whether incr should be added to each cell count of all studies if a continuity correction was used for at least one study (only considered if method = "Inverse"). If FALSE (default), incr is used as continuity correction only for studies with zero events or zero non-events in any of the treatment arms.

addincr A logical indicating whether incr should be added to each cell count of all studies, irrespective of zero cell counts (only considered if method = "Inverse").

allstudies A logical indicating whether studies with zero events or non-events in all treatment arms should be included in an inverse variance meta-analysis (applies only if method = "Inverse" and sm is equal to either "RR" or "OR").

level The level used to calculate confidence intervals for individual studies.

level.ma The level used to calculate confidence intervals for network estimates.

common A logical indicating whether a common effects network meta-analysis should be conducted.

random A logical indicating whether a random effects network meta-analysis should be conducted.

prediction A logical indicating whether a prediction interval should be printed (only considered if method = "Inverse").

level.predict The level used to calculate prediction interval for a new study (only considered if method = "Inverse").


baseline.reference A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa. This argument is only considered if reference.group has been specified.

all.treatments A logical or "NULL". If TRUE, matrices with all treatment effects, and confidence limits will be printed.

seq A character or numerical vector specifying the sequence of treatments in printouts.

tau.preset An optional value for manually setting the square-root of the between-study variance \( \tau^2 \) (only considered if method = "Inverse").

tol.multiarm A numeric for the tolerance for consistency of treatment estimates in multi-arm studies which are consistent by design (only considered if method = "Inverse").

tol.multiarm.se A numeric for the tolerance for consistency of standard errors in multi-arm studies which are consistent by design (only considered if the argument is not NULL and method = "Inverse").

details.chkmultiarm A logical indicating whether treatment estimates and / or variances of multi-arm studies with inconsistent results or negative multi-arm variances should be printed (only considered if method = "Inverse").

details.chkdata A logical indicating whether number of events and participants of studies with inconsistent data should be printed.
sep.trts  A character used in comparison names as separator between treatment labels.
nchar.trts  A numeric defining the minimum number of characters used to create unique treatment names (see Details).
func.inverse  R function used to calculate the pseudoinverse of the Laplacian matrix L (see netmeta).
backtransf  A logical indicating whether results should be back transformed in printouts and forest plots. If backtransf = TRUE, results for sm = "OR" are presented as odds ratios rather than log odds ratios, for example.
title  Title of meta-analysis / systematic review.
keepdata  A logical indicating whether original data (set) should be kept in netmeta object.
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.
...  Additional arguments (to catch deprecated arguments).

Details

This function implements three models for the network meta-analysis of binary data:

- The Mantel-Haenszel network meta-analysis model, as described in Efthimiou et al. (2019) (method = "MH");
- a network meta-analysis model using the non-central hypergeometric distribution with the Breslow approximation, as described in Stijnen et al. (2010) (method = "NCH");
- the inverse variance method for network meta-analysis (method = "Inverse"), also provided by netmeta.

Comparisons belonging to multi-arm studies are identified by identical study labels (argument studlab). It is therefore important to use identical study labels for all comparisons belonging to the same multi-arm study.

Data entry for this function is in contrast-based format, that is, each line of the data corresponds to a single pairwise comparison between two treatments (arguments treat1, treat2, event1, n1, event2, and n2). If data are provided in arm-based format, that is, number of events and participants are given for each treatment arm separately, function pairwise can be used to transform the data to contrast-based format (see help page of function pairwise).

Note, all pairwise comparisons must be provided for a multi-arm study. Consider a multi-arm study of p treatments with known variances. For this study, the number of events and observations must be provided for each treatment, for each of p(p - 1) / 2 possible comparisons in separate lines in the data. For instance, a three-arm study contributes three pairwise comparisons, a four-arm study even six pairwise comparisons. Function pairwise automatically calculates all pairwise comparisons for multi-arm studies.

For method = "Inverse", both common and random effects models are calculated regardless of values choosen for arguments common and random. Accordingly, the network estimates for the random effects model can be extracted from component TE.random of an object of class "netmeta"
even if argument random = FALSE. However, all functions in R package netmeta will adequately consider the values for common and random. E.g. function print.summary.netmeta will not print results for the random effects model if random = FALSE.

For the random-effects model, the direct treatment estimates are based on the common between-study variance \( \tau^2 \) from the network meta-analysis.

For method = "MH" and method = "NCH", only a common effects model is available.

By default, treatment names are not abbreviated in printouts. However, in order to get more concise printouts, argument nchar.trts can be used to define the minimum number of characters for abbreviated treatment names (see abbreviate, argument minlength). R function treats is utilised internally to create abbreviated treatment names.

Names of treatment comparisons are created by concatenating treatment labels of pairwise comparisons using sep.trts as separator (see paste). These comparison names are used in the covariance matrices Cov.common and Cov.random and in some R functions, e.g. decomp.design. By default, a colon is used as the separator. If any treatment label contains a colon the following characters are used as separator (in consecutive order): ",","_","/","+",".","|", and "*". If all of these characters are used in treatment labels, a corresponding error message is printed asking the user to specify a different separator.

Value

An object of class netmetabin and netmeta with corresponding print, summary, forest, and netrank functions. The object is a list containing the following components:

- studlab, treat1, treat2
  As defined above.
- n1, n2, event1, event2
  As defined above.
- TE
  Estimate of treatment effect, i.e. difference between first and second treatment (e.g. log odds ratio).
- seTE
  Standard error of treatment estimate.
- k
  Total number of studies.
- m
  Total number of pairwise comparisons.
- n
  Total number of treatments.
- d
  Total number of designs (corresponding to the unique set of treatments compared within studies).
- trts
  Treatments included in network meta-analysis.
- k.trts
  Number of studies evaluating a treatment.
- n.trts
  Number of observations receiving a treatment.
- events.trts
  Number of events observed for a treatment.
- studies
  Study labels coerced into a factor with its levels sorted alphabetically.
- narms
  Number of arms for each study.
- designs
  Unique list of designs present in the network. A design corresponds to the set of treatments compared within a study.
**TE.common**, seTE.common

$n \times n$ matrix with estimated overall treatment effects and standard errors for common effects model.

**lower.common**, upper.common

$n \times n$ matrices with lower and upper confidence interval limits for common effects model.

**statistic.common**, pval.common

$n \times n$ matrices with z-value and p-value for test of overall treatment effect under common effects model.

**TE.random**, seTE.random

$n \times n$ matrix with estimated overall treatment effects and standard errors for random effects model (only available if method = "Inverse").

**lower.random**, upper.random

$n \times n$ matrices with lower and upper confidence interval limits for random effects model (only available if method = "Inverse").

**statistic.random**, pval.random

$n \times n$ matrices with z-value and p-value for test of overall treatment effect under random effects model (only available if method = "Inverse").

**TE.direct.common**, seTE.direct.common

$n \times n$ matrix with estimated treatment effects and standard errors from direct evidence under common effects model.

**lower.direct.common**, upper.direct.common

$n \times n$ matrices with lower and upper confidence interval limits from direct evidence under common effects model.

**statistic.direct.common**, pval.direct.common

$n \times n$ matrices with z-value and p-value for test of overall treatment effect from direct evidence under common effects model.

**TE.direct.random**, seTE.direct.random

$n \times n$ matrix with estimated treatment effects and standard errors from direct evidence under random effects model (only available if method = "Inverse").

**lower.direct.random**, upper.direct.random

$n \times n$ matrices with lower and upper confidence interval limits from direct evidence under random effects model (only available if method = "Inverse").

**statistic.direct.random**, pval.direct.random

$n \times n$ matrices with z-value and p-value for test of overall treatment effect from direct evidence under random effects model (only available if method = "Inverse").

**Q**  
Overall heterogeneity / inconsistency statistic. (only available if method = "Inverse")

**df.Q**  
Degrees of freedom for test of heterogeneity / inconsistency.

**pval.Q**  
P-value for test of heterogeneity / inconsistency.

**I2**, lower.I2, upper.I2  
I-squared, lower and upper confidence limits (only available if method = "Inverse").

**tau**  
Square-root of between-study variance (only available if method = "Inverse").

**Q.heterogeneity**  
Overall heterogeneity statistic. (only available if method = "Inverse")

**df.Q.heterogeneity**  
Degrees of freedom for test of overall heterogeneity.
pval.Q.heterogeneity
P-value for test of overall heterogeneity.

Q.inconsistency
Overall inconsistency statistic.

df.Q.inconsistency
Degrees of freedom for test of overall inconsistency.

pval.Q.inconsistency
P-value for test of overall inconsistency.

A.matrix
Adjacency matrix (n x n).

H.matrix
Hat matrix (m x m)

n.matrix
n x n matrix with number of observations in direct comparisons.

events.matrix
n x n matrix with number of events in direct comparisons.

sm, method, level, level.ma
As defined above.

incr, allincr, addincr, allstudies, cc.pooled
As defined above.

common, random
As defined above.

prediction, level.predict
As defined above.

reference.group, baseline.reference, all.treatments
As defined above.

seq, tau.preset, tol.multiarm, tol.multiarm.se
As defined above.

details.chkmultiarm, details.chkdata
As defined above.

sep.trts, nchar.trts
As defined above.

backtransf, title, warn, warn.deprecated
As defined above.

data
Data set (in contrast-based format).

data.design
List with data in arm-based format (each list element corresponds to a single design).

call
Function call.

version
Version of R package netmeta used to create object.

Author(s)
Orestis Efthimiou <oremiou@gmail.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References


See Also

`pairwise, netmeta`

Examples

data(Dong2013)

# Only consider first ten studies (to reduce runtime of example)
#
first10 <- subset(Dong2013, id <= 10)

# 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.
#
p1 <- pairwise(treatment, death, randomized, studlab = id,
               data = first10, sm = "OR")

# Conduct Mantel-Haenszel network meta-analysis (without continuity
# correction)
#
nb1 <- netmetabin(p1, ref = "plac")
nb1

# Obtain the league table
#
netleague(nb1)

## Not run:
# Conduct Mantel-Haenszel network meta-analysis for the whole
# dataset
#
p2 <- pairwise(treatment, death, randomized, studlab = id,
               data = Dong2013, sm = "OR")
netmetabin(p2, ref = "plac")

# Conduct network meta-analysis using the non-central
# hypergeometric model (without continuity correction)
#
netmetabin(p2, ref = "plac", method = "NCH")

# Conduct Mantel-Haenszel network meta-analysis (with continuity
# correction of 0.5; include all studies)
#
netmetabin(p2, ref = "plac", cc.pooled = TRUE)
```r
data(Gurusamy2011)

p3 <- pairwise(treatment, death, n, studlab = study,
               data = Gurusamy2011, sm = "OR")

# Conduct Mantel-Haenszel network meta-analysis (without continuity
# correction)
#
# netmetabin(p3, ref = "cont")

## End(Not run)
```

---

**netpairwise**

Conduct pairwise meta-analyses for all comparisons with direct evidence in a network meta-analysis.

### Description

Conduct pairwise meta-analyses for all comparisons with direct evidence in a network meta-analysis.

### Usage

```r
netpairwise(
  x,
  separate = FALSE,
  common = x$common,
  random = x$random,
  level = x$level,
  level.ma = x$level.ma,
  prediction = x$prediction,
  level.predict = x$level.predict,
  reference.group = if (missing(order)) x$reference.group else "",
  baseline.reference = x$baseline.reference,
  method.tau = x$method.tau,
  order = NULL,
  sep.trts = x$sep.trts,
  nchar.trts = x$nchar.trts,
  backtransf = x$backtransf,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

## S3 method for class 'netpairwise'

print(x, ...)

## S3 method for class 'netpairwise'

summary(object, ...)
## S3 method for class 'summary.netpairwise'
print(x, ...)

## S3 method for class 'netpairwise'
forest(x, ...)

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

## S3 method for class 'netpairwise'
funnel(x, k.min = 3, ...)

## S3 method for class 'netpairwise'
radiat(x, k.min = 3, ...)

## S3 method for class 'netpairwise'
baujat(x, k.min = 3, ...)

## S3 method for class 'netpairwise'
metabias(x, k.min = 10, ...)

## S3 method for class 'metabias.netpairwise'
print(x, ...)

## S3 method for class 'netpairwise'
trimfill(x, k.min = 3, ...)

## S3 method for class 'trimfill.netpairwise'
print(x, ...)

## S3 method for class 'netpairwise'
metainf(x, k.min = 2, ...)

## S3 method for class 'metainf.netpairwise'
print(x, ...)

## S3 method for class 'netpairwise'
metacum(x, k.min = 2, ...)

## S3 method for class 'metacum.netpairwise'
print(x, ...)

## S3 method for class 'netpairwise'
metareg(x, ..., k.min = 2)

## S3 method for class 'metareg.netpairwise'
print(x, ...)
Arguments

x          An object of class netmeta or netpairwise.
separate   A logical indicating whether results for pairwise comparisons should be printed as separate meta-analyses or as subgroups which is more concise.
common     A logical indicating whether a common effects network meta-analysis should be conducted.
random     A logical indicating whether a random effects network meta-analysis should be conducted.
level      The level used to calculate confidence intervals for individual comparisons.
level.ma   The level used to calculate confidence intervals for pooled estimates.
prediction  A logical indicating whether prediction intervals should be printed.
level.predict The level used to calculate prediction intervals for a new study.
baseline.reference A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa. This argument is only considered if reference.group has been specified.
method.tau  A character string indicating which method is used to estimate the between-study variance $\tau^2$ and its square root $\tau$. Either "DL", "REML", or "ML", can be abbreviated.
order      An optional character or numerical vector specifying the order of treatments.
sep.trts   A character used in comparison names as separator between treatment labels.
nchar.trts A numeric defining the minimum number of characters used to create unique treatment names (see Details).
backtransf A logical indicating whether results should be back transformed in printouts and forest plots. If backtransf = TRUE, results for sm = "OR" are presented 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 metagen or print functions and to catch deprecated arguments).
object     An object of class netpairwise.
k.min      Minimum number of studies in pairwise comparison to show funnel plot, radial plot or conduct test for funnel plot asymmetry.

Details

Conduct pairwise meta-analyses for all comparisons with direct evidence in a network meta-analysis. In contrast to netmeta and netsplit, unadjusted standard errors are used in the calculations and the between-study heterogeneity variance is allowed to differ between comparisons.

The R function metagen is called internally.
Value

Either a single `metagen` object with pairwise comparisons as subgroups or a list with `metagen` objects for each direct pairwise comparison.

Note

This function must not be confused with `pairwise` which can be used as a pre-processing step to convert data from arm-based to contrast-based format by calculating all pairwise comparisons within a study.

Author(s)

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

See Also

`netmeta`, `netsplit`, `pairwise`

Examples

```r
# Transform data from arm-based format to contrast-based format
p1 <- pairwise(list(treat1, treat2, treat3),
               event = list(event1, event2, event3),
               n = list(n1, n2, n3),
               data = smokingcessation, sm = "OR")

# Conduct random effects network meta-analysis
net1 <- netmeta(p1, common = FALSE)

# Calculate and print concise results for all pairwise meta-analyses
np1 <- netpairwise(net1)
np1
print(np1, details.method = FALSE)
```

## Not run:
```r
data(Senn2013)

# Random effects model
net2 <- netmeta(TE, seTE, treat1.long, treat2.long, studlab,
                data = Senn2013, sm = "MD", common = FALSE, reference = "plac")

# Calculate and print concise results for all pairwise meta-analyses
```
## netposet

Partial order of treatments in network meta-analysis

### Description

Partial order of treatments in network meta-analysis. The set of treatments in a network is called a partially ordered set (in short, a *poset*), if different outcomes provide different treatment ranking lists.

### Usage

```r
netposet(
  ..., 
  outcomes, 
  treatments, 
  small.values, 
  common, 
  random, 
  fixed, 
  comb.fixed, 
  comb.random
)
```

## S3 method for class 'netposet'

```r
print(x, pooled = ifelse(x$random, "random", "common"), ...)
```
Arguments

... See details.
outcomes A character vector with outcome names.
treatments A character vector with treatment names.
small.values See details.
common A logical indicating whether to show results for the common effects model.
random A logical indicating whether to show results for the random effects model.
fixed Ignored deprecated argument (replaced by common).
comb.fixed Ignored deprecated argument (replaced by common).
comb.random Ignored deprecated argument (replaced by random).
x An object of class netposet.
pooled A character string indicating whether Hasse diagram should be drawn for common ("common") or random effects model ("random"). Can be abbreviated.

Details

In network meta-analysis, frequently different outcomes are considered which may each provide a different ordering of treatments. The concept of a partially ordered set (in short, a poset, Carlsen & Bruggemann, 2014) of treatments can be used to gain further insights in situations with apparently conflicting orderings. This implementation for rankings in network meta-analysis is described in Rücker & Schwarzer (2017).

In function netposet, argument ...() can be any of the following:

- arbitrary number of netrank objects providing P-scores;
- arbitrary number of netmeta objects;
- single ranking matrix with each column providing P-scores (Rücker & Schwarzer 2015) or SUCRA values (Salanti et al. 2011) for an outcome and rows corresponding to treatments.

Note, albeit in general a ranking matrix is not constrained to have values between 0 and 1, netposet stops with an error in this case as this function expects a matrix with P-scores or SUCRA values.

Argument outcomes can be used to label outcomes. If argument outcomes is missing,

- column names of the ranking matrix are used as outcome labels (if first argument is a ranking matrix and column names are available);
- capital letters 'A', 'B', ... are used as outcome labels and a corresponding warning is printed.

Argument treatments can be used to provide treatment labels if the first argument is a ranking matrix. If argument treatment is missing,

- row names of the ranking matrix are used as treatment labels (if available);
- letters 'a', 'b', ... are used as treatment labels and a corresponding warning is printed.
If argument ...{} consists of netmeta objects, netrank is called internally to calculate P-scores. In this case, argument small.values can be used to specify for each outcome whether small values are beneficial ("desirable") or harmful ("undesirable"); see netrank. This argument is ignored for a ranking matrix and netrank objects.

Arguments common and random can be used to define whether results should be printed and plotted for common and random effects model. If netmeta and netrank objects are provided in argument ...{}, values for common and random within these objects are considered; if these values are not unique, argument common or random are set to TRUE.

In function print.netposet, argument ...{} is passed on to the printing function.

Value

An object of class netposet with corresponding print, plot, and hasse functions. The object is a list containing the following components:

- **P.common** Ranking matrix with rows corresponding to treatments and columns corresponding to outcomes (common effects model).
- **M0.common** Hasse matrix skipping unnecessary paths (common effects model).
- **M.common** "Full" Hasse matrix (common effects model).
- **O.common** Matrix with information about partial ordering (common effects model).
- **P.random** Ranking matrix with rows corresponding to treatments and columns corresponding to outcomes (random effects model).
- **M0.random** Hasse matrix skipping unnecessary paths (random effects model).
- **M.random** "Full" Hasse matrix (random effects model).
- **O.random** Matrix with information about partial ordering (random effects model).
- **small.values, common, random** As defined above.
- **call** Function call.
- **version** Version of R package netmeta used to create object.

Author(s)

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

References


See Also

netmeta, netrank, plot.netrank, hasse, plot.netposet

Examples

```r
## Not run:
# Use depression dataset
# data(Linde2015)

# Define order of treatments
#
trts <- c("TCA", "SSRI", "SNRI", "NRI",
  "Low-dose SARI", "NaSSa", "rMAO-A", "Hypericum", "Placebo")

# Outcome labels
#
outcomes <- c("Early response", "Early remission")

# (1) Early response
#
p1 <- pairwise(treat = list(treatment1, treatment2, treatment3),
  event = list(resp1, resp2, resp3), n = list(n1, n2, n3),
  studlab = id, data = Linde2015, sm = "OR")

# net1 <- netmeta(p1, common = FALSE,
  seq = trts, ref = "Placebo", small.values = "undesirable")

# (2) Early remission
#
p2 <- pairwise(treat = list(treatment1, treatment2, treatment3),
  event = list(remi1, remi2, remi3), n = list(n1, n2, n3),
  studlab = id, data = Linde2015, sm = "OR")

# net2 <- netmeta(p2, common = FALSE,
  seq = trts, ref = "Placebo", small.values = "undesirable")

# Partial order of treatment rankings (two outcomes)
#
po <- netposet(netrank(net1), netrank(net2), outcomes = outcomes)

# Hasse diagram
#
hasse(po)
```

# Outcome labels
#
outcomes <- c("Early response", "Early remission",
  "Lost to follow-up", "Lost to follow-up due to AEs",
  "Adverse events (AEs)")
# (3) Loss to follow-up
#
p3 <- pairwise(treat = list(treatment1, treatment2, treatment3),
   event = list(loss1, loss2, loss3), n = list(n1, n2, n3),
   studlab = id, data = Linde2015, sm = "OR")
#
net3 <- netmeta(p3, common = FALSE,
   seq = trts, ref = "Placebo", small.values = "desirable")
#
# (4) Loss to follow-up due to adverse events
#
p4 <- pairwise(treat = list(treatment1, treatment2, treatment3),
   event = list(loss.ae1, loss.ae2, loss.ae3), n = list(n1, n2, n3),
   studlab = id, data = subset(Linde2015, id != 55), sm = "OR")
#
net4 <- netmeta(p4, common = FALSE,
   seq = trts, ref = "Placebo", small.values = "desirable")
#
# (5) Adverse events
#
p5 <- pairwise(treat = list(treatment1, treatment2, treatment3),
   event = list(ae1, ae2, ae3), n = list(n1, n2, n3),
   studlab = id, data = Linde2015, sm = "OR")
#
net5 <- netmeta(p5, common = FALSE,
   seq = trts, ref = "Placebo", small.values = "desirable")

# Partial order of treatment rankings (all five outcomes)
#
po.ranks <- netposet(netrank(net1), netrank(net2),
   netrank(net3), netrank(net4), netrank(net5), outcomes = outcomes)
#
# Same result
#
po.nets <- netposet(net1, net2, net3, net4, net5,
   outcomes = outcomes)
#
all.equal(po.ranks, po.nets)

# Print matrix with P-scores (random effects model)
#
po.nets$P.random

# Hasse diagram for all outcomes (random effects model)
#
hasse(po.ranks)

# Hasse diagram for outcomes early response and early remission
#
p012 <- netposet(netrank(net1), netrank(net2),
   outcomes = outcomes[1:2])
hasse(p012)
# Scatter plot
#
oldpar <- par(pty = "s")
plot(po12)
par(oldpar)

## End(Not run)

# Example using ranking matrix with P-scores
#
# What is the best treatment of locally advanced nasopharyngeal
# carcinoma? An individual patient data network meta-analysis.
# Journal of Clinical Oncology, 35, 498-505
#
outcomes <- c("OS", "PFS", "LC", "DC")
treatments <- c("RT", "IC-RT", "IC-CRT", "CRT",
               "CRT-AC", "RT-AC", "IC-RT-AC")
#
# P-scores (from Table 1)
#
pscore.os <- c(15, 33, 63, 70, 96, 28, 45) / 100
pscore.pfs <- c(4, 46, 79, 52, 94, 36, 39) / 100
pscore.lc <- c(9, 27, 47, 37, 82, 58, 90) / 100
pscore.dc <- c(16, 76, 95, 48, 72, 32, 10) / 100
#
pscore.matrix <- data.frame(pscore.os, pscore.pfs, pscore.lc, pscore.dc)
rownames(pscore.matrix) <- treatments
colnames(pscore.matrix) <- outcomes
pscore.matrix
#
po <- netposet(pscore.matrix)
po12 <- netposet(pscore.matrix[, 1:2])
po
po12
#
hasse(po)
hasse(po12)
#
oldpar <- par(pty = "s")
plot(po12)
par(oldpar)

netrank

---

Frequentist method to rank treatments in network

Description

Ranking treatments in frequentist network meta-analysis with and without resampling methods.
Usage

\texttt{netrank(}
\begin{verbatim}
x, small.values = x$small.values, method, nsim, common = x$common, random = x$random,
warn.deprecated = gs("warn.deprecated"), ...
\end{verbatim}
\texttt{)}

\texttt{## S3 method for class 'netrank'
print(}
\begin{verbatim}
x, common = x$common, random = x$random, sort = TRUE,
digits = gs("digits.prop"), warn.deprecated = gs("warn.deprecated"), ...
\end{verbatim}
\texttt{)}

Arguments

\begin{itemize}
\item \texttt{x} \hspace{1cm} An object of class \texttt{netmeta} or \texttt{rankogram}.
\item \texttt{small.values} \hspace{1cm} A character string specifying whether small treatment effects indicate a beneficial ("desirable") or harmful ("undesirable") effect, can be abbreviated.
\item \texttt{method} \hspace{1cm} A character string specifying whether the "P-score" or "SUCRA" ranking metric will be calculated.
\item \texttt{nsim} \hspace{1cm} Number of simulations to calculate SUCRAs.
\item \texttt{common} \hspace{1cm} A logical indicating whether to print P-scores or SUCRAs for the common effects model.
\item \texttt{random} \hspace{1cm} A logical indicating whether to print P-scores or SUCRAs for the random effects model.
\item \texttt{warn.deprecated} \hspace{1cm} A logical indicating whether warnings should be printed if deprecated arguments are used.
\item ... \hspace{1cm} Additional arguments passed on to \texttt{print.data.frame} function (used internally).
\item \texttt{sort} \hspace{1cm} A logical indicating whether printout should be sorted by decreasing P-score.
\item \texttt{digits} \hspace{1cm} Minimal number of significant digits, see \texttt{print.default}.
\end{itemize}

Details

Treatments are ranked based on a network meta-analysis. Ranking is performed by a ranking metric: P-score or SUCRA.
P-scores are based solely on the point estimates and standard errors of the network estimates. They measure the extent of certainty that a treatment is better than another treatment, averaged over all competing treatments (Rücker and Schwarzer 2015).

The Surface Under the Cumulative RAnking curve (SUCRA) is the rank of treatment $i$ within the range of treatments, measured on a scale from 0 (worst) to 1 (best) (Salanti et al. 2011). A resampling method is used to calculate SUCRAs for frequentist network meta-analysis. The number of simulations is determined by argument $nsim$.

The interpretation of P-scores and SUCRAs is comparable.

The P-score of treatment $i$ is defined as the mean of all $1 - P[j]$ where $P[j]$ denotes the one-sided P-value of accepting the alternative hypothesis that treatment $i$ is better than one of the competing treatments $j$. Thus, if treatment $i$ is better than many other treatments, many of these P-values will be small and the P-score will be large. Vice versa, if treatment $i$ is worse than most other treatments, the P-score is small.

The P-score of treatment $i$ can be interpreted as the mean extent of certainty that treatment $i$ is better than another treatment.

Value

An object of class netrank with corresponding print function. The object is a list containing the following components:

- `ranking.common`: A named numeric vector with P-scores or SUCRAs for the common effects model.
- `Pmatrix.common`: Numeric matrix based on pairwise one-sided p-values for the common effects model.
- `ranking.random`: A named numeric vector with P-scores or SUCRAs for the random effects model.
- `Pmatrix.random`: Numeric matrix based on pairwise one-sided p-values of the random effects model.
- `small.values`, `method`, `x`:
  As defined above.
- `version`: Version of R package netmeta used to create object.

Author(s)

Gerta Rücker <gerta.ruecker@uniklinik-freiburg.de>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>, Theodoros Papakonstantinou <dev@tpapak.com>

References


See Also

- `netmeta`, `rankogram`, `plot.rankogram`
Examples

data(smokingcessation)

p1 <- pairwise(list(treat1, treat2, treat3),
    event = list(event1, event2, event3), n = list(n1, n2, n3),
    data = smokingcessation, sm = "OR")
net1 <- netmeta(p1)

netrank(net1)

## Not run:
data(Senn2013)

net2 <- netmeta(TE, seTE, treat1, treat2, studlab,
    data = Senn2013, sm = "MD", random = FALSE)

nr2 <- netrank(net2)
nr2
print(nr2, sort = FALSE)

net3 <- netmeta(TE, seTE, treat1, treat2, studlab,
    data = Senn2013, sm = "MD")

nr3 <- netrank(net3)
nr3
print(nr3, sort = "common")
pn(print(nr3, sort = FALSE)

net4 <- netmeta(TE, seTE, treat1, treat2, studlab,
    data = Senn2013, sm = "MD")

nr4 <- netrank(net4, method = "SUCRA", nsim = 100)
nr4
print(nr4, sort = "common")
pn(print(nr4, sort = FALSE)

## End(Not run)

 netsplit

Split direct and indirect evidence in network meta-analysis

Description

Methods to split network estimates into the contribution of direct and indirect evidence and to test
for local inconsistency in network meta-analysis.
Usage

```r
etsplit(
  x,
  method,
  upper = TRUE,
  reference.group = x$reference.group,
  baseline.reference = x$baseline.reference,
  order = NULL,
  sep.trts = x$sep.trts,
  quote.trts = "",
  tol.direct = 5e-04,
  common = x$common,
  random = x$random,
  backtransf = x$backtransf,
  warn = FALSE,
  warn.deprecated = gs("warn.deprecated"),
  verbose = FALSE,
  ...
)
```

```r
## S3 method for class 'netsplit'
print(
  x,
  common = x$x$common,
  random = x$x$random,
  show = "all",
  overall = TRUE,
  ci = FALSE,
  test = show %in% c("all", "with.direct", "both"),
  only.reference = FALSE,
  sortvar = NULL,
  nchar.trts = x$nchar.trts,
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = gs("digits.pval"),
  digits.prop = max(gs("digits.pval") - 2, 2),
  text.NA = ".",
  backtransf = x$backtransf,
  scientific.pval = gs("scientific.pval"),
  big.mark = gs("big.mark"),
  legend = TRUE,
  indent = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```
Arguments

x An object of class netmeta or netsplit.
method A character string indicating which method to split direct and indirect evidence is to be used. Either "Back-calculation" or "SIDDE", can be abbreviated. See Details.
upper A logical indicating whether treatment comparisons should be selected from the lower or upper triangle of the treatment effect matrices (see list elements TE.common and TE.random in the netmeta object). Ignored if argument order is provided.
reference.group Reference treatment. Ignored if argument order is provided.
baseline.reference A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment or vice versa. This argument is only considered if reference.group is not equal to "" and argument order is not provided.
order A optional character or numerical vector specifying the order of treatments in comparisons.
sep.trts A character string used in comparison names as separator between treatment labels, e.g., " vs ".
quote.trts A character used to print around treatment labels.
tol.direct A numeric defining the maximum deviation of the direct evidence proportion from 0 or 1 to classify a comparison as providing only indirect or direct evidence, respectively.
common A logical indicating whether results for the common effects network meta-analysis should be printed.
random A logical indicating whether results for the random effects network meta-analysis should be printed.
backtransf A logical indicating whether printed results should be back transformed. For example, if backtransf = TRUE, results for sm = "OR" are printed as odds ratios rather than log odds ratios.
warn A logical indicating whether warnings should be printed.
warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments are used.
verbose A logical indicating whether progress information should be printed.
... Additional arguments.
show A character string indicating which comparisons should be printed (see Details).
overall A logical indicating whether estimates from network meta-analysis should be printed in addition to direct and indirect estimates.
Ci A logical indicating whether confidence intervals should be printed in addition to treatment estimates.
netsplit

A logical indicating whether results of a test comparing direct and indirect estimates should be printed.

A logical indicating whether only comparisons with the reference group should be printed.

An optional vector used to sort comparisons (must be of same length as the total number of comparisons).

A numeric defining the minimum number of characters used to create unique treatment names.

Minimal number of significant digits, see print.default.

Minimal number of significant digits for z-value of test of agreement between direct and indirect evidence, see print.default.

Minimal number of significant digits for p-value of test of agreement between direct and indirect evidence, see print.default.

Minimal number of significant digits for direct evidence proportions, see print.default.

A character string specifying text printed for missing values.

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

A character used as thousands separator.

A logical indicating whether a legend should be printed.

A logical indicating whether items in the legend should be indented.

Details

A comparison of direct and indirect treatment estimates can serve as check for consistency of network meta-analysis (Dias et al., 2010).

This function provides two methods to derive indirect estimates:

- Separate Indirect from Direct Evidence (SIDE) using a back-calculation method. The direct evidence proportion as described in König et al. (2013) is used in the calculation of the indirect evidence;
- Separate Indirect from Direct Design Evidence (SIDDE) as described in Efthimiou et al. (2019).

Note, for the back-calculation method, indirect treatment estimates are already calculated in netmeta and this function combines and prints these estimates in a user-friendly way. Furthermore, this method is not available for the Mantel-Haenszel and non-central hypergeometric distribution approach implemented in netmetabin.

For the random-effects model, the direct treatment estimates are based on the common between-study variance $\tau^2$ from the network meta-analysis, i.e. the square of list element x$\tau$.

Argument show determines which comparisons are printed:

- “all” All comparisons
- “both” Only comparisons contributing both direct and indirect evidence
- “with.direct” Comparisons providing direct evidence
- “direct.only” Comparisons providing only direct evidence
- “indirect.only” Comparisons providing only indirect evidence
The SIDDE approach can be compute-intensive in large networks. Crude information on the computation progress is printed for SIDDE if argument `verbose` is `TRUE`. In addition, computation times are printed if R package `tictoc` is installed.

Value

An object of class `netsplit` with corresponding `print` and `forest` functions. The object is a list containing the following components:

- `common`, `random` As defined above.
- `comparison` A vector with treatment comparisons.
- `prop.common`, `prop.random` A vector with direct evidence proportions (common / random effects model).
- `common`, `random` Results of network meta-analysis (common / random effects model), i.e., data frame with columns `comparison`, `TE`, `seTE`, `lower`, `upper`, `z`, and `p`.
- `direct.common`, `direct.random` Network meta-analysis results based on direct evidence (common / random effects model), i.e., data frame with columns `comparison`, `TE`, `seTE`, `lower`, `upper`, `z`, and `p`.
- `indirect.common`, `indirect.random` Network meta-analysis results based on indirect evidence (common / random effects model), i.e., data frame with columns `comparison`, `TE`, `seTE`, `lower`, `upper`, `z`, and `p`.
- `compare.common`, `compare.random` Comparison of direct and indirect evidence in network meta-analysis (common / random effects model), i.e., data frame with columns `comparison`, `TE`, `seTE`, `lower`, `upper`, `z`, and `p`.
- `sm` A character string indicating underlying summary measure
- `level.ma` The level used to calculate confidence intervals for pooled estimates.
- `tictoc` Computation times for SIDDE approach (if R package `tictoc` is installed).
- `version` Version of R package `netmeta` used to create object.

Author(s)

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>, Gerta Rücker <gerta.ruecker@uniklinik-freiburg.de>, Orestis Efthimiou <oremiou@gmail.com>

References


See Also

forest.netsplit, netmeta, netmetabin, netmeasures

Examples

data(Woods2010)
# p1 <- pairwise(treatment, event = r, n = N,
#    studlab = author, data = Woods2010, sm = "OR")
# net1 <- netmeta(p1)
# print(netsplit(net1), digits = 2)

# Not run:
print(netsplit(net1), digits = 2,
    backtransf = FALSE, common = FALSE)

# Sort by increasing number of studies in direct comparisons
print(netsplit(net1), digits = 2, sortvar = k)
# Sort by decreasing number of studies in direct comparisons
print(netsplit(net1), digits = 2, sortvar = -k)

# Sort by increasing evidence proportion under common effects model
print(netsplit(net1), digits = 2, sortvar = prop.common)
# Sort by decreasing evidence proportion under common effects model
print(netsplit(net1), digits = 2, sortvar = -prop.common)

# Sort by decreasing evidence proportion under common effects model
# and number of studies
print(netsplit(net1), digits = 2, sortvar = cbind(-prop.common, -k))

data(Senn2013)
# net2 <- netmeta(TE, seTE, treat1.long, treat2.long,
#    studlab, data = Senn2013)
# print(netsplit(net2), digits = 2)
# Layout of Puhan et al. (2014), Table 1
print(netsplit(net2), digits = 2, ci = TRUE, test = FALSE)

data(Dong2013)
p3 <- pairwise(treatment, death, randomized, studlab = id,
    data = Dong2013, sm = "OR")
net3 <- netmetabin(p3)
netsplit(net3)

## End(Not run)
### nettable

**Table with network meta-analysis results**

#### Description

Construct a table with network, direct and indirect estimates from one or more network meta-analyses.

#### Usage

```r
nettable(
  ..., name = NULL, method = NULL, order = NULL, common, random, upper = TRUE, reference.group = NULL, baseline.reference = NULL, backtransf = NULL, digits = gs("digits"), digits.I2 = gs("digits.I2"), digits.pval = gs("digits.pval"), scientific.pval = gs("scientific.pval"), zero.pval = gs("zero.pval"), JAMA.pval = gs("JAMA.pval"), big.mark = gs("big.mark"), text.NA = ".", bracket = gs("CIbracket"), separator = gs("CIseparator"), lower.blank = gs("CILower.blank"), upper.blank = gs("CIUpper.blank"), tol.direct = 5e-04, writexl = !missing(path), path = "nettable.xlsx", overwrite = FALSE, warn = FALSE, verbose = FALSE
)
```

#### Arguments

- `...`: Any number of network meta-analysis objects or a single list with network meta-analyses.
name
An optional character vector providing descriptive names for network meta-
analysis objects.

method
A character string indicating which method to split direct and indirect evidence
is to be used. Either "Back-calculation" or "SIDDE", can be abbreviated. See
Details.

order
A optional character or numerical vector specifying the order of treatments in
comparisons.

common
A logical indicating whether table for the common effects network meta-analysis
should be printed.

random
A logical indicating whether table for the random effects network meta-analysis
should be printed.

upper
A logical indicating whether treatment comparisons should be selected from
the lower or upper triangle of the treatment effect matrices (see list elements
TE.common and TE.random in the netmeta object). Ignored if argument order
is provided.

reference.group
Reference treatment. Ignored if argument order is provided.

baseline.reference
A logical indicating whether results should be expressed as comparisons of other
treatments versus the reference treatment or vice versa. This argument is only
considered if reference.group is not equal to "" and argument order is not
provided.

backtransf
A logical indicating whether printed results should be back transformed. For
example, if backtransf = TRUE, results for sm = "OR" are printed as odds ratios
rather than log odds ratios.

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

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

digits.pval
Minimal number of significant digits for p-value of test of agreement between
direct and indirect evidence, 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.

text.NA
A character string specifying text printed for missing values.

bracket
A character with bracket symbol to print lower confidence interval: "[", "(", 
""]=".

separator
A character string with information on separator between lower and upper con-
fidence 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.
tol.direct A numeric defining the maximum deviation of the direct evidence proportion from 0 or 1 to classify a comparison as providing only indirect or direct evidence, respectively.
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.
warn A logical indicating whether warnings should be printed.
verbose A logical indicating whether progress information should be printed.
x An object of class nettable.

Details

Construct a table with network, direct and indirect estimates from one or more network meta-analyses. The table looks very similar to the statistical part of a GRADE table for a network meta-analysis (Puhan et al., 2014).

By default, an R object with the network tables is generated. Alternatively, an Excel file is created if argument writexl = TRUE.

Two methods to derive indirect estimates are available:

- Separate Indirect from Direct Evidence (SIDE) using a back-calculation method. The direct evidence proportion as described in König et al. (2013) is used in the calculation of the indirect evidence;
- Separate Indirect from Direct Design Evidence (SIDDE) as described in Efthimiou et al. (2019).

Note, for the back-calculation method, indirect treatment estimates are already calculated in netmeta and this function combines and prints these estimates in a user-friendly way. Furthermore, this method is not available for the Mantel-Haenszel and non-central hypergeometric distribution approach implemented in netmetabin.

For the random-effects model, the direct treatment estimates are based on the common between-study variance $\tau^2$ from the network meta-analysis, i.e. the square of list element x$\tau$.

The SIDDE approach can be compute-intensive in large networks. Crude information on the computation progress is printed for SIDDE if argument verbose is TRUE.

Value

An object of class nettable with corresponding print function if argument writexl = FALSE. The object is a list containing the network tables in list elements 'common' and 'random'. An Excel file is created if writexl = TRUE. In this case, NULL is returned in R.

Author(s)

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


See Also

`netsplit, netmeta, netmetabin, netmeasures`

Examples

data(Woods2010)

#
p1 <- pairwise(treatment, event = r, n = N,

studlab = author, data = Woods2010, sm = "OR")

#
net1 <- netmeta(p1)

#
nt1 <- nettable(net1, digits = 2)

nt1

print(nt1, common = FALSE)

print(nt1, random = FALSE)

## Not run:
# Create a CSV file with network table from random effects model
#
table1 <- nettable(net1, digits = 2, bracket = "(" , separator = " to ")
#
write.table(table1$random, file = "table1-random.csv",

row.names = FALSE, col.names = TRUE, sep = ",")
#
# Create Excel files with network tables
# (if R package writexl is available)
#
nettable(net1, digits = 2, bracket = ",", separator = ",",

path = tempfile(fileext = ".xlsx" ))

## End(Not run)
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 function `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

```r
pairwise(
    treat,
    event,
    n,
    mean,
    sd,
    TE,
    seTE,
    time,
    data = NULL,
    studlab,
    incr = gs("incr"),
    allincr = gs("allincr"),
    addincr = gs("addincr"),
    allstudies = gs("allstudies"),
    reference.group,
    keep.all.comparisons,
    warn = FALSE,
    ...
)
```

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).
data An optional data frame containing the study information.
studlab A vector with study labels (optional).
inr A numerical value which is added to cell frequencies for studies with a zero cell count, see Details.
allincr A logical indicating if incr is added to cell frequencies of all studies if at least one study has a zero cell count. If FALSE (default), incr is added only to cell frequencies of studies with a zero cell count.
addincr A logical indicating if incr is added to cell frequencies of all studies irrespective of zero cell counts.
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’).
warn A logical indicating whether warnings should be printed (e.g., if studies are excluded due to only providing a single treatment arm).
... Additional arguments passed-through to the functions to calculate effects.

Details

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.

The pairwise function transforms data given in (wide or long) arm-based format into the contrast-based format which consists of pairwise comparisons and is needed as input to the netmeta function.

The pairwise function can transform data with binary outcomes (using the metabin function from R package meta), 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`).

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 `...`. The most important argument is `sm` defining the summary measure. More information on this and other arguments is given in the help pages of R functions `metabin`, `metacont`, `metainc`, and `metagen`, respectively.

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 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 data set 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`.

**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).

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.de>

**References**

See Also

netmeta, metacont, metagen, metabin, metainc, netgraph.netmeta, pairwise

Examples

# Example using continuous outcomes (internal call of function
# metacont)
#
# data(Franchini2012)
# Transform data from arm-based format to contrast-based format
p1 <- pairwise(list(Treatment1, Treatment2, Treatment3),
               n = list(n1, n2, n3),
               mean = list(y1, y2, y3), sd = list(sd1, sd2, sd3),
               data = Franchini2012, studlab = Study)
p1

## Not run:
# Conduct network meta-analysis
#
net1 <- netmeta(p1)
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)
p2 <- 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")
p2

# Compare pairwise objects p1 (based on continuous outcomes) and p2
# (based on generic outcomes)
# all.equal(p1[, c("TE", "seTE", "studlab", "treat1", "treat2")],
p2[, c("TE", "seTE", "studlab", "treat1", "treat2")])

# Same result as network meta-analysis based on continuous outcomes
# (object net1)
net2 <- netmeta(p2)
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.
# p3 <- pairwise(list(treat1, treat2, treat3),
# list(event1, event2, event3), list(n1, n2, n3),
# data = smokingcessation,
# sm = "OR")
p3

# Conduct network meta-analysis
#
net3 <- netmeta(p3)
net3

# Example with incidence rates
#
data(dietaryfat)
# Transform data from arm-based format to contrast-based format
# p4 <- pairwise(list(treat1, treat2, treat3),
# list(d1, d2, d3), time = list(years1, years2, years3),
# studlab = ID,
# data = dietaryfat)
p4

# 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(p4, sm = "IRR")
summary(net4)

# Example with long data format
#
data(Woods2010)

# Transform data from long arm-based format to contrast-based # format Argument 'sm' has to be used for odds ratio as summary
plot.netposet

Scatter plot or biplot showing partially order of treatment ranks

Description

This function generates a scatter plot or biplot of P-scores with an overlay describing partial order of treatment ranks.

Usage

## S3 method for class 'netposet'
plot(
  x,
  plottype = "scatter",
pooled = ifelse(x$random, "random", "common"),
dim = "2d",
  sel.x = 1,
  sel.y = 2,
  sel.z = 3,
  cex = 1,
  col = "black",
  cex.text = cex,
  col.text = col,
  adj.x = 0,
  adj.y = 1,
  offset.x = 0.005,
  offset.y = -0.005,
  pch = NULL,
  cex.points = cex,
  col.points = col,
  col.lines = "black",
  lty.lines = 1,
  lwd.lines = 1,
  arrows = FALSE,
length = 0.05,
grid = TRUE,
col.grid = "gray",
lty.grid = 2,
lwd.grid = 1,
... )

Arguments

x An object of class netmeta (mandatory).
plottype A character string indicating whether a scatter plot or biplot should be produced, either "scatter" or "biplot". Can be abbreviated.
pooled A character string indicating whether scatter plot should be drawn for common ("common") or random effects model ("random"). Can be abbreviated.
dim A character string indicating whether a 2- or 3-dimensional plot should be produced, either "2d" or "3d". Can be abbreviated.
 sel.x A numeric specifying number of outcome to use for the x-axis in a scatterplot (argument is not considered for a biplot).
 sel.y A numeric specifying number of outcome to use for the y-axis in a scatterplot (argument is not considered for a biplot).
 sel.z A numeric specifying number of outcome to use for the z-axis in a scatterplot (argument is not considered for a biplot).
cex The magnification to be used for treatment labels and points.
col Colour(s) of treatment labels and points.
cex.text The magnification to be used for treatment labels.
col.text Colour(s) of treatment labels.
adj.x Value(s) in [0, 1] to specify adjustment of treatment labels on x-axis (only considered in 2-D plots); see text.
adj.y Value(s) in [0, 1] to specify adjustment of treatment labels on y-axis (only considered in 2-D plots); see text.
offset.x Offset(s) of treatment labels on x-axis (only considered in 2-D plots).
offset.y Offset(s) of treatment labels on y-axis (only considered in 2-D plots).
pch Plot symbol(s) for points; no points printed if equal to NULL.
cex.points Magnification(s) to be used for points.
col.points Colour(s) of points.
col.lines Line colour.
lty.lines Line type.
lwd.lines Line width.
arrows A logical indicating whether arrows should be printed (only considered in 2-D plots).
length Length of arrows; see arrows.
grid  A logical indicating whether grid lines should be added to plot.
col.grid Colour of grid lines.
lty.grid Line type of grid lines.
lwd.grid Line width of grid lines.
... Additional graphical arguments.

Details

By default (arguments plotttype = "scatter" and dim = "2d"), a scatter plot is created showing P-scores (see netrank) for the first two outcomes considered in the generation of a partially ordered set of treatment ranks (using netposet). In addition to the P-scores, the partially order of treatment ranks is shown as lines connecting treatments which is analogous to a Hasse diagram. If argument dim = "3d"), a 3-D scatter plot is generated showing P-scores for the first three outcomes.

To overcome the restriction of two or three dimension, a biplot (Gabriel, 1971) can be generated using argument plotttype = "biplot". This is essentially a scatter plot using the first two (dim = "2d") or three (dim = "3d") components in a principal components analysis (using prcomp). Note, if only two / three outcomes are considered in a netposet object, a 2-D / 3-D scatter plot is generated instead of a biplot as a principal component analysis is superfluous in such a situation.

Arguments sel.x and sel.y can be used to select different outcomes to show on x- and y-axis in a 2-D scatter plot; argument sel.z can be used accordingly in a 3-D scatter plot. These arguments are ignored for a biplot.

Note, in order to generate 3-D plots (argument dim = "3d"), R package rgl is necessary. Note, under macOS the X.Org X Window System must be available (see https://www.xquartz.org).

Author(s)

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

References


See Also

netposet, hasse, netrank, netmeta

Examples

## Not run:
data(Linde2015)

# Define order of treatments
#
trts <- c("TCA", "SSRI", "SNRI", "NRI",
    "Low-dose SARI", "NaSSa", "rMAO-A", "Hypericum", "Placebo")
# Outcome labels
#
# outcomes <- c("Early response", "Early remission")

# (1) Early response
#
p1 <- pairwise(treat = list(treatment1, treatment2, treatment3),
   event = list(resp1, resp2, resp3), n = list(n1, n2, n3),
   studlab = id, data = Linde2015, sm = "OR")
#
# net1 <- netmeta(p1, common = FALSE,
#    seq = trts, ref = "Placebo", small.values = "undesirable")

# (2) Early remission
#
p2 <- pairwise(treat = list(treatment1, treatment2, treatment3),
   event = list(remi1, remi2, remi3), n = list(n1, n2, n3),
   studlab = id, data = Linde2015, sm = "OR")
#
# net2 <- netmeta(p2, common = FALSE,
#    seq = trts, ref = "Placebo", small.values = "undesirable")

# Partial order of treatment rankings
#
# po2 <- netposet(netrank(net1), netrank(net2), outcomes = outcomes)

# Scatter plot
#
# plot(po2)

# Same scatter plot as only two outcomes considered in netposet()
#
# plot(po2, "biplot")

# Consider three outcomes
#
# Outcome labels
#
# outcomes <- c("Early response", "Early remission", "Lost to follow-up")

# (3) Loss to follow-up
#
p3 <- pairwise(treat = list(treatment1, treatment2, treatment3),
   event = list(loss1, loss2, loss3), n = list(n1, n2, n3),
   studlab = id, data = Linde2015, sm = "OR")
#
# net3 <- netmeta(p3, common = FALSE,
#    seq = trts, ref = "Placebo", small.values = "desirable")

# Partial order of treatment rankings (with three outcomes)
#
\texttt{po3 <- netposet(netrank(net1), netrank(net2), netrank(net3),}
\texttt{  outcomes = outcomes)}

\# Hasse diagram
\# hasse(po3)

\# Scatter plot
\# plot(po3)

\# Biplot (reverse limits of y-axis as biplot is upside down)
\# plot(po3, "bi", xlim = c(-1, 1.7), ylim = c(2.5, -2.5))

## End(Not run)

\section*{plot.netrank}

\textit{Plot treatment ranking(s) of network meta-analyses}

\subsection*{Description}
Produce an image plot of treatment ranking(s) generated with R function \texttt{netrank}.

\subsection*{Usage}
\#
\# S3 method for class 'netrank'
\# plot(
\#    ..., 
\#    name, 
\#    common, 
\#    random, 
\#    seq, 
\#    low = "red", 
\#    mid = "yellow", 
\#    high = "green", 
\#    col = "black", 
\#    main, 
\#    main.size = 14, 
\#    main.col = col, 
\#    main.face = "bold", 
\#    legend = TRUE, 
\#    axis.size = 12, 
\#    axis.col = col, 
\#    axis.face = "plain", 
\#    na.value = "grey50", 
\#    angle = 45,
hjust.x = 1,
vjust.x = 1,
hjust.y = 1,
vjust.y = 0,
nchar.trts,
digits = 3,
fixed,
combi.fixed,
combi.random,
warn.deprecated = gs("warn.deprecated")
)

Arguments
...
name
An optional character vector providing descriptive names for the network meta-
analysis objects.
common
A logical indicating whether results for the common effects model should be
plotted.
random
A logical indicating whether results for the random effects model should be
plotted.
seq
A character or numerical vector specifying the sequence of treatments on the
x-axis.
low
A character string defining the colour for a P-score of 0, see scale_fill_gradient2.
mid
A character string defining the colour for a P-score of 0.5, see scale_fill_gradient2.
high
A character string defining the colour for a P-score of 1, see scale_fill_gradient2.
col
Colour of text.
main
Title.
main.size
Font size of title, see element_text.
main.col
Colour of title, see element_text.
main.face
Font face of title, see element_text.
legend
A logical indicating whether a legend should be printed.
axis.size
Font size of axis text, see element_text.
axis.col
Colour of axis text, see element_text.
axis.face
Font face of axis text, see element_text.
nan.value
Colour for missing values, see scale_fill_gradient2.
angle
Angle for text on x-axis, see element_text.
hjust.x
A numeric between 0 and 1 with horizontal justification of text on x-axis, see
element_text.
vjust.x
A numeric between 0 and 1 with vertical justification of text on x-axis, see
element_text.
hjust.y
A numeric between 0 and 1 with horizontal justification of text on y-axis, see
element_text.


**vjust.y**
A numeric between 0 and 1 with vertical justification of text on y-axis, see `element_text`.

**nchar.trts**
A numeric defining the minimum number of characters used to create unique treatment names.

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

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

**comb.fixed**
Deprecated argument (replaced by 'common').

**comb.random**
Deprecated argument (replaced by 'random').

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

**Details**

This function produces an image plot of network rankings (Palpacuer et al., 2018, Figure 4). Note, a scatter plot of two network rankings can be generated with `plot.netposet`.

By default, treatments are ordered by decreasing P-scores of the first network meta-analysis object. Argument seq can be used to specify a different treatment order.

**Value**

A ggplot2 object or NULL if no ranking was conducted.

**Author(s)**

Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>, Clément Palpacuer <clementpalpacuer@gmail.com>

**References**


**See Also**

`netrank`, `netmeta`, `netposet`, `hasse`

**Examples**

```r
## Not run:
# Use depression dataset
#
data(Linde2015)

# Define order of treatments
#
trts <- c("TCA", "SSRI", "SNRI", "NRI","
"Low-dose SARI", "NaSSa", "rMAO-A", "Hypericum", "Placebo")

# Outcome labels

# outcomes <- c("Early response", "Early remission")

# (1) Early response

# p1 <- pairwise(treat = list(treatment1, treatment2, treatment3),
#                event = list(resp1, resp2, resp3), n = list(n1, n2, n3),
#                studlab = id, data = Linde2015, sm = "OR")
# net1 <- netmeta(p1, common = FALSE, seq = trts, ref = "Placebo")

# (2) Early remission

# p2 <- pairwise(treat = list(treatment1, treatment2, treatment3),
#                event = list(remi1, remi2, remi3), n = list(n1, n2, n3),
#                studlab = id, data = Linde2015, sm = "OR")
# net2 <- netmeta(p2, common = FALSE, seq = trts, ref = "Placebo")

# Image plot of treatment rankings (two outcomes)

# plot(netrank(net1, small.values = "undesirable"),
#      netrank(net2, small.values = "undesirable"),
#      name = outcomes, digits = 2)

# Outcome labels

# outcomes <- c("Early response", "Early remission",
#                "Lost to follow-up", "Lost to follow-up due to AEs",
#                "Adverse events (AEs)")

# (3) Loss to follow-up

# p3 <- pairwise(treat = list(treatment1, treatment2, treatment3),
#                event = list(loss1, loss2, loss3), n = list(n1, n2, n3),
#                studlab = id, data = Linde2015, sm = "OR")
# net3 <- netmeta(p3, common = FALSE, seq = trts, ref = "Placebo")

# (4) Loss to follow-up due to adverse events

# p4 <- pairwise(treat = list(treatment1, treatment2, treatment3),
#                event = list(loss.ae1, loss.ae2, loss.ae3), n = list(n1, n2, n3),
#                studlab = id, data = subset(Linde2015, id != 55), sm = "OR")
# net4 <- netmeta(p4, common = FALSE, seq = trts, ref = "Placebo")
# (5) Adverse events
#
p5 <- pairwise(treat = list(treatment1, treatment2, treatment3),
               event = list(ae1, ae2, ae3), n = list(n1, n2, n3),
               studlab = id, data = Linde2015, sm = "OR")
#
net5 <- netmeta(p5, common = FALSE, seq = trts, ref = "Placebo")

# Image plot of treatment rankings (two outcomes)
#
plot(netrank(net1, small.values = "undesirable"),
     netrank(net2, small.values = "undesirable"),
     netrank(net3, small.values = "desirable"),
     netrank(net4, small.values = "desirable"),
     netrank(net5, small.values = "desirable"),
     name = outcomes, digits = 2)

## End(Not run)

---

**plot.rankogram**  
*Plot rankograms*

---

**Description**

This function produces a rankogram, i.e., an image plot of ranking probabilities for all treatments.

**Usage**

```
## S3 method for class 'rankogram'
plot(
    x,
    type = if (cumulative.rankprob) "step" else "bar",
    pooled = ifelse(x$random, "random", "common"),
    sort = TRUE,
    trts,
    cumulative.rankprob = x$cumulative.rankprob,
    ylim,
    ylab,
    nchar.trts = x$nchar.trts,
    ...
)
```

**Arguments**

- `x`  
  An object of class `rankogram`.
- `type`  
  A character string specifying whether a "bar" chart, a "line" graph, or "step" functions should be drawn. Can be abbreviated.
pooled  A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.

sort    A logical indicating whether treatments should be sorted by decreasing SUCRAs.

trts    Treatment(s) to show in rankogram.

cumulative.rankprob A logical indicating whether cumulative ranking probabilities should be shown.

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

ylab    A label for the y-axis.

nchar.trts A numeric defining the minimum number of characters used to create unique treatment names.

...    Additional graphical arguments (ignored at the moment).

Details

This function produces an image plot of (cumulative) ranking probabilities for all treatments as a bar graph, a line graph or as step functions (argument type).

By default (argument pooled), results for the random effects model are shown if a network meta-analysis was conducted for both the common and random effects model.

Treatments are sorted according to their mean effects if argument sort = TRUE (default). A subset of treatments can be specified using argument trts.

Cumulative ranking probabilities are shown if cumulative.rankprob = TRUE. By default, step functions are shown for cumulative ranking probabilities.

Author(s)

Theodoros Papakonstantinou <dev@tpapak.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References


See Also

rankogram

Examples

data(Woods2010)
p1 <- pairwise(treatment, event = r, n = N, studlab = author, data = Woods2010, sm = "OR")
net1 <- netmeta(p1, small.values = "good")

ran1 <- rankogram(net1, nsim = 100)
ran1
plot(ran1)
plot(ran1, type = "l")
plot(ran1, cumulative.rankprob = TRUE)

---

**print.decomp.design**  
*Print method for objects of class decomp.design*

**Description**

Print method for objects of class decomp.design.

**Usage**

```r
## S3 method for class 'decomp.design'
print(
  x,
  digits.Q = gs("digits.Q"),
  showall = FALSE,
  digits.pval.Q = gs("digits.pval.Q"),
  digits.tau2 = gs("digits.tau2"),
  scientific.pval = gs("scientific.pval"),
  big.mark = gs("big.mark"),
  nchar.trts = x$nchar.trts,
  sort = TRUE,
  legend = TRUE,
  ...
)
```

**Arguments**

- `x`  
  An object of class decomp.design.
- `digits.Q`  
  Minimal number of significant digits for Q statistics, see `print.default`.
- `showall`  
  A logical indicating whether results should be shown for all designs or only designs contributing to chi-squared statistics (default).
- `digits.pval.Q`  
  Minimal number of significant digits for p-value of heterogeneity tests, see `print.default`.
- `digits.tau2`  
  Minimal number of significant digits for between-study 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.
- `nchar.trts`  
  A numeric defining the minimum number of characters used to create unique treatment names.
sort A logical indicating whether to sort results by p-values.
legend A logical indicating whether a legend should be printed.
... Additional arguments (ignored at the moment).

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>, Ulrike Krahn <ulrike.krahn@bayer.com>

See Also
decomp.design

Examples
data(Senn2013)

# Only consider first five studies (to reduce runtime of example)
# studies <- unique(Senn2013$studlab)
Senn2013.5 <- subset(Senn2013, studlab %in% studies[1:5])

# Conduct network meta-analysis with placebo as reference treatment
# net1 <- netmeta(TE, seTE, treat1, treat2, studlab,
# data = Senn2013.5, sm = "MD", reference = "plac")

# Decomposition of Cochran's Q
# decomp.design(net1)
Arguments

- **x**: An object of class `netbind` or `summary.netbind`.
- **common**: A logical indicating whether results for the common effects model should be printed.
- **random**: A logical indicating whether results for the random effects model should be printed.
- **warn.deprecated**: A logical indicating whether warnings should be printed if deprecated arguments are used.
- ... Additional arguments (to catch deprecated arguments).

Author(s)

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

See Also

- `netbind`

Examples

```r
data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce # runtime of example)
# face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Standard random effects NMA model (with placebo as reference # treatment)
# net1 <- netmeta(lnOR, selnOR, treat1, treat2, id, #
# data = face, reference.group = "placebo", #
# sm = "OR", common = FALSE)

# Additive CNMA model with placebo as inactive component and # reference
# ncl <- netcomb(net1, inactive = "placebo")

# Combine results of standard NMA and CNMA
# nb1 <- netbind(ncl, net1, #
# name = c("Additive CNMA", "Standard NMA"), #
# col.study = c("red", "black"), col.square = c("red", "black"))

nb1
print(nb1, common = TRUE)
```
Print method for objects of class netcomb

Description

Print method for objects of class netcomb.

Usage

```r
## S3 method for class 'netcomb'
print(
  x,
  common = x$common,
  random = x$random,
  backtransf = x$backtransf,
  nchar.comps = x$nchar.comps,
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = gs("digits.pval"),
  digits.pval.Q = max(gs("digits.pval.Q"), 2),
  digits.Q = gs("digits.Q"),
  digits.tau2 = gs("digits.tau2"),
  digits.tau = gs("digits.tau"),
  digits.I2 = gs("digits.I2"),
  scientific.pval = gs("scientific.pval"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  big.mark = gs("big.mark"),
  text.tau2 = gs("text.tau2"),
  text.tau = gs("text.tau"),
  text.I2 = gs("text.I2"),
  legend = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

Arguments

- `x`: An object of class netcomb or summary.netcomb.
- `common`: A logical indicating whether results for the common effects model should be printed.
- `random`: A logical indicating whether results for the random effects model should be printed.
- `backtransf`: A logical indicating whether results should be back transformed in printouts and forest plots. If `backtransf = TRUE`, results for `sm = "OR"` are presented as odds ratios rather than log odds ratios, for example.
nchar.comps A numeric defining the minimum number of characters used to create unique names for components.
digits Minimal number of significant digits, see print.default.
digits.stat Minimal number of significant digits for z- or t-value, 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 tests, see print.default.
digits.Q Minimal number of significant digits for heterogeneity statistics, 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.
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 component or combination effect should be printed according to JAMA reporting standards.
big.mark A character used as thousands separator.
text.tau2 Text printed to identify between-study variance $\tau^2$.
text.tau Text printed to identify $\tau$, the square root of the between-study variance $\tau^2$.
text.I2 Text printed to identify heterogeneity statistic $I^2$.
legend A logical indicating whether a legend should be printed.
warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments are used.
...
Additional arguments (to catch deprecated arguments).

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also
netcomb, discomb

Examples
data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce
# runtime of example)
#
face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
# net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
# data = face, reference.group = "placebo",
# sm = "OR", common = FALSE)

# Additive model for treatment components
# nc1 <- netcomb(net1)
nc1
print(nc1, digits = 2, digits.stat = 3)

## Not run:
# Conduct random effects network meta-analysis
# net2 <- netmeta(lnOR, selnOR, treat1, treat2, id,
# data = Linde2016, reference.group = "placebo",
# sm = "OR", common = FALSE)

# Additive model for treatment components
# nc2 <- netcomb(net2)
nc2
print(nc2, digits = 2, digits.stat = 3)

## End(Not run)

---

**print.netimpact**

*Print method for objects of class netimpact*

**Description**

Print method for objects of class netimpact.

**Usage**

```r
## S3 method for class 'netimpact'
print(
  x,
  common = x$x$common,
  random = x$x$random,
  digits = gs("digits.prop"),
  legend = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```
**Arguments**

- **x**: An object of class `netimpact`.
- **common**: A logical indicating whether results for the common effects model should be printed.
- **random**: A logical indicating whether results for the random effects model should be printed.
- **digits**: Minimal number of significant digits.
- **legend**: A logical indicating whether a legend should be printed.
- **warn.deprecated**: A logical indicating whether warnings should be printed if deprecated arguments are used.
- **...**: Additional arguments (to catch deprecated arguments).

**Author(s)**

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

**See Also**

`netimpact`

**Examples**

```r
data(Franchini2012)

# Only consider first two studies (to reduce runtime of example)
#
# studies <- unique(Franchini2012$Study)
pl <- pairwise(list(Treatment1, Treatment2, Treatment3),
               n = list(n1, n2, n3),
               mean = list(y1, y2, y3),
               sd = list(sd1, sd2, sd3),
               data = subset(Franchini2012, Study %in% studies[1:2]),
               studlab = Study)

net1 <- netmeta(pl)
ni <- netimpact(net1, verbose = TRUE)
ni
```

**Description**

Print detailed information for component network meta-analysis.
## S3 method for class 'summary.netcomb'
print(
  x,
  common = x$x$common,
  random = x$x$random,
  backtransf = x$x$backtransf,
  nchar.comps = x$x$nchar.comps,
  digits = gs("digits"),
  digits.stat = gs("digits.stat"),
  digits.pval = gs("digits.pval"),
  digits.pval.Q = max(gs("digits.pval.Q"), 2),
  digits.Q = gs("digits.Q"),
  scientific.pval = gs("scientific.pval"),
  zero.pval = gs("zero.pval"),
  JAMA.pval = gs("JAMA.pval"),
  big.mark = gs("big.mark"),
  legend = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)

### Arguments

- **x**
  - An object of class `summary.netcomb`

- **common**
  - A logical indicating whether results for the common effects model should be printed.

- **random**
  - A logical indicating whether results for the random effects model should be printed.

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

- **nchar.comps**
  - A numeric defining the minimum number of characters used to create unique component names.

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

- **digits.stat**
  - Minimal number of significant digits for z- or t-value, 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 tests, see `print.default`.

- **digits.Q**
  - Minimal number of significant digits for heterogeneity statistics, 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.
print.summary.netmeta

JAMA.pval  A logical specifying whether p-values for test of effects should be printed according to JAMA reporting standards.
big.mark  A character used as thousands separator.
legend  A logical indicating whether a legend should be printed.
warn.deprecated  A logical indicating whether warnings should be printed if deprecated arguments are used.
...  Additional arguments.

Author(s)

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

See Also

netcomb, discomb, summary.netcomb

Examples

data(Linde2016)

# Only consider studies including Face-to-face PST (to reduce
# runtime of example)
#
face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

# Conduct random effects network meta-analysis
#
net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
data = face, reference.group = "placebo",
sm = "OR", common = FALSE)

# Additive model for treatment components
#
ncl <- netcomb(net1)
print(summary(ncl), digits = 2)
## Usage

```r
## S3 method for class 'summary.netmeta'
print(
  x,
  sortvar,
  common = x$x$common,
  random = x$x$random,
  prediction = x$x$prediction,
  reference.group = x$x$reference.group,
  baseline.reference = x$x$baseline.reference,
  all.treatments = x$x$all.treatments,
  details = TRUE,
  nma = TRUE,
  backtransf = x$x$backtransf,
  nchar.trts = x$x$nchar.trts,
  nchar.studlab = x$x$nchar.studlab,
  digits = gs("digits"),
  digits.se = gs("digits.se"),
  digits.pval.Q = max(gs("digits.pval.Q"), 2),
  digits.Q = gs("digits.Q"),
  digits.tau2 = gs("digits.tau2"),
  digits.I2 = gs("digits.I2"),
  scientific.pval = gs("scientific.pval"),
  big.mark = gs("big.mark"),
  truncate,
  text.truncate = "*** Output truncated ***",
  legend = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

## Arguments

- **x**
  - An object of class `summary.netmeta`.

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

- **common**
  - A logical indicating whether results for the common effects model should be printed.

- **random**
  - A logical indicating whether results for the random effects model should be printed.

- **prediction**
  - A logical indicating whether prediction intervals should be printed.

- **reference.group**
  - Reference treatment.

- **baseline.reference**
  - A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa. This argument is only considered if `reference.group` has been specified.
all.treatments A logical or "NULL". If TRUE, matrices with all treatment effects, and confidence limits will be printed.
details A logical indicating whether further details for individual studies should be printed.
nma A logical indicating whether summary results of network meta-analysis should be printed.
backtransf A logical indicating whether results should be back transformed in printouts and forest plots. If backtransf = TRUE, results for sm = "OR" are presented as odds ratios rather than log odds ratios, for example.
nchar.trts A numeric defining the minimum number of characters used to create unique treatment names.
nchar.studlab A numeric defining the minimum number of characters used to create unique study labels.
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.pval.Q Minimal number of significant digits for p-value of heterogeneity tests, see print.default.
digits.Q Minimal number of significant digits for heterogeneity statistics, see print.default.
digits.tau2 Minimal number of significant digits for 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.
truncatem A character string printed if study results were truncated from the printout.
legend A logical indicating whether a legend should be printed.
warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments are used.
...

Author(s)
Guido Schwarzer &lt;guido.schwarzer@uniklinik-freiburg.de&gt;

See Also

netmeta, summary.netmeta
Examples

```r
data(smokingcessation)

# Transform data from arm-based format to contrast-based format
#
p1 <- pairwise(list(treat1, treat2, treat3),
    event = list(event1, event2, event3), n = list(n1, n2, n3),
    data = smokingcessation, sm = "OR")

# Conduct random effects network meta-analysis and print detailed summary
#
net1 <- netmeta(p1, common = FALSE)
summary(net1)

## Not run:
data(Senn2013)

# Conduct common effects network meta-analysis
#
net2 <- netmeta(TE, seTE, treat1, treat2, studlab,
    data = Senn2013, sm = "MD", random = FALSE, ref = "plac")
snet2 <- summary(net2)
print(snet2, digits = 3)

# Only show individual study results for multi-arm studies
#
print(snet2, digits = 3, truncate = multiarm)

# Only show first three individual study results
#
print(snet2, digits = 3, truncate = 1:3)

# Only show individual study results for Kim2007 and Willms1999
#
print(snet2, digits = 3, truncate = c("Kim2007", "Willms1999"))

# Only show individual study results for studies starting with the letter "W"
#
print(snet2, ref = "plac", digits = 3,
    truncate = substring(studlab, 1, 1) == "W")

# Conduct random effects network meta-analysis
#
net3 <- netmeta(TE, seTE, treat1, treat2, studlab,
    data = Senn2013, sm = "MD", common = FALSE, ref = "plac")
print(summary(net3), digits = 3)

## End(Not run)
```
Draw a 'comparison-adjusted' radial plot to assess funnel plot asymmetry in network meta-analysis.

## Usage

```r
## S3 method for class 'netmeta'
radial(
  x,
  order,
  pooled = ifelse(x$random, "random", "common"),
  level = x$level,
  pch,
  col = "black",
  legend = FALSE,
  pos.legend = "topright",
  pos.tests = "topleft",
  lump.comparator = FALSE,
  text.comparator = "comparator",
  method.bias,
  text.linreg = "(Egger)",
  text.rank = "(Begg-Mazumdar)",
  text.mm = "(Thompson-Sharp)",
  sep.trts = x$sep.trts,
  nchar.trts = x$nchar.trts,
  digits.pval = gs("digits.pval"),
  warn.deprecated = gs("warn.deprecated"),
  linreg = FALSE,
  rank = FALSE,
  mm = FALSE,
  ...
)
```

## Arguments

- `x`: An object of class `netmeta`.
- `order`: A mandatory character or numerical vector specifying the order of treatments or list of comparators (see Details).
- `pooled`: A character string indicating whether results for the common ("common") or random effects model ("random") should be plotted. Can be abbreviated.
- `level`: The confidence level utilised in the plot.
- `pch`: The plotting symbol(s) used for individual studies within direct comparisons.
col

The colour(s) used for individual studies within direct comparisons.

legend

A logical indicating whether a legend with information on direct comparisons should be added to the plot.

pos.legend

The position of the legend describing plotting symbols and colours for direct comparisons.

pos.tests

The position of results for test(s) of funnel plot asymmetry.

lump.comparator

A logical indicating whether comparators should be lumped, e.g., to specify inactive treatments. Information on direct comparisons should be added to the plot.

text.comparator

A character string used in the plot to label the comparator if lump.comparator is TRUE.

method.bias

A character vector indicating which test(s) for funnel plot asymmetry to use. Admissible values are "Begg", "Egger", and "Thompson", can be abbreviated. See function metabias.

text.linreg

A character string used in the plot to label the Egger test for funnel plot asymmetry.

text.rank

A character string used in the plot to label the Begg test for funnel plot asymmetry.

text.mm

A character string used in the plot to label the Thompson-Sharp test for funnel plot asymmetry.

sep.trts

A character used in comparison names as separator between treatment labels.

nchar.trts

A numeric defining the minimum number of characters used to create unique treatment names (see netmeta).

digits.pval

Minimal number of significant digits for p-value of test(s) for funnel plot asymmetry.

warn.deprecated

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

linreg

 Deprecated argument (replaced by method.bias).

rank

 Deprecated argument (replaced by method.bias).

mm

 Deprecated argument (replaced by method.bias).

... Additional graphical arguments passed as arguments to radial.meta.

Details

A 'comparison-adjusted' radial plot is drawn in the active graphics window. The idea of this radial plot is similar to a 'comparison-adjusted' funnel plot (Chaimani & Salanti, 2012).

Argument order is mandatory to determine the order of treatments (Chaimani et al., 2013):

"Before using this plot, investigators should order the treatments in a meaningful way and make assumptions about how small studies differ from large ones. For example, if they anticipate that newer treatments are favored in small trials, then they could name the treatments from oldest to
newest so that all comparisons refer to ‘old versus new intervention’. Other possibilities include defining the comparisons so that all refer to an active treatment versus placebo or sponsored versus non-sponsored intervention.”

Alternatively, it is possible to only provide a single or few treatment name(s) in argument order to define the comparator(s). In this case only comparisons with this / these treatment(s) will be considered. If argument `lump.comparator` is `TRUE`, all comparators will be lumped into a single group. The text for this group can be specified with argument `text.comparator`.

Value

A data frame with the following columns:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>studlab</td>
<td>Study label.</td>
</tr>
<tr>
<td>treat1</td>
<td>Label/Number for first treatment.</td>
</tr>
<tr>
<td>treat2</td>
<td>Label/Number for second treatment.</td>
</tr>
<tr>
<td>comparison</td>
<td>Treatment comparison.</td>
</tr>
<tr>
<td>TE</td>
<td>Estimate of treatment effect, e.g., log odds ratio.</td>
</tr>
<tr>
<td>TE.direct</td>
<td>Pooled estimate from direct evidence.</td>
</tr>
<tr>
<td>TE.adj</td>
<td>‘Comparison-adjusted’ treatment effect (TE - TE.direct).</td>
</tr>
<tr>
<td>seTE</td>
<td>Standard error of treatment estimate.</td>
</tr>
<tr>
<td>pch</td>
<td>Plotting symbol(s).</td>
</tr>
<tr>
<td>col</td>
<td>Colour of plotting symbol(s).</td>
</tr>
</tbody>
</table>

Author(s)

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

References


See Also

netmeta, radial.meta, funnel.meta, metabias

Examples

```r
## Not run:
data(Senn2013)
net1 <- netmeta(TE, seTE, treat1, treat2, studlab, data = Senn2013, sm = "MD")

# 'Comparison-adjusted' radial plot not created as argument 'order'
```
# is missing
#
try(radial(net1))

# Only show comparisons with placebo
#
radial(net1, order = "pl")

# Add result for Egger test of funnel plot asymmetry
#
radial(net1, order = "pl", method.bias = "Egger",
 digits.pval = 2)

# (Non-sensical) alphabetic order of treatments with placebo as
# last treatment
#
ord <- c("a", "b", "me", "mi", "pi", "r", "si", "su", "v", "pl")
radial(net1, order = ord)

# Add results for tests of funnel plot asymmetry and use different
# plotting symbols and colours
#
radial(net1, order = ord,
 pch = rep(c(15:18, 1), 3), col = 1:3,
 method.bias = c("Egger", "Begg", "Thompson"), digits.pval = 2)

# Same results for tests of funnel plot asymmetry using reversed
# order of treatments
#
radial(net1, order = rev(ord),
 pch = rep(c(15:18, 1), 3), col = 1:3,
 method.bias = c("Egger", "Begg", "Thompson"), digits.pval = 2)

# Calculate tests for funnel plot asymmetry
#
f1 <- radial(net1, order = ord)
#
metabias(metagen(TE.adj, seTE, data = f1))
metabias(metagen(TE.adj, seTE, data = f1), method = "Begg")
metabias(metagen(TE.adj, seTE, data = f1), method = "Thompson")

## End(Not run)

---

**rankogram**

### Calculate rankogram

**Description**

This function calculates the probabilities of each treatment being at each possible rank and the SUCRAs (Surface Under the Cumulative RAoking curve) in frequentist network meta-analysis.
Usage

```r
rankogram(
  x,
  nsim = 1000,
  common = x$common,
  random = x$random,
  small.values = x$small.values,
  cumulative.rankprob = FALSE,
  nchar.trts = x$nchar.trts,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

## S3 method for class 'rankogram'

```r
print(
  x,
  common = x$common,
  random = x$random,
  cumulative.rankprob = x$cumulative.rankprob,
  nchar.trts = x$nchar.trts,
  digits = gs("digits.prop"),
  legend = TRUE,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

Arguments

- **x**: An object of class `netmeta`.
- **nsim**: Number of simulations.
- **common**: A logical indicating to compute ranking probabilities and SUCRAs for the common effects model.
- **random**: A logical indicating to compute ranking probabilities and SUCRAs for the random effects model.
- **small.values**: A character string specifying whether small treatment effects indicate a beneficial ("desirable") or harmful ("undesirable") effect, can be abbreviated.
- **cumulative.rankprob**: A logical indicating whether cumulative ranking probabilities should be printed.
- **nchar.trts**: A numeric defining the minimum number of characters used to create unique treatment names.
- **warn.deprecated**: A logical indicating whether warnings should be printed if deprecated arguments are used.
- **digits**: Minimal number of significant digits, see `print.default`.
- **legend**: A logical indicating whether a legend should be printed.
- **...**: Additional arguments for printing.
Details

We derive a matrix showing the probability of each treatment being at each possible rank. To this aim, we use resampling from a multivariate normal distribution with estimated network effects as means and corresponding estimated variance covariance matrix. We then summarise them using the ranking metric SUCRAs (Surface Under the Cumulative RAnking curve).

Value

An object of class `rankogram` with corresponding `print` and `plot` function. The object is a list containing the following components:

- `ranking.matrix.common` Numeric matrix giving the probability of each treatment being at each possible rank for the common effects model.
- `ranking.common` SUCRA values for the common effects model.
- `ranking.matrix.random` Numeric matrix giving the probability of each treatment being at each possible rank for the random effects model.
- `ranking.random` SUCRA values for the random effects model.
- `cumrank.matrix.common` Numeric matrix giving the cumulative ranking probability of each treatment for the common effects model.
- `cumrank.matrix.random` Numeric matrix giving the cumulative ranking probability of each treatment for the random effects model.
- `nsim, common, random` As defined above

Author(s)

Theodoros Papakonstantinou <dev@tpapak.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

References


See Also

`netmeta`, `netrank`, `plot.rankogram`
Examples

data(Woods2010)
p1 <- pairwise(treatment, event = r, n = N, studlab = author, 
    data = Woods2010, sm = "OR")
net1 <- netmeta(p1, small.values = "desirable")

ran1 <- rankogram(net1, nsim = 100)
ran1
print(ran1, cumulative.rankprob = TRUE)
plot(ran1)

Senn2013

Network meta-analysis in diabetes

Description

Network meta-analysis in diabetes comparing effects of a number of drugs on the HbA1c value.
These data are used as an example in Senn et al. (2013) and have been preprocessed for use in R package netmeta.

Format

A data frame with the following columns:

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TE</td>
<td>treatment effect</td>
</tr>
<tr>
<td>seTE</td>
<td>standard error of treatment effect</td>
</tr>
<tr>
<td>treat1</td>
<td>treatment 1</td>
</tr>
<tr>
<td>treat2</td>
<td>treatment 2</td>
</tr>
<tr>
<td>treat1.long</td>
<td>treatment 1 (full treatment names)</td>
</tr>
<tr>
<td>treat2.long</td>
<td>treatment 2 (full treatment names)</td>
</tr>
<tr>
<td>studlab</td>
<td>Study label</td>
</tr>
</tbody>
</table>

Details

Treatment labels provided by columns treat1 and treat2 have been abbreviated:

- acar = Acarbose
- benf = Benfluorex
- metf = Metformin
- migl = Miglitol
- piog = Pioglitazone
- plac = Placebo
- rosi = Rosiglitazone
• sita = Sitagliptin
• sulf = Sulfonylurea
• vild = Vildagliptin

Full treatment names are available in columns treat1.long and treat2.long.

Source


See Also

`netmeta`

Examples

data(Senn2013)
head(Senn2013)

```r
## Not run:
# Common effects model
#
net1 <- netmeta(TE, seTE, treat1.long, treat2.long, studlab,
    data = Senn2013, sm = "MD", random = FALSE, nchar.trts = 4)
net1
net1$Q.decomp

# Forest plot
#
forest(net1, ref = "plac")

# Comparison with reference group
#
netmeta(TE, seTE, treat1.long, treat2.long,
    studlab, data = Senn2013, reference = "plac")

# Random effects model
#
net2 <- netmeta(TE, seTE, treat1.long, treat2.long, studlab,
    data = Senn2013, common = FALSE)
net2
forest(net2, ref = "plac")

## End(Not run)
```
smokingcessation

Network meta-analysis of interventions for smoking cessation

Description

Network meta-analysis comparing the effects of a number of interventions for smoking cessation. These data are used as an example in Dias et al. (2013), page 651.

Format

A data frame with the following columns:

- **event1**: number of individuals with successful smoking cessation in arm 1
- **n1**: number of individuals in arm 1
- **event2**: number of individuals with successful smoking cessation in arm 2
- **n2**: number of individuals in arm 2
- **event3**: number of individuals with successful smoking cessation in arm 3
- **n3**: number of individuals in arm 3
- **treat1**: treatment 1
- **treat2**: treatment 2
- **treat3**: treatment 3

Source


See Also

pairwise, metabin, netmeta, netgraph.netmeta

Examples

data(smokingcessation)

# Transform data from 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.
#
p1 <- pairwise(list(treat1, treat2, treat3),
   event = list(event1, event2, event3), n = list(n1, n2, n3),
   data = smokingcessation, sm = "OR")
p1

# Conduct network meta-analysis
#

```r
net1 <- netmeta(p1)
net1

# Draw network graph
#
netgraph(net1, points = TRUE, cex.points = 3, cex = 1.25)
tname <- c("No intervention", "Self-help", "Individual counselling", "Group counselling")
netgraph(net1, points = TRUE, cex.points = 3, cex = 1.25, labels = tname)
```

---

**Stowe2010**

**Network meta-analysis of adjuvant treatments to levodopa therapy for Parkinson’s disease**

**Description**

This data set contains data from a Cochrane review assessing efficacy and safety of three drug classes as adjuvant treatment to levodopa therapy in patients with Parkinson’s disease and motor complications (Stowe et al., 2010). The authors conducted three pairwise meta-analyses comparing dopamine agonists, catechol-O-methyl transferase inhibitors (COMTIs), and monoamine oxidase type B inhibitors (MAOBIAs), respectively, with placebo.

The primary outcome was the mean reduction of the time spent in a relatively immobile ‘off’ phase (mean off-time), calculated in hours per day. Relative treatment effects were expressed as mean difference. Data on this outcome were available for 5,331 patients from 28 studies comparing an active treatment with placebo and one three-arm study comparing two active treatments with placebo.

**Format**

A data frame with the following columns:

- **study**: study label
- **id**: study id
- **t1**: treatment 1
- **y1**: treatment effect arm 1
- **sd1**: Standard deviation arm 1
- **n1**: Sample size arm 1
- **t2**: treatment 2
- **y2**: treatment effect arm 2
- **sd2**: Standard deviation arm 2
- **n2**: Sample size arm 2
- **t3**: treatment 3
- **y3**: treatment effect arm 3
- **sd3**: Standard deviation arm 3
- **n3**: Sample size arm 3
Source


See Also

`pairwise`, `metacont`, `netmeta`, `netgraph.netmeta`

Examples

data(Stowe2010)

# Transform data from arm-based format to contrast-based format
#
p1 <- pairwise(list(t1, t2, t3),
n = list(n1, n2, n3),
mean = list(y1, y2, y3), sd = list(sd1, sd2, sd3),
data = Stowe2010, studlab = study)
p1

# Conduct network meta-analysis
net1 <- netmeta(p1, ref = "plac")
net1

---

subset.pairwise  
**Return subset of pairwise object**

Description

The subset method returns a subset of a pairwise object.

Usage

```r
## S3 method for class 'pairwise'
subset(x, subset, ...)
```

Arguments

- **x**: An object of class `pairwise`.
- **subset**: A logical expression indicating elements or rows to keep: missing values are taken as false.
- **...**: Additional arguments.

Value

A pairwise object is returned.
Summary method for objects of class `netcomb`

**Usage**

```r
## S3 method for class 'netcomb'
summary(
  object,
  common = object$common,
  random = object$random,
  backtransf = object$backtransf,
  nchar.comps = object$nchar.comps,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

**Arguments**

- **object**: An object of class `netcomb`.
- **common**: A logical indicating whether results for the common effects model should be printed.
- **random**: A logical indicating whether results for the random effects model should be printed.
backtransf  A logical indicating whether results should be back transformed in printouts and forest plots.
nchar.comps  A numeric defining the minimum number of characters used to create unique component names.
warn.deprecated  A logical indicating whether warnings should be printed if deprecated arguments are used.
...  Additional arguments (to catch deprecated arguments).

Value
A list is returned with the same elements as a netcomb object.

Author(s)
Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also
netcomb, discomb

Examples

data(Linde2016)

  # Only consider studies including Face-to-face PST (to reduce # runtime of example)
  #
  # face <- subset(Linde2016, id %in% c(16, 24, 49, 118))

  # Conduct random effects network meta-analysis
  #
  net1 <- netmeta(lnOR, selnOR, treat1, treat2, id,
                  data = face, reference.group = "placebo",
                  sm = "OR", common = FALSE)

  # Additive model for treatment components
  #
  nc1 <- netcomb(net1)
  summary(nc1)
  print(summary(nc1), digits = 2, digits.stat = 3)

  ## Not run:
  # Conduct random effects network meta-analysis
  #
  net2 <- netmeta(lnOR, selnOR, treat1, treat2, id,
                  data = Linde2016, reference.group = "placebo",
                  sm = "OR", common = FALSE)

  # Additive model for treatment components
  #
`nc2 <- netcomb(net2)`

`summary(nc2)`

`print(summary(nc2), digits = 2, digits.stat = 3)`

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

---

### summary.netmeta

**Summary method for objects of class netmeta**

**Description**

Summary method for objects of class netmeta.

**Usage**

````
## S3 method for class 'netmeta'
summary(
  object,
  common = object$common,
  random = object$random,
  prediction = object$prediction,
  reference.group = object$reference.group,
  baseline.reference = object$baseline.reference,
  all.treatments = object$all.treatments,
  backtransf = object$backtransf,
  nchar.trts = object$nchar.trts,
  warn.deprecated = gs("warn.deprecated"),
  ...
)
```

**Arguments**

- **object**: An object of class netmeta.
- **common**: A logical indicating whether results for the common effects model should be printed.
- **random**: A logical indicating whether results for the random effects model should be printed.
- **prediction**: A logical indicating whether prediction intervals should be printed.
- **reference.group**: A logical indicating whether results should be expressed as comparisons of other treatments versus the reference treatment (default) or vice versa. This argument is only considered if `reference.group` has been specified.
- **baseline.reference**: Reference treatment.
all.treatments A logical or "NULL". If TRUE, matrices with all treatment effects, and confidence limits will be printed.
backtransf A logical indicating whether results should be back transformed in printouts and forest plots.
nchar.trts A numeric defining the minimum number of characters used to create unique treatment names (see Details).
warn.deprecated A logical indicating whether warnings should be printed if deprecated arguments are used.

Value

A list is returned with the following elements:

- **comparison**: Results for pairwise comparisons (data frame with columns studlab, treat1, treat2, TE, seTE, lower, upper, z, p).
- **comparison.nma.common**: Results for pairwise comparisons based on common effects model (data frame with columns studlab, treat1, treat2, TE, seTE, lower, upper, z, p, leverage).
- **comparison.nma.random**: Results for pairwise comparisons based on random effects model (data frame with columns studlab, treat1, treat2, TE, seTE, lower, upper, z, p).
- **common**: Results for common effects model (a list with elements TE, seTE, lower, upper, z, p).
- **random**: Results for random effects model (a list with elements TE, seTE, lower, upper, z, p).
- **predict**: Prediction intervals (a list with elements seTE, lower, upper).
- **studies**: Study labels coerced into a factor with its levels sorted alphabetically.
- **narms**: Number of arms for each study.
- **k**: Total number of studies.
- **m**: Total number of pairwise comparisons.
- **n**: Total number of treatments.
- **d**: Total number of designs (corresponding to the unique set of treatments compared within studies).
- **Q**: Overall heterogeneity / inconsistency statistic.
- **df.Q**: Degrees of freedom for test of heterogeneity / inconsistency.
- **pval.Q**: P-value for test of heterogeneity / inconsistency.
- **I2, lower.I2, upper.I2**: I-squared, lower and upper confidence limits.
- **tau**: Square-root of between-study variance.
- **Q.heterogeneity**: Overall heterogeneity statistic.
df.Q.heterogeneity
   Degrees of freedom for test of overall heterogeneity.

pval.Q.heterogeneity
   P-value for test of overall heterogeneity.

Q.inconsistency
   Overall inconsistency statistic.

df.Q.inconsistency
   Degrees of freedom for test of overall inconsistency.

pval.Q.inconsistency
   P-value for test of overall inconsistency.

sm
   A character string indicating underlying summary measure.

method
   A character string indicating which method is to be used for pooling of studies.

level
   The level used to calculate confidence intervals for individual studies.

level.ma
   The level used to calculate confidence intervals for pooled estimates.

common, random
   As defined above.

prediction, level.predict
   As defined above.

reference.group, baseline.reference
   As defined above.

all.treatments, backtransf
   As defined above.

ci.lab
   Label for confidence interval.

seq
   A character specifying the sequence of treatments.

tau.preset
   An optional value for the square-root of the between-study variance $\tau^2$.

sep.trts
   A character used in comparison names as separator between treatment labels.

nchar.trts
   A numeric defining the minimum number of characters used to create unique
treatment names.

title
   Title of meta-analysis / systematic review.

call
   Function call.

version
   Version of R package netmeta used to create object.

Author(s)

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

See Also

netmeta
### Examples

```r
data(smokingcessation)

p1 <- pairwise(list(treat1, treat2, treat3),
    event = list(event1, event2, event3), n = list(n1, n2, n3),
    data = smokingcessation, sm = "OR")
net1 <- netmeta(p1)

summary(net1)

## Not run:
data(Senn2013)

# Conduct common effects network meta-analysis
# net2 <- netmeta(TE, seTE, treat1, treat2, studlab, data = Senn2013, sm = "MD", random = FALSE)
print(net2, ref = "plac", digits = 3)
summary(net2)

# Conduct random effects network meta-analysis
# net3 <- netmeta(TE, seTE, treat1, treat2, studlab, data = Senn2013, sm = "MD", common = FALSE)
print(net3, ref = "plac", digits = 3)
summary(net3)

## End(Not run)
```

### treats

*Abbreviate treatment names*

#### Description

Auxiliary functions to create uniquely abbreviated treatment names.

#### Usage

```r
treats(x, nchar.trts = 8, row = TRUE)
comps(x, trts, sep.trts, nchar.trts = 8, row = TRUE)
```

#### Arguments

- **x**: A vector with treatment or comparison names or a matrix with treatment or comparison names as row and / or column names.
- **nchar.trts**: A numeric defining the minimum number of characters used to create unique treatment names.
row A logical indicating whether row or column names should be used (only considered if argument x is a matrix).

trts A character vector with treatment names.

sep.trts A character used in comparison names as separator between treatment labels.

Details

These auxiliary functions can be used to create uniquely abbreviated treatment names (and are used internally in several R functions for this purpose).

In order to construct uniquely abbreviated treatment names, `treats` uses `substring` to extract the first `nchar.trts` characters. If these abbreviated treatment names are not unique, `abbreviate` with argument `minlength = nchar.trts` is used.

In order to construct comparisons with uniquely abbreviated treatment names, `comps` calls `treats` internally.

Author(s)

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

See Also

`netmeta`, `print.netmeta`, `print.summary.netmeta`

Examples

data(smokingcessation)

# Transform data from arm-based format to contrast-based format
#
p1 <- pairwise(list(treat1, treat2, treat3),
    event = list(event1, event2, event3), n = list(n1, n2, n3),
    data = smokingcessation, sm = "OR")

# Conduct random effects network meta-analysis and show data frame
#
net1 <- netmeta(p1, common = FALSE)

# Full treatment names
#
net1$trts

# Treatment names with maximal four characters
#
treats(net1$trts, nchar.trts = 4)

## Not run:
data(Senn2013)
#
net2 <- netmeta(TE, seTE, treat1.long, treat2.long, studlab, data = Senn2013)
# Full treatment names
#
net2$trts

# Treatment names with four characters
#
treats(net2$trts, nchar.trts = 4)

# With two characters
#
treats(net2$trts, nchar.trts = 2)

# With one character (if possible)
#
treats(net2$trts, nchar.trts = 1)

# Full comparison names
#
net2$comparisons

# Abbreviated comparison names
#
with(net2, comps(comparisons, trts, sep.trts, nchar = 4))

## End(Not run)

---

**Woods2010**  

*Count statistics of survival data*

---

**Description**

Count mortality statistics in randomised controlled trials of treatments for chronic obstructive pulmonary disease (Woods et al. (2010), Table 1).

**Format**

A data frame with the following columns:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>author</strong></td>
<td>first author / study name</td>
</tr>
<tr>
<td><strong>treatment</strong></td>
<td>treatment</td>
</tr>
<tr>
<td><strong>r</strong></td>
<td>number of deaths in treatment arm</td>
</tr>
<tr>
<td><strong>N</strong></td>
<td>number of patients in treatment arm</td>
</tr>
</tbody>
</table>

**Source**

See Also

`pairwise`, `metabin`, `netmeta`

Examples

data(Woods2010)

# 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.
#
p1 <- pairwise(treatment, event = r, n = N,
          studlab = author, data = Woods2010, sm = "OR")
p1

# Conduct network meta-analysis
#
net1 <- netmeta(p1)
net1

## Not run:
# Show forest plot
#
forest(net1, ref = "Placebo", drop = TRUE,
       leftlabs = "Contrast to Placebo")

## End(Not run)
## Index

* **contribution**
  - netcontrib. 84
  
* **datagen**
  - pairwise. 158

* **datasets**
  - Baker2009, 9
  - dietaryfat, 13
  - Dogliotti2014, 22
  - Dong2013, 23
  - Franchini2012, 42
  - Gurusamy2011, 48
  - Linde2015, 58
  - Linde2016, 59
  - Senn2013, 193
  - smokingcessation, 195
  - Stowe2010, 196
  - Woods2010, 205

* **hplot**
  - forest.netbind, 24
  - forest.netcomb, 26
  - forest.netcomparison, 29
  - forest.netcomplex, 32
  - forest.netmeta, 34
  - forest.netsplit, 38
  - funnel.netmeta, 43
  - heatplot, 54
  - heatplot.netmeta, 55
  - netgraph, 88
  - netgraph.discomb, 90
  - netgraph.netcomb, 91
  - netgraph.netconnection, 92
  - netgraph.netimpact, 94
  - netgraph.netmeta, 96
  - netheat, 104
  - plot.netposet, 165
  - plot.netrank, 169
  - radial.netmeta, 187

* **package**
  - netmeta-package, 4

* **plot**
  - hasse, 49

* **print**
  - netleague, 108
  - print.decomp.design, 175
  - print.netbind, 176
  - print.netcomb, 178
  - print.netimpact, 180
  - print.summary.netcomb, 181
  - print.summary.netmeta, 183
  - summary.netcomb, 198

* **summary**
  - summary.netmeta, 200

abbreviate, 17, 67, 75, 79, 86, 123, 133, 204
arrows, 166
as.data.frame.netconnection, 7
as.data.frame.netmeta, 8

Baker2009, 9
baujat.netpairwise (netpairwise), 137
colours, 94
coms (treats), 203
decomp.design, 4, 10, 123, 133, 176
dietaryfat, 13
discomb, 4, 14, 28, 31, 33, 65, 71, 74, 76, 78, 80, 81, 83, 90, 179, 183, 199
Dogliotti2014, 22
Dong2013, 23
eigen, 99
element_text, 170, 171
factor, 114
forest.meta, 25–28, 30–33, 36, 37, 40, 41
forest.netbind, 5, 24, 65
forest.netcomb, 4, 21, 26, 71
forest.netcomparison, 29
forest.netcomplex, 32
INDEX

forest.netcom (forest.netcomb), 26
forest.netmeta, 4, 34, 128
forest.netpairwise (netpairwise), 137
forest.netsplit, 38, 154
Franchini2012, 42
funnel.meta, 45, 46, 189
funnel.netmeta, 5, 43, 62
funnel.netpairwise (netpairwise), 137
Gurusamy2011, 48
hasse, 4, 49, 49, 144, 167, 171
hatmatrix, 51, 85
heatplot, 54
heatplot.crossnma, 54
heatplot.netmeta, 54, 55
I, 100, 101
invmat, 56
Linde2015, 58
Linde2016, 59
metabias, 45, 46, 62, 188, 189
metabias.meta, 61
metabias.netmeta, 60
metabias.netpairwise (netpairwise), 137
metabin, 10, 22, 23, 48, 58, 123, 160, 161, 163, 195, 206
metacont, 43, 123, 160, 161, 163, 197
metacum.netpairwise (netpairwise), 137
metagen, 123, 128, 139, 140, 160, 161, 163
metainc, 14, 123, 160, 161, 163
metainf.netpairwise (netpairwise), 137
metareg.netpairwise (netpairwise), 137
netbind, 5, 26, 63, 177
netcomb, 4, 17, 21, 26, 28, 31, 33, 60, 65, 65, 74, 76, 78, 80, 92, 179, 183, 199
netcomparison, 5, 20, 21, 31, 71, 73, 79, 80
netcomplex, 5, 20, 21, 33, 71, 75, 76, 77
netconnection, 5, 7, 21, 88, 93
netconnection (netconnection.default), 81
netconnection.default, 81
netcontrib, 5, 53, 84
netdistance, 83, 87
netgraph, 4, 88
netgraph.discomb, 89, 90
netgraph.netcomb, 89, 91
netgraph.netconnection, 89, 92
netgraph.netimpact, 89, 94, 108
netgraph.netmeta, 14, 43, 89–93, 95, 96, 114, 163, 195, 197
netheat, 4, 11, 12, 53, 104
netimpact, 5, 95, 107, 181
netleague, 4, 108
netmatrix, 113
netmeasures, 5, 36, 115, 154, 158
netmeta-package, 4
netmetabin, 4, 10, 22, 23, 48, 107, 108, 129, 152, 154, 157, 158
netpairwise, 5, 137, 162
netposet, 4, 50, 58, 111, 141, 167, 171
netrank, 4, 35, 36, 50, 110, 111, 120, 128, 143, 144, 146, 167, 171, 192
netsplit, 4, 139, 140, 149, 158
nettable, 4, 155
order, 28, 36
pairwise, 10, 14, 15, 22, 23, 43, 48, 58, 82, 120, 121, 123, 127, 128, 132, 136, 140, 158, 163, 193, 197, 198, 206
parkinson (Franchini2012), 42
paste, 123, 133
plot.netbind (forest.netbind), 24
plot.netcomb (forest.netcomb), 26
plot.netcomparison
(forest.netcomparison), 29
plot.netcomplex (forest.netcomplex), 32
plot.netmeta (forest.netmeta), 34
plot.netpairwise (netpairwise), 137
plot.netposet, 4, 144, 165, 171
plot.netrank, 50, 144, 169
plot.netsplit (forest.netsplit), 38
plot.rankogram, 148, 173, 192
prcomp, 99, 167
print.data.frame, 147
print.decomp.design, 175
print.default, 35, 82, 147, 191
print.hatmatrix (hatmatrix), 51
INDEX

print.metabias.netpairwise (netpairwise), 137
print.metacum.netpairwise (netpairwise), 137
print.metainf.netpairwise (netpairwise), 137
print.metareg.netpairwise (netpairwise), 137
print.netbind, 176
print.netcomb, 71, 178
print.netcomparison (netcomparision), 73
print.netcomplex (netcomplex), 77
print.netconnection (netconnection.default), 81
print.netcontrib (netcontrib), 84
print.netimpact, 108, 180
print.netleague (netleague), 108
print.netmeta, 204
print.netmeta (netmeta), 118
print.netpairwise (netpairwise), 137
print.netposet (netposet), 141
print.netrank (netrank), 146
print.netsplit (netsplit), 149
print.nettable (nettable), 155
print.rankogram (rankogram), 190
print.summary.netcomb, 181
print.summary.netmeta, 123, 133, 183, 204
print.summary.netpairwise (netpairwise), 137
print.trimfill.netpairwise (netpairwise), 137
radial.meta, 188, 189
radial.netmeta, 187
radial.netpairwise (netpairwise), 137
rainbow, 101
rankogram, 4, 148, 173, 174, 190
rma.mv, 121, 127
scale_fill_gradient2, 170
Senn2013, 193
sequential_hcl, 101
smokingcessation, 195
solve, 57
sqrt, 100, 101
Stowe2010, 196
subset.pairwise, 197
substring, 204
summary.netcomb, 21, 183, 198

summary.netmeta, 185, 200
summary.netpairwise (netpairwise), 137
text, 166
treats, 17, 67, 75, 79, 86, 123, 133, 203
trimfill.netpairwise (netpairwise), 137
Woods2010, 205