Package 'crossnma'

May 8, 2026

Title Cross-Design & Cross-Format Network Meta-Analysis and Regression

Version 1.3.0

Date 2024-11-28

Depends R (>= 3.5), meta (>= 8.0-1), netmeta (>= 2.8-0)

Imports rjags, coda, dplyr, plyr, rlang, magrittr, tidyr, ggplot2

Suggests rmarkdown, knitr

Author Tasnim Hamza [aut] (ORCID: <https://orcid.org/0000-0002-4700-6990>),
Guido Schwarzer [aut, cre] (ORCID:
<https://orcid.org/0000-0001-6214-9087>),
Georgia Salanti [aut] (ORCID: <https://orcid.org/0000-0002-3830-8508>)

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

URL https://github.com/htx-r/crossnma

Description Network meta-analysis and meta-regression (allows
including up to three covariates) for individual participant data,
aggregate data, and mixtures of both formats using the three-level
hierarchical model. Each format can come from randomized controlled
trials or non-randomized studies or mixtures of both. Estimates are
generated in a Bayesian framework using JAGS. The implemented models
are described by Hamza et al. 2023 <DOI:10.1002/jrsm.1619>.

License GPL (>= 2)

Encoding UTF-8

LazyData true

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2024-11-28 13:50:22 UTC

Contents

`crossnma-package` *crossnma: An R package for synthesizing cross-design evidence and cross-format data using Bayesian methods in network meta-analysis and network meta-regression*

Description

An R package **crossnma** for performing (network) meta-analysis and (network) meta-regression (allows including up to 3 covariates) of individual participant data and aggregate data or combination of both (Hamza et al., 2024). Each format can come from randomized controlled trials or non-randomized studies. Estimates are generated in a Bayesian framework using JAGS. The implemented models are described by Hamza et al. (2023).

Details

The evidence in network meta-analysis (NMA) typically comes from randomized controlled trials (RCT) where aggregate data (AD) are extracted from published reports. Retrieving individual participant data (IPD) allows considering participant covariates to explain some of the heterogeneity/inconsistency in the network and identify effect modifiers. Additionally, evidence from non-randomized studies (NRS) reflects the reality in clinical practice and bridges the efficacy-effectiveness gap. The cross-NMA/NMR model is a Bayesian suite for evidence synthesis which extends and integrates four different approaches that combine RCT and NRS evidence into a three-level hierarchical model for the synthesis of IPD and AD. The four approaches account for differences in the design and risk of bias in the RCT and NRS evidence. These four approaches variously ignoring differences in risk of bias, using NRS to construct penalized treatment effect priors and

bias-adjustment models that control the contribution of information from high risk of bias studies in two different ways.

Further details:

- To have a list of all R functions available in **crossnma** type `help(package = "crossnma")`
- Hamza et al. (2024) is the preferred citation in publications for **crossnma**. Type `citation("crossnma")` for a BibTeX entry of this publication.
- To report problems and bugs send an email to <hamza.a.tasnim@gmail.com>
- The development version of **crossnma** is available on GitHub https://github.com/htx-r/crossnma.

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>, Georgia Salanti <georgia.salanti@ispm.unibe.ch>

References

Dias S, Welton NJ, Marinho VCC et al. (2010): Estimation and adjustment of bias in randomized evidence by using mixed treatment comparison meta-analysis. *Journal of the Royal Statistical Society: Series A*, **173**, 613-29

Hamza T, Chalkou K, Pellegrini F et al. (2023): Synthesizing cross-design evidence and cross-format data using network meta-regression. *Research Synthesis Methods*, **14**, 283-300

Hamza T, Schwarzer G, Salanti G (2024): crossnma: An R Package to Synthesize Cross-Design Evidence and Cross-Format Data Using Network Meta-Analysis and Network Meta-Regression. *BMC Medical Research Methodology*, **24**, 169.

Plummer M (2003): JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling

Saramago P, Sutton AJ, Cooper NJ, Manca A (2012): Mixed treatment comparisons using aggregate and individual participant level data. *Statistics in Medicine*, **31**, 3516-36

Tramacere I, Del Giovane C, Salanti G et al. (2015): Immunomodulators and immunosuppressants for relapsing-remitting multiple sclerosis: a network meta-analysis. *Cochrane Database of Systematic Reviews*, **9**, John Wiley & Sons, Ltd. doi:10.1002/14651858.CD011381.pub2

Verde PE (2021): A bias-corrected meta-analysis model for combining studies of different types and quality. *Biometrical Journal*, **63**, 406-22

See Also

Useful links:

- https://github.com/htx-r/crossnma

crossnma *Run JAGS to fit cross NMA and NMR*

Description

This function takes the JAGS model from an object produced by `crossnma.model` and runs it using `jags.model` from R package **rjags**.

Usage

```
crossnma(
  x,
  inits = NULL,
  n.adapt = 1000,
  n.burnin = floor(n.iter/2),
  n.iter = 2000,
  thin = max(1, floor((n.iter - n.burnin)/1000)),
  n.chains = 2,
  monitor = NULL,
  level.ma = x$level.ma,
  backtransf = x$backtransf,
  quiet = TRUE,
  n.thin = NULL
)
```

Arguments

`x`	An object produced by `crossnma.model`.
`inits`	A list of lists with `n.chains` elements; each element contains initial values for each model parameter or a function that generates starting values. Default is different numbers in `.RNG.seed` and `.RNG.name = "base::Mersenne-Twister"`.
`n.adapt`	Number of adaptations for the MCMC chains.
`n.burnin`	Number of burnin iterations for the MCMC chains. Default is `n.iter / 2` which discards the first half of the iterations.
`n.iter`	Number of iterations to run each MCMC chain.
`thin`	Thinning for the MCMC chains. Default is max(1, floor((n.iter - n.burnin) / 1000)), that is only thinning if there are more than 2000 iterations.
`n.chains`	Number of MCMC chains.
`monitor`	A character vector of the names of the parameters to be monitored. Basic parameters (depends on the analysis) will be automatically monitored and only additional parameters need to be specified.
`level.ma`	The level used to calculate credible intervals for network estimates.
`backtransf`	A logical indicating whether results should be back transformed in printouts. If `backtransf = TRUE`, results for `sm = "OR"` are presented as odds ratios rather than log odds ratios, for example.

quiet	A logical passed on to jags.model.
n.thin	Deprecated argument (replaced by thin).

Value

An object of class crossnma which is a list containing the following components:

jagsfit	An "rjags" object produced when rjags package used to run the JAGS model.
model	The crossnma.model object obtained from crossnma.model which was used to run JAGS.
trt.key	A table of treatment names and their correspondence to integers used in the JAGS model.
inits, n.adapt, n.burnin, n.iter	As defined above.
thin, n.chains	As defined above.
call	Function call.
version	Version of R package **crossnma** used to create object.

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

crossnma.model, jags.model

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Display the output
summary(fit)
plot(fit)

## End(Not run)
```

`crossnma.model` *Create JAGS model and data to perform cross network meta analysis or meta-regression*

Description

This function creates a JAGS model and the needed data for cross-design and cross-format network meta-analysis or meta-regression for different types of outcome

Usage

```
crossnma.model(
  trt,
  study,
  outcome,
  n,
  design,
  se,
  cov1 = NULL,
  cov2 = NULL,
  cov3 = NULL,
  bias = NULL,
  unfav = NULL,
  bias.covariate = NULL,
  bias.group = NULL,
  prt.data = NULL,
  std.data = NULL,
  sm,
  reference = NULL,
  trt.effect = "random",
  level.ma = gs("level.ma"),
  sucra = FALSE,
  small.values = NULL,
  cov1.value = NULL,
  cov2.value = NULL,
  cov3.value = NULL,
  cov1.ref = NULL,
  cov2.ref = NULL,
  cov3.ref = NULL,
  reg0.effect = "independent",
  regb.effect = "random",
  regw.effect = "random",
  split.regcoef = TRUE,
  method.bias = NULL,
  bias.type = NULL,
  bias.effect = "common",
  down.wgt = NULL,
```

```
  prior.tau.trt = NULL,
  prior.tau.reg0 = NULL,
  prior.tau.regb = NULL,
  prior.tau.regw = NULL,
  prior.tau.bias = NULL,
  prior.pi.high.rct = NULL,
  prior.pi.low.rct = NULL,
  prior.pi.high.nrs = NULL,
  prior.pi.low.nrs = NULL,
  run.nrs.var.infl = 1,
  run.nrs.mean.shift = 0,
  run.nrs.trt.effect = "common",
  run.nrs.n.adapt = 1000,
  run.nrs.n.iter = 10000,
  run.nrs.n.burnin = 4000,
  run.nrs.thin = 1,
  run.nrs.n.chains = 2,
  backtransf = gs("backtransf"),
  run.nrs.n.thin = NULL
)
```

Arguments

`trt`	Treatment variable in `prt.data` and `std.data`.
`study`	Study variable in `prt.data` and `std.data`.
`outcome`	Outcome variable in `prt.data` and `std.data`.
`n`	Number of participants in `std.data`.
`design`	Design variable in `prt.data` and `std.data`.
`se`	Standard error variable in `std.data` (required only for continuous outcome when `sm = "MD"` or `"SMD"`).
`cov1`	Optional first covariate in `prt.data` and `std.data` to conduct network meta-regression (see Details).
`cov2`	Optional second covariate in `prt.data` and `std.data` to conduct network meta-regression (see Details).
`cov3`	Optional third covariate in `prt.data` and `std.data` to conduct network meta-regression (see Details).
`bias`	Optional variable with information on risk of bias in `prt.data` and `std.data`. Possible values for this variable are "low", "high" or "unclear" (can be abbreviated). These values must be identical for all participants from the same study. Either this variable or bias.covariate variable should be provided when method.bias = "adjust1" or "adjust2".
`unfav`	An optional variable in `prt.data` and `std.data` indicating the unfavored treatment in each study (should be provided when method.bias = "adjust1" or "adjust2"). The entries of this variable are either 0 (unfavored treatment) or 1 (favorable treatment or treatments). Each study should include only one 0 entry. The values need to be repeated for participants who take the same treatment.

`bias.covariate` An optional variable in `prt.data` and `std.data` indicate the covariate used to estimate the probability of bias. Either this variable or bias variable should be provided when method.bias = "adjust1" or "adjust2".

`bias.group` An optional variable in `prt.data` and `std.data` that indicates the bias effect in each study (can be provided when method.bias = "adjust1" or "adjust2"). The entries of these variables should be either 1 (study has inactive treatment and its estimate should be adjusted for bias effect), 2 (study has only active treatments and its estimate should be adjusted for bias effect (different from inactive bias effect) or 0 (study does not need any bias adjustment). The values need to be repeated for the participants assigned to the same treatment. Default is 1.

`prt.data` An object of class data.frame containing the individual participant dataset. Each row contains the data of a single participant. The dataset needs to have the following columns: treatment, study identification, outcome (event and non-event), design. Additional columns might be required for certain analyses.

`std.data` An object of class data.frame containing the study-level dataset. Each row represents the information of study arm. The dataset needs to have the following columns: treatment, study identification, outcome (number of events), sample size and design. Additional columns might be required for certain analyses.

`sm` A character indicating the underlying summary measure. Options are: Odds Ratio "OR" (default), Risk Ratio "RR", Mean Difference "MD" or Standardised Mean Difference "SMD".

`reference` A character indicating the name of the reference treatment. When the reference is not specified, the first alphabetic treatment will be used as a reference in the analysis.

`trt.effect` A character defining the model for the study-specific treatment effects. Options are "random" (default) or "common".

`level.ma` The level used to calculate credible intervals for network estimates.

`sucra` Logical. If TRUE SUCRA (Surface Under the Cumulative Ranking) values will be calculated within JAGS.

`small.values` A character string specifying whether small treatment effects indicate a beneficial (`"desirable"`) or harmful (`"undesirable"`) effect, can be abbreviated. This argument is required when `sucra` is TRUE.

`cov1.value` The participant covariate value of `cov1` for which to report the results. Must be specified for network meta-regression, `sucra` is TRUE and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.

`cov2.value` The participant covariate value of `cov2` for which to report the results. Must be specified for network meta-regression, `sucra` is TRUE and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.

`cov3.value` The participant covariate value of `cov3` for which to report the results. Must be specified for network meta-regression, `sucra` is TRUE and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.

`cov1.ref` An optional value to center the first covariate which is only useful for a continuous covariate. Dichotomous covariates should be given NA value. The default is the overall minimum covariate value from all studies.

`cov2.ref` An optional value to center the second covariate which is only useful for a continuous covariate. Dichotomous covariates should be given NA value. The default is the overall minimum covariate value from all studies.

`cov3.ref` An optional value to center the third covariate which is only useful for a continuous covariate. Dichotomous covariates should be given NA value. The default is the overall minimum covariate value from all studies.

`reg0.effect` An optional character (can by provided when at least `cov1` is not NULL) indicating the relationship across studies for the prognostic effects expressed by the regression coefficient, (β_0), in a study j. Options are "independent" or "random". We recommend using "independent" (default).

`regb.effect` An optional character (can by provided when at least `cov1` is not NULL) indicating the relationship across treatments for the between-study regression coefficient (β^B). This parameter quantifies the treatment-mean covariate interaction. Options are "independent", "random" or "common". Default is "random".

`regw.effect` An optional character (can by provided when at least `cov1` is not NULL) indicating the relationship across treatments for the within-study regression coefficient (β^W). This parameter quantifies the treatment-covariate interaction effect at the individual level. Options are "independent", "random" and "common". Default is "random".

`split.regcoef` A logical value (needed when at least `cov1` is not NULL). If TRUE (default) the within- and between-study coefficients will be splitted in the analysis of `prt.data`. When the split.regcoef = FALSE, only a single regression coefficient will be estimated to represent both the between-studies and within-studies covariate effects. In this case, both arguments `regb.effect` and `regw.effect` need to be given the same option to model the single regression effect.

`method.bias` A character for defining the method to combine randomized clinical trials (RCT) and non-randomized studies (NRS). Options are "naive" for naive or unadjusted synthesize, "prior" for using NRS evidence to construct priors for the relative treatment effects in RCTs analysis, or "adjust1" and "adjust2" to allow a bias adjustment. When only one design is available (either rct or nrs), this argument needs also to be specified to indicate whether unadjusted (naive) or bias-adjusted analysis (adjust1 or adjust2) should be applied.

`bias.type` An optional character defining the relationship between the bias effect and the treatment effect (required when method.bias = "adjust1"). Three options are possible: "add" to add the additive bias effect, "mult" for multiplicative bias effect and "both" includes both an additive and a multiplicative terms.

`bias.effect` An optional character indicating the relationship for the bias coefficients across studies. Options are "random" or "common" (default). It can be provided when method.bias = "adjust1" or "adjust2".

`down.wgt` An optional numeric indicating the percent to which studies at high risk of bias will be downweighted on average. The value ranges between 0 and 1. It can be provided when method.bias = "adjust1" or "adjust2".

`prior.tau.trt` Optional string to specify the prior for the between-study heterogeneity in treatment effects in JAGS model (when trt.effect="random"). The default prior is constructed from the data (see Details).

`prior.tau.reg0` Optional string to specify the prior for the between-study heterogeneity in prognostic effects in JAGS model (when reg0.effect="random"). The default prior is constructed from the data (see Details).

`prior.tau.regb` Optional string to specify the prior for the between-study heterogeneity in between-study covariate effects in JAGS model (when regb.effect="random"). The default prior is constructed from the data (see Details).

`prior.tau.regw` Optional string to specify the prior for the between-study heterogeneity in within-study covariate effects in JAGS model (when regw.effect="random"). The default prior is constructed from the data (see Details).

`prior.tau.bias` Optional string to specify the prior for the between-study heterogeneity in bias effects in JAGS model (when bias.effect="random").

`prior.pi.high.rct`
Optional string to provide the prior for the bias probability of randomised clinical trials (RCT) with high risk of bias in JAGS model (when the method.bias = "adjust1" or "adjust2" and the variable "bias" is provided). The default is the beta distribution "dbeta(10,1)".

`prior.pi.low.rct`
Optional string to provide the prior for the bias probability of randomised clinical trials (RCT) with low risk of bias in JAGS model (when the method.bias = "adjust1" or "adjust2" and the variable "bias" is provided). The default is the beta distribution "dbeta(1,10)".

`prior.pi.high.nrs`
Optional string to provide the prior for the bias probability of non-randomised studies (NRS) with high risk of bias in JAGS model (when the method.bias = "adjust1" or "adjust2" and the variable "bias" is provided). The default is the beta distribution "dbeta(30,1)".

`prior.pi.low.nrs`
Optional string to provide the prior for the bias probability of non-randomised studies (NRS) with low risk of bias in JAGS model (when the method.bias = "adjust1" or "adjust2" and the variable "bias" is provided). The default is the beta distribution "dbeta(1,30)".

`run.nrs.var.infl`
Optional numeric controls the common inflation of the variance of NRS estimates (w) and its values range between 0 (NRS does not contribute at all and the prior is vague) and 1 (the NRS evidence is used at face value, default approach). This argument can be provided when the NRS used as a prior (method.bias = "prior").

`run.nrs.mean.shift`
Optional numeric controls the bias shift (ζ) to be added / subtracted from the estimated mean treatment effects (on the log-scale when `sm = "OR"` or `"RR"`) from NRS network (0 is the default). This argument can be provided when the NRS used as a prior (`method.bias = "prior"`).

`run.nrs.trt.effect`
: Optional character indicates how to combine treatment effects across NRS studies. Options are "random" or "common" (default). This argument can be provided when the NRS used as a prior (method.bias = "prior").

`run.nrs.n.adapt`
: Optional numeric specifies the number of iterations for adaptation. This determines how many steps the algorithm takes to adjust its parameters before starting the main sampling process. Default is 1000. This argument can be provided when the NRS used as a prior (method.bias = "prior").

`run.nrs.n.iter`
: Optional numeric specifies the number of iterations to run MCMC chains for NRS network. Default is 10000. This argument can be provided when the NRS used as a prior (method.bias = "prior").

`run.nrs.n.burnin`
: Optional numeric specifies the number of burn-in to run MCMC chains for NRS network. Default is 4000. This argument can be provided when the NRS used as a prior (method.bias = "prior").

`run.nrs.thin`
: Optional numeric specifying thinning to run MCMC chains for NRS network. Default is 1. This argument can be provided when the NRS used as a prior (method.bias = "prior").

`run.nrs.n.chains`
: Optional numeric specifies the number of chains to run MCMC chains for NRS network. Default is 2. This argument can be provided when the NRS used as a prior (method.bias = "prior").

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

`run.nrs.n.thin`
: Deprecated argument (replaced by `run.nrs.thin`).

Details

This function creates a JAGS model and the needed data. The JAGS code is created from the internal function `crossnma.code`.

Covariates provided in arguments `cov1`, `cov2` and `cov3` can be either numeric or dichotomous (should be provided as factor or character) variables. By default, no covariate adjustment is applied (network meta-analysis).

The default prior for the between-study heterogeneity parameters (prior.tau.trt, prior.tau.reg0, prior.tau.regb, prior.tau.regw and prior.tau.bias) is a uniform distribution over the range 0 to ML, where ML is the largest maximum likelihood estimates of all relative treatment effects in all studies.

Value

An object of class `crossnma.model` containing information on the JAGS model, which is a list containing the following components:

`model`
: A long character string containing JAGS code that will be run in `jags.parallel`.

`data`
: The data to be used to run JAGS model.

`trt.key`	A table of the treatments and its mapped integer number (as used in JAGS model).
`study.key`	A table of the studies and its mapped integer number (as used in JAGS model).
`trt.effect`	A character defining the model for the study-specific treatment effects.
`method.bias`	A character for defining the method to analyse combine randomized clinical trials (RCT) or V and non-randomized studies (NRS).
`covariate`	A vector of the the names of the covariates (`cov1`, `cov2` and `cov3`) in prt.data and std.data used in network meta-regression.
`cov.ref`	A vector of values of `cov1.ref`, `cov2.ref`, `cov3.ref` to center continuous covariates. Dichotomous covariates take NA.
`dich.cov.labels`	A matrix with the levels of each dichotomous covariate and the corresponding assigned 0 / 1 values.
`split.regcoef`	A logical value. If FALSE the within- and between-study regression coefficients will be considered equal.
`regb.effect`	A character indicating the model for the between-study regression coefficients across studies.
`regw.effect`	A character indicating the model for the within-study regression coefficients across studies.
`bias.effect`	A character indicating the model for the bias coefficients across studies.
`bias.type`	A character indicating the effect of bias on the treatment effect; additive ("add") or multiplicative ("mult") or both ("both").
`all.data.ad`	A data.frame object with the prt.data (after it is aggregated) and std.data in a single dataset.
`call`	Function call.
`version`	Version of R package **crossnma** used to create object.

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

crossnma, jags.parallel

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
```

```
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Print call of JAGS model
mod

# Print JAGS code
summary(mod)

# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Display the output
summary(fit)
plot(fit)

## End(Not run)
```

`heatplot.crossnma` *Heat Plot*

Description

Produces a heat plot that contain point estimates of relative effects for all possible pairs of treatments along with credible intervals obtained with the quantile method.

Usage

```
## S3 method for class 'crossnma'
heatplot(
  x,
  median = TRUE,
  backtransf = x$model$backtransf,
  seq = NULL,
  low.colour = "red",
  mid.colour = "white",
  high.colour = "springgreen4",
  cov1.value = NULL,
  cov2.value = NULL,
  cov3.value = NULL,
  size = 6,
  size.trt = 20,
  size.axis = 12,
  digits = gs("digits.forest"),
  exp = backtransf,
  ...
)
```

Arguments

x	An object created with crossnma.
median	A logical indicating whether to use the median (default) or mean to measure relative treatment effects.
backtransf	A logical indicating whether results should be back transformed. If backtransf = TRUE, results for sm = "OR" are presented as odds ratios rather than log odds ratios, for example.
seq	A vector of treatment names (character) representing the order in which to display these treatments.
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).
cov1.value	The participant covariate value of cov1 for which to report the results. Must be specified for network meta-regression and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.
cov2.value	The participant covariate value of cov2 for which to report the results. Must be specified for network meta-regression and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.
cov3.value	The participant covariate value of cov3 for which to report the results. Must be specified for network meta-regression and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.
size	The size of cell entries with the relative treatment effect and 95% credible 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	The number of digits to be used when displaying the results.
exp	Deprecated argument (replaced by backtransf).
...	Additional arguments (ignored at the moment).

Value

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

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>

See Also

crossnma

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model. The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Create a heat plot
heatplot(fit)

## End(Not run)
```

---

ipddata *Simulated individual participant dataset.*

Description

A dataset containing 1944 participants who are treated in four different treatments: A, B, C and D. The dataset includes four studies. The outcome is binary. There are 10 attributes on individual level.

Usage

```
ipddata
```

Format

A data frame with 1944 rows and 10 variables:

id numeric, study identifier

relapse binary data, respond indicator, 0=no relapse and 1=relapse

treat character, indicating the assigned treatment to each participant

design character, design of the study, either 'rct' or 'nrs'

age numeric, age of the participant

sex binary data, sex of the participant, 0=female and 1=male

rob character, the risk of bias of the study, 'low', 'high','unclear'

unfavored numeric, the indicator of the unfavored treatment in each study, values are 0 or 1

bias.group numeric, the bias effect of the study, 1 = if the study has inactive treatment and adjust for bias effect, 2= if the study has active treatments and it is assumed another bias effect, 0=no bias adjustment

year numeric, the year study published

`league.crossnma` *League Table*

Description

Produces a league table that contains point estimates of relative effects for all possible pairs of treatments along with 95% credible intervals obtained with the quantile method.

Usage

```
## S3 method for class 'crossnma'
league(
  x,
  median = TRUE,
  backtransf = x$model$backtransf,
  order = NULL,
  cov1.value = NULL,
  cov2.value = NULL,
  cov3.value = NULL,
  digits = gs("digits"),
  direction = "wide",
  exp = backtransf,
  ...
)

league(x, ...)

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

Arguments

`x`	An object created with `crossnma`.
`median`	A logical indicating whether to use the median (default) or mean to measure relative treatment effects.

backtransf	A logical indicating whether results should be back transformed. If backtransf = TRUE, results for sm = "OR" are presented as odds ratios rather than log odds ratios, for example.
order	A vector of treatment names (character) representing the order in which to display these treatments.
cov1.value	The participant covariate value of cov1 for which to report the results. Must be specified for network meta-regression and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.
cov2.value	The participant covariate value of cov2 for which to report the results. Must be specified for network meta-regression and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.
cov3.value	The participant covariate value of cov3 for which to report the results. Must be specified for network meta-regression and when individual participant dataset is used in the analysis. For dichotomous covariates, a character of the level (used in the data) should be indicated.
digits	The number of digits to be used when displaying the results.
direction	The format to display the league table. Two options "wide" (default) and "long".
exp	Deprecated argument (replaced by backtransf).
...	Additional arguments (ignored at the moment).

Value

A league table. Row names indicate comparator treatments. The table will be displayed in a long or wide formatting.

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>

See Also

crossnma

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
```

```
  reference = "A", trt.effect = "random", method.bias = "naive")

# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Create league tables
league(fit)                     #  wide format
league(fit, direction = "long") #  long format

## End(Not run)
```

`netconnection.crossnma` *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. 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

Usage

```
## S3 method for class 'crossnma'
netconnection(data, ...)
```

Arguments

`data`	An object produced by `crossnma`.
`...`	... Additional arguments (passed on to `netconnection`).

Value

An object of class `netconnection` with corresponding `print` function. The object is a list containing 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)

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

See Also

netconnection

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Check network connectivity
netconnection(fit)

## End(Not run)
```

---

netconnection.crossnma.model

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. 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

Usage

```
## S3 method for class 'crossnma.model'
netconnection(data, ...)
```

Arguments

data	An object produced by crossnma.model.
...	... Additional arguments (passed on to netconnection).

Value

An object of class netconnection with corresponding print function. The object is a list containing 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)

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

See Also

netconnection

Examples

```
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")
```

```
# Check network connectivity
netconnection(mod)
```

`netgraph.crossnma` *Produce a network plot*

Description

Create a network plot of the cross network meta-analysis or meta-regression

Usage

```
## S3 method for class 'crossnma'
netgraph(
  x,
  labels,
  adj = NULL,
  offset = if (!is.null(adj) && all(unique(adj) == 0.5)) 0 else 0.0175,
  points = !missing(cex.points),
  cex.points = 1,
  ...
)
```

Arguments

`x`	An object produced by `crossnma`.
`labels`	An optional vector with 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.
`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).
`points`	A logical indicating whether points should be printed at nodes (i.e. treatments) of the network graph.
`cex.points`	Corresponding size for points. Can be a vector with length equal to the number of treatments.
`...`	... Additional arguments (passed on to `netgraph.netmeta`).

Value

A data frame containing the following columns:

`labels`	Treatment labels.
`seq`	Sequence of treatment labels.
`xpos`	Position of treatment / edge on x-axis.
`ypos`	Position of treatment / edge on y-axis.
`zpos`	Position of treatment / edge on z-axis (for 3-D plots).
`xpos.labels`	Position of treatment labels on x-axis (for 2-D plots).
`ypos.labels`	Position of treatment labels on y-axis (for 2-D plots).
`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).

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>

See Also

netgraph.netmeta

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Create network plot
netgraph(fit, plastic = FALSE, cex.points = 7, adj = 0.5)

## End(Not run)
```

`netgraph.crossnma.model` *Produce a network plot*

Description

Create a network plot of the cross network meta-analysis or meta-regression

Usage

```
## S3 method for class 'crossnma.model'
netgraph(
  x,
  labels,
  adj = NULL,
  offset = if (!is.null(adj) && all(unique(adj) == 0.5)) 0 else 0.0175,
  points = !missing(cex.points),
  cex.points = 1,
  ...
)
```

Arguments

`x`	An object produced by `crossnma.model`.
`labels`	An optional vector with 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.
`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).
`points`	A logical indicating whether points should be printed at nodes (i.e. treatments) of the network graph.
`cex.points`	Corresponding size for points. Can be a vector with length equal to the number of treatments.
`...`	. . . Additional arguments (passed on to `netgraph.netmeta`).

Value

A data frame containing the following columns:

`labels`	Treatment labels.
`seq`	Sequence of treatment labels.
`xpos`	Position of treatment / edge on x-axis.
`ypos`	Position of treatment / edge on y-axis.

zpos	Position of treatment / edge on z-axis (for 3-D plots).
xpos.labels	Position of treatment labels on x-axis (for 2-D plots).
ypos.labels	Position of treatment labels on y-axis (for 2-D plots).
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).

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

netgraph.netmeta

Examples

```
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Create network plot
netgraph(mod, plastic = FALSE, cex.points = 7, adj = 0.5)
```

---

plot.crossnma *Trace plot of MCMC output*

Description

Produces a separate plot for each parameter in the JAGS model. Each plot shows iterations vs sampled values.

Usage

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

Arguments

x	An object generated by crossnma.
...	Additional arguments (passed on to traceplot)

Value

No return value (plot function).

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>

See Also

crossnma, traceplot

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Trace plot of model parameters
plot(fit)

## End(Not run)
```

---

print.crossnma *Print results of cross-design & -format network meta-analysis or regression*

Description

Print call used to create JAGS model for cross-design & -format network meta-analysis or regression

Usage

```
## S3 method for class 'crossnma'
print(x, backtransf = x$model$backtransf, digits = gs("digits"), ...)
```

Arguments

x	An object of class crossnma.
backtransf	A logical indicating whether results should be back transformed. If backtransf = TRUE, results for sm = "OR" are presented as odds ratios rather than log odds ratios, for example.
digits	The number of significant digits printed.
...	Additional arguments.

Value

No return value (print function).

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

crossnma

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Fit JAGS model
# (suppress warning 'Adaptation incomplete' due to n.adapt = 20)
fit <-
  suppressWarnings(crossnma(mod))
fit

## End(Not run)
```

`print.crossnma.model` *Print call used to create JAGS model for cross-design & -format network meta-analysis or regression*

Description

Print call used to create JAGS model for cross-design & -format network meta-analysis or regression

Usage

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

Arguments

`x`	An object of class `crossnma.model`.
`...`	Additional arguments (ignored).

Value

No return value (print function).

Author(s)

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

See Also

`crossnma.model`

Examples

```
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")
mod
```

print.summary.crossnma *Print summary of cross-design & -format network meta-analysis or regression*

Description

Print results of cross-design and cross-format network meta-analysis or meta-regression. In addition, the call used to create the JAGS model is printed.

Usage

```
## S3 method for class 'summary.crossnma'
print(x, digits = gs("digits"), ...)
```

Arguments

x	An object of class crossnma.
digits	The number of significant digits printed. The default value is 3.
...	Additional arguments.

Value

No return value (print function).

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

summary.crossnma

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")
```

```
# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Display the output (with 5 digits)
print(summary(fit), digits = 5)

## End(Not run)
```

`print.summary.crossnma.model` *Print code of JAGS model for cross-design & -format network meta-analysis or regression*

Description

Print code of JAGS model for cross-design & -format network meta-analysis or regression

Usage

```
## S3 method for class 'summary.crossnma.model'
print(x, ...)
```

Arguments

`x`	An object of class `summary.crossnma.model`.
`...`	Additional arguments (ignored).

Value

No return value (print function).

Author(s)

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

See Also

`summary.crossnma.model`

Examples

```
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data
```

```
# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

summary(mod)
```

---

stddata *Simulated aggregate dataset.*

Description

The dataset includes two randomized-controlled trials (RCTs), comparing treatments A and C. The outcome is binary represented as the number of participants with at least one relapse.

Usage

```
stddata
```

Format

A data frame with 4 rows and 11 variables:

id numeric, study identifier

n numeric, the sample size

relapse numeric, the number of relapses

treat character, indicating the assigned treatment to participants in each study arm

design character, design of the study, either 'rct' or 'nrs'

age numeric, the mean age of participants in each study

sex numeric, the proportion of females on each study

rob character, the risk of bias of the study, 'low', 'high','unclear'

unfavored numeric, the indicator of the unfavored treatment in each study, values are 0 or 1

bias.group numeric, the bias effect of the study, 1 = study has inactive treatment and adjust for bias effect, 2= study has active treatments and another adjustment for bias effect, 0=no bias adjustment

year numeric, the year published of the study

summary.crossnma *Summary function for crossnma object*

Description

This function creates posterior summary statistics for the fitted cross network meta-analysis / meta-regression model

Usage

```
## S3 method for class 'crossnma'
summary(
  object,
  quantiles = object$model$quantiles,
  backtransf = object$model$backtransf,
  exp = backtransf,
  ...
)
```

Arguments

object	An object generated by the crossnma.
quantiles	A numeric vector of probabilities to present posterior summaries. The default value is c(0.025, 0.5, 0.975) for the 95% credible interval and the median.
backtransf	A logical value indicating whether to exponentiate the parameters of relative treatment effect and covariate effect.
exp	Deprecated argument (replaced by backtransf).
...	Additional arguments to be passed to summary() function

Value

crossnma.summary returns a matrix containing the following summary statistics (in columns) for each estimated parameter:

Mean the mean of the posterior distribution

SD the standard deviation of the posterior distribution

2.5% (default) the 2.5% quantile of the posterior distribution (the lower bound of the 95% credible interval)

50% (default) the median of the posterior distribution

97.5% (default) the 97.5% quantile of the posterior distribution (the upper bound of the 95% credible interval)

Rhat Gelman-Rubin statistic. The further the value of Rhat from 1, the worse the mixing of chains and so the convergence.

n.eff An estimate of the effective sample size. The smaller the value of n.eff the greater the uncertainty associated with the corresponding parameter.

Author(s)

Tasnim Hamza <hamza.a.tasnim@gmail.com>, Guido Schwarzer <guido.schwarzer@uniklinik-freiburg.de>

See Also

print.summary.crossnma

Examples

```
## Not run:
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

# Fit JAGS model
set.seed(1909)
fit <- crossnma(mod)

# Display the output
summary(fit)

## End(Not run)
```

---

summary.crossnma.model

Summary function for crossnma.model object

Description

Summary function for crossnma.model object

Usage

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

Arguments

object	An object generated by the crossnma.model.
...	Additional arguments (ignored)

Author(s)

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

See Also

print.summary.crossnma.model

Examples

```
# We conduct a network meta-analysis assuming a random-effects
# model.
# The data comes from randomized-controlled trials and
# non-randomized studies (combined naively)
head(ipddata) # participant-level data
stddata # study-level data

# Create a JAGS model
mod <- crossnma.model(treat, id, relapse, n, design,
  prt.data = ipddata, std.data = stddata,
  reference = "A", trt.effect = "random", method.bias = "naive")

summary(mod)
```

Index