Package 'MBNMAtime'

March 4, 2020

Type Package

Title Run Time-Course MBNMA Models

Version 0.1.3

Language en-GB

Date 2020-03-04

Maintainer Hugo Pedder <hugopedder@gmail.com>

URL https://doi.org/10.1002/jrsm.1351

Description Fits Bayesian time-course models for model-based network meta-analysis (MBNMA) and model-based meta-analysis (MBMA) that account for repeated measures over time within studies by applying different time-course functions, following the method of Pedder et al. (2019) <doi:10.1002/jrsm.1351>. The method allows synthesis of studies with multiple follow-up measurements that can account for time-course for a single or multiple treatment comparisons. Several general time-course functions are provided; others may be added by the user. Various characteristics can be flexibly added to the models, such as correlation between time points and shared class effects. The consistency of direct and indirect evidence in the network can be assessed using unrelated mean effects models and/or by node-splitting.

License GPL-3

Depends R (>= 3.5.0)

Imports grDevices, stats, graphics, utils, dplyr (>= 0.7.4), R2jags (>= 0.5-7), rjags (>= 4-8), reshape2 (>= 1.4.3), magrittr (>= 1.5), checkmate (>= 1.8.5), Rdpack (>= 0.10-1)

Suggests knitr, scales (>= 1.0.0), overlapping (>= 1.5.0), ggplot2 (>= 2.2.1), igraph (>= 1.1.2), rmarkdown, testthat (>= 1.0.2), RColorBrewer (>= 1.1-2), mcmcplots (>= 0.4.3)

SystemRequirements JAGS (>= 4.3.0) (http://mcmc-jags.sourceforge.net)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

VignetteBuilder knitr

RdMacros Rdpack

NeedsCompilation no

Author Hugo Pedder [aut, cre],
Nicky Welton [ctb, rev],
Sofia Dias [ctb, rev],
Meg Bennetts [ctb, rev],
Martin Boucher [ctb, rev]

Repository CRAN

Date/Publication 2020-03-04 15:50:06 UTC

R topics documented:

`add_index` *Add follow-up time and arm indices to a dataset*

Description

Adds follow-up time (`fups`, `fupcount`) and arm (`arms`, `narms`) indices to a dataset.

Usage

```
add_index(data.ab, reference = 1)
```

Arguments

data.ab	A data frame of arm-level data in "long" format containing the columns: • studyID Study identifiers • time Numeric data indicating follow-up times • y Numeric data indicating the mean response for a given observation • se Numeric data indicating the standard error for a given observation • treatment Treatment identifiers (can be numeric, factor or character) • class An optional column indicating a particular class code. Treatments with the same identifier must also have the same class code.
reference	A number or character (depending on the format of treatment within data.ab) indicating the reference treatment in the network (i.e. those for which estimated relative treatment effects estimated by the model will be compared to).

Value

A data frame similar to data.ab but with additional columns:

- arm Arm identifiers coded for each study
- fupcount Follow-up identifiers coded for each study
- fups The total number of follow-up measurements in each study
- narm The total number of arms in each study

If treatment or class are non-numeric or non-sequential (i.e. with missing numeric codes), treatments/classes in the returned data frame will be numbered and recoded to enforce sequential numbering (a warning will be shown stating this).

Examples

```
# Add indices to osteoarthritis pain dataset
data.ab <- add_index(osteopain)

# Add indices to dataset using different network reference treatment
data.ab <- add_index(osteopain, reference=3)
```

`alog_pcfb` *Studies of alogliptin for lowering blood glucose concentration in patients with type II diabetes*

Description

A dataset from a systematic review of Randomised-Controlled Trials (RCTs) comparing different doses of alogliptin with placebo (Langford et al. 2016). The systematic review was simply performed and was intended to provide data to illustrate a statistical methodology rather than for clinical inference. Alogliptin is a treatment aimed at reducing blood glucose concentration in type II diabetes. The outcome is continuous, and aggregate data responses correspond to the mean change in HbA1c from baseline to follow-up. The dataset includes 14 Randomised-Controlled Trials (RCTs), comparing 5 different doses of alogliptin with placebo, leading to 6 different treatments (combination of dose and agent) within the network.

Usage

```
alog_pcfb
```

Format

A data frame in long format (one row per arm and study), with 46 rows and 6 variables:

- `studyID` Study identifiers
- `agent` Character data indicating the agent to which participants were randomised
- `dose` Numeric data indicating the standardised dose received
- `treatment` Character data indicating the treatment (combination of agent and dose) to which participants were randomised
- `y` Numeric data indicating the mean change from baseline in blood glucose concentration (mg/dL) in a study arm
- `se` Numeric data indicating the standard error for the mean change from baseline in blood glucose concentration (mg/dL) in a study arm
- `time` Numeric data indicating the time at which the observation was measured (given in weeks)

Details

`alog_pcfb` is a data frame in long format (one row per observation, arm and study), with the variables `studyID`, `agent`, `dose`, `treatment`, `y`, `se`, `N` and `time`.

References

Langford O, Aronson JK, van Valkenhoef G, Stevens RJ (2016). "Methods for meta-analysis of pharmacodynamic dose-response data with application to multi-arm studies of alogliptin." *Stat Methods Med Res*. ISSN 1477-0334 (Electronic) 0962-2802 (Linking), doi: 10.1177/0962280216637093, https://www.ncbi.nlm.nih.gov/pubmed/26994216.

compound.beta *Prepares beta time-course parameters for mb.write()*

Description

Checks that beta time-course parameters have been specified correctly and converts them to the correct format for `mb.write()` and other subsequent functions.

Usage

```
compound.beta(beta.1)
```

Arguments

`beta.1`	A two-element list whose elements have the following names: • `pool` Can take either `"rel"`, `"arm"` or `"const"` • `method` Can take either `"common"`, `"random"`, or be assigned a numeric value

devplot *Plot deviance contributions from an MBNMA model*

Description

Plot deviance contributions from an MBNMA model

Usage

```
devplot(mbnma, dev.type = "resdev", plot.type = "scatter",
  xaxis = "time", facet = TRUE, n.iter = mbnma$BUGSoutput$n.iter,
  n.thin = mbnma$BUGSoutput$n.thin, ...)
```

Arguments

`mbnma`	An S3 object of class `"mbnma"` generated by running a time-course MBNMA model
`dev.type`	Deviances to plot - can be either residual deviances (`"resdev"`, the default) or deviances (`"dev"`)
`plot.type`	Deviances can be plotted either as scatter points (`"scatter"` - using `geom_point()`, the default) or as boxplots (`"box"`)
`xaxis`	A character object that indicates whether deviance contributions should be plotted by time (`"time"`) or by follow-up count (`"fup"`)
`facet`	A boolean object that indicates whether or not to facet by treatment

n.iter	The number of iterations to update the model whilst monitoring additional parameters (if necessary). Must be a positive integer. Default is the value used in mbnma.
n.thin	The thinning rate. Must be a positive integer. Default is the value used in mbnma.
...	Arguments to be sent to ggplot2::geom_point() or ggplot2::geom_boxplot

Details

Deviances should only be plotted for models that have converged successfully. If deviance contributions have not been monitored in mbnma$parameters.to.save then additional iterations will have to be run to get results for these.

Deviance contributions cannot be calculated for models with a multivariate likelihood (i.e. those that account for correlation between observations) because the covariance matrix in these models is treated as unknown (if rho="estimate") and deviance contributions will be correlated.

Value

Generates a plot of deviance contributions and returns a list containing the plot (as an object of class c("gg","ggplot")), and a data.frame of posterior mean deviance/residual deviance contributions for each observation.

Examples

```
# Make network
network <- mb.network(alog_pcfb)

# Run MBNMA
mbnma <- mb.quadratic(network)

# Plot residual deviance contributions in a scatterplot
devplot(mbnma)

# Plot deviance contributions in boxplots at each follow-up measurement
# Monitor for 500 additional iterations
devplot(mbnma, dev.type="dev", plot.type="box", xaxis="fup", n.iter=500)
```

---

fitplot *Plot fitted values from MBNMA model*

Description

Plot fitted values from MBNMA model

Usage

```
fitplot(mbnma, treat.labs = NULL, disp.obs = TRUE,
  n.iter = mbnma$BUGSoutput$n.iter, n.thin = mbnma$BUGSoutput$n.thin,
  ...)
```

Arguments

mbnma	An S3 object of class "mbnma" generated by running a time-course MBNMA model
treat.labs	A character vector of treatment labels with which to name graph panels. Can use mb.network()[["treatments"]] with original dataset if in doubt.
disp.obs	A boolean object to indicate whether raw data responses should be plotted as points on the graph
n.iter	number of total iterations per chain (including burn in; default: 2000)
n.thin	thinning rate. Must be a positive integer. Set n.thin > 1 to save memory and computation time if n.iter is large. Default is max(1,floor(n.chains * (n.iter-n.burnin) / 1000)) which will only thin if there are at least 2000 simulations.
...	Arguments to be sent to ggplot2::geom_point() or ggplot2::geom_line()

Details

Fitted values should only be plotted for models that have converged successfully. If fitted values (theta) have not been monitored in mbnma$parameters.to.save then additional iterations will have to be run to get results for these.

Value

Generates a plot of fitted values from the MBNMA model and returns a list containing the plot (as an object of class c("gg","ggplot")), and a data.frame of posterior mean fitted values for each observation.

Examples

```
# Make network
network <- mb.network(osteopain)

# Run MBNMA
mbnma <- mb.emax(network,
  emax=list(pool="rel", method="common"),
  et50=list(pool="const", method="common"))

# Plot fitted values from the model with treatment labels
# Monitor fitted values for 500 additional iterations
fitplot(mbnma, treat.labs=network$treatments, n.iter=500)
```

gen.parameters.to.save

Automatically generate parameters to save for a dose-response MBNMA model

Description

Identical to `gen.parameters.to.save()` in MBNMAdose

Usage

```
gen.parameters.to.save(model.params, model)
```

Arguments

`model.params`	A character or numeric vector containing the names of the dose-response parameters in the model
`model`	A JAGS model written as a character object

genmaxcols

Get large vector of distinct colours using Rcolorbrewer

Description

Get large vector of distinct colours using Rcolorbrewer

Usage

```
genmaxcols()
```

get.earliest.time

Create a dataset with the earliest time point only

Description

Takes the earliest time point from each arm in each study within an `mb.network` object. Useful for network plots.

Usage

```
get.earliest.time(network)
```

Arguments

network	An object of class mb.network.

Value

A data frame in long format of responses at the earliest time point in each arm of each study.

---

get.latest.time *Create a dataset with the latest time point only*

Description

Takes the latest time point from each arm in each study within an mb.network object. Useful for network plots.

Usage

```
get.latest.time(network)
```

Arguments

network	An object of class mb.network.

Value

A data frame in long format of responses at the latest time point in each arm of each study.

Examples

```
# Using the alogliptin dataset
network <- mb.network(alog_pcfb)

# Generate a data frame with only the latest time point included in each study
get.latest.time(network)
```

`get.model.vals` *Get MBNMA model values*

Description

Extracts specific information required for prediction from a time-course MBNMA model

Usage

```
get.model.vals(mbnma, timecourse, beta.incl, E0 = 0)
```

Arguments

`mbnma`	An S3 object of class `"mbnma"` generated by running a time-course MBNMA model
`timecourse`	A character object that specifies the time-course used in `mbnma` (in terms of alpha, beta, and time), as generated by `init.predict()`
`beta.incl`	A numeric vector that indicates the time-course parameters that were included in `mbnma`, as generated by `init.predict()`
`E0`	An object to indicate the value(s) to use for the response at time = 0 in the prediction. This can take a number of different formats depending on how it will be used/calculated. The default is `0` but this may lead to non-sensical predictions. • `numeric()` A single numeric value representing the deterministic response at time = 0, given. • `character()` A single string representing a stochastic distribution for the response at time = 0. This is specified as a random number generator (RNG) given as a string, and can take any RNG distribution for which a function exists in R. For example: `"rnorm(n,7,0.5)"`.

Value

A list containing named elements that correspond to different time-course parameters in `mbnma`. These elements contain MCMC results either taken directly from `mbnma` or (in the case of random time-course parameters specified as `method="random"`) randomly generated using parameter values estimated in `mbnma`.

Additional elements contain the following values:

- `timecourse` A character object that specifies the time-course used in `mbnma` in terms of alpha, beta, mu, d and time. Consistency relative time-course parameters are specified in terms of mu and d.
- `mu.prior` A character vector that indicates for which time-course parameters a network reference treatment effect will be required for prediction.
- `time.params` A character vector that indicates the different time-course parameters that are required for the prediction

@noRd

`get.prior` *Get current priors from JAGS model code*

Description

Identical to `get.prior()` in MBNMAdose. This function takes JAGS model presented as a string and identifies what prior values have been used for calculation.

Usage

```
get.prior(model)
```

Arguments

`model`	A character object of JAGS MBNMA model code

Details

Even if an MBNMA model that has not initialised successfully and results have not been calculated, the JAGS model for it is saved in `MBNMA$model.arg$jagscode` and therefore priors can still be obtained. This allows for priors to be changed even in failing models, which may help solve issues with initialisation.

Value

A character vector, each element of which is a line of JAGS code corresponding to a prior in the JAGS code.

Examples

```
# Create mb.network object using an MBNMAtime dataset
network <- mb.network(osteopain)

# Create mb.network object using an MBNMAdose dataset

# Run linear MBNMA
result <- mb.linear(network,
  slope=list(pool="rel", method="random"))

# Obtain model prior values
get.prior(result$model.arg$jagscode)

# ...also equivalent to
print(result$model.arg$priors)
```

getjagsdata *Prepares data for JAGS*

Description

Converts MBNMA data frame to a list for use in JAGS model

Usage

```
getjagsdata(data.ab, class = FALSE, rho = NULL, covstruct = "CS")
```

Arguments

data.ab	A data frame of arm-level data in "long" format containing the columns: • studyID Study identifiers • time Numeric data indicating follow-up times • y Numeric data indicating the aggregate response for a given observation (e.g. mean) • se Numeric data indicating the standard error for a given observation • treatment Treatment identifiers (can be numeric, factor or character) • class An optional column indicating a particular class identifier. Observations with the same treatment identifier must also have the same class identifier. • N An optional column indicating the number of participants used to calculate the response at a given observation
class	A boolean object indicating whether or not data.ab contains information on different classes of treatments
rho	The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.
covstruct	A character to indicate the covariance structure required for modelling correlation between time points (if any), since this determines some of the data. Can be either "CS" (compound symmetry) or "AR1" (autoregressive AR1).

Value

A named list of numbers, vector, matrices and arrays to be sent to JAGS. List elements are:

- y An array of mean responses for each observation in each arm within each study
- se An array of standard errors for each observation in each arm within each study
- time A matrix of follow-up times within each study

- fups A numeric vector with the number of follow-up measurements per study
- narm A numeric vector with the number of arms per study
- NS The total number of studies in the dataset
- NT The total number of treatments in the dataset
- treat A matrix of treatment codes within each study
- Nclass Optional. The total number of classes in the dataset
- class Optional. A matrix of class codes within each study
- classkey Optional. A vector of class codes that correspond to treatment codes. Same length as the number of treatment codes.
- mat.triangle Optional. A matrix with number indicating how to fill covariance matrices within the JAGS code.
- mat.order Optional. A matrix with number indicating what order to fill covariance matrices within the JAGS code.
- timedif.0 Optional. A vector of the difference in times between the first and second follow-up time in each study.

Examples

```
# Using the alogliptin dataset
network <- mb.network(alog_pcfb)
jagsdat <- getjagsdata(network$data.ab)


# Get JAGS data with class
netclass <- mb.network(goutSUA_CFBcomb)
jagsdat <- getjagsdata(netclass$data.ab, class=TRUE)


# Get JAGS data that allows for modelling correlation between time points
painnet <- mb.network(osteopain)
jagsdat <- getjagsdata(painnet$data.ab, rho="estimate", covstruct="AR1")
```

---

goutSUA_CFB *Studies of treatments for reducing serum uric acid in patients with gout*

Description

A dataset from a systematic review of interventions for lowering Serum Uric Acid (SUA) concentration in patients with gout **(not published previously)**. The outcome is continuous, and aggregate data responses correspond to the mean change from baseline in SUA in mg/dL. Overall there are 41 treatments of 8 agents in the network. Standard deviations have been imputed for 181 observations.

Usage

```
goutSUA_CFB
```

Format

A data frame with 224 rows and 7 variables:

- `studyID` Study identifiers
- `time` Numeric data indicating follow-up times
- `y` Numeric data indicating the mean response for a given observation
- `se` Numeric data indicating the standard error for a given observation
- `treatment` Treatment identifiers as factors. Labels are shortened treatment names.
- `treatname` Character data giving the full names of each treatment in the format agent_dose
- `class` Shortened agent names stored as factors.

Details

`goutSUA_CFB` is a data frame in long format (one row per observation, arm and study), with the variables `studyID`, `time`, `y`, `se`, `treatment`, `treatname` and `class`.

Source

Pfizer Ltd.

`goutSUA_CFBcomb` *Studies of combined treatments for reducing serum uric acid in patients with gout*

Description

A dataset from a systematic review of interventions for lowering Serum Uric Acid (SUA) concentration in patients with gout **(not published previously)**. The outcome is continuous, and aggregate data responses correspond to the mean change from baseline in SUA in mg/dL. Treatments with similar doses have been pooled together to improve network connectivity and facilitate evidence synthesis, resulting in 19 treatments of 7 agents included in the network. Standard deviations have been imputed for 181 observations.

Usage

```
goutSUA_CFBcomb
```

Format

A data frame with 224 rows and 7 variables:

- `studyID` Study identifiers
- `time` Numeric data indicating follow-up times
- `y` Numeric data indicating the mean response for a given observation
- `se` Numeric data indicating the standard error for a given observation
- `treatment` Treatment identifiers as factors. Labels are shortened treatment names.
- `treatname` Character data giving the full names of each treatment in the format agent_dose
- `class` Shortened agent names stored as factors.

Details

goutSUA_CFBcomb is a data frame in long format (one row per observation, arm and study), with the variables studyID, time, y, se, treatment, treatname and class.

Source

Pfizer Ltd.

---

inconsistency.loops *Identify comparisons in loops that fulfill criteria for node-splitting*

Description

Identify comparisons informed by both direct and indirect evidence from independent sources, which therefore fulfill the criteria for testing for inconsistency via node-splitting. Follows the method of van Valkenhoef van Valkenhoef et al. (2016).

Usage

```
inconsistency.loops(data)
```

Arguments

data	A data frame containing variables studyID and treatment (as numeric codes) that indicate which treatments are used in which studies.

Details

Similar to mtc.nodesplit but uses a fixed reference treatment and therefore suggests fewer loops in which to test for inconsistency. Heterogeneity can also be parameterised as inconsistency and so testing for inconsistency in additional loops whilst changing the reference treatment would also be identifying heterogeneity. Depends on igraph.

Value

A data frame of comparisons that are informed by direct and indirect evidence from independent sources. Each row of the data frame is a different treatment comparison. Numerical codes in t1 and t2 correspond to treatment codes.

References

van Valkenhoef G, Dias S, Ades AE, Welton NJ (2016). "Automated generation of node-splitting models for assessment of inconsistency in network meta-analysis." *Res Synth Methods*, **7**(1), 80-93. ISSN 1759-2887 (Electronic) 1759-2879 (Linking), doi: 10.1002/jrsm.1167, https://www.ncbi.nlm.nih.gov/pubmed/26461181.

Examples

```
data <- data.frame(studyID=c(1,1,2,2,3,3,4,4,5,5,5),
  treatment=c(1,2,1,3,2,3,3,4,1,2,4)
  )

# Identify comparisons informed by direct and indirect evidence
inconsistency.loops(data)
```

`mb.comparisons` *Identify unique comparisons within a network (identical to MBNMA-dose)*

Description

Identify unique contrasts within a network that make up all the head-to-head comparisons. Repetitions of the same treatment comparison are grouped together.

Usage

```
mb.comparisons(data)
```

Arguments

`data`	A data frame containing variables `studyID` and `treatment` (as numeric codes) that indicate which treatments are used in which studies.

Value

A data frame of unique comparisons in which each row represents a different comparison. `t1` and `t2` indicate the treatment codes that make up the comparison. `nr` indicates the number of times the given comparison is made within the network.

If there is only a single observation for each study within the dataset (i.e. as for standard network meta-analysis) `nr` will represent the number of studies that compare treatments `t1` and `t2`.

If there are multiple observations for each study within the dataset (as in time-course MBNMA) `nr` will represent the number of time points in the dataset in which treatments `t1` and `t2` are compared.

Examples

```
data <- data.frame(studyID=c(1,1,2,2,3,3,4,4,5,5,5),
  treatment=c(1,2,1,3,2,3,3,4,1,2,4)
  )

# Identify comparisons informed by direct and indirect evidence
mb.comparisons(data)
```

`mb.emax` *Run MBNMA model with an Emax time-course function (without a Hill parameter)*

Description

Fits a Bayesian model-based network meta-analysis (MBNMA) with a defined time-course function. This function accounts for repeated measures over time within studies by applying an Emax time-course function. Follows the methods of Pedder et al. (2019). This function acts as a wrapper for `mb.run()` that allows for more clearly defined parameter names.

Usage

```
mb.emax(network, emax = list(pool = "rel", method = "common"),
  et50 = list(pool = "rel", method = "common"), alpha = "study",
  positive.scale = FALSE, intercept = TRUE, rho = NULL,
  covar = NULL, var.scale = NULL, class.effect = list(),
  UME = FALSE, pd = "pv", parallel = TRUE, priors = NULL, ...)
```

Arguments

`network`	An object of class `mb.network`.
`emax`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`et50`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`alpha`	Refers to the baseline mean response and is a character object that can take either: • `"study"` to constrain baseline to be equal for all arms within a study (`i` index is added)) • `"arm"` to allow baseline to vary between arms within a study (`i,k` index is added)).
`positive.scale`	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.
`intercept`	A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (`FALSE`) or not (setting it to `FALSE` removes the intercept, `alpha`, from the model).
`rho`	The correlation coefficient when modelling correlation between time points. If left as `NULL` (the default) then this implies modelling no correlation between time points. Can either be assigned the string `"estimate"` to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes `rho` in the model to the assigned value, either for when `rho` is calculated externally or for use in deterministic sensitivity analyses.

`covar` A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either `"CS"` (compound symmetry) or `"AR1"` (autoregressive AR1).

`var.scale` A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.

`class.effect` A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be (`"common"` or `"random"`). For example: `list("beta.2"="common","beta.3"="random")`.

`UME` Can take either `TRUE` or `FALSE` (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: `c("beta.1","beta.2")`.

`pd` Can take either:

- `pv` only pV will be reported (as automatically outputted by R2jags).
- `plugin` calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models.
- `pd.kl` calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result.
- `popt` calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)

`parallel` A boolean value that indicates whether JAGS should be run in parallel (`TRUE`) or not (`FALSE`). If `TRUE` then the number of cores to use is automatically calculated.

`priors` A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.

`...` Arguments to be sent to `mb.run()`

Value

An object of S3 class `c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. `d.et50` or `d.1`):

- `d` The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if `pool="rel"`
- `sd.d` The between-study SD (heterogeneity) for relative effects, reported if `pool="rel"` and `method="random"`
- `D` The class effect for a given class compared to the network reference class for a particular time-course parameter

- sd.D The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- beta If pool="const" then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If pool="arm" then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment
- sd.beta Reported if method="random" and pool is either "const" or "arm". If pool="const" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If pool="arm" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- rho The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- totresdev The residual deviance of the model
- deviance The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and model.arg, a list of arguments provided to mb.run() which includes jagscode, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements pool and method.

pool is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- "rel" indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).
- "arm" indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- "const" indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

method is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- "common" implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- "random" implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.

- `numeric()` Assigned a numeric value - this can only be used if `pool="const"`. It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Correlation between observations

When modelling correlation between observations using `rho`, values for `rho` must imply a positive semidefinite covariance matrix. If estimating `rho` from the data (by assigning it `"estimate"`), the default prior distribution (`dunif(-1,1)`) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to `dunif(0,1)`) using the `priors` argument (see `get.prior`)

References

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

result <- mb.emax(network,
  emax=list(pool="rel", method="random"),
  et50=list(pool="const", method="common"))


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
mcmcplots::denplot(result, parms=c("deviance", "sd.emax", "d.emax[2]", "d.emax[3]"))
```

```
# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "d.emax")


########## Output ############

# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)


###### Additional model arguments ######

# Fit model with correlation between time points
mb.emax(network, alpha="study",
  parameters.to.save=c("d.emax", "d.et50", "rho"),
  emax=list(pool="rel", method="random"),
  et50=list(pool="rel", method="common"),
  rho="estimate", covar="CS"
  )
```

`mb.emax.hill` *Run MBNMA model with an Emax time-course function and a Hill parameter*

Description

Fits a Bayesian model-based network meta-analysis (MBNMA) with a defined time-course function. This function accounts for repeated measures over time within studies by applying an Emax time-course function with a Hill parameter. Follows the methods of Pedder et al. (2019). This function acts as a wrapper for `mb.run()` that allows for more clearly defined parameter names.

Usage

```
mb.emax.hill(network, fun = "emax.hill", emax = list(pool = "rel",
  method = "common"), et50 = list(pool = "rel", method = "common"),
  hill = list(pool = "const", method = "common"), alpha = "study",
  positive.scale = FALSE, intercept = TRUE, rho = NULL,
  covar = NULL, var.scale = NULL, class.effect = list(),
  UME = FALSE, pd = "pv", parallel = TRUE, priors = NULL, ...)
```

Arguments

network	An object of class mb.network.
fun	is a character specifying a functional form to be assigned to the time-course. Options are given in details.
emax	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
et50	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
hill	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
alpha	Refers to the baseline mean response and is a character object that can take either: • "study" to constrain baseline to be equal for all arms within a study (i index is added)) • "arm" to allow baseline to vary between arms within a study (i,k index is added)).
positive.scale	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.
intercept	A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (FALSE) or not (setting it to FALSE removes the intercept, alpha, from the model).
rho	The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.
covar	A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either "CS" (compound symmetry) or "AR1" (autoregressive AR1).
var.scale	A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.
class.effect	A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").
UME	Can take either TRUE or FALSE (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: c("beta.1","beta.2").
pd	Can take either:

- pv only pV will be reported (as automatically outputted by R2jags).
- plugin calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models.
- pd.kl calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result.
- popt calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)

`parallel` A boolean value that indicates whether JAGS should be run in parallel (`TRUE`) or not (`FALSE`). If `TRUE` then the number of cores to use is automatically calculated.

`priors` A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.

`...` Arguments to be sent to `mb.run()`

Value

An object of S3 class `c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. `d.et50` or `d.1`):

- d The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if `pool="rel"`
- sd.d The between-study SD (heterogeneity) for relative effects, reported if `pool="rel"` and `method="random"`
- D The class effect for a given class compared to the network reference class for a particular time-course parameter
- sd.D The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- beta If `pool="const"` then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If `pool="arm"` then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment
- sd.beta Reported if `method="random"` and pool is either `"const"` or `"arm"`. If `pool="const"` this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If `pool="arm"` this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- rho The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- totresdev The residual deviance of the model
- deviance The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and `model.arg`, a list of arguments provided to `mb.run()` which includes `jagscode`, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements `pool` and `method`.

`pool` is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- `"rel"` indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).
- `"arm"` indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- `"const"` indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

`method` is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- `"common"` implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- `"random"` implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.
- `numeric()` Assigned a numeric value - this can only be used if `pool="const"`. It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Correlation between observations

When modelling correlation between observations using `rho`, values for `rho` must imply a positive semidefinite covariance matrix. If estimating `rho` from the data (by assigning it `"estimate"`), the default prior distribution (`dunif(-1,1)`) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to `dunif(0,1)`) using the `priors` argument (see `get.prior`)

References

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Fit model with random consistency effects on emax, fixed consistency effects on et50, and a
#Hill parameter of 0.2 across the network
result <- mb.emax.hill(network,
  emax=list(pool="rel", method="random"),
  et50=list(pool="rel", method="common"),
  hill=list(pool="const", method=0.2))


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
mcmcplots::denplot(result, c("d.et50[2]", "deviance", "sd.emax", "totresdev"))

# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "d.emax")


########## Output ###########

# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)
```

```
###### Additional model arguments ######

# Create mb.network object containing class variable
network <- mb.network(goutSUA_CFB)

# Fit model with class effects and correlation between time points and user-defined priors
mb.emax.hill(network, alpha="study",
  parameters.to.save=c("d.emax", "d.et50", "beta.hill"),
  emax=list(pool="rel", method="random"),
  et50=list(pool="rel", method="common"),
  hill=list(pool="const", method="common"),
  rho="estimate", covar="AR1",
  priors=list("rho"="dunif(0,1)"),
  class.effect=list("et50"="random")
  )
```

`mb.exponential` *Run MBNMA model with an exponential time-course function*

Description

Fits a Bayesian model-based network meta-analysis (MBNMA) with a defined time-course function. This function accounts for repeated measures over time within studies by applying an exponential time-course function. Follows the methods of Pedder et al. (2019). This function acts as a wrapper for `mb.run()` that allows for more clearly defined parameter names.

Usage

```
mb.exponential(network, lambda = list(pool = "rel", method = "common"),
  alpha = "study", positive.scale = FALSE, intercept = TRUE,
  rho = NULL, covar = NULL, var.scale = NULL,
  class.effect = list(), UME = FALSE, pd = "pv", parallel = TRUE,
  priors = NULL, ...)
```

Arguments

`network`	An object of class `mb.network`.
`lambda`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`alpha`	Refers to the baseline mean response and is a character object that can take either: • `"study"` to constrain baseline to be equal for all arms within a study (`i` index is added)) • `"arm"` to allow baseline to vary between arms within a study (`i,k` index is added)).

`positive.scale` A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.

`intercept` A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (`FALSE`) or not (setting it to `FALSE` removes the intercept, `alpha`, from the model).

`rho` The correlation coefficient when modelling correlation between time points. If left as `NULL` (the default) then this implies modelling no correlation between time points. Can either be assigned the string `"estimate"` to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes `rho` in the model to the assigned value, either for when `rho` is calculated externally or for use in deterministic sensitivity analyses.

`covar` A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either `"CS"` (compound symmetry) or `"AR1"` (autoregressive AR1).

`var.scale` A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.

`class.effect` A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be (`"common"` or `"random"`). For example: `list("beta.2"="common","beta.3"="random")`.

`UME` Can take either `TRUE` or `FALSE` (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: `c("beta.1","beta.2")`.

`pd` Can take either:

- `pv` only pV will be reported (as automatically outputted by R2jags).
- `plugin` calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models.
- `pd.kl` calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result.
- `popt` calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)

`parallel` A boolean value that indicates whether JAGS should be run in parallel (`TRUE`) or not (`FALSE`). If `TRUE` then the number of cores to use is automatically calculated.

`priors` A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.

`...` Arguments to be sent to `mb.run()`

Value

An object of S3 class `c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. `d.et50` or `d.1`):

- `d` The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if `pool="rel"`
- `sd.d` The between-study SD (heterogeneity) for relative effects, reported if `pool="rel"` and `method="random"`
- `D` The class effect for a given class compared to the network reference class for a particular time-course parameter
- `sd.D` The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- `beta` If `pool="const"` then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If `pool="arm"` then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment
- `sd.beta` Reported if `method="random"` and `pool` is either `"const"` or `"arm"`. If `pool="const"` this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If `pool="arm"` this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- `rho` The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- `totresdev` The residual deviance of the model
- `deviance` The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and `model.arg`, a list of arguments provided to `mb.run()` which includes `jagscode`, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements `pool` and `method`.

`pool` is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- `"rel"` indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).

- `"arm"` indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- `"const"` indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

`method` is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- `"common"` implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- `"random"` implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.
- `numeric()` Assigned a numeric value - this can only be used if `pool="const"`. It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Correlation between observations

When modelling correlation between observations using `rho`, values for `rho` must imply a positive semidefinite covariance matrix. If estimating `rho` from the data (by assigning it `"estimate"`), the default prior distribution (`dunif(-1,1)`) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to `dunif(0,1)`) using the `priors` argument (see `get.prior`)

References

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Fit exponential time-course with random consistency treatment effects
result <- mb.exponential(network,
  lambda=list(pool="rel", method="random"))


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
mcmcplots::denplot(result, parms=c("sd.lambda", "d.lambda[2]"))

# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "d.lambda")


########## Output ############

# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)


###### Additional model arguments ######

# Fit model with unrelated mean effects that saves residual deviance contributions
mb.exponential(network, alpha="study",
  parameters.to.save=c("d.lambda", "resdev"),
  lambda=list(pool="rel", method="random"),
  UME=TRUE
  )
```

`mb.fract.first` *Run MBNMA model with a first-order fractional polynomial time-course function*

Description

Fits a Bayesian model-based network meta-analysis (MBNMA) with a defined time-course function. This function accounts for repeated measures over time within studies by applying a first-order

fractional polynomial time-course function (Jansen et al. 2015). Follows the methods of Pedder et al. (2019). This function acts as a wrapper for `mb.run()` that allows for more clearly defined parameter names.

Usage

```
mb.fract.first(network, slope = list(pool = "rel", method = "common"),
  power = list(pool = "const", method = "common"), alpha = "study",
  positive.scale = FALSE, intercept = TRUE, rho = NULL,
  covar = NULL, var.scale = NULL, class.effect = list(),
  UME = FALSE, pd = "pv", parallel = TRUE, priors = NULL, ...)
```

Arguments

`network`	An object of class `mb.network`.
`slope`	list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`power`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details). For this parameter, `pool` must be set to `"arm"` or `"const"` (i.e. it cannot be `"rel"`).
`alpha`	Refers to the baseline mean response and is a character object that can take either: • `"study"` to constrain baseline to be equal for all arms within a study (`i` index is added)) • `"arm"` to allow baseline to vary between arms within a study (`i,k` index is added)).
`positive.scale`	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.
`intercept`	A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (`FALSE`) or not (setting it to `FALSE` removes the intercept, `alpha`, from the model).
`rho`	The correlation coefficient when modelling correlation between time points. If left as `NULL` (the default) then this implies modelling no correlation between time points. Can either be assigned the string `"estimate"` to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes `rho` in the model to the assigned value, either for when `rho` is calculated externally or for use in deterministic sensitivity analyses.
`covar`	A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either `"CS"` (compound symmetry) or `"AR1"` (autoregressive AR1).
`var.scale`	A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.

class.effect	A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").
UME	Can take either TRUE or FALSE (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: c("beta.1","beta.2").
pd	Can take either: • pv only pV will be reported (as automatically outputted by R2jags). • plugin calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models. • pd.kl calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result. • popt calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)
parallel	A boolean value that indicates whether JAGS should be run in parallel (TRUE) or not (FALSE). If TRUE then the number of cores to use is automatically calculated.
priors	A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.
...	Arguments to be sent to mb.run()

Value

An object of S3 class c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. d.et50 or d.1):

- d The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if pool="rel"
- sd.d The between-study SD (heterogeneity) for relative effects, reported if pool="rel" and method="random"
- D The class effect for a given class compared to the network reference class for a particular time-course parameter
- sd.D The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- beta If pool="const" then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If pool="arm" then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment

- sd.beta Reported if method="random" and pool is either "const" or "arm". If pool="const" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If pool="arm" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- rho The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- totresdev The residual deviance of the model
- deviance The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and model.arg, a list of arguments provided to mb.run() which includes jagscode, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements pool and method.

pool is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- "rel" indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).
- "arm" indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- "const" indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

method is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- "common" implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- "random" implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.
- numeric() Assigned a numeric value - this can only be used if pool="const". It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Correlation between observations

When modelling correlation between observations using `rho`, values for `rho` must imply a positive semidefinite covariance matrix. If estimating `rho` from the data (by assigning it `"estimate"`), the default prior distribution (`dunif(-1,1)`) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to `dunif(0,1)`) using the `priors` argument (see `get.prior`)

References

Jansen JP, Vieira MC, Cope S (2015). "Network meta-analysis of longitudinal data using fractional polynomials." *Stat Med*, **34**(15), 2294-311. ISSN 1097-0258 (Electronic) 0277-6715 (Linking), doi: 10.1002/sim.6492, https://www.ncbi.nlm.nih.gov/pubmed/25877808.

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Fit 1st order fractional polynomial time-course with random consistency treatment effects
#on the slope and a common parameter for power estimated across the network
result <- mb.fract.first(network,
  slope=list(pool="rel", method="random"),
  power=list(pool="const", method="common"))


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
```

```
mcmcplots::denplot(result, parms=c("sd.slope", "deviance", "beta.power"))

# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "d.slope")


########## Output ############

# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)


###### Additional model arguments ######

# Fit model with unrelated mean effects and correlation between time points
mb.fract.first(network, alpha="study",
  slope=list(pool="rel", method="random"),
  power=list(pool="const", method="common"),
  rho=0.5, covar="AR1",
  UME="slope"
  )
```

`mb.fract.second` *Run MBNMA model with a second-order fractional polynomial time-course function*

Description

Fits a Bayesian model-based network meta-analysis (MBNMA) with a defined time-course function. This function accounts for repeated measures over time within studies by applying a second-order fractional polynomial time-course function (Jansen et al. 2015). Follows the methods of Pedder et al. (2019). This function acts as a wrapper for `mb.run()` that allows for more clearly defined parameter names.

Usage

```
mb.fract.second(network, slope.1 = list(pool = "rel", method = "common"),
  slope.2 = list(pool = "rel", method = "common"), power.1 = list(pool
  = "const", method = "common"), power.2 = list(pool = "const", method =
  "common"), alpha = "study", positive.scale = FALSE,
  intercept = TRUE, rho = NULL, covar = NULL, var.scale = NULL,
  class.effect = list(), UME = FALSE, pd = "pv", parallel = TRUE,
  priors = NULL, ...)
```

Arguments

network	An object of class mb.network.
slope.1	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
slope.2	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
power.1	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details). For this parameter, pool must be set to "arm" or "const" (i.e. it cannot be "rel").
power.2	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details). For this parameter, pool must be set to "arm" or "const" (i.e. it cannot be "rel").
alpha	Refers to the baseline mean response and is a character object that can take either: • "study" to constrain baseline to be equal for all arms within a study (i index is added)) • "arm" to allow baseline to vary between arms within a study (i,k index is added)).
positive.scale	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.
intercept	A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (FALSE) or not (setting it to FALSE removes the intercept, alpha, from the model).
rho	The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.
covar	A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either "CS" (compound symmetry) or "AR1" (autoregressive AR1).
var.scale	A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.
class.effect	A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").
UME	Can take either TRUE or FALSE (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: c("beta.1","beta.2").

pd	Can take either: • pv only pV will be reported (as automatically outputted by R2jags). • plugin calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models. • pd.kl calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result. • popt calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)
parallel	A boolean value that indicates whether JAGS should be run in parallel (TRUE) or not (FALSE). If TRUE then the number of cores to use is automatically calculated.
priors	A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.
...	Arguments to be sent to mb.run()

Value

An object of S3 class c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. d.et50 or d.1):

- d The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if pool="rel"
- sd.d The between-study SD (heterogeneity) for relative effects, reported if pool="rel" and method="random"
- D The class effect for a given class compared to the network reference class for a particular time-course parameter
- sd.D The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- beta If pool="const" then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If pool="arm" then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment
- sd.beta Reported if method="random" and pool is either "const" or "arm". If pool="const" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If pool="arm" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- rho The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- totresdev The residual deviance of the model
- deviance The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and `model.arg`, a list of arguments provided to `mb.run()` which includes `jagscode`, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements `pool` and `method`.

`pool` is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- `"rel"` indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).
- `"arm"` indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- `"const"` indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

`method` is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- `"common"` implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- `"random"` implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.
- `numeric()` Assigned a numeric value - this can only be used if `pool="const"`. It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Correlation between observations

When modelling correlation between observations using `rho`, values for `rho` must imply a positive semidefinite covariance matrix. If estimating `rho` from the data (by assigning it `"estimate"`), the default prior distribution (`dunif(-1,1)`) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to `dunif(0,1)`) using the `priors` argument (see `get.prior`)

References

Jansen JP, Vieira MC, Cope S (2015). "Network meta-analysis of longitudinal data using fractional polynomials." *Stat Med*, **34**(15), 2294-311. ISSN 1097-0258 (Electronic) 0277-6715 (Linking), doi: 10.1002/sim.6492, https://www.ncbi.nlm.nih.gov/pubmed/25877808.

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Fit 2nd order fractional polynomial time-course with fixed consistency treatment effects on
#beta.1 and beta.2, absolute time-course parameters estimated by treatment for power.1,
#and an exchangeable parameter for power estimated across the network
result <- mb.fract.second(network,
  slope.1=list(pool="rel", method="common"),
  slope.2=list(pool="rel", method="common"),
  power.1=list(pool="arm", method="common"),
  power.2=list(pool="arm", method="random"))


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
mcmcplots::denplot(result, parms=c("beta.power.1", "beta.power.2"))

# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "beta.power.1")


########## Output ############
```

```
# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)


###### Additional model arguments ######

# Fit model with correlation between time points
mb.fract.second(network, alpha="study",
  slope.1=list(pool="rel", method="common"),
  slope.2=list(pool="rel", method="common"),
  power.1=list(pool="const", method="common"),
  power.2=list(pool="const", method="common"),
  rho=0.5, covar="AR1"
  )
```

`mb.linear` *Run MBNMA model with a linear time-course function*

Description

Fits a Bayesian model-based network meta-analysis (MBNMA) with a defined time-course function. This function accounts for repeated measures over time within studies by applying a linear time-course function. Follows the methods of Pedder et al. (2019). This function acts as a wrapper for `mb.run()` that allows for more clearly defined parameter names.

Usage

```
mb.linear(network, slope = list(pool = "rel", method = "common"),
  alpha = "study", positive.scale = FALSE, intercept = TRUE,
  rho = NULL, covar = NULL, var.scale = NULL,
  class.effect = list(), UME = FALSE, pd = "pv", parallel = TRUE,
  priors = NULL, ...)
```

Arguments

`network`	An object of class `mb.network`.
`slope`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`alpha`	Refers to the baseline mean response and is a character object that can take either: • `"study"` to constrain baseline to be equal for all arms within a study (`i` index is added))

- "arm" to allow baseline to vary between arms within a study (i,k index is added)).

positive.scale A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.

intercept A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (FALSE) or not (setting it to FALSE removes the intercept, alpha, from the model).

rho The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.

covar A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either "CS" (compound symmetry) or "AR1" (autoregressive AR1).

var.scale A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.

class.effect A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").

UME Can take either TRUE or FALSE (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: c("beta.1","beta.2").

pd Can take either:

- pv only pV will be reported (as automatically outputted by R2jags).
- plugin calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models.
- pd.kl calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result.
- popt calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)

parallel A boolean value that indicates whether JAGS should be run in parallel (TRUE) or not (FALSE). If TRUE then the number of cores to use is automatically calculated.

priors A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.

... Arguments to be sent to mb.run()

Value

An object of S3 class `c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. `d.et50` or `d.1`):

- `d` The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if `pool="rel"`
- `sd.d` The between-study SD (heterogeneity) for relative effects, reported if `pool="rel"` and `method="random"`
- `D` The class effect for a given class compared to the network reference class for a particular time-course parameter
- `sd.D` The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- `beta` If `pool="const"` then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If `pool="arm"` then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment
- `sd.beta` Reported if `method="random"` and `pool` is either `"const"` or `"arm"`. If `pool="const"` this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If `pool="arm"` this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- `rho` The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- `totresdev` The residual deviance of the model
- `deviance` The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and `model.arg`, a list of arguments provided to `mb.run()` which includes `jagscode`, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements `pool` and `method`.

`pool` is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- `"rel"` indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).

- `"arm"` indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- `"const"` indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

`method` is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- `"common"` implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- `"random"` implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.
- `numeric()` Assigned a numeric value - this can only be used if `pool="const"`. It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Correlation between observations

When modelling correlation between observations using `rho`, values for `rho` must imply a positive semidefinite covariance matrix. If estimating `rho` from the data (by assigning it `"estimate"`), the default prior distribution (`dunif(-1,1)`) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to `dunif(0,1)`) using the `priors` argument (see `get.prior`)

References

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Fit linear time-course with random consistency treatment effects on the slope
result <- mb.linear(network,
  slope=list(pool="rel", method="random"))


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
mcmcplots::denplot(result, parms="sd.slope")

# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "d.slope")


########## Output ############

# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)


###### Additional model arguments ######

# Fit model with unrelated mean effects and correlation between time points
mb.linear(network, alpha="study",
  slope=list(pool="rel", method="random"),
  rho=0.5, covar="AR1",
  UME=TRUE
  )
```

`mb.make.contrast` *Convert arm-based MBNMA data to contrast data*

Description

Converts an object of class `mb.network` from arm-based long MBNMA data to a data frame with contrast data (a separate contrast for each treatment comparison at each time point within each study). Data can be either long or wide.

Usage

```
mb.make.contrast(network, datatype = NULL, format = "wide")
```

Arguments

network	An object of class mb.network
datatype	A string indicating the data type. Can be binomial or normal
format	A string indicating the data format. Can be wide (two additional columns for each variable - contrast arms) or long.

Value

A data frame with the following columns. In wide format, some columns are given the indices 1 and 2 to indicate each arm in a given treatment comparison.:

- t The treatment in each arm
- TE The treatment effect (mean difference, log-odds) for the treatment in arm 1 versus the treatment in arm 2
- seTE The standard error for the treatment effect (mean difference, log-odds) for the treatment in arm 1 versus the treatment in arm 2
- y The mean response in each arm
- se The standard error of the mean in each arm
- r The number of responders in each arm
- n The total number of participants in each arm
- fupcount Follow-up identifier
- time The time the data are reported
- studyID Study identifier

Examples

```
# Create mb.network
network <- mb.network(osteopain)

# Convert to wide contrast data
mb.make.contrast(network, format="wide")

# Convert to long contrast data
mb.make.contrast(network, format="long")
```

`mb.nodesplit` *Perform node-splitting on a MBNMA time-course network*

Description

Within a MBNMA time-course network, split contributions into direct and indirect evidence and test for consistency between them. Closed loops of treatments in which it is possible to test for consistency are those in which direct and indirect evidence are available from independent sources van Valkenhoef van Valkenhoef et al. (2016).

Usage

```
mb.nodesplit(network, comparisons = mb.nodesplit.comparisons(network),
  nodesplit.parameters = "all", fun = "linear", user.fun = NULL,
  beta.1 = list(pool = "rel", method = "common"), beta.2 = NULL,
  beta.3 = NULL, beta.4 = NULL, ...)

## S3 method for class 'mb.nodesplit'
plot(x, plot.type = NULL, params = NULL, ...)
```

Arguments

`network`	An object of class `mb.network`.
`comparisons`	A data frame specifying the comparisons to be split (one row per comparison). The frame has two columns indicating each treatment for each comparison: `t1` and `t2`.
`nodesplit.parameters`	A character vector of named time-course parameters on which to node-split (e.g. c("beta.1", "beta.2")). Can use "all" to split on all time-course parameters.
`fun`	is a character specifying a functional form to be assigned to the time-course. Options are given in `details`.
`user.fun`	A formula specifying any relationship including `time` and one/several of: `beta.1`, `beta.2`, `beta.3`, `beta.4`.
`beta.1`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.2`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.3`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.4`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`...`	Arguments to be sent to `mb.run()`
`x`	An object of `class("mb.nodesplit")`

`plot.type` A character string that can take the value of `"forest"` to plot only forest plots, `"density"` to plot only density plots, or left as `NULL` (the default) to plot both types of plot.

`params` A character vector corresponding to a time-course parameter(s) for which to plot results. If left as `NULL` (the default), nodes-split results for all time-course parameters will be plotted.

Details

The S3 method `plot()` on an `mb.nodesplit` object generates either forest plots of posterior medians and 95% credible intervals, or density plots of posterior densities for direct and indirect evidence.

Value

A an object of `class("mb.nodesplit")` that is a list containing elements `d.X.Y` (treatment 1 = `X`, treatment 2 = `Y`). Each element (corresponding to each comparison) contains additional numbered elements corresponding to each parameter in the time-course function on which node splitting was performed. These elements then contain:

- `overlap` `matrix` MCMC results for the difference between direct and indirect evidence
- `p.values` Bayesian p-value for the test of consistency between direct and indirect evidence
- `quantiles`
- `forest.plot`
- `density.plot`
- `direct` MCMC results for the direct evidence
- `indirect` MCMC results for the indirect evidence

Plots the desired graph(s) and returns an object (or list of objects if `plot.type=NULL`) of `class(c("gg","ggplot"))`, which can be edited using `ggplot` commands.

Methods (by generic)

- `plot`: Plot outputs from nodesplit models

References

van Valkenhoef G, Dias S, Ades AE, Welton NJ (2016). "Automated generation of node-splitting models for assessment of inconsistency in network meta-analysis." *Res Synth Methods*, **7**(1), 80-93. ISSN 1759-2887 (Electronic) 1759-2879 (Linking), doi: 10.1002/jrsm.1167, https://www.ncbi.nlm.nih.gov/pubmed/26461181.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Identify comparisons informed by direct and indirect evidence
```

```
splits <- mb.nodesplit.comparisons(network)

# Fit an exponential time-course MBNMA
result <- mb.nodesplit(network, comparisons=splits, nodesplit.parameters="all",
  fun="exponential",
  beta.1=list(pool="rel", method="common"))

# Fit an emax time-course MBNMA with a node-split on emax parameters only
result <- mb.nodesplit(network, comparisons=splits, nodesplit.parameters="beta.1",
  fun="emax",
  beta.1=list(pool="rel", method="random"),
  beta.2=list(pool="rel", method="common")
  )

# Inspect results
print(result)
summary(result)

# Plot results
plot(result)
```

`mb.nodesplit.comparisons` *Identify comparisons in time-course MBNMA datasets that fulfill criteria for node-splitting*

Description

Identify comparisons informed by both direct and indirect evidence from independent sources in MBNMA datasets with repeated measurements in each study. These comparisons are therefore those which fulfill the criteria for testing for inconsistency via node-splitting, following the method of van Valkenhoef van Valkenhoef et al. (2016).

Usage

```
mb.nodesplit.comparisons(network)
```

Arguments

`network`	An object of class `mb.network`.

Details

Similar to `mtc.nodesplit` but uses a fixed reference treatment and therefore suggests fewer loops in which to test for inconsistency. Heterogeneity can also be parameterised as inconsistency and so testing for inconsistency in additional loops whilst changing the reference treatment would also be identifying heterogeneity. Depends on `igraph`.

Value

A data frame of comparisons that are informed by direct and indirect evidence from independent sources. Each row of the data frame is a different treatment comparison. Numerical codes in `t1` and `t2` correspond to treatment codes.

References

van Valkenhoef G, Dias S, Ades AE, Welton NJ (2016). "Automated generation of node-splitting models for assessment of inconsistency in network meta-analysis." *Res Synth Methods*, **7**(1), 80-93. ISSN 1759-2887 (Electronic) 1759-2879 (Linking), doi: 10.1002/jrsm.1167, https://www.ncbi.nlm.nih.gov/pubmed/26461181.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Identify comparisons informed by direct and indirect evidence
mb.nodesplit.comparisons(network)
```

---

mb.piecelinear *Run MBNMA model with a piecewise linear time-course function*

Description

Fits a Bayesian model-based network meta-analysis (MBNMA) with a defined time-course function. This function accounts for repeated measures over time within studies by applying a piecewise linear time-course function. Follows the methods of Pedder et al. (2019). This function acts as a wrapper for `mb.run()` that allows for more clearly defined parameter names.

Usage

```
mb.piecelinear(network, slope.1 = list(pool = "rel", method = "common"),
  slope.2 = list(pool = "rel", method = "common"), knot = list(pool =
  "const", method = "common"), alpha = "study", positive.scale = FALSE,
  intercept = TRUE, rho = NULL, covar = NULL, var.scale = NULL,
  class.effect = list(), UME = FALSE, pd = "pv", parallel = TRUE,
  priors = NULL, ...)
```

Arguments

`network`	An object of class `mb.network`.
`slope.1`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`slope.2`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).

knot — A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details). For this parameter, pool must be set to "arm" or "const" (i.e. it cannot be "rel").

alpha — Refers to the baseline mean response and is a character object that can take either:

- "study" to constrain baseline to be equal for all arms within a study (i index is added))
- "arm" to allow baseline to vary between arms within a study (i,k index is added)).

positive.scale — A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.

intercept — A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (FALSE) or not (setting it to FALSE removes the intercept, alpha, from the model).

rho — The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.

covar — A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either "CS" (compound symmetry) or "AR1" (autoregressive AR1).

var.scale — A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.

class.effect — A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").

UME — Can take either TRUE or FALSE (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: c("beta.1","beta.2").

pd — Can take either:

- pv only pV will be reported (as automatically outputted by R2jags).
- plugin calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models.
- pd.kl calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result.

- popt calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)

parallel — A boolean value that indicates whether JAGS should be run in parallel (TRUE) or not (FALSE). If TRUE then the number of cores to use is automatically calculated.

priors — A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.

... — Arguments to be sent to mb.run()

Value

An object of S3 class c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. d.et50 or d.1):

- d The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if pool="rel"
- sd.d The between-study SD (heterogeneity) for relative effects, reported if pool="rel" and method="random"
- D The class effect for a given class compared to the network reference class for a particular time-course parameter
- sd.D The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- beta If pool="const" then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If pool="arm" then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment
- sd.beta Reported if method="random" and pool is either "const" or "arm". If pool="const" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If pool="arm" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- rho The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- totresdev The residual deviance of the model
- deviance The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and model.arg, a list of arguments provided to mb.run() which includes jagscode, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements `pool` and `method`.

`pool` is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- `"rel"` indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).
- `"arm"` indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- `"const"` indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

`method` is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- `"common"` implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- `"random"` implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.
- `numeric()` Assigned a numeric value - this can only be used if `pool="const"`. It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Correlation between observations

When modelling correlation between observations using `rho`, values for `rho` must imply a positive semidefinite covariance matrix. If estimating `rho` from the data (by assigning it `"estimate"`), the default prior distribution (`dunif(-1,1)`) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to `dunif(0,1)`) using the `priors` argument (see `get.prior`)

References

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Fit piecewise linear time-course with random consistency treatment effects on slope.1,
#absolute time-course parameters by treatment on slope.2, and an exchangeable parameter
#for knot estimated across the network
result <- mb.piecelinear(network,
  slope.1=list(pool="rel", method="random"),
  slope.2=list(pool="arm", method="common"),
  knot=list(pool="const", method="common"))


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
mcmcplots::denplot(result, parms=c("beta.slope.2", "beta.knot"))

# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "beta.slope.2")


########## Output ###########

# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)
```

```
###### Additional model arguments ######

# Fit model with unrelated mean effects on beta.1 and correlation between time points
mb.piecelinear(network, alpha="study",
  slope.1=list(pool="rel", method="random"),
  slope.2=list(pool="rel", method="common"),
  knot=list(pool="const", method=1),
  rho="estimate", covar="CS",
  UME="slope.1"
  )
```

`mb.quadratic` *Run MBNMA model with a quadratic time-course function*

Description

Fits a Bayesian model-based network meta-analysis (MBNMA) with a defined time-course function. This function accounts for repeated measures over time within studies by applying a quadratic time-course function. Follows the methods of Pedder et al. (2019). This function acts as a wrapper for `mb.run()` that allows for more clearly defined parameter names.

Usage

```
mb.quadratic(network, beta.1 = list(pool = "rel", method = "common"),
  beta.2 = list(pool = "rel", method = "common"), alpha = "study",
  positive.scale = FALSE, intercept = TRUE, rho = NULL,
  covar = NULL, var.scale = NULL, class.effect = list(),
  UME = FALSE, pd = "pv", parallel = TRUE, priors = NULL, ...)
```

Arguments

`network`	An object of class `mb.network`.
`beta.1`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.2`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`alpha`	Refers to the baseline mean response and is a character object that can take either: • `"study"` to constrain baseline to be equal for all arms within a study (`i` index is added)) • `"arm"` to allow baseline to vary between arms within a study (`i,k` index is added)).

`positive.scale` A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.

`intercept` A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (`FALSE`) or not (setting it to `FALSE` removes the intercept, `alpha`, from the model).

`rho` The correlation coefficient when modelling correlation between time points. If left as `NULL` (the default) then this implies modelling no correlation between time points. Can either be assigned the string `"estimate"` to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes `rho` in the model to the assigned value, either for when `rho` is calculated externally or for use in deterministic sensitivity analyses.

`covar` A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either `"CS"` (compound symmetry) or `"AR1"` (autoregressive AR1).

`var.scale` A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.

`class.effect` A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be (`"common"` or `"random"`). For example: `list("beta.2"="common","beta.3"="random")`.

`UME` Can take either `TRUE` or `FALSE` (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: `c("beta.1","beta.2")`.

`pd` Can take either:

- `pv` only pV will be reported (as automatically outputted by R2jags).
- `plugin` calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models.
- `pd.kl` calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result.
- `popt` calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)

`parallel` A boolean value that indicates whether JAGS should be run in parallel (`TRUE`) or not (`FALSE`). If `TRUE` then the number of cores to use is automatically calculated.

`priors` A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.

`...` Arguments to be sent to `mb.run()`

Value

An object of S3 class `c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. `d.et50` or `d.1`):

- `d` The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if `pool="rel"`
- `sd.d` The between-study SD (heterogeneity) for relative effects, reported if `pool="rel"` and `method="random"`
- `D` The class effect for a given class compared to the network reference class for a particular time-course parameter
- `sd.D` The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- `beta` If `pool="const"` then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If `pool="arm"` then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment
- `sd.beta` Reported if `method="random"` and `pool` is either `"const"` or `"arm"`. If `pool="const"` this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If `pool="arm"` this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- `rho` The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- `totresdev` The residual deviance of the model
- `deviance` The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and `model.arg`, a list of arguments provided to `mb.run()` which includes `jagscode`, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements `pool` and `method`.

`pool` is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- `"rel"` indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).

- `"arm"` indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- `"const"` indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

`method` is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- `"common"` implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- `"random"` implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.
- `numeric()` Assigned a numeric value - this can only be used if `pool="const"`. It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Correlation between observations

When modelling correlation between observations using `rho`, values for `rho` must imply a positive semidefinite covariance matrix. If estimating `rho` from the data (by assigning it `"estimate"`), the default prior distribution (`dunif(-1,1)`) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to `dunif(0,1)`) using the `priors` argument (see `get.prior`)

References

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Fit quadratic time-course with fixed consistency treatment effects on beta.1 and
#random consistency treatment effects on beta.2
result <- mb.quadratic(network,
  beta.1=list(pool="rel", method="common"),
  beta.2=list(pool="rel", method="random"))


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
mcmcplots::denplot(result, parms=c("sd.2", "d.1[3]", "d.2[3]", "totresdev"))

# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "d.2")


########## Output ###########

# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)


###### Additional model arguments ######

# Fit model with unrelated mean effects on beta.1
mb.quadratic(network, alpha="study",
  beta.1=list(pool="rel", method="random"),
  beta.2=list(pool="const", method="common"),
  UME="beta.1"
  )
```

---

mb.run *Run MBNMA time-course models*

Description

Fits a Bayesian time-course model for model-based network meta-analysis (MBNMA) that can account for repeated measures over time within studies by applying a desired time-course function.

Follows the methods of Pedder et al. (2019).

Usage

```
mb.run(network, parameters.to.save = NULL, fun = "linear",
  user.fun = NULL, alpha = "study", beta.1 = list(pool = "rel",
  method = "common"), beta.2 = NULL, beta.3 = NULL, beta.4 = NULL,
  positive.scale = FALSE, intercept = TRUE, rho = NULL,
  covar = NULL, var.scale = NULL, class.effect = list(),
  UME = FALSE, pd = "pv", parallel = TRUE, priors = NULL,
  n.iter = 10000, n.chains = 3, n.burnin = floor(n.iter/2),
  n.thin = max(1, floor((n.iter - n.burnin)/1000)), model.file = NULL,
  arg.params = NULL, ...)
```

Arguments

network	An object of class mb.network.
parameters.to.save	A character vector containing names of parameters to monitor in JAGS
fun	is a character specifying a functional form to be assigned to the time-course. Options are given in details.
user.fun	A formula specifying any relationship including time and one/several of: beta.1, beta.2, beta.3, beta.4.
alpha	Refers to the baseline mean response and is a character object that can take either: • "study" to constrain baseline to be equal for all arms within a study (i index is added)) • "arm" to allow baseline to vary between arms within a study (i,k index is added)).
beta.1	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.2	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.3	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.4	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
positive.scale	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.
intercept	A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (FALSE) or not (setting it to FALSE removes the intercept, alpha, from the model).

rho — The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.

covar — A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either "CS" (compound symmetry) or "AR1" (autoregressive AR1).

var.scale — A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.

class.effect — A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").

UME — Can take either TRUE or FALSE (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: c("beta.1","beta.2").

pd — Can take either:

- pv only pV will be reported (as automatically outputted by R2jags).
- plugin calculates pD by the plug-in method (Spiegelhalter et al. 2002). It is faster, but may output negative non-sensical values, due to skewed deviances that can arise with non-linear models.
- pd.kl calculates pD by the Kullback–Leibler divergence (Plummer 2008). This will require running the model for additional iterations but will always produce a positive result.
- popt calculates pD using an optimism adjustment which allows for calculation of the penalized expected deviance (Plummer 2008)

parallel — A boolean value that indicates whether JAGS should be run in parallel (TRUE) or not (FALSE). If TRUE then the number of cores to use is automatically calculated.

priors — A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.

n.iter — number of total iterations per chain (including burn in; default: 15000)

n.chains — number of Markov chains (default: 3)

n.burnin — length of burn in, i.e. number of iterations to discard at the beginning. Default is n.iter/2``,that is,discarding the first half of the simulations. If n.burnin‘ is 0, jags() will run 100 iterations for adaption.

n.thin — thinning rate. Must be a positive integer. Set n.thin > 1`` to save memory and computation time if n.iteris large. Default ismax(1, floor(n.chains * (n.iter-n.burnin) / 1000))“ which will only thin if there are at least 2000 simulations.

model.file	A JAGS model written as a character object that can be used to overwrite the JAGS model that is automatically written based on the specified options. Useful when amending priors using replace.prior()
arg.params	Contains a list of arguments sent to mb.run() by time-course specific wrapper functions
...	Arguments to be sent to R2jags.

Value

An object of S3 class c("mbnma","rjags")`` containing parameter results from the model. Can be summarized by print()and can check traceplots usingR2jags::traceplot()or various functions from the packagemcmcplots‘.

Nodes that are automatically monitored (if present in the model) have the following interpretation. They will have an additional suffix that relates to the name/number of the time-course parameter to which they correspond (e.g. d.et50 or d.1):

- d The pooled relative effect for a given treatment compared to the network reference treatment for a particular time-course parameter, reported if pool="rel"
- sd.d The between-study SD (heterogeneity) for relative effects, reported if pool="rel" and method="random"
- D The class effect for a given class compared to the network reference class for a particular time-course parameter
- sd.D The standard deviation for the pooled relative effects of treatments within a given class from a model with a random class effect.
- beta If pool="const" then only a single node will be present in the output, which corresponds to the absolute value of a particular time-course parameter across the network, If pool="arm" then for the relevant time-course parameter there will be one node for each treatment, which represents the absolute value of the time-course parameter for each treatment
- sd.beta Reported if method="random" and pool is either "const" or "arm". If pool="const" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable across the network. If pool="arm" this represents the between-study SD for the absolute value of a particular time-course parameter exchangeable by treatment
- rho The correlation coefficient for correlation between time-points. Its interpretation will differ depending on the covariance structure used
- totresdev The residual deviance of the model
- deviance The deviance of the model

If there are errors in the JAGS model code then the object will be a list consisting of two elements - an error message from JAGS that can help with debugging and model.arg, a list of arguments provided to mb.run() which includes jagscode, the JAGS code for the model that can help users identify the source of the error.

Time-course parameters

Time-course parameters in the model must be provided as a list with named elements pool and method.

`pool` is used to define the approach used for pooling of a given time-course parameter and can take any of the following values:

- `"rel"` indicates that relative effects should be pooled for this time-course parameter. This preserves randomisation within included studies, are likely to vary less between studies (only due to effect modification), and allow for testing of consistency between direct and indirect evidence. Pooling follows the general approach for Network Meta-Analysis proposed by Lu and Ades (2004).
- `"arm"` indicates that study arms should be pooled within each treatment in the network for this time-course parameter. This allows estimation of absolute time-course parameter values, but makes stronger assumptions regarding similarity of studies.
- `"const"` indicates that study arms should be pooled across the whole network for this time-course parameter *independently of assigned treatment*. This implies using a single value across the network for this time-course parameter, and may therefore be making very strong assumptions of similarity.

`method` is used to define the model used for meta-analysis for a given time-course parameter and can take any of the following values:

- `"common"` implies that all studies estimate the same true effect (akin to a "fixed effects" meta-analysis)
- `"random"` implies that all studies estimate a separate true effect, but that each of these true effects vary randomly around a true mean effect. This approach allows for modelling of between-study heterogeneity.
- `numeric()` Assigned a numeric value - this can only be used if `pool="const"`. It indicates that this time-course parameter should not be estimated from the data but should be assigned the numeric value determined by the user. This can be useful for fixing specific time-course parameters (e.g. Hill parameters in Emax functions or knot location in piecewise functions).

When relative effects are modelled on more than one time-course parameter, correlation between the time-course parameters is automatically estimated using a vague Wishart prior. This prior can be made slightly more informative by specifying the relative scale of variances between the time-course parameters using `var.scale`.

Time-course function

Several general time-course functions are provided, but a user-defined time-course relationship can instead be used.

Built-in time-course functions are:

- `"linear"`: `beta.1` refers to the gradient
- `"quadratic"`: `beta.1` refers to the gradient, `beta.2` refers to the change in gradient
- `"exponential"`: `beta.1` refers to the rate of gain/decay
- `"emax"` (emax without a Hill parameter): `beta.1` refers to Emax parameter, `beta.2` refers to ET50 parameter
- `"emax.hill"` (emax with a Hill parameter): `beta.1` refers to Emax parameter, `beta.2` refers to ET50 parameter, `beta.3` refers to Hill parameter

- "fract.poly.first" (first-order fractional polynomial): beta.1 refers to the slope parameter, beta.3 refers to the power parameter
- "fract.poly.second" (second-order fractional polynomial): beta.1 refers to the first slope parameter, beta.2 refers to the first power parameter, beta.3 refers to the first power parameter, beta.4 refers to the second power parameter
- "piecelinear" piecewise linear: beta.1 refers to the gradient of the first linear piece, beta.2 refers to the gradient of the second linear piece, beta.3 refers to the knot location (the time at which the two pieces are joined)
- "user" (user-defined function: user.fun must be specified in arguments)

Correlation between observations

When modelling correlation between observations using rho, values for rho must imply a positive semidefinite covariance matrix. If estimating rho from the data (by assigning it "estimate"), the default prior distribution (dunif(-1,1)) may include values that exclude a positive semidefinite covariance matrix. This prior can be restricted (e.g. to dunif(0,1)) using the priors argument (see get.prior)

References

Lu G, Ades AE (2004). "Combination of direct and indirect evidence in mixed treatment comparisons." *Stat Med*, **23**(20), 3105-24. ISSN 0277-6715 (Print) 0277-6715 (Linking), doi: 10.1002/sim.1875, https://www.ncbi.nlm.nih.gov/pubmed/15449338.

Pedder H, Dias S, Bennetts M, Boucher M, Welton NJ (2019). "Modelling time-course relationships with multiple treatments: Model-Based Network Meta-Analysis for continuous summary outcomes." *Res Synth Methods*, **10**(2), 267-286.

Plummer M (2008). "Penalized loss functions for Bayesian model comparison." *Biostatistics*, **9**(3), 523-39. ISSN 1468-4357 (Electronic) 1465-4644 (Linking), https://www.ncbi.nlm.nih.gov/pubmed/18209015.

Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). "Bayesian measures of model complexity and fit." *J R Statistic Soc B*, **64**(4), 583-639.

Examples

```
# Create mb.network object
network <- mb.network(osteopain)

# Fit a linear time-course MBNMA with random consistency treatment effects on beta.1 (slope)
#and equal baselines
#in study arms
mb.run(network, fun="linear",
  alpha="study", beta.1=list(pool="rel", method="random"))

# Fit an emax time-course MBNMA with fixed consistency treatment effects on beta.1 (emax),
#a common parameter estimated across the network for beta.2 (et50) and a Hill parameter of 0.5
```

```
#across the network on data reported as change from baseline
result <- mb.run(network, fun="emax.hill",
  beta.1=list(pool="rel", method="common"),
  beta.2=list(pool="const", method="common"),
  beta.3=list(pool="const", method=0.5),
  intercept=TRUE)


####### Examine MCMC diagnostics (using mcmcplots package) #######

# Density plots
mcmcplots::denplot(result, c("beta.2", "deviance", "totresdev"))

# Traceplots
mcmcplots::traplot(result)

# Caterpillar plots
mcmcplots::caterplot(result, "d.1")


########## Output ###########

# Print R2jags output and summary
print(result)
summary(result)

# Plot forest plot of results
plot(result)


###### Additional model arguments ######

# Fit a linear time-course MBNMA that accounts for correlation between time points
mb.run(network, fun="linear",
  beta.1=list(pool="rel", method="common"),
  rho="estimate", covar="CS")

# Define a user-defined time-course relationship for use in mb.run
time.fun <- ~alpha + exp(beta.1 * time) + (time^beta.2)

# Run model using Kullback-Liebler divergence to calculate pD
mb.run(network, fun="user", user.fun=time.fun,
  beta.1=list(pool="rel", method="random"),
  beta.2=list(pool="rel", method="common"),
  pd="pd.kl")
```

---

mb.update *Update MBNMA to obtain deviance contributions or fitted values*

Description

Update MBNMA to obtain deviance contributions or fitted values

Usage

```
mb.update(mbnma, param = "theta", n.iter = mbnma$BUGSoutput$n.iter,
  n.thin = mbnma$BUGSoutput$n.thin)
```

Arguments

mbnma	An S3 object of class "mbnma" generated by running a time-course MBNMA model
param	A character object that represents the parameter within the model to monitor when updating. Can currently only be used for monitoring fitted values and deviance contributions and so can take either "dev" (for deviance contributions), "resdev" (for residual deviance contributions) or "theta" (for fitted values).
n.iter	The number of iterations to update the model whilst monitoring additional parameters (if necessary). Must be a positive integer. Default is the value used in mbnma.
n.thin	The thinning rate. Must be a positive integer. Default is the value used in mbnma.

Value

A data frame containing posterior means for the specified param at each observation, arm and study.

Examples

```
# Using the alogliptin dataset
network <- mb.network(alog_pcfb)

# Run Emax model
emax <- mb.emax(network)

# Update model for 500 iterations to monitor fitted values
mb.update(emax, param="theta", n.iter=500)

# Update model for 500 iterations to monitor residual deviance contributions
mb.update(emax, param="resdev", n.iter=500)

# Update model for 500 iterations to monitor deviance contributions
mb.update(emax, param="dev", n.iter=500)
```

`mb.validate.data` *Validates that a dataset fulfills requirements for MBNMA*

Description

Validates that a dataset fulfills requirements for MBNMA

Usage

```
mb.validate.data(data.ab, single.arm = FALSE, CFB = TRUE)
```

Arguments

`data.ab`	A data frame of arm-level data in "long" format containing the columns: • `studyID` Study identifiers • `time` Numeric data indicating follow-up times • `y` Numeric data indicating the aggregate response for a given observation (e.g. mean) • `se` Numeric data indicating the standard error for a given observation • `treatment` Treatment identifiers (can be numeric, factor or character) • `class` An optional column indicating a particular class identifier. Observations with the same treatment identifier must also have the same class identifier. • `N` An optional column indicating the number of participants used to calculate the response at a given observation
`single.arm`	A boolean object to indicate whether or not function should allow singe arm studies to be allowed in the network without returning an error. Default is not to allow their inclusion (`single.arm=FALSE`)
`CFB`	A boolean object to indicate if the dataset is composed of studies measuring change from baseline (`TRUE`) or not (`FALSE`). It is not essential to specify this correctly but failing to do so may lead to warnings.

Details

Checks done within the validation:

- Checks data.ab has required column names
- Checks there are no NAs
- Checks that all SEs are positive
- Checks that studies have baseline measurement (unless change from baseline data is being used)
- Checks that arms are balanced at each time point
- Checks that class codes are consistent within each treatment
- Checks that treatment codes are consistent across different time points within a study
- Checks that studies have at least two arms (if `single.arm = FALSE`)

Value

An error or warnings if checks are not passed. Runs silently if checks are passed

---

`mb.write` *Write MBNMA time-course models JAGS code*

Description

Writes JAGS code for a Bayesian time-course model for model-based network meta-analysis (MBNMA).

Usage

```
mb.write(fun = "linear", user.fun = NULL, alpha = "arm",
  beta.1 = "rel.common", beta.2 = NULL, beta.3 = NULL,
  beta.4 = NULL, positive.scale = TRUE, intercept = TRUE,
  rho = NULL, covar = NULL, var.scale = NULL,
  class.effect = list(), UME = FALSE)
```

Arguments

`fun`	is a character specifying a functional form to be assigned to the time-course. Options are given in `details`.
`user.fun`	A formula specifying any relationship including `time` and one/several of: `beta.1`, `beta.2`, `beta.3`, `beta.4`.
`alpha`	Refers to the baseline mean response and is a character object that can take either: • `"study"` to constrain baseline to be equal for all arms within a study (`i` index is added)) • `"arm"` to allow baseline to vary between arms within a study (`i,k` index is added)).
`beta.1`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.2`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.3`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.4`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`positive.scale`	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.

`intercept` A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (`FALSE`) or not (setting it to `FALSE` removes the intercept, `alpha`, from the model).

`rho` The correlation coefficient when modelling correlation between time points. If left as `NULL` (the default) then this implies modelling no correlation between time points. Can either be assigned the string `"estimate"` to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes `rho` in the model to the assigned value, either for when `rho` is calculated externally or for use in deterministic sensitivity analyses.

`covar` A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either `"CS"` (compound symmetry) or `"AR1"` (autoregressive AR1).

`var.scale` A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.

`class.effect` A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be (`"common"` or `"random"`). For example: `list("beta.2"="common","beta.3"="random")`.

`UME` Can take either `TRUE` or `FALSE` (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: `c("beta.1","beta.2")`.

Value

A single long character string containing the JAGS model generated based on the arguments passed to the function.

Examples

```
# Write a linear time-course MBNMA with random treatment effects on beta.1 and equal baselines
#in study arms
model <- mb.write(fun="linear", alpha="study", beta.1="rel.random")
cat(model) # Concatenates model representations making code more easily readable

# Write an emax time-course MBNMA with a Hill parameter of 0.5 with no intercept
model <- mb.write(fun="emax.hill",
  beta.1="rel.common", beta.2="const.common", beta.3=0.5,
  intercept=TRUE)
cat(model) # Concatenates model representations making code more easily readable

# Write an exponential time-course MBNMA that accounts for correlation between time points
model <- mb.write(fun="exponential",
  alpha="arm", beta.1="rel.common",
  rho="estimate", covar="AR1")
cat(model)
```

```
# Define a user-defined time-course relationship for the MBNMA JAGS model
time.fun <- ~ alpha + (exp(beta.1 * time) / (beta.2 * time))
model <- mb.write(fun="user", user.fun=time.fun,
  beta.1="rel.random", beta.2="rel.common")
cat(model)
```

---

obesityBW_CFB *Studies of treatments for reducing body weight in patients with obesity*

Description

A dataset from a systematic review of pharmacological treatments for reducing body weight in patients with obesity. The outcome is continuous, and aggregate data responses are given as mean change from baseline in body weight (KG). Overall there are 35 RCTs investigating 26 treatments of 16 agents (/combinations of agents) in the network. Standard deviations have been imputed for 421 observations.

Usage

```
obesityBW_CFB
```

Format

A data frame with 710 rows and 7 variables:

- `studyID` Study identifiers
- `time` Numeric data indicating follow-up times
- `y` Numeric data indicating the mean response for a given observation
- `se` Numeric data indicating the standard error for a given observation
- `N` Numeric data indicating the number of participants used to calculate means for each observation
- `treatment` Treatment identifiers as factors. Labels are shortened treatment names.
- `treatname` Character data giving the full names of each treatment in the format agent_dose
- `agent` Agent (drug) names stored as characters
- `agentclass` The drug class of the agent (a broader category than `agent`) stored as characters

Details

`obesityBW_CFB` is a data frame in long format (one row per observation, arm and study), with the variables `studyID`, `time`, `y`, `se`, `N`, `treatment`, `treatname`, `agent` and `agentclass`.

Source

Pfizer Ltd.

osteopain *Studies of pain relief medications for osteoarthritis*

Description

A dataset containing results on the WOMAC pain scale (0-10) over time for studies investigating 29 treatments for pain relief in patients with osteoarthritis. Standard deviations have been imputed for 269 observations.

Usage

```
osteopain
```

Format

A data frame with 417 rows and 7 variables:

- `studyID` Study identifiers
- `time` Numeric data indicating follow-up times
- `y` Numeric data indicating the mean response for a given observation
- `se` Numeric data indicating the standard error for a given observation
- `treatment` Treatment identifiers as factors. Labels are shortened treatment names.
- `arm` Arm identifiers coded for each study
- `treatname` Character data giving the full names of each treatment

Details

`osteopain` is a data frame in long format (one row per observation, arm and study), with the variables `studyID`, `time`, `y`, `se`, `treatment`, `arm` and `treatname`.

Source

Pfizer Ltd.

pDcalc *Calculate plugin pD from a JAGS model with univariate likelihood for studies with repeated measurements*

Description

Uses results from MBNMA JAGS models to calculate pD via the plugin method (Spiegelhalter et al. 2002). Can only be used for models with known standard errors or covariance matrices (typically univariate).

Usage

```
pDcalc(obs1, obs2, fups = NULL, narm, NS, theta.result, resdev.result,
  likelihood = "normal", type = "time")
```

Arguments

obs1	A matrix (study x arm) or array (study x arm x time point) containing observed data for y (normal likelihood) or r (binomial or Poisson likelihood) in each arm of each study. This will be the same array used as data for the JAGS model.
obs2	A matrix (study x arm) or array (study x arm x time point) containing observed data for se (normal likelihood), N (binomial likelihood) or E (Poisson likelihood) in each arm of each study. This will be the same array used as data for the JAGS model.
fups	A numeric vector of length equal to the number of studies, containing the number of follow-up mean responses reported in each study. Required for time-course MBNMA models (if type="time")
narm	A numeric vector of length equal to the number of studies, containing the number of arms in each study.
NS	A single number equal to the number of studies in the dataset.
theta.result	A matrix (study x arm) or array (study x arm x time point) containing the posterior mean predicted means/probabilities/rate in each arm of each study. This will be estimated by the JAGS model.
resdev.result	A matrix (study x arm) or array (study x arm x time point) containing the posterior mean residual deviance contributions in each arm of each study. This will be estimated by the JAGS model.
likelihood	A character object of any of the following likelihoods: • univariate • binomial (does not work with time-course MBNMA models) • multivar.normal (does not work with time-course MBNMA models)
type	The type of MBNMA model fitted. Can be either "time" or "dose"

Details

Method for calculating pD via the plugin method proposed by (Spiegelhalter et al. 2002). Standard errors / covariance matrices must be assumed to be known. To obtain values for theta.result and resdev.result these parameters must be monitored when running the JAGS model.

For non-linear time-course MBNMA models residual deviance contributions may be skewed, which can lead to non-sensical results when calculating pD via the plugin method. Alternative approaches are to use pV (pv) as an approximation (Plummer REF) or pD calculated by Kullback–Leibler divergence (pd.kl) or using an optimism adjustment (popt) (REF).

References

TO ADD pV REF

Examples

```
# Using the alogliptin dataset
network <- mb.network(alog_pcfb)

# Run Emax model saving predicted means and residual deviance contributions
emax <- mb.emax(network, parameters.to.save=c("theta", "resdev"))

# Get matrices of observed data
jagsdat <- getjagsdata(network$data.ab)

# Plugin estimation of pD is problematic with non-linear models as it often leads to
#negative values, hence use of pV, pd.kl and popt as other measures for the effective
#number of parameters
pDcalc(obs1=jagsdat$y, obs2=jagsdat$se,
  fups=jagsdat$fups, narm=jagsdat$narm, NS=jagsdat$NS,
  theta.result = emax$BUGSoutput$mean$theta,
  resdev.result = emax$BUGSoutput$mean$resdev
  )
```

`plot.mb.network` *Create an mb.network object*

Description

Creates an object of class `mb.network`. Various MBNMA functions can subsequently be applied to this object.

Usage

```
## S3 method for class 'mb.network'
plot(x, edge.scale = 1, label.distance = 0,
  level = "treatment", remove.loops = FALSE, v.color = "connect",
  v.scale = NULL, layout = igraph::in_circle(), ...)

mb.network(data.ab, reference = 1, description = "Network")
```

Arguments

`x`	An object of class `mb.network`.
`edge.scale`	A number to scale the thickness of connecting lines (edges). Line thickness is proportional to the number of studies for a given comparison. Set to `0` to make thickness equal for all comparisons.
`label.distance`	A number scaling the distance of labels from the nodes to improve readability. The labels will be directly on top of the nodes if the default of `0` is used. Option only applicable if `layout_in_circle` is set to `TRUE`.

`level` A string indicating whether nodes/facets should represent `treatment` or `class` in the plot. Can be used to examine the expected impact of modelling class/agent effects.

`remove.loops` A boolean value indicating whether to include loops that indicate comparisons within a node.

`v.color` Can take either `"connect"` (the default) to indicate that nodes should only be coloured if they are connected to the network reference treatment (indicates network connectivity) or `"class"` to colour nodes by class (this requires that the variable `class` be included in the dataset).

`v.scale` A number with which to scale the size of the nodes. If the variable `N` (to indicate the numbers of participants at each observation) is included in the dataset then the size of the nodes will be proportional to the number of participants within a treatment/class in the network *at the earliest time point reported in each study*.

`layout` An igraph layout specification. This is a function specifying an igraph layout that determines the arrangement of the vertices (nodes). The default `igraph::as_circle()` arranged vertices in a circle. Two other useful layouts for network plots are: `igraph::as_star()`, `igraph::with_fr()`. Others can be found in `layout_`

`...` Options for plotting in `igraph`.

`data.ab` A data frame of arm-level data in "long" format containing the columns:

- `studyID` Study identifiers
- `time` Numeric data indicating follow-up times
- `y` Numeric data indicating the aggregate response for a given observation (e.g. mean)
- `se` Numeric data indicating the standard error for a given observation
- `treatment` Treatment identifiers (can be numeric, factor or character)
- `class` An optional column indicating a particular class identifier. Observations with the same treatment identifier must also have the same class identifier.
- `N` An optional column indicating the number of participants used to calculate the response at a given observation

`reference` A number or character (depending on the format of `treatment` within `data.ab`) indicating the reference treatment in the network (i.e. those for which estimated relative treatment effects estimated by the model will be compared to).

`description` Optional. Short description of the network.

Details

The S3 method `plot()` on an `mb.network` object generates a network plot that shows how different treatments are connected within the network via study comparisons. This can be used to identify how direct and indirect evidence are informing different treatment comparisons. Depends on `igraph`.

Missing values (`NA`) cannot be included in the dataset. Studies must have a baseline measurement and more than a single follow-up time (unless change from baseline data are being used). Data must be present for all arms within a study at each follow-up time.

Value

An object of class `mb.network` which is a list containing:

- `description` A short description of the network
- `data.ab` A data frame containing the arm-level network data (treatment identifiers will have been recoded to a sequential numeric code)
- `treatments` A character vector indicating the treatment identifiers that correspond to the new treatment codes.
- `classes` A character vector indicating the class identifiers (if included in the original data) that correspond to the new class codes.

Methods (by generic)

- `plot`: Generate a network plot

Examples

```
# Create an mb.network object from the data
network <- mb.network(osteopain)

# Arrange network plot in a star with the reference treatment in the centre
plot(network, layout=igraph::as_star())

# Generate a network plot at the class level that removes loops indicating comparisons
#within a node
goutnet <- mb.network(goutSUA_CFB)
plot(goutnet, level="class", remove.loops=TRUE)

# Generate a network plot at the treatment level that colours nodes by class
plot(goutnet, v.color="class", remove.loops=TRUE)

# Plot network in which node size is proportional to number of participants
alognet <- mb.network(alog_pcfb)
plot(alognet, v.scale=2)

# Using the osteoarthritis dataset
print(osteopain)

# Define network
network <- mb.network(osteopain, description="Osteoarthritis Dataset")

# Define network with different network reference treatment
network <- mb.network(osteopain, reference="Ce_200")


# Using the alogliptin dataset
network <- mb.network(alog_pcfb, description="Alogliptin Dataset")

# Examine networks
print(network)
plot(network)
```

plot.mb.predict *Plots predicted responses from a time-course MBNMA model*

Description

Plots predicted responses from a time-course MBNMA model

Usage

```
## S3 method for class 'mb.predict'
plot(x, disp.obs = FALSE, overlay.ref = TRUE,
  col = "blue", max.col.scale = NULL, treat.labs = NULL, ...)
```

Arguments

x	An object of class "mb.predict" generated by predict("mbnma")
disp.obs	A boolean object to indicate whether to show shaded sections of the plot for where there is observed data (TRUE) or not (FALSE)
overlay.ref	A boolean object indicating whether to overlay a line showing the median network reference treatment response over time on the plot (TRUE) or not (FALSE). The network reference treatment (treatment 1. must be included in predict for this to display the network reference treatment properly.
col	A character indicating the colour to use for shading if disp.obs is set to TRUE. Can be either "blue", "green", or "red"
max.col.scale	Rarely requires adjustment. The maximum count of observations (therefore the darkest shaded color) only used if disp.obs is used. This allows consistency of shading between multiple plotted graphs. It should always be at least as high as the maximum count of observations plotted
treat.labs	A vector of treatment labels in the same order as treatment codes. Easiest to use treatment labels stored by mb.network()
...	Arguments for ggplot

Details

For the S3 method plot(), if disp.obs is set to TRUE it is advisable to ensure predictions in predict are estimated using an even sequence of time points to avoid misrepresentation of shaded densities. Shaded counts of observations will be relative to the treatment plotted in each panel rather than to the network reference treatment if disp.obs is set to TRUE.

Examples

```
# Create an mb.network object from a dataset
network <- mb.network(alog_pcfb)

# Run an MBNMA model with an Emax time-course
emax <- mb.emax(network,
  emax=list(pool="rel", method="common"),
  et50=list(pool="rel", method="common"))

# Predict responses using the original dataset to estimate the network reference
#treatment response
df.ref <- alog_pcfb[alog_pcfb$treatment=="placebo",]
predict <- predict(emax, times=c(0:15), baseline=10, ref.estimate=df.ref)

# Plot the predicted responses with observations displayed on plot as green shading
plot(predict, disp.obs=TRUE, overlay.ref=FALSE, col="green")

# Plot the predicted responses with the median network reference treatment response overlayed
#on the plot
plot(predict, disp.obs=FALSE, overlay.ref=TRUE)
```

`plot.mb.rank` *Plot histograms of rankings from MBNMA models*

Description

Plot histograms of rankings from MBNMA models

Usage

```
## S3 method for class 'mb.rank'
plot(x, params = NULL, treat.labs = NULL, ...)
```

Arguments

`x`	An object of class `"mb.rank"` generated by `rank.mbnma()`
`params`	A character vector containing any model parameters monitored in `mbnma` for which ranking is desired (e.g. `"beta.1"`, `"d.emax"`). Parameters must vary by treatment for ranking to be possible. Can include `"auc"` (see details).
`treat.labs`	A vector of treatment labels in the same order as treatment codes. Easiest to use treatment labels stored by `mb.network()`
`...`	Arguments to be sent to `ggplot::geom_bar()`

Value

A series of histograms that show rankings for each treatment/agent/prediction, with a separate panel for each parameter The object returned is a list containing a separate element for each parameter in `params` which is an object of class `c("gg","ggplot")`.

Examples

```
# Create an mb.network object from a dataset
network <- mb.network(osteopain)

# Run an MBNMA model with an Emax time-course
emax <- mb.emax(network,
  emax=list(pool="rel", method="common"),
  et50=list(pool="arm", method="common"),
  positive.scale=TRUE)

# Calculate treatment rankings
ranks <- rank(emax,
  param=c("auc", "d.emax", "beta.et50"),
  int.range=c(0,15),
  treats=c(1:10), n.iter=500)

# Plot histograms for ranking by AUC
plot(ranks, param="auc")

# Plot histograms for ranking by d.emax
plot(ranks, param="d.emax")
```

`plot.mbnma` *Forest plot for results from time-course MBNMA models*

Description

Generates a forest plot for time-course parameters of interest from results from time-course MBNMA models.

Usage

```
## S3 method for class 'mbnma'
plot(x, params = NULL, treat.labs = NULL,
  class.labs = NULL, ...)
```

Arguments

x	An S3 object of class "mbnma" generated by running a time-course MBNMA model
params	A character vector of time-course parameters to plot. Parameters must be given the same name as monitored nodes in mbnma and must vary by treatment or class. Can be set to NULL to include all available time-course parameters estimated by mbnma.
treat.labs	A character vector of treatment labels. If left as NULL (the default) then labels will be used as defined in the data.
class.labs	A character vector of class labels if mbnma was modelled using class effects If left as NULL (the default) then labels will be used as defined in the data.
...	Arguments to be sent to ggplot

Value

A forest plot of class c("gg","ggplot") that has separate panels for different time-course parameters

Examples

```
# Create an mb.network object from a dataset
network <- mb.network(alog_pcfb)

# Run an MBNMA model with an Emax time-course
emax <- mb.emax(network,
  emax=list(pool="rel", method="common"),
  et50=list(pool="rel", method="common"))

# Generate forest plot
plot(emax)

# Plot results for only one time-course parameter
plot(emax, params="d.emax")
```

---

predict.mbnma *Predict responses over time in a given population based on MBNMA time-course models*

Description

Used to predict responses over time for different treatments or to predict the results of a new study. For MBNMA models that include consistency relative effects on time-course parameters, this is calculated by combining relative treatment effects with a given reference treatment response (specific to the population of interest).

Usage

```
## S3 method for class 'mbnma'
predict(object, times = c(0:max(object$model$data()$time,
  na.rm = TRUE)), E0 = 0, treats = NULL, ref.resp = NULL,
  synth = "fixed", ...)
```

Arguments

`object`	An S3 object of class `"mbnma"` generated by running a time-course MBNMA model
`times`	A sequence of positive numbers indicating which time points to predict mean responses for
`E0`	An object to indicate the value(s) to use for the response at time = 0 in the prediction. This can take a number of different formats depending on how it will be used/calculated. The default is `0` but this may lead to non-sensical predictions. • `numeric()` A single numeric value representing the deterministic response at time = 0, given. • `character()` A single string representing a stochastic distribution for the response at time = 0. This is specified as a random number generator (RNG) given as a string, and can take any RNG distribution for which a function exists in R. For example: `"rnorm(n,7,0.5)"`.
`treats`	A character vector of treatment names or a numeric vector of treatment codes (as coded in `mbnma`) that indicate which treatments to calculate predictions for. If left 'NULL" then predictions will be calculated for all treatments.
`ref.resp`	An object to indicate the value(s) to use for the reference treatment response in MBNMA models in which the reference treatment response is not estimated within the model (i.e. those that model any time- course paramters using `pool="rel"`). This can take a number of different formats depending on how it will be used/calculated. There are two approaches for this: 1. The reference response can be estimated from a dataset of studies investigating the reference treatment using meta-analysis. This dataset could be a set of observational studies that are specific to the population on which to make predictions, or it could be a subset of the study arms within the MBNMA dataset that investigate the reference treatment. The data should be provided to `ref.resp` as a `data.frame()` containing the data in long format (one row per observation). See `ref.synth()` 2. Values for the reference treatment response can be assigned to different time-course parameters within the model that have been modelled using consistency relative effects (`pool="rel"`). These are given as a list, in which each named element corresponds to a time-course parameter modelled in `mbnma`. Their values can be either of the following: • `numeric()` A single numeric value representing the deterministic value of the time-course parameter in question in individuals given the reference treatment. `0` is used as the default, though this may produce nonsensical predictions as this typically assumes no effect of time on the reference treatment.

- `character()` A single string representing a stochastic distribution for the value of the time-course parameter in question. This is specified as a random number generator (RNG) given as a string, and can take any RNG distribution for which a function exists in R. For example: `"rnorm(n,-3,0.2)"`.

`synth` A character object that can take the value `"fixed"` or `"random"` that specifies the the type of pooling to use for synthesis of `ref.resp`. Using `"random"` rather than `"fixed"` for `synth` will result in wider 95% CrI for predictions.

`...` Arguments to be sent to R2jags for synthesis of the network reference treatment effect (using `ref.synth()`)

Details

`ref.resp` only needs to be specified if `mbnma` has been estimated using consistency relative effects (`pool="rel"`) for any time-course parameters, as these inform the absolute values of the network reference treatment parameters which can then be added to the relative effects to calculate specific predictions.

Value

An S3 object of class `mb.predict` that contains the following elements:

- `summary` A named list of data frames. Each data frame contains a summary of predicted responses at follow-up times specified in `times` for each treatment specified in `treats`
- `pred.mat` A named list of matrices. Each matrix contains the MCMC results of predicted responses at follow-up times specified in `times` for each treatment specified in `treats`

Examples

```
# Create an mb.network object from a dataset
network <- mb.network(osteopain)

# Run an MBNMA model with an Emax time-course
emax <- mb.emax(network,
  emax=list(pool="rel", method="common"),
  et50=list(pool="const", method="common"),
  positive.scale=TRUE)

# Predict responses using a stochastic baseline (E0) and a distribution for the
#network reference treatment
preds <- predict(emax, times=c(0:10),
  E0="rnorm(n, 7, 0.5)",
  ref.resp=list("emax"="rnorm(n, -0.5, 0.05)"))
summary(preds)

# Predict responses using the original dataset to estimate the network reference
#treatment response
paindata.ref <- osteopain[osteopain$treatname=="Placebo_0",]
preds <- predict(emax, times=c(5:15),
  E0=10,
```

```
  ref.resp=paindata.ref)
summary(preds)

# Repeat the above prediction but using a random effects meta-analysis of the
#network reference treatment response
preds <- predict(emax, times=c(5:15),
  E0=10,
  ref.resp=paindata.ref,
  synth="random")
summary(preds)
```

`print.mb.network` *Print mb.network information to the console*

Description

Print mb.network information to the console

Usage

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

Arguments

`x`	An object of class `mb.network`.
`...`	further arguments passed to or from other methods

`print.mb.nodesplit` *Prints basic results from a node-split to the console*

Description

Prints basic results from a node-split to the console

Usage

```
## S3 method for class 'mb.nodesplit'
print(x, groupby = "time.param", ...)
```

Arguments

x	An object of class "mb.nodesplit" generated by mb.nodeplit()
groupby	A character object that can take the value "time.param" to present results grouped by time-course parameter (the default) or "comparison" to present results grouped by treatment comparison.
...	further arguments passed to or from other methods

---

print.mb.predict *Print summary information from an mb.predict object*

Description

Print summary information from an mb.predict object

Usage

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

Arguments

x	An object of class("mb.predict") generated by predict.mbnma()
...	further arguments passed to or from other methods

---

print.mb.rank *Prints a summary of rankings for each parameter*

Description

Prints a summary of rankings for each parameter

Usage

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

Arguments

x	An object of class "mb.rank" generated by rank.mbnma()
...	further arguments passed to or from other methods

radian.rescale *Calculate position of label with respect to vertex location within a circle*

Description

Useful for graphs drawn using `igraph` to reposition labels relative to vertices when vertices are laid out in a circle (as is common in network plots). `igraph` interprets position within `vertex.label.degree` as radians, so it is necessary to convert locations into radian values. This is the main role of this function.

Usage

```
radian.rescale(x, start = 0, direction = 1)
```

Arguments

`x`	A numeric vector of positions around a circle, typically sequentially numbered.
`start`	A number giving the offset from 12 o'clock in radians for the label locations.
`direction`	Either `1` for clockwise numbering (based on the order of `x`) or `-1` for anti-clockwise.

References

https://gist.github.com/kjhealy/834774/a4e677401fd6e4c319135dabeaf9894393f9392c

Examples

```
MBNMAtime:::radian.rescale(c(1:10), start=0, direction=1)
```

rank *Set rank as a method*

Description

Set rank as a method

Usage

```
rank(x, ...)
```

Arguments

`x`	An object on which to apply the rank method
`...`	Arguments to be passed to methods

rank.mbnma *Rank parameters from a time-course MBNMA*

Description

Ranks desired parameters saved from a time-course MBNMA model from "best" to "worst".

Usage

```
## S3 method for class 'mbnma'
rank(x, params = "auc", direction = 1, treats = NULL,
  int.range = NULL, level = "treatment",
  n.iter = x$BUGSoutput$n.sims, ...)
```

Arguments

x	An object of class "mb.predict" generated by predict("mbnma")
params	A character vector containing any model parameters monitored in mbnma for which ranking is desired (e.g. "beta.1", "d.emax"). Parameters must vary by treatment for ranking to be possible. Can include "auc" (see details).
direction	Indicates whether positive responses are better (taking the value 1) or negative responses are better (taking the value -1)
treats	A character vector of treatment/class names (depending on the value of level) or a numeric vector of treatment/class codes (as coded in mbnma) that indicate which treatments/classes to calculate rankings for. If left 'NULL" then rankings will be calculated for all treatments/classes.
int.range	A numeric vector with two elements that indicates the range over which to calculate AUC. Takes the form c(lower bound, upper bound). If left as NULL (the default) then the range will be between zero and the maximum follow-up time in the data for the treatments specified in treats.
level	A character object to indicate whether the parameters to be ranked are at the treatment level ("treatment") or class level ("class").
n.iter	The number of iterations for which to calculate AUC (if "auc" is included in params). Must be a positive integer. Default is the value used in mbnma.
...	Arguments to be sent to integrate()

Details

"auc" can be included in params to rank treatments based on Area Under the Curve (AUC). This accounts for the effect of multiple time-course parameters simultaneously on the treatment response, but will be impacted by the range of time over which AUC is calculated (int.range). Currently "auc" cannot be ranked for class effect models.

As with other post-estimation functions, rank() should only be performed on models which have successfully converged. Note that rankings can be very sensitive to even small changes in treatment effects and therefore failure to converge in only one parameter may have substantial impact on rankings.

Value

A named list whose elements correspond to parameters given in `params`. Each element contains:

- `summary.rank` A data frame containing mean, sd, and quantiles for the ranks of each treatment given in `treats`
- `prob.matrix` A matrix of the proportions of MCMC results for which each treatment in `treats` ranked in which position for the given parameter
- `rank.matrix` A matrix of the ranks of MCMC results for each treatment in `treats` for the given parameter.

Examples

```
# Create an mb.network object from a dataset
network <- mb.network(alog_pcfb)

# Run an MBNMA model with an Emax time-course
emax <- mb.emax(network,
  emax=list(pool="rel", method="common"),
  et50=list(pool="rel", method="random"))

# Rank treatments by time-course parameter from the model with lower scores being better
rank(emax, params=c("d.emax", "d.et50"), direction=-1)

# Rank treatments by AUC
rank(emax, params="auc", treats=c(1:3), direction=-1,
  int.range=c(0,20))
```

`rankauc` *Calculates ranking probabilities for AUC from a time-course MBNMA*

Description

Calculates ranking probabilities for AUC from a time-course MBNMA

Usage

```
rankauc(mbnma, decreasing = FALSE, treats = NULL, int.range = NULL,
  n.iter = mbnma$BUGSoutput$n.sims, ...)
```

Arguments

`mbnma`	An S3 object of class `"mbnma"` generated by running a time-course MBNMA model
`decreasing`	A boolean object to indicate whether higher values are better (`decreasing=TRUE`) or worse (`decreasing=FALSE`).

treats	A character vector of treatment/class names (depending on the value of level). If left NULL`` then rankings will be calculated for all treatments/classes. Note that unlike rank.mbnma()‘ this argument cannot take a numeric vector.
int.range	A numeric vector with two elements that indicates the range over which to calculate AUC. Takes the form c(lower bound, upper bound). If left as NULL (the default) then the range will be between zero and the maximum follow-up time in the data for the treatments specified in treats.
n.iter	The number of iterations for which to calculate AUC (if "auc" is included in params). Must be a positive integer. Default is the value used in mbnma.
...	Arguments to be sent to integrate()

Details

"auc" can be included in params to rank treatments based on Area Under the Curve (AUC). This accounts for the effect of multiple time-course parameters simultaneously on the treatment response, but will be impacted by the range of time over which AUC is calculated (int.range). Currently "auc" cannot be ranked for class effect models.

As with other post-estimation functions, rank() should only be performed on models which have successfully converged. Note that rankings can be very sensitive to even small changes in treatment effects and therefore failure to converge in only one parameter may have substantial impact on rankings.

Value

A named list whose elements correspond to parameters given in params. Each element contains:

- summary.rank A data frame containing mean, sd, and quantiles for the ranks of each treatment given in treats
- prob.matrix A matrix of the proportions of MCMC results for which each treatment in treats ranked in which position for the given parameter
- rank.matrix A matrix of the ranks of MCMC results for each treatment in treats for the given parameter.

---

ref.comparisons *Identify unique comparisons relative to study reference treatment within a network*

Description

Identify unique contrasts relative to each study reference within a network. Repetitions of the same treatment comparison are grouped together.

Usage

```
ref.comparisons(data)
```

Arguments

data	A data frame containing variables studyID and treatment (as numeric codes) that indicate which treatments are used in which studies.

Value

A data frame of unique comparisons in which each row represents a different comparison. t1 and t2 indicate the treatment codes that make up the comparison. nr indicates the number of times the given comparison is made within the network.

If there is only a single observation for each study within the dataset (i.e. as for standard network meta-analysis) nr will represent the number of studies that compare treatments t1 and t2.

If there are multiple observations for each study within the dataset (as in MBNMAtime) nr will represent the number of time points in the dataset in which treatments t1 and t2 are compared.

Examples

```
data <- data.frame(studyID=c(1,1,2,2,3,3,4,4,5,5,5),
  treatment=c(1,2,1,3,2,3,3,4,1,2,4)
  )

# Identify comparisons informed by direct and indirect evidence
MBNMAtime:::ref.comparisons(data)
```

---

ref.synth *Synthesise single arm studies with repeated observations of the same treatment over time*

Description

Synthesises single arm studies with repeated measures by applying a particular time-course function. Used in predicting mean responses from a time-course MBNMA. The same parameterisation of the time course must be used as in the MBNMA.

Usage

```
ref.synth(data.ab, mbnma, synth = "random",
  n.iter = mbnma$BUGSoutput$n.iter,
  n.burnin = mbnma$BUGSoutput$n.burnin,
  n.thin = mbnma$BUGSoutput$n.thin,
  n.chains = mbnma$BUGSoutput$n.chains, ...)
```

Arguments

data.ab	A data frame of arm-level data in "long" format containing the columns: • studyID Study identifiers • time Numeric data indicating follow-up times

- y Numeric data indicating the mean response for a given observation
- se Numeric data indicating the standard error for a given observation

mbnma	An S3 object of class "mbnma" generated by running a time-course MBNMA model
synth	A character object that can take the value "fixed" or "random" that specifies the the type of pooling to use for synthesis of ref.resp. Using "random" rather than "fixed" for synth will result in wider 95% CrI for predictions.
n.iter	number of total iterations per chain (including burn in; default: 2000)
n.burnin	length of burn in, i.e. number of iterations to discard at the beginning. Default is n.iter/2, that is, discarding the first half of the simulations. If n.burnin is 0, jags() will run 100 iterations for adaption.
n.thin	thinning rate. Must be a positive integer. Set n.thin > 1 to save memory and computation time if n.iter is large. Default is max(1,floor(n.chains * (n.iter-n.burnin) / 1000)) which will only thin if there are at least 2000 simulations.
n.chains	number of Markov chains (default: 3)
...	Arguments to be sent to R2jags for synthesis of the network reference treatment effect (using ref.synth())

Details

data.ab can be a collection of studies that closely resemble the population of interest intended for the prediction, which could be different to those used to estimate the MBNMA model, and could be include single arms of RCTs or observational studies. If other data is not available, the data used to estimate the MBNMA model can be used by selecting only the studies and arms that specify the network reference treatment responses.

Value

A list of named elements corresponding to each time-course parameter within an MBNMA model that contain the median posterior value for the network reference treatment response.

Examples

```
# Create an mb.network object from a dataset
network <- mb.network(osteopain)

# Run an MBNMA model with an Emax time-course
emax <- mb.emax(network,
  emax=list(pool="rel", method="common"),
  et50=list(pool="rel", method="random"),
  positive.scale=TRUE)

# Generate a set of studies with which to estimate the network reference treatment response
paindata.ref <- osteopain[osteopain$treatname=="Placebo_0",]

# Estimate the network reference treatment effect using fixed effects meta-analysis
```

```
ref.synth(data.ab=paindata.ref, mbnma=emax, synth="fixed")

# Estimate the network reference treatment effect using random effects meta-analysis
ref.synth(data.ab=paindata.ref, mbnma=emax, synth="random")
```

---

ref.validate *Checks the validity of ref.resp if given as data frame*

Description

Ensures `ref.resp` takes the correct form to allow for synthesis of network reference treatment response if data is provided for meta-analysis

Usage

```
ref.validate(data.ab)
```

Arguments

`data.ab`	A data frame of arm-level data in "long" format containing the columns: • `studyID` Study identifiers • `time` Numeric data indicating follow-up times • `y` Numeric data indicating the mean response for a given observation • `se` Numeric data indicating the standard error for a given observation

---

replace.prior *Replace original priors in an MBNMA model with new priors*

Description

Identical to `replace.prior()` in MBNMAdose.

Usage

```
replace.prior(priors, model = NULL, mbnma = NULL)
```

Arguments

`priors`	A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.
`model`	A character object of JAGS MBNMA model code
`mbnma`	An S3 object of class `c("mbnma","rjags")` generated by running a time-course MBNMA model.

Details

This function takes new priors, as specified by the user, and adds them to the JAGS code from an MBNMA model. New priors replace old priors in the JAGS model.

Values in `priors` can include any JAGS functions/distributions (e.g. censoring/truncation).

Value

A character object of JAGS MBNMA model code that includes the new priors in place of original priors

---

`summary.mb.nodesplit` *Takes node-split results and produces summary data frame*

Description

Takes node-split results and produces summary data frame

Usage

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

Arguments

`object`	An object of class `"mb.nodesplit"` generated by `mb.nodeplit()`
`...`	further arguments passed to or from other methods

Value

A data frame of summary node-split results with the following variables:

- `Comparison` The treatment comparison on which a node-split has been performed
- `Time.Param` The time-course parameter on which a node-split has been performed
- `Evidence` The evidence contribution for the given comparison (either "Direct" or "Indirect")
- `Median` The posterior median
- `2.5%` The lower 95
- `97.5%` The upper 95
- `p.value` The Bayesian p-value for the overlap between direct and indirect evidence for the given comparison (it will therefore have an identical value for direct and indirect evidence within a particular comparison and time-course parameter)

`summary.mb.predict` *Prints summary of mb.predict object*

Description

Prints a summary table of the mean of MCMC iterations at each time point for each treatment

Usage

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

Arguments

`object`	An object of class `"mb.predict"`
`...`	further arguments passed to or from other methods

Value

A matrix containing times at which responses have been predicted (`time`) and an additional column for each treatment for which responses have been predicted. Each row represents mean MCMC predicted responses for each treatment at a particular time.

Examples

```
# Define network
network <- mb.network(obesityBW_CFB, reference="plac")

# Run an MBNMA with a quadratic time-course function
quad <- mb.quadratic(network,
  beta.1=list(pool="rel", method="common"),
  beta.2=list(pool="rel", method="common"),
  intercept=TRUE)

# Predict responses
pred <- predict(quad, times=c(0:50), treats=c(1:5),
  ref.estimate = network$data.ab[network$data.ab$treatment==1,],
  baseline=10)

# Generate summary of predictions
summary(pred)
```

summary.mbnma *Print summary MBNMA results to the console*

Description

Print summary MBNMA results to the console

Usage

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

Arguments

object	An S3 object of class "mbnma" generated by running a time-course MBNMA model
...	further arguments passed to or from other methods

time.fun *Write time-course function component of JAGS code for MBNMA time-course models*

Description

Writes a single line of JAGS code representing the time-course function component of an MBNMA time-course model, outputted as a single character string.

Usage

```
## S3 method for class 'fun'
time(fun = "linear", user.fun = NULL, alpha = "arm",
  beta.1 = "rel.common", beta.2 = NULL, beta.3 = NULL,
  beta.4 = NULL)
```

Arguments

fun	is a character specifying a functional form to be assigned to the time-course. Options are given in details.
user.fun	A formula specifying any relationship including time and one/several of: beta.1, beta.2, beta.3, beta.4.
alpha	Refers to the baseline mean response and is a character object that can take either: • "study" to constrain baseline to be equal for all arms within a study (i index is added))

- "arm" to allow baseline to vary between arms within a study (i,k index is added)).

beta.1	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.2	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.3	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.4	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).

Value

A single character string containing JAGS model representing the time-course function component of an MBNMA time-course model, generated based on the arguments passed to the function.

---

timeplot *Plot raw responses over time by treatment or class*

Description

Plot raw responses over time by treatment or class

Usage

```
timeplot(network, level = "treatment", plotby = "arm", ...)
```

Arguments

network	An object of class mb.network.
level	A string indicating whether nodes/facets should represent treatment or class in the plot. Can be used to examine the expected impact of modelling class/agent effects.
plotby	A character object that can take either "arm" to indicate that raw responses should be plotted separately for each study arm, or "rel" to indicate that the relative effects within each study should be plotted. In this way the time-course of both the absolute effects and the relative effects can be examined.
...	Arguments to be sent to ggplot

Details

Plots can be faceted by either treatment (level="treatment") or class (level="class") to investigate similarity of treatment responses within classes/agents. Points represent observed responses and lines connect between observations within the same study and arm.

Value

The function returns an object of `class(c("gg","ggplot")`. Characteristics of the object can therefore be amended as with other plots generated by `ggplot()`. A message will indicate if data are assumed to be change from baseline (i.e. if there are no responses in the data at time=0). In this case responses will be set to 0 at time=0.

Examples

```
# Make network
network <- mb.network(goutSUA_CFB)

# Use timeplot to plot responses grouped by treatment
timeplot(network)

# Use timeplot ot plot resposes grouped by class
timeplot(network, level="class")
```

---

write.alpha *Adds sections of JAGS code for an MBNMA model that correspond to alpha parameters*

Description

Adds sections of JAGS code for an MBNMA model that correspond to alpha parameters

Usage

```
write.alpha(model, timecourse, intercept, positive.scale)
```

Arguments

`model`	A character string representing the MBNMA model in JAGS code
`timecourse`	A character object that contains JAGS code for the time-course component of the model
`intercept`	A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (`FALSE`) or not (setting it to `FALSE` removes the intercept, `alpha`, from the model).
`positive.scale`	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.

Value

A list of named elements: `model` is a character object of JAGS MBNMA model code that includes alpha parameter components of the model `timecourse` is a character object that contains JAGS code for the time-course component of the model, for which alpha will be indexed correctly

`write.beta` *Adds sections of JAGS code for an MBNMA model that correspond to beta parameters*

Description

Adds sections of JAGS code for an MBNMA model that correspond to beta parameters

Usage

```
write.beta(model, timecourse, beta.1, beta.2 = NULL, beta.3 = NULL,
  beta.4 = NULL, UME, class.effect)
```

Arguments

`model`	A character object of JAGS MBNMA model code
`timecourse`	A character object representing the time-course used in the MBNMA model
`beta.1`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.2`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.3`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`beta.4`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).
`UME`	Can take either `TRUE` or `FALSE` (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: `c("beta.1","beta.2")`.
`class.effect`	A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be (`"common"` or `"random"`). For example: `list("beta.2"="common","beta.3"="random")`.

Value

A character object of JAGS MBNMA model code that includes beta parameter components of the model

write.check *Checks validity of arguments for mb.write*

Description

Checks validity of arguments for mb.write

Usage

```
write.check(fun = "linear", user.fun = NULL, alpha = "arm",
  beta.1 = "rel.common", beta.2 = NULL, beta.3 = NULL,
  beta.4 = NULL, positive.scale = TRUE, intercept = TRUE,
  rho = NULL, covar = NULL, var.scale = NULL,
  class.effect = list(), UME = FALSE)
```

Arguments

fun	is a character specifying a functional form to be assigned to the time-course. Options are given in details.
user.fun	A formula specifying any relationship including time and one/several of: beta.1, beta.2, beta.3, beta.4.
alpha	Refers to the baseline mean response and is a character object that can take either: • "study" to constrain baseline to be equal for all arms within a study (i index is added)) • "arm" to allow baseline to vary between arms within a study (i,k index is added)).
beta.1	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.2	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.3	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.4	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
positive.scale	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.
intercept	A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (FALSE) or not (setting it to FALSE removes the intercept, alpha, from the model).

rho	The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.
covar	A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either "CS" (compound symmetry) or "AR1" (autoregressive AR1).
var.scale	A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.
class.effect	A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").
UME	Can take either TRUE or FALSE (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: c("beta.1","beta.2").

Details

Used to check if the arguments given to mb.write are valid. The function will return informative errors if arguments are mispecified and will return an object that indicates whether the arguments imply modelling a correlation between time points if it passes.

Value

A boolean object that indicates whether the arguments imply modelling correlation between time points.

---

write.cor *Adds correlation between time-course relative effects*

Description

This uses a Wishart prior as default for modelling the correlation

Usage

```
write.cor(model, var.scale = NULL, class.effect = list())
```

Arguments

model	A character object of JAGS MBNMA model code
var.scale	A numeric vector indicating the relative scale of variances between correlated time-course parameters when relative effects are modelled on more than one time-course parameter(see Details LINK). Each element of the vector refers to the relative scale of each of the time-course parameters that is modelled using relative effects.
class.effect	A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").

---

write.fract.poly *Adds additional sections of JAGS code for an MBNMA model required for models with a fractional polynomial time-course*

Description

Adds additional sections of JAGS code for an MBNMA model required for models with a fractional polynomial time-course

Usage

```
write.fract.poly(model, timecourse)
```

Arguments

model	A character object of JAGS MBNMA model code
timecourse	A character object representing the time-course used in the MBNMA model

Details

FUNCTION IS DEPRECATED

Value

A character object of JAGS MBNMA model code that includes fractional polynomial components of the model

write.inserts *Writes insert points for RegEx in MBNMA JAGS code*

Description

Writes insert points for RegEx in MBNMA JAGS code

Usage

```
write.inserts()
```

Value

A list with named elements containing character strings that match points in MBNMA JAGS code. These points can therefore be used to insert other lines of JAGS code into the correct section within the overall model code.

write.likelihood *Adds sections of JAGS code for an MBNMA model that correspond to the likelihood*

Description

Adds sections of JAGS code for an MBNMA model that correspond to the likelihood

Usage

```
write.likelihood(model, timecourse, rho = NULL, covar = NULL)
```

Arguments

model	A character object of JAGS MBNMA model code
timecourse	A character object representing the time-course used in the MBNMA model
rho	The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.
covar	A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either "CS" (compound symmetry) or "AR1" (autoregressive AR1).

Value

A character object of JAGS MBNMA model code that includes likelihood components of the model

`write.model` *Write the basic JAGS model code for MBNMA to which other lines of model code can be added*

Description

Write the basic JAGS model code for MBNMA to which other lines of model code can be added

Usage

```
write.model()
```

Value

A character object of JAGS model code

`write.piece.fract` *Adds additional sections of JAGS code for an MBNMA model required for models with a piecewise linear or fractional polynomial time-course*

Description

Adds additional sections of JAGS code for an MBNMA model required for models with a piecewise linear or fractional polynomial time-course

Usage

```
write.piece.fract(model, fun, beta.3)
```

Arguments

`model`	A character object of JAGS MBNMA model code
`fun`	is a character specifying a functional form to be assigned to the time-course. Options are given in `details`.
`beta.3`	A list with named elements `pool` and `method` that refers to time-course parameter(s) specified within the time-course function (see details).

Value

A character object of JAGS MBNMA model code that includes piecewise linear components of the model

write.piecelinear *Adds additional sections of JAGS code for an MBNMA model required for models with a piecewise linear time-course*

Description

Adds additional sections of JAGS code for an MBNMA model required for models with a piecewise linear time-course

Usage

```
write.piecelinear(model, beta.1, beta.3)
```

Arguments

model	A character object of JAGS MBNMA model code
beta.1	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.3	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).

Details

FUNCTION DEPRECATED

Value

A character object of JAGS MBNMA model code that includes piecewise linear components of the model

write.ref.synth *Write MBNMA time-course models JAGS code for synthesis of studies investigating reference treatment*

Description

Writes JAGS code for a Bayesian time-course model for model-based network meta-analysis (MBNMA) that pools reference treatment effects from different studies. This model only pools single study arms and therefore does not pool relative effects.

Usage

```
write.ref.synth(fun = "linear", user.fun = NULL, alpha = "arm",
  beta.1 = "rel.common", beta.2 = NULL, beta.3 = NULL,
  beta.4 = NULL, positive.scale = TRUE, intercept = TRUE,
  rho = NULL, covar = NULL, mu.synth = "random",
  class.effect = list(), UME = FALSE, priors = NULL)
```

Arguments

fun	is a character specifying a functional form to be assigned to the time-course. Options are given in details.
user.fun	A formula specifying any relationship including time and one/several of: beta.1, beta.2, beta.3, beta.4.
alpha	Refers to the baseline mean response and is a character object that can take either: • "study" to constrain baseline to be equal for all arms within a study (i index is added)) • "arm" to allow baseline to vary between arms within a study (i,k index is added)).
beta.1	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.2	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.3	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
beta.4	A list with named elements pool and method that refers to time-course parameter(s) specified within the time-course function (see details).
positive.scale	A boolean object that indicates whether all continuous mean responses (y) are positive and therefore whether the baseline response should be given a prior that constrains it to be positive.
intercept	A boolean object that indicates whether an intercept is to be included in the model. Can be used to imply whether mean responses in data are change from baseline (FALSE) or not (setting it to FALSE removes the intercept, alpha, from the model).
rho	The correlation coefficient when modelling correlation between time points. If left as NULL (the default) then this implies modelling no correlation between time points. Can either be assigned the string "estimate" to indicate that rho should be estimated from the data, or assigned a numeric value, which fixes rho in the model to the assigned value, either for when rho is calculated externally or for use in deterministic sensitivity analyses.
covar	A character specifying the covariance structure to use for the multivariate normal likelihood. Can currently take either "CS" (compound symmetry) or "AR1" (autoregressive AR1).
mu.synth	A string that takes the value fixed or random, indicating the type of synthesis model to use
class.effect	A list of named strings that determines which time-course parameters to model with a class effect and what that effect should be ("common" or "random"). For example: list("beta.2"="common","beta.3"="random").
UME	Can take either TRUE or FALSE (for an unrelated mean effects model on all or no time-course parameters respectively) or can be a vector of parameter name strings to model as UME. For example: c("beta.1","beta.2").
priors	A named list of parameter values (without indices) and replacement prior distribution values given as strings **using distributions as specified in JAGS syntax**.

Value

A character object of JAGS MBNMA model code that includes beta parameter components of the model

Examples

```
# Write an exponential time-course MBNMA synthesis model
model <- write.ref.synth(fun="exponential",
  alpha="arm", beta.1="rel.common", mu.synth="fixed")
cat(model) # Concatenates model representations making code more easily readable
```

---

write.remove.loops *Removes any loops from MBNMA model JAGS code that do not contain any expressions*

Description

Removes any loops from MBNMA model JAGS code that do not contain any expressions

Usage

```
write.remove.loops(model)
```

Arguments

model	A character object of JAGS MBNMA model code

Value

A character object of JAGS MBNMA model code that has had empty loops removed from it

Index