Package: bayesmeta (via r-universe)

October 13, 2024

Type Package

Title Bayesian Random-Effects Meta-Analysis and Meta-Regression

Version 3.4

Date 2024-02-15

Maintainer Christian Roever <christian.roever@med.uni-goettingen.de>

Depends forestplot (>= 2.0.0), metafor (>= 2.0-0), mvtnorm (>= 1.1-1), numDeriv (>= 2016.8-1.1)

Suggests compute.es, knitr, rmarkdown, R.rsp

Description A collection of functions allowing to derive the posterior distribution of the model parameters in random-effects meta-analysis or meta-regression, and providing functionality to evaluate joint and marginal posterior probability distributions, predictive distributions, shrinkage effects, posterior predictive p-values, etc.; For more details, see also Roever C (2020) <doi:10.18637/jss.v093.i06>, or Roever C and Friede T (2022) <doi:10.1016/j.cmpb.2022.107303>.

License GPL (>= 2)

URL https://gitlab.gwdg.de/croever/bayesmeta

VignetteBuilder knitr, R.rsp

NeedsCompilation no

Author Christian Roever [aut, cre] (<https://orcid.org/0000-0002-6911-698X>), Tim Friede [ctb] (<https://orcid.org/0000-0001-5347-7441>)

Date/Publication 2024-02-15 18:10:02 UTC

Repository https://christianroever.r-universe.dev

RemoteUrl https://github.com/cran/bayesmeta

RemoteRef HEAD

RemoteSha e7eaba2af611b8cace198c31ee107ef7bda58b55

Contents

bayesmeta-package *Bayesian Random-Effects Meta-Analysis and Meta-Regression*

Description

A collection of functions allowing to derive the posterior distribution of the model parameters in random-effects meta-analysis or meta-regression, and providing functionality to evaluate joint and marginal posterior probability distributions, predictive distributions, shrinkage effects, posterior predictive p-values, etc.

Details

Package:	bayesmeta
Type:	Package
Version:	3.4
Date:	2024-02-15
License:	GPL (>=2)

The main functionality is provided by the `bayesmeta()` and `bmr()` function. It takes the data (estimates and associated standard errors) and prior information (effect and heterogeneity priors), and returns an object containing functions that allow to derive posterior quantities like joint or marginal densities, quantiles, etc. The `bmr()` function extends the approach to meta-regression by allowing to specify covariables (moderators) in addition.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

See Also

`forestplot.bayesmeta`, `plot.bayesmeta`, `bmr`.

Examples

```
# example data by Snedecor and Cochran:
data("SnedecorCochran")

## Not run:
# analysis using improper uniform prior
# (may take a few seconds to compute!):
```

```
bma <- bayesmeta(y=SnedecorCochran[,"mean"],
                 sigma=sqrt(SnedecorCochran[,"var"]),
                 label=SnedecorCochran[,"no"])

# show some summary statistics:
bma

# show a bit more details:
summary(bma)

# show a forest plot:
forestplot(bma)

# show some more plots:
plot(bma)

## End(Not run)
```

`BaetenEtAl2013` *Ankylosing spondylitis example data*

Description

Numbers of cases (patients) and events (responders) in the placebo control groups of eight studies.

Usage

```
data("BaetenEtAl2013")
```

Format

The data frame contains the following columns:

study	`character`	study label
year	`numeric`	publication year
events	`numeric`	number of responders
total	`numeric`	total number of patients

Details

A study was conducted in order to investigate a novel treatment in ankylosing spondylitis (Baeten et al., 2013). The primary endpoint related to *treatment response*. In order to formulate an informative prior distribution for the response rate to be expected in the control group of the new study, a systematic review of previous studies was consulted (McLeod et al., 2007), and, after a meta-analysis of the estimated response probabilities, the predictive distribution for the new study's response probability was derived. The predictive distribution here constitutes the *meta-analytic-predictive (MAP) prior* distribution (Schmidli et al., 2014). The data set contains the relevant data from the eight "historical" studies' placebo groups.

Note that the original analysis (Baeten et al., 2013) involved a binomial model, and the resulting posterior predictive distribution was eventually approximated by a mixture of beta distributions.

Source

D. Baeten et al. Anti-interleukin-17A monoclonal antibody secukinumab in treatment of ankylosing spondylitis: a randomised, double-blind, placebo-controlled trial. *The Lancet*, **382**(9906):1705-1713, 2013. doi:10.1016/S01406736(13)611344.

References

C. McLeod et al. Adalimumab, etanercept, and infliximab for the treatment of ankylosing spondylitis: a systematic review and economic evaluation. *Health Technology Assessment*, **11**(28), 2007. doi:10.3310/hta11280.

H. Schmidli, S. Gsteiger, S. Roychoudhury, A. O'Hagan, D. Spiegelhalter, B. Neuenschwander. Robust meta-analytic-predictive priors in clinical trials with historical control information. *Biometrics*, **70**(4):1023-1032, 2014. doi:10.1111/biom.12242.

See Also

uisd, ess.

Examples

```
# load data:
data("BaetenEtAl2013")

# show data:
BaetenEtAl2013

## Not run:
# compute effect sizes (logarithmic odds) from the count data:
as <- escalc(xi=events, ni=total, slab=study,
             measure="PLO", data=BaetenEtAl2013)

# compute the unit information standard deviation (UISD):
uisd(as)

# perform meta-analysis
# (using uniform priors for effect and heterogeneity):
bm <- bayesmeta(as)

# show results (log-odds):
forestplot(bm, xlab="log-odds", zero=NA)
# show results (odds):
forestplot(bm, exponentiate=TRUE, xlab="odds", zero=NA)

# show posterior predictive distribution --
# in terms of log-odds:
bm$summary[,"theta"]
logodds <- bm$summary[c(2,5,6), "theta"]
logodds
# in terms of odds:
exp(logodds)
# in terms of probabilities:
```

```
(exp(logodds) / (exp(logodds) + 1))

# illustrate MAP prior density:
x <- seq(-3, 1, by=0.01)
plot(x, bm$dposterior(theta=x, predict=TRUE), type="l",
     xlab="log-odds (response)", ylab="posterior predictive density")
abline(h=0, v=0, col="grey")

## End(Not run)
```

---

bayesmeta *Bayesian random-effects meta-analysis*

Description

This function allows to derive the posterior distribution of the two parameters in a random-effects meta-analysis and provides functions to evaluate joint and marginal posterior probability distributions, etc.

Usage

```
  bayesmeta(y, ...)
  ## Default S3 method:
bayesmeta(y, sigma, labels = names(y),
          tau.prior = "uniform",
          mu.prior = c("mean"=NA,"sd"=NA),
          mu.prior.mean = mu.prior[1], mu.prior.sd = mu.prior[2],
          interval.type = c("shortest", "central"),
          delta = 0.01, epsilon = 0.0001,
          rel.tol.integrate = 2^16*.Machine$double.eps,
          abs.tol.integrate = 0.0,
          tol.uniroot = rel.tol.integrate, ...)
  ## S3 method for class 'escalc'
bayesmeta(y, labels = NULL, ...)
```

Arguments

y	vector of estimates *or* an escalc object.
sigma	vector of standard errors associated with y.
labels	(optional) a vector of labels corresponding to y and sigma.
tau.prior	a function returning the prior density for the heterogeneity parameter (τ) *or* a character string specifying one of the *default 'non-informative' priors*; possible choices for the latter case are: • "uniform": a uniform prior in τ • "sqrt": a uniform prior in $\sqrt{\tau}$ • "Jeffreys": the Jeffreys prior for τ

- "BergerDeely": the prior due to Berger and Deely (1988)
- "conventional": the conventional prior
- "DuMouchel": the DuMouchel prior
- "shrinkage": the 'uniform shrinkage' prior
- "I2": a uniform prior on the 'relative heterogeneity' I^2

The default is "uniform" (which should be used with caution; see remarks below). The above priors are described in some more detail below.

`mu.prior` the mean and standard deviation of the normal prior distribution for the effect μ. If unspecified, an (improper) uniform prior is used.

`mu.prior.mean`, `mu.prior.sd`
alternative parameters to specify the prior distribution for the effect μ (see above).

`interval.type` the type of (credible, prediction, shrinkage) interval to be returned by default; either "shortest" for shortest intervals, or "central" for central, equal-tailed intervals.

`delta`, `epsilon` the parameters specifying the desired accuracy for approximation of the μ posterior(s), and with that determining the number of τ support points being used internally. Smaller values imply greater accuracy and greater computational burden (Roever and Friede, 2017).

`rel.tol.integrate`, `abs.tol.integrate`, `tol.uniroot`
the `rel.tol`, `abs.tol` and `tol` 'accuracy' arguments that are passed to the `integrate()` or `uniroot()` functions for internal numerical integration or root finding (see also the help there).

`...` other `bayesmeta` arguments.

Details

The common random-effects meta-analysis model may be stated as

$$y_i|\mu,\sigma_i,\tau \;\sim\; \mathrm{Normal}(\mu,\, \sigma_i^2+\tau^2)$$

where the data (y, σ) enter as y_i, the i-th estimate, that is associated with standard error $\sigma_i > 0$, and $i = 1, ..., k$. The model includes two unknown parameters, namely the (mean) effect μ, and the heterogeneity τ. Alternatively, the model may also be formulated via an intermediate step as

$$y_i|\theta_i,\sigma_i \;\sim\; \mathrm{Normal}(\theta_i,\, \sigma_i^2),$$

$$\theta_i|\mu,\tau \;\sim\; \mathrm{Normal}(\mu,\, \tau^2),$$

where the θ_i denote the *trial-specific* means that are then measured through the estimate y_i with an associated measurement uncertainty σ_i. The θ_i again differ from trial to trial and are distributed around a common mean μ with standard deviation τ.

The `bayesmeta()` function utilizes the fact that the joint posterior distribution $p(\mu,\tau|y,\sigma)$ may be partly analytically integrated out to determine the integrals necessary for coherent Bayesian inference on one or both of the parameters.

As long as we assume that the prior probability distribution may be factored into independent marginals $p(\mu,\tau) = p(\mu) \times p(\tau)$ and either an (improper) uniform or a normal prior is used for the effect μ, the joint likelihood $p(y|\mu,\tau)$ may be analytically marginalized over μ, yielding the

marginal likelihood function $p(y|\tau)$. Using any prior $p(\tau)$ for the heterogeneity, the 1-dimensional marginal posterior $p(\tau|y) \propto p(y|\tau) \times p(\tau)$ may then be treated numerically. As the *conditional* posterior $p(\mu|\tau, y)$ is a normal distribution, inference on the remaining joint ($p(\mu, \tau|y)$) or marginal ($p(\mu|y)$) posterior may be approached numerically (using the DIRECT algorithm) as well. Accuracy of the computations is determined by the `delta` (maximum divergence δ) and `epsilon` (tail probability ϵ) parameters (Roever and Friede, 2017).

What constitutes a sensible prior for both μ and τ depends (as usual) very much on the context. Potential candidates for informative (or weakly informative) heterogeneity (τ) priors may include the *half-normal*, *half-Student-t*, *half-Cauchy*, *exponential*, or *Lomax* distributions; we recommend the use of heavy-tailed prior distributions if in case of prior/data conflict the prior is supposed to be discounted (O'Hagan and Pericchi, 2012). A sensible informative prior might also be a *log-normal* or a *scaled inverse* χ^2 distribution. One might argue that an uninformative prior for τ may be uniform or monotonically decreasing in τ. Another option is to use the *Jeffreys prior* (see the `tau.prior` argument above). The Jeffreys prior implemented here is the variant also described by Tibshirani (1989) that results from fixing the location parameter (μ) and considering the Fisher information element corresponding to the heterogeneity τ only. This prior also constitutes the *Berger/Bernardo reference prior* for the present problem (Bodnar *et al.*, 2016). The *uniform shrinkage* and *DuMouchel* priors are described in Spiegelhalter *et al.* (2004, Sec. 5.7.3). The procedure is able to handle improper priors (and does so by default), but of course the usual care must be taken here, as the resulting posterior *may* or *may not* be a proper probability distribution.

Note that a wide range of different types of endpoints may be treated on a continuous scale after an appropriate transformation; for example, count data may be handled by considering corresponding logarithmic odds ratios. Many such transformations are implemented in the **metafor** package's `escalc` function and may be directly used as an input to the `bayesmeta()` function; see also the example below. Alternatively, the **compute.es** package also provides a range of effect sizes to be computed from different data types.

The `bayesmeta()` function eventually generates some basic summary statistics, but most importantly it provides an object containing a range of `functions` allowing to evaluate posterior distributions; this includes joint and marginal posteriors, prior and likelihood, predictive distributions, densities, cumulative distributions and quantile functions. For more details see also the documentation and examples below. Use of the `individual` argument allows to access posteriors of study-specific (*shrinkage*-) effects (θ_i). The `predict` argument may be used to access the predictive distribution of a future study's effect (θ_{k+1}), facilitating a *meta-analytic-predictive (MAP)* approach (Neuenschwander *et al.*, 2010).

Prior specification details: When specifying the `tau.prior` argument as a `character` string (and not as a prior density `function`), then the actual prior probability density functions corresponding to the possible choices of the `tau.prior` argument are given by:

- `"uniform"` - the (improper) uniform prior in τ:

$$p(\tau) \propto 1$$

- `"sqrt"` - the (improper) uniform prior in $\sqrt{\tau}$:

$$p(\tau) \propto \tau^{-1/2} = \frac{1}{\sqrt{\tau}}$$

- `"Jeffreys"` - *Tibshirani's noninformative prior*, a variation of the ('general' or 'overall')

Jeffreys prior, which here also constitutes the *Berger/Bernardo reference prior* for τ:

$$p(\tau) \;\propto\; \sqrt{\sum_{i=1}^{k}\Big(\frac{\tau}{\sigma_i^2+\tau^2}\Big)^2}$$

This is also an improper prior whose density does not integrate to 1. This prior results from applying *Jeffreys' general rule* (Kass and Wasserman, 1996), and in particular considering that location parameters (here: the effect μ) should be treated separately (Roever, 2020).

- `"overallJeffreys"` - the 'general' or 'overall' form of the *Jeffreys prior*; this is derived based on the Fisher information terms corresponding to *both* the effect (μ) and heterogeneity (τ). The resulting (improper) prior density is

$$p(\tau) \;\propto\; \sqrt{\sum_{i=1}^{k}\frac{1}{\sigma_i^2+\tau^2}\;\sum_{i=1}^{k}\Big(\frac{\tau}{\sigma_i^2+\tau^2}\Big)^2}$$

  Use of this specification is generally *not* recommended; see, e.g., Jeffreys (1946), Jeffreys (1961, Sec. III.3.10), Berger (1985, Sec. 3.3.3) and Kass and Wasserman (1996, Sec. 2.2). Since the effect μ constitutes a *location parameter* here, it should be treated separately (Roever, 2020).
- `"BergerDeely"` - the (improper) *Berger/Deely* prior:

$$p(\tau) \;\propto\; \prod_{i=1}^{k}\Big(\frac{\tau}{\sigma_i^2+\tau^2}\Big)^{1/k}$$

  This is a variation of the above *Jeffreys* prior, and both are equal in case all standard errors (σ_i) are the same.
- `"conventional"` - the (proper) *conventional prior*:

$$p(\tau) \;\propto\; \prod_{i=1}^{k}\Big(\frac{\tau}{(\sigma_i^2+\tau^2)^{3/2}}\Big)^{1/k}$$

  This is a proper variation of the above *Berger/Deely* prior intended especially for testing and model selection purposes.
- `"DuMouchel"` - the (proper) *DuMouchel* prior:

$$p(\tau) \;=\; \frac{s_0}{(s_0+\tau)^2}$$

  where $s_0=\sqrt{s_0^2}$ and s_0^2 again is the harmonic mean of the standard errors (as above).
- `"shrinkage"` - the (proper) *uniform shrinkage* prior:

$$p(\tau) \;=\; \frac{2s_0^2\tau}{(s_0^2+\tau^2)^2}$$

  where $s_0^2=\frac{k}{\sum_{i=1}^{k}\sigma_i^{-2}}$ is the harmonic mean of the squared standard errors σ_i^2.
- `"I2"` - the (proper) uniform prior in I^2:

$$p(\tau) \;=\; \frac{2\hat{\sigma}^2\tau}{(\hat{\sigma}^2+\tau^2)^2}$$

where $\hat{\sigma}^2 = \frac{(k-1)\ \sum_{i=1}^{k} \sigma_i^{-2}}{\left(\sum_{i=1}^{k} \sigma_i^{-2}\right)^2 - \sum_{i=1}^{k} \sigma_i^{-4}}$. This prior is similar to the uniform shrinkage prior, except for the use of $\hat{\sigma}^2$ instead of s_0^2.

For more details on the above priors, see Roever (2020). For more general information especially on (weakly) informative heterogeneity prior distributions, see Roever *et al.* (2021). Regarding empirically motivated informative heterogeneity priors see also the `TurnerEtAlPrior()` and `RhodesEtAlPrior()` functions, or Roever *et al.* (2023) and Lilienthal *et al.* (2023).

Credible intervals: Credible intervals (as well as prediction and shrinkage intervals) may be determined in different ways. By default, *shortest* intervals are returned, which for unimodal posteriors (the usual case) is equivalent to the *highest posterior density region* (Gelman *et al.*, 1997, Sec. 2.3). Alternatively, central (equal-tailed) intervals may also be derived. The default behaviour may be controlled via the `interval.type` argument, or also by using the `method` argument with each individual call of the `$post.interval()` function (see below). A third option, although not available for prediction or shrinkage intervals, and hence not as an overall default choice, but only for the `$post.interval()` function, is to determine the *evidentiary* credible interval, which has the advantage of being parameterization invariant (Shalloway, 2014).

Bayes factors: Bayes factors (Kass and Raftery, 1995) for the two hypotheses of $\tau = 0$ and $\mu = 0$ are provided in the `$bayesfactor` element; *low* or *high* values here constitute evidence *against* or *in favour of* the hypotheses, respectively. Bayes factors are based on marginal likelihoods and can only be computed if the priors for heterogeneity and effect are proper. Bayes factors for other hypotheses can be computed using the marginal likelihood (as provided through the `$marginal` element) and the `$likelihood()` function. Bayes factors must be interpreted with very much caution, as they are susceptible to *Lindley's paradox* (Lindley, 1957), which especially implies that variations of the prior specifications that have only minuscule effects on the posterior distribution may have a substantial impact on Bayes factors (via the marginal likelihood). For more details on the problems and challenges related to Bayes factors see also Gelman *et al.* (1997, Sec. 7.4).

Besides the 'actual' Bayes factors, *minimum Bayes factors* are also provided (Spiegelhalter *et al.*, 2004; Sec. 4.4). The minimum Bayes factor for the hypothesis of $\mu = 0$ constitutes the minimum across all possible priors for μ and hence gives a measure of how much (or how little) evidence *against* the hypothesis is provided by the data *at most*. It is independent of the particular effect prior used in the analysis, but still dependent on the heterogeneity prior. Analogously, the same is true for the heterogeneity's minimum Bayes factor. A minimum Bayes factor can also be computed when only one of the priors is proper.

Numerical accuracy: Accuracy of the numerical results is determined by four parameters, namely, the accuracy of numerical integration as specified through the `rel.tol.integrate` and `abs.tol.integrate` arguments (which are internally passed on to the `integrate` function), and the accuracy of the grid approximation used for integrating out the heterogeneity as specified through the `delta` and `epsilon` arguments (Roever and Friede, 2017). As these may also heavily impact on the computation time, be careful when changing these from their default values. You can monitor the effect of different settings by checking the number and range of support points returned in the `$support` element.

Study weights: Conditional on a given τ value, the posterior expectations of the overall effect (μ) as well as the shrinkage estimates (θ_i) result as convex combinations of the estimates y_i. The *weights* associated with each estimate y_i are commonly quoted in frequentist meta-analysis

results in order to quantify (arguably somewhat heuristically) each study's contribution to the overall estimates, often in terms of percentages.

In a Bayesian meta-analysis, these numbers to not immediately arise, since the heterogeneity is marginalized over. However, due to linearity, the posterior mean effects may still be expressed in terms of linear combinations of initial estimates y_i, with weights now given by the *posterior mean weights*, marginalized over the heterogeneity τ (Roever and Friede, 2020). The posterior mean weights are returned in the `$weights` and `$weights.theta` elements, for the overall effect μ as well as for the shrinkage estimates θ_i.

Value

A list of class `bayesmeta` containing the following elements:

`y`	vector of estimates (the input data).
`sigma`	vector of standard errors corresponding to `y` (input data).
`labels`	vector of labels corresponding to `y` and `sigma`.
`k`	number of data points (in `y`).
`tau.prior`	the prior probability density function for τ.
`mu.prior.mean`	the prior mean of μ.
`mu.prior.sd`	the prior standard deviation of μ.
`dprior`	a `function(tau=NA, mu=NA, log=FALSE)` of two parameters, `tau` and/or `mu`, returning either the joint or marginal prior probability density, depending on which parameter(s) is/are provided.
`tau.prior.proper`	a `logical` flag indicating whether the heterogeneity prior appears to be proper (which is judged based on an attempted numerical integration of the density function).
`likelihood`	a `function(tau=NA, mu=NA, log=FALSE)` of two parameters, `tau` and/or `mu`, returning either the joint or marginal likelihood, depending on which parameter(s) is/are provided.
`dposterior`	a `function(tau=NA, mu=NA, theta=mu, log=FALSE, predict=FALSE, individual=FALSE)` of two parameters, `tau` and/or `mu`, returning either the joint or marginal posterior probability density, depending on which parameter(s) is/are provided. Using the argument `predict=TRUE` yields the *posterior predictive distribution* for θ. Using the `individual` argument, you can request individual effects' (θ_i) posterior distributions. May be an integer number (`1,...,k`) giving the index, or a character string giving the label.
`pposterior`	a `function(tau=NA, mu=NA, theta=mu, predict=FALSE, individual=FALSE)` of one parameter (either `tau` or `mu`) returning the corresponding marginal posterior cumulative distribution function. Using the argument `predict=TRUE` yields the posterior predictive distribution for θ. Using the `individual` argument, you can request individual effects' (θ_i) posterior distributions. May be an integer number (`1,...,k`) giving the index, or a character string giving the label.
`qposterior`	a `function(tau.p=NA, mu.p=NA, theta.p=mu.p, predict=FALSE, individual=FALSE)` of one parameter (either `tau.p` or `mu.p`) returning the corresponding marginal

posterior quantile function. Using the argument `predict=TRUE` yields the posterior predictive distribution for θ. Using the `individual` argument, you can request individual effects' (θ_i) posterior distributions. May be an integer number (`1`, ..., `k`) giving the index, or a character string giving the label.

`rposterior` a `function(n=1, predict=FALSE, individual=FALSE, tau.sample=TRUE)` generating `n` independent random draws from the (2-dimensional) posterior distribution. Using the argument `predict=TRUE` yields the posterior predictive distribution for θ. Using the `individual` argument, you can request individual effects' (θ_i) posterior distributions. May be an integer number (`1`, ..., `k`) giving the index, or a character string giving the label. In general, this via the inversion method, so it is rather slow. However, if one is not interested in sampling the heterogeneity parameter (τ), using '`tau.sample=FALSE`' will speed up the function substantially.

`post.interval` a `function(tau.level=NA, mu.level=NA, theta.level=mu.level, method=c("shortest","centr predict=FALSE, individual=FALSE)` returning a credible interval, depending on which of the two parameters is provided (either `tau.level` or `mu.level`). The additional parameter `method` may be used to specify the desired type of interval: `method = "shortest"` returns the shortest interval, `method = "central"` returns a central interval, and `method = "evidentiary"` returns an evidentiary interval (Shalloway, 2014); the former is the default option. Using the argument `predict=TRUE` yields a posterior predictive interval for θ. Using the `individual` argument, you can request individual effects' (θ_i) posterior distributions. May be an integer number (`1`, ..., `k`) giving the index, or a character string giving the label.

`cond.moment` a `function(tau, predict=FALSE, individual=FALSE, simplify=TRUE)` returning conditional moments (mean and standard deviation) of μ as a function of τ. Using the argument `predict=TRUE` yields (conditional) posterior predictive moments for θ. Using the `individual` argument, you can request individual effects' (θ_i) posterior distributions. May be an integer number (`1`, ..., `k`) giving the index, or a character string giving the label.

`I2` a `function(tau)` returning the 'relative' heterogeneity I^2 as a function of τ.

`summary` a `matrix` listing some summary statistics, namely marginal posterior mode, median, mean, standard deviation and a (shortest) 95% credible intervals, of the marginal posterior distributions of τ and μ, and of the posterior predictive distribution of θ.

`interval.type` the `interval.type` input argument specifying the type of interval to be returned by default.

`ML` a `matrix` giving joint and marginal maximum-likelihood estimates of (τ, μ).

`MAP` a `matrix` giving joint and marginal maximum-a-posteriori estimates of (τ, μ).

`theta` a `matrix` giving the 'shrinkage estimates', i.e, summary statistics of the trial-specific means θ_i.

`weights` a `vector` giving the posterior expected *inverse-variance weights* for each study (and for the effect prior mean, if the effect prior was proper).

`weights.theta` a `matrix` whose columns give the posterior expected weights of each study (and of the effect prior mean, if the effect prior was proper) for all shrinkage estimates.

`marginal.likelihood`
: the marginal likelihood of the data (this number is only computed if proper effect and heterogeneity priors are specified).

`bayesfactor`
: Bayes factors and minimum Bayes factors for the two hypotheses of $\tau = 0$ and $\mu = 0$. These depend on the marginal likelihood and hence can only be computed if proper effect and/or heterogeneity priors are specified; see also remark above.

`support`
: a `matrix` giving the τ support points used internally in the grid approximation, along with their associated weights, conditional mean and standard deviation of μ, and the standard deviation of the (conditional) predictive distribution of θ.

`delta`, `epsilon`
: the '`delta`' and '`epsilon`' input parameter determining numerical accuracy.

`rel.tol.integrate`, `abs.tol.integrate`, `tol.uniroot`
: the input parameters determining the numerical accuracy of the internally used `integrate()` and `uniroot()` functions.

`call`
: an object of class `call` giving the function call that generated the `bayesmeta` object.

`init.time`
: the computation time (in seconds) used to generate the `bayesmeta` object.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

C. Roever, T. Friede. Discrete approximation of a mixture distribution via restricted divergence. *Journal of Computational and Graphical Statistics*, **26**(1):217-222, 2017. doi:10.1080/10618600.2016.1276840.

C. Roever, T. Friede. Bounds for the weight of external data in shrinkage estimation. *Biometrical Journal*, **65**(5):1131-1143, 2021. doi:10.1002/bimj.202000227.

J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. 2nd edition. Springer-Verlag, 1985. doi:10.1007/9781475742862.

J.O. Berger, J. Deely. A Bayesian approach to ranking and selection of related means with alternatives to analysis-of-variance methodology. *Journal of the American Statistical Association*, **83**(402):364-373, 1988. doi:10.1080/01621459.1988.10478606.

O. Bodnar, A. Link, C. Elster. Objective Bayesian inference for a generalized marginal random effects model. *Bayesian Analysis*, **11**(1):25-45, 2016. doi:10.1214/14BA933.

A. Gelman, J.B. Carlin, H.S. Stern, D.B. Rubin. *Bayesian data analysis*. Chapman & Hall / CRC, Boca Raton, 1997.

A. Gelman. Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*, **1**(3):515-534, 2006. doi:10.1214/06BA117A.

J. Hartung, G. Knapp, B.K. Sinha. *Statistical meta-analysis with applications*. Wiley, Hoboken, 2008.

L.V. Hedges, I. Olkin. *Statistical methods for meta-analysis*. Academic Press, San Diego, 1985

H. Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of london, Series A*, **186**(1007):453-462, 1946. doi:10.1098/rspa.1946.0056.

H. Jeffreys. *Theory of Probability*. 3rd edition. Clarendon Press, Oxford, 1961.

A.E. Kass, R.E. Raftery. Bayes factors. *Journal of the American Statistical Association*, **90**(430):773-795, 1995. doi:10.2307/2291091.

A.E. Kass, L. Wasserman. The selection of prior distributions by formal rules. *Journal of the American Statistical Association*. **91**(453):1243-1370, 1996. doi:10.1080/01621459.1996.10477003.

J. Lilienthal, S. Sturtz, C. Schuermann, M. Maiworm, C. Roever, T. Friede, R. Bender. Bayesian random-effects meta-analysis with empirical heterogeneity priors for application in health technology assessment with very few studies. *Research Synthesis Methods*, 2023. doi:10.1002/jrsm.1685.

D.V. Lindley. A statistical paradox. *Biometrika*, **44**(1/2):187-192, 1957. doi:10.1093/biomet/44.1-2.187.

B. Neuenschwander, G. Capkun-Niggli, M. Branson, D.J. Spiegelhalter. Summarizing historical information on controls in clinical trials. *Trials*, **7**(1):5-18, 2010. doi:10.1177/1740774509356002.

A. O'Hagan, L. Pericchi. Bayesian heavy-tailed models and conflict resolution: A review. *Brazilian Journal of Probability and Statistics*, **26**(4):372-401, 2012. doi:10.1214/11BJPS164.

C. Roever, S. Sturtz, J. Lilienthal, R. Bender, T. Friede. Summarizing empirical information on between-study heterogeneity for Bayesian random-effects meta-analysis. *Statistics in Medicine*, **42**(14):2439-2454, 2023. doi:10.1002/sim.9731.

S. Shalloway. The evidentiary credible region. *Bayesian Analysis*, **9**(4):909-922, 2014. doi:10.1214/14BA883.

D.J. Spiegelhalter, K.R. Abrams, J.P.Myles. *Bayesian approaches to clinical trials and health-care evaluation*. Wiley & Sons, 2004.

R. Tibshirani. Noninformative priors for one parameter of many. *Biometrika*, **76**(3):604-608, 1989. doi:10.1093/biomet/76.3.604.

W. Viechtbauer. Conducting meta-analyses in R with the metafor package. *Journal of Statistical Software*, **36**(3):1-48, 2010. doi:10.18637/jss.v036.i03.

See Also

`forestplot.bayesmeta`, `plot.bayesmeta`, `escalc`, `bmr`, `compute.es`.

Examples

```
########################################
# example data by Snedecor and Cochran:
data("SnedecorCochran")

## Not run:
# analysis using improper uniform prior
# (may take a few seconds to compute!):
ma01 <- bayesmeta(y=SnedecorCochran[,"mean"], sigma=sqrt(SnedecorCochran[,"var"]),
```

```
                  label=SnedecorCochran[,"no"])

# analysis using an informative prior
# (normal for mu and half-Cauchy for tau (scale=10))
# (may take a few seconds to compute!):
ma02 <- bayesmeta(y=SnedecorCochran[,"mean"], sigma=sqrt(SnedecorCochran[,"var"]),
                  label=SnedecorCochran[,"no"],
                  mu.prior.mean=50, mu.prior.sd=50,
                  tau.prior=function(x){return(dhalfcauchy(x, scale=10))})

# show some summary statistics:
print(ma01)
summary(ma01)

# show some plots:
forestplot(ma01)
plot(ma01)

# compare resulting marginal densities;
# the effect parameter (mu):
mu <- seq(30, 80, le=200)
plot(mu, ma02$dposterior(mu=mu), type="l", lty="dashed",
     xlab=expression("effect "*mu),
     ylab=expression("marginal posterior density"),
     main="Snedecor/Cochran example")
lines(mu, ma01$dposterior(mu=mu), lty="solid")

# the heterogeneity parameter (tau):
tau <- seq(0, 50, le=200)
plot(tau, ma02$dposterior(tau=tau), type="l", lty="dashed",
     xlab=expression("heterogeneity "*tau),
     ylab=expression("marginal posterior density"),
     main="Snedecor/Cochran example")
lines(tau, ma01$dposterior(tau=tau), lty="solid")

# compute posterior median relative heterogeneity I-squared:
ma01$I2(tau=ma01$summary["median","tau"])

# determine 90 percent upper limits on the heterogeneity tau:
ma01$qposterior(tau=0.90)
ma02$qposterior(tau=0.90)
# determine shortest 90 percent credible interval for tau:
ma01$post.interval(tau.level=0.9, method="shortest")
## End(Not run)


####################################
# example data by Sidik and Jonkman:
data("SidikJonkman2007")
# add log-odds-ratios and corresponding standard errors:
sj <- SidikJonkman2007
sj <- cbind(sj, "log.or"=log(sj[,"lihr.events"])-log(sj[,"lihr.cases"]-sj[,"lihr.events"])
                    -log(sj[,"oihr.events"])+log(sj[,"oihr.cases"]-sj[,"oihr.events"]),
```

```
           "log.or.se"=sqrt(1/sj[,"lihr.events"] + 1/(sj[,"lihr.cases"]-sj[,"lihr.events"])
                      + 1/sj[,"oihr.events"] + 1/(sj[,"oihr.cases"]-sj[,"oihr.events"])))

## Not run:
# analysis using weakly informative half-normal prior
# (may take a few seconds to compute!):
ma03a <- bayesmeta(y=sj[,"log.or"], sigma=sj[,"log.or.se"],
                   label=sj[,"id.sj"],
                   tau.prior=function(t){dhalfnormal(t,scale=1)})

# alternatively: may utilize "metafor" package's "escalc()" function
# to compute log-ORs and standard errors:
require("metafor")
es <- escalc(measure="OR",
             ai=lihr.events, n1i=lihr.cases,
             ci=oihr.events, n2i=oihr.cases,
             slab=id, data=SidikJonkman2007)
# apply "bayesmeta()" function directly to "escalc" object:
ma03b <- bayesmeta(es, tau.prior=function(t){dhalfnormal(t,scale=1)})
# "ma03a" and "ma03b" should be identical:
print(ma03a$summary)
print(ma03b$summary)
# compare to metafor's (frequentist) random-effects meta-analysis:
rma03a <- rma.uni(es)
rma03b <- rma.uni(es, method="EB", knha=TRUE)
# compare mu estimates (estimate and confidence interval):
plot(ma03b, which=3)
abline(v=c(rma03a$b, rma03a$ci.lb, rma03a$ci.ub), col="red", lty=c(1,2,2))
abline(v=c(rma03b$b, rma03b$ci.lb, rma03b$ci.ub), col="green3", lty=c(1,2,2))
# compare tau estimates (estimate and confidence interval):
plot(ma03b, which=4)
abline(v=confint(rma03a)$random["tau",], col="red", lty=c(1,2,2))
abline(v=confint(rma03b)$random["tau",], col="green3", lty=c(1,3,3))

# show heterogeneity's posterior density:
plot(ma03a, which=4, main="Sidik/Jonkman example")

# show some numbers (mode, median and mean):
abline(v=ma03a$summary[c("mode","median","mean"),"tau"], col="blue")

# compare with Sidik and Jonkman's estimates:
sj.estimates <- sqrt(c("MM"  = 0.429,   # method of moments estimator
                       "VC"  = 0.841,   # variance component type estimator
                       "ML"  = 0.562,   # maximum likelihood estimator
                       "REML"= 0.598,   # restricted maximum likelihood estimator
                       "EB"  = 0.703,   # empirical Bayes estimator
                       "MV"  = 0.818,   # model error variance estimator
                       "MVvc"= 0.747))  # a variation of the MV estimator
abline(v=sj.estimates, col="red", lty="dashed")
## End(Not run)


############################
```

```
# example data by Cochran:
data("Cochran1954")

## Not run:
# analysis using improper uniform prior
# (may take a few seconds to compute!):
ma04 <- bayesmeta(y=Cochran1954[,"mean"], sigma=sqrt(Cochran1954[,"se2"]),
                  label=Cochran1954[,"observer"])

# show joint posterior density:
plot(ma04, which=2, main="Cochran example")
# show (known) true parameter value:
abline(h=161)

# pick a point estimate for tau:
tau <- ma04$summary["median","tau"]
# highlight two point hypotheses (fixed vs. random effects):
abline(v=c(0, tau), col="orange", lty="dotted", lwd=2)

# show marginal posterior density:
plot(ma04, which=3)
abline(v=161)
# show the conditional distributions of the effect mu
# at two tau values corresponding to fixed and random effects models:
cm <- ma04$cond.moment(tau=c(0,tau))
mu <- seq(130,200, le=200)
lines(mu, dnorm(mu, mean=cm[1,"mean"], sd=cm[1,"sd"]), col="orange", lwd=2)
lines(mu, dnorm(mu, mean=cm[2,"mean"], sd=cm[2,"sd"]), col="orange", lwd=2)

# determine a range of tau values:
tau <- seq(0, ma04$qposterior(tau=0.99), length=100)
# compute conditional posterior moments:
cm.overall <- ma04$cond.moment(tau=tau)
# compute study-specific conditional posterior moments:
cm.indiv <- ma04$cond.moment(tau=tau, individual=TRUE)
# show forest plot along with conditional posterior means:
par(mfrow=c(1,2))
  plot(ma04, which=1, main="Cochran 1954 example")
  matplot(tau, cm.indiv[,"mean",], type="l", lty="solid", col=1:ma04$k,
          xlim=c(0,max(tau)*1.2), xlab=expression("heterogeneity "*tau),
          ylab=expression("(conditional) shrinkage estimate E["*
                          theta[i]*"|"*list(tau, y, sigma)*"]"))
  text(rep(max(tau)*1.01, ma04$k), cm.indiv[length(tau),"mean",],
       ma04$label, col=1:ma04$k, adj=c(0,0.5))
  lines(tau, cm.overall[,"mean"], lty="dashed", lwd=2)
  text(max(tau)*1.01, cm.overall[length(tau),"mean"],
       "overall", adj=c(0,0.5))
par(mfrow=c(1,1))

# show the individual effects' posterior distributions:
theta <- seq(120, 240, le=300)
plot(range(theta), c(0,0.1), type="n", xlab=expression(theta[i]), ylab="")
for (i in 1:ma04$k) {
```

```
  # draw estimate +/- uncertainty as a Gaussian:
  lines(theta, dnorm(theta, mean=ma04$y[i], sd=ma04$sigma[i]), col=i+1, lty="dotted")
  # draw effect's posterior distribution:
  lines(theta, ma04$dposterior(theta=theta, indiv=i), col=i+1, lty="solid")
}
abline(h=0)
legend(max(theta), 0.1, legend=ma04$label, col=(1:ma04$k)+1, pch=15, xjust=1, yjust=1)

## End(Not run)
```

---

bmr *Bayesian random-effects meta-regression*

Description

This function allows to derive the posterior distribution of the parameters in a random-effects meta-regression and provides functions to evaluate joint and marginal posterior probability distributions, etc.

Usage

```
  bmr(y, ...)
  ## Default S3 method:
bmr(y, sigma, labels = names(y),
    X = matrix(1.0, nrow=length(y), ncol=1,
               dimnames=list(labels,"intercept")),
    tau.prior = "uniform",
    beta.prior.mean = NULL,
    beta.prior.sd = NULL,
    beta.prior.cov = diag(beta.prior.sd^2,
                          nrow=length(beta.prior.sd),
                          ncol=length(beta.prior.sd)),
    interval.type = c("shortest", "central"),
    delta = 0.01, epsilon = 0.0001,
    rel.tol.integrate = 2^16*.Machine$double.eps,
    abs.tol.integrate = 0.0,
    tol.uniroot = rel.tol.integrate, ...)
  ## S3 method for class 'escalc'
bmr(y, labels = NULL, ...)
```

Arguments

y	vector of estimates, *or* an escalc object.
sigma	vector of standard errors associated with y.
labels	(optional) a vector of labels corresponding to y and sigma.
X	(optional) the *regressor matrix* for the regression.

`tau.prior` a `function` returning the prior density for the heterogeneity parameter (τ) *or* a `character` string specifying one of the *default 'non-informative' priors*; possible choices for the latter case are:

- `"uniform"`: a uniform prior in τ
- `"sqrt"`: a uniform prior in $\sqrt{\tau}$
- `"Jeffreys"`: the Jeffreys prior for τ
- `"BergerDeely"`: the prior due to Berger and Deely (1988)
- `"conventional"`: the conventional prior
- `"DuMouchel"`: the DuMouchel prior
- `"shrinkage"`: the 'uniform shrinkage' prior
- `"I2"`: a uniform prior on the 'relative heterogeneity' I^2

The default is `"uniform"` (which should be used with caution). The above priors are described in some more detail in the `bayesmeta()` help.

`beta.prior.mean`, `beta.prior.sd`, `beta.prior.cov`
the mean and standard deviations, or covariance of the normal prior distribution for the effects β. If unspecified, an (improper) uniform prior is used.

`interval.type` the type of (credible, prediction, shrinkage) interval to be returned by default; either `"shortest"` for shortest intervals, or `"central"` for central, equal-tailed intervals.

`delta`, `epsilon` the parameters specifying the desired accuracy for approximation of the β posterior(s), and with that determining the number of τ support points being used internally. Smaller values imply greater accuracy and greater computational burden (Roever and Friede, 2017).

`rel.tol.integrate`, `abs.tol.integrate`, `tol.uniroot`
the `rel.tol`, `abs.tol` and `tol` 'accuracy' arguments that are passed to the `integrate()` or `uniroot()` functions for internal numerical integration or root finding (see also the help there).

`...` other `bmr` arguments.

Details

The random-effects meta-regression model may be stated as

$$y_i|x_i,\beta,\sigma_i,\tau \;\sim\; \mathrm{Normal}(\beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots + \beta_d x_{i,d},\; \sigma_i^2 + \tau^2)$$

where the data (y, σ) enter as y_i, the i-th estimate, that is associated with standard error $\sigma_i > 0$, where $i = 1, ..., k$. In addition to estimates and standard errors for the ith observation, a set of covariables $x_{i,j}$ with $j = 1, ..., d$ are available for each estimate y_i.

The model includes $d + 1$ unknown parameters, namely, the d coefficients $(\beta_1, ..., \beta_d)$, and the heterogeneity τ. Alternatively, the model may also be formulated via an intermediate step as

$$y_i|\theta_i,\sigma_i \;\sim\; \mathrm{Normal}(\theta_i,\; \sigma_i^2),$$

$$\theta_i|\beta,x_i,\tau \;\sim\; \mathrm{Normal}(\beta_1 x_{i,1} + \ldots + \beta_d x_{i,d},\; \tau^2),$$

where the θ_i denote the *trial-specific* means that are then measured through the estimate y_i with an associated measurement uncertainty σ_i. The θ_i again differ from trial to trial (even for identical

covariable vectors x_i) and are distributed around a mean of $\beta_1 x_{i,1} + \ldots + \beta_d x_{i,d}$ with standard deviation τ.

It if often convenient to express the model in matrix notation, i.e.,

$$y|\theta, \sigma \sim \text{Normal}(\theta, \Sigma)$$

$$\theta|X, \beta, \tau \sim \text{Normal}(X\beta, \tau I)$$

where y, σ, β and θ now denote k-dimensional vectors, X is the $(k \times d)$ *regressor matrix*, and Σ is a $(k \times k)$ diagonal covariance matrix containing the σ_i^2 values, while I is the $(k \times k)$ identity matrix. The regressor matrix X plays a crucial role here, as the 'X' argument (with rows corresponding to studies, and columns corresponding to covariables) is required to specify the exact regression setup.

Meta-regression allows the consideration of (study-level) covariables in a meta-analysis. Quite often, these may also be indicator variables ('zero/one' variables) simply identifying subgroups of studies. See also the examples shown below.

Connection to the simple random-effects model: The meta-regression model is a generalisation of the 'simple' random-effects model that is implemented in the `bayesmeta()` function. Meta-regression reduces to the estimation of a single "intercept" term when the regressor matrix (X) consists of a single column of ones (which is also the default setting in case the 'X' argument is left unspecified). The single regression coefficient β_1 then is equivalent to the μ parameter from the simple random effects model (see also the 'Examples' section below).

Specification of the regressor matrix: The actual regression model is specified through the regressor matrix X, which is supplied via the 'X' argument, and which often may be specified in different ways. There usually is no unique solution, and what serves the present purpose best then depends on the context; see also the examples below. Sensible column names should be specified for `X`, as these will subsequently determine the labels for the associated parameters later on. Model specification via the regressor matrix has the advantage of being very explicit and transparent; if one prefers a `formula` interface instead, a regressor matrix may be generated via the '`model.matrix()`' function.

Prior specification: Priors for β and τ are assumed to factor into into independent marginals $p(\beta, \tau) = p(\beta) \times p(\tau)$ and either (improper) uniform or a normal priors may be specified for the regression coefficients β. For sensible prior choices for the heterogeneity parameter τ, see also Roever (2020), Roever *et al.* (2021) and the '`bayesmeta()`' function's help.

Accessing posterior density functions, etc.: Within the `bayesmeta()` function, access to posterior density, cumulative distribution function, quantile functtion, random number generation and posterior inverval computation is implemented via the `$dposterior()`, `$dposterior()`, `$pposterior()`, `$qposterior()`, `$rposterior()` and `$post.interval()` functions that are accessible as elements in the returned `list` object. Prediction and shrinkage estimation are available by setting additional arguments in the above functions.

In the meta-regression context things get slightly more complicated, as the β parameter may be of higher dimension. Hence, in the `bmr()` function, the three different types of distributions related to *posterior distribution*, *prediction* and *shrinkage* are split up into three groups of functions. For example, the posterior density is accessible via the `$dposterior()` function, the predictive distribution via the `$dpredict()` function, and the shrinkage estimates via the `$dshrink()` function. Analogous functions are returned for cumulative distribution, quantile function, etc.; see also the 'Value' section below.

Computation: The `bmr()` function utilizes the same computational method as the `bayesmeta()` function to derive the posterior distribution, namely, the DIRECT algorithm. Numerical accuracy of the computations is determined by the 'delta' and 'epsilon' arguments (Roever and Friede, 2017).

A slight difference between the `bayesmeta()` and `bmr()` implementations exists in the determination of the grid approximation within the DIRECT algorithm. While `bmr()` considers divergences w.r.t. the conditional posterior distributions $p(\beta|\tau)$, `bayesmeta()` in addition considers divergences w.r.t. the shrinkage estimates, which in general leads to a denser binning (as one can see from the numbers of bins required; see the example below). A denser binning within the `bmr()` function may be achieved by reducing the 'delta' argument.

Value

A list of class `bmr` containing the following elements:

`y`	vector of estimates (the input data).
`sigma`	vector of standard errors corresponding to `y` (input data).
`X`	the regressor matrix.
`k`	number of data points (length of `y`, or rows of X).
`d`	number of coefficients (columns of `X`).
`labels`	vector of labels corresponding to `y` and `sigma`.
`variables`	variable names for the β coefficients (determined by the column names of the supplied `X` argument).
`tau.prior`	the prior probability density function for τ.
`tau.prior.proper`	a `logical` flag indicating whether the heterogeneity prior appears to be proper (which is judged based on an attempted numerical integration of the density function).
`beta.prior`	a `list` containing the prior mean vector and covariance matrix for the coefficients β.
`beta.prior.proper`	a `logical` vector (of length d) indicating whether the corresponding β coefficient's prior is proper (i.e., finite prior mean and variance were specified).
`dprior`	a `function(tau, beta, which.beta, log=FALSE)` of τ and/or β parameters, returning either the joint or marginal prior probability density, depending on which parameter(s) is/are provided.
`likelihood`	a `function(tau, beta, which.beta)` τ and/or β, returning either the joint or marginal likelihood, depending on which parameter(s) is/are provided.
`dposterior`, `pposterior`, `qposterior`, `rposterior`, `post.interval`	functions of τ and/or β parameters, returning either the joint or marginal posterior probability density, (depending on which parameter(s) is/are provided), or cumulative distribution function, quantile function, random numbers or posterior intervals.

`dpredict`, `ppredict`, `qpredict`, `rpredict`, `pred.interval`
: functions of β returning density, cumulative distribution function, quantiles, random numbers, or intervals for the *predictive distribution*. This requires specification of x values to indicate what covariable values to consider. Use of '`mean=TRUE`' (the default) yields predictions for the *mean* ($x'\beta$ values), setting it to `FALSE` yields *predictions* (θ values).

`dshrink`, `pshrink`, `qshrink`, `rshrink`, `shrink.interval`
: functions of θ yielding density, cumulative distribution, quantiles, random numbers or posterior intervals for the *shrinkage estimates* of the individual θ_i parameters corresponding to the supplied y_i data values ($i = 1, \ldots, k$). May be identified using the '`which`' argument via its index (i) or a character string giving the corresponding study label.

`post.moments`
: a `function(tau)` returning conditional posterior moments (mean and covariance) of β as a function of τ.

`pred.moments`
: a `function(tau, x, mean=TRUE)` returning conditional posterior predictive moments (means and standard deviations) as a function of τ.

`shrink.moments`
: a `function(tau, which)` returning conditional moments (means and standard deviations of shrinkage distributions) as a function of τ.

`summary`
: a `matrix` listing some summary statistics, namely marginal posterior mode, median, mean, standard deviation and a (shortest) 95% credible intervals, of the marginal posterior distributions of τ and β_i.

`interval.type`
: the `interval.type` input argument specifying the type of interval to be returned by default.

`ML`
: a `matrix` giving joint and marginal maximum-likelihood estimates of (τ, β).

`MAP`
: a `matrix` giving joint and marginal maximum-a-posteriori estimates of (τ, β).

`theta`
: a `matrix` giving the 'shrinkage estimates', i.e, summary statistics of the trial-specific means θ_i.

`marginal.likelihood`
: the marginal likelihood of the data (this number can only be computed if proper effect and heterogeneity priors are specified).

`bayesfactor`
: Bayes factors and minimum Bayes factors for the hypotheses of $\tau = 0$ and $\beta_i = 0$. These depend on the marginal likelihood and hence can only be computed if proper effect and/or heterogeneity priors are specified.

`support`
: a `list` giving the τ support points used internally in the grid approximation, along with their associated weights, and conditional mean and covariance of β.

`delta`, `epsilon`
: the '`delta`' and '`epsilon`' input parameter determining numerical accuracy.

`rel.tol.integrate`, `abs.tol.integrate`, `tol.uniroot`
: the input parameters determining the numerical accuracy of the internally used `integrate()` and `uniroot()` functions.

`call`
: an object of class `call` giving the function call that generated the `bmr` object.

`init.time`
: the computation time (in seconds) used to generate the `bmr` object.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

C. Roever, T. Friede. Discrete approximation of a mixture distribution via restricted divergence. *Journal of Computational and Graphical Statistics*, **26**(1):217-222, 2017. doi:10.1080/10618600.2016.1276840.

A. Gelman, J.B. Carlin, H.S. Stern, D.B. Rubin. *Bayesian data analysis*. Chapman & Hall / CRC, Boca Raton, 1997.

See Also

bayesmeta, escalc, model.matrix, CrinsEtAl2014, RobergeEtAl2017.

Examples

```
## Not run:
#########################################################################
# (1)  A simple example with two groups of studies

# load data:
data("CrinsEtAl2014")
# compute effect measures (log-OR):
crins.es <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)
# show data:
crins.es[,c("publication", "IL2RA", "exp.AR.events", "exp.total",
            "cont.AR.events", "cont.total", "yi", "vi")]

# specify regressor matrix:
X <- cbind("bas"=as.numeric(crins.es$IL2RA=="basiliximab"),
           "dac"=as.numeric(crins.es$IL2RA=="daclizumab"))
print(X)
print(cbind(crins.es[,c("publication", "IL2RA")], X))
# NB: regressor matrix specifies individual indicator covariates
#     for studies with "basiliximab" and "daclizumab" treatment.

# perform regression:
bmr01 <- bmr(y=crins.es$yi, sigma=sqrt(crins.es$vi),
             labels=crins.es$publication, X=X)

# alternatively, one may simply supply the "escalc" object
# (yields identical results):
bmr01 <- bmr(crins.es, X=X)
```

```
# show results:
bmr01
bmr01$summary
plot(bmr01)
pairs(bmr01)

# NOTE: there are many ways to set up the regressor matrix "X"
# (also affecting the interpretation of the involved parameters).
# See the above specification and check out the following alternatives:
X <- cbind("bas"=1, "offset.dac"=c(1,0,1,0,0,0))
X <- cbind("intercept"=1, "offset"=0.5*c(1,-1,1,-1,-1,-1))
# One may also use the "model.matrix()" function
# to specify regressor matrices via the "formula" interface; e.g.:
X <- model.matrix( ~ IL2RA, data=crins.es)
X <- model.matrix( ~ IL2RA - 1, data=crins.es)


#########################################################################
# (2)  A simple example reproducing a "bayesmeta" analysis:

data("CrinsEtAl2014")
crins.es <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)

# a "simple" meta-analysis:
bma02 <- bayesmeta(crins.es,
                   tau.prior=function(t){dhalfnormal(t, scale=0.5)},
                   mu.prior.mean=0, mu.prior.sd=4)

# the equivalent "intercept-only" meta-regression:
bmr02 <- bmr(crins.es,
             tau.prior=function(t){dhalfnormal(t, scale=0.5)},
             beta.prior.mean=0, beta.prior.sd=4)
# the corresponding (default) regressor matrix:
bmr02$X

# compare computation time and numbers of bins used internally:
cbind("seconds" = c("bayesmeta" = unname(bma02$init.time),
                    "bmr"       = unname(bmr02$init.time)),
      "bins"    = c("bayesmeta" = nrow(bma02$support),
                    "bmr"       = nrow(bmr02$support$tau)))

# compare heterogeneity estimates:
rbind("bayesmeta"=bma02$summary[,1], "bmr"=bmr02$summary[,1])

# compare effect estimates:
rbind("bayesmeta"=bma02$summary[,2], "bmr"=bmr02$summary[,2])


#########################################################################
# (3)  An example with binary as well as continuous covariables:
```

```
# load data:
data("RobergeEtAl2017")
str(RobergeEtAl2017)
head(RobergeEtAl2017)
?RobergeEtAl2017

# compute effect sizes (log odds ratios) from count data:
es.pe  <- escalc(measure="OR",
                 ai=asp.PE.events,  n1i=asp.PE.total,
                 ci=cont.PE.events, n2i=cont.PE.total,
                 slab=study, data=RobergeEtAl2017,
                 subset=complete.cases(RobergeEtAl2017[,7:10]))

# show "bubble plot" (bubble sizes are
# inversely proportional to standard errors):
plot(es.pe$dose, es.pe$yi, cex=1/sqrt(es.pe$vi),
     col=c("blue","red")[as.numeric(es.pe$onset)],
     xlab="dose (mg)", ylab="log-OR (PE)", main="Roberge et al. (2017)")
legend("topright", col=c("blue","red"), c("early onset", "late onset"), pch=1)

# set up regressor matrix:
# (individual intercepts and slopes for two subgroups):
X <- model.matrix(~ -1 + onset + onset:dose, data=es.pe)
colnames(X) <- c("intEarly", "intLate", "slopeEarly", "slopeLate")
# check out regressor matrix (and compare to original data):
print(X)

# perform regression:
bmr03 <- bmr(es.pe, X=X)
bmr03$summary

# derive predictions from the model;
# specify corresponding "regressor matrices":
newx.early <- cbind(1, 0, seq(50, 150, by=5), 0)
newx.late  <- cbind(0, 1, 0, seq(50, 150, by=5))
# (note: columns correspond to "beta" parameters)

# compute predicted medians and 95 percent intervals:
pred.early <- cbind("median"=bmr03$qpred(0.5, x=newx.early),
                    bmr03$pred.interval(x=newx.early))
pred.late <- cbind("median"=bmr03$qpred(0.5, x=newx.late),
                    bmr03$pred.interval(x=newx.late))

# draw "bubble plot":
plot(es.pe$dose, es.pe$yi, cex=1/sqrt(es.pe$vi),
     col=c("blue","red")[as.numeric(es.pe$onset)],
     xlab="dose (mg)", ylab="log-OR (PE)", main="Roberge et al. (2017)")
legend("topright", col=c("blue","red"), c("early onset", "late onset"), pch=1)
# add predictions to bubble plot:
matlines(newx.early[,3], pred.early, col="blue", lty=c(1,2,2))
matlines(newx.late[,4], pred.late, col="red", lty=c(1,2,2))
```

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

`BucherEtAl1997` *Direct and indirect comparison example data*

Description

Numbers of subjects and events in the different treatment arms of 22 studies.

Usage

```
data("BucherEtAl1997")
```

Format

The data frame contains the following columns:

study	`character`	publication identifier (first author and publication year)
treat.A	`factor`	treatment in first study arm ("TMP-SMX" or "AP")
treat.B	`factor`	treatment in second study arm ("D/P" or "AP")
events.A	`numeric`	number of events in first study arm
events.B	`numeric`	number of events in second study arm
total.A	`numeric`	total number of patients in first study arm
total.B	`numeric`	total number of patients in second study arm

Details

Bucher *et al.* (1997) discussed the example case of the comparison of *sulphametoxazole-trimethoprim (TMP-SMX)* versus *dapsone/pyrimethamine (D/P)* for the prophylaxis of *Pneumocystis carinii* pneumonia in HIV patients. Eight studies had undertaken a head-to-head comparison of both medications, but an additional 14 studies were available investigating one of the two medications with *aerosolized pentamidine (AP)* as a comparator. Nine studies compared TMP-SMX vs. AP, and five studies compared D/P vs. AP. Together these provide *indirect* evidence on the effect of TMP-SMX compared to D/P (Kiefer *et al.*, 2015).

The example constitutes a simple case of a *network meta-analysis (NMA)* setup, where only two-armed studies are considered, and analysis is based on pairwise comparisons of treatments (or *contrasts*). In this case, the joint analysis of *direct* and *indirect* evidence may be implemented as a special case of a meta-regression (Higgins *et al.*, 2019; Sec. 11.4.2). The original data in fact included some three-armed studies, in which case one of the arms was deliberately omitted (Bucher *et al.*; 1997).

Source

H.C. Bucher, G.H. Guyatt, L.E. Griffith, S.D. Walter. The results of direct and indirect treatment comparisons in meta-analysis of randomized controlled trials. *Journal of Clinical Epidemiology*, **50**(6):683-691, 1997. doi:10.1016/S08954356(97)000498.

References

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

J.P.T. Higgins, J. Thomas, J. Chandler, M. Cumpston, T. Li, M.J. Page, V.A. Welch (eds.). *Cochrane handbook for systematic reviews of interventions*. Wiley and Sons, 2nd edition, 2019. doi:10.1002/9781119536604. http://training.cochrane.org/handbook.

C. Kiefer, S. Sturtz, R. Bender. Indirect comparisons and network meta-analyses. *Deutsches Aerzteblatt International*, **112**(47):803-808, 2015. doi:10.3238/arztebl.2015.0803.

Examples

```
# load data:
data("BucherEtAl1997")

# show data:
head(BucherEtAl1997)

## Not run:
# compute effect sizes (log-ORs for pairwise comparisons)
# from the count data:
es <- escalc(measure="OR",
             ai=events.A, n1i=total.A,   # "exposure group"
             ci=events.B, n2i=total.B,   # "control group"
             slab=study, data=BucherEtAl1997)

# specify regressor matrix:
X <- cbind("TMP.DP" = rep(c(1, 0, 1), c(8,5,9)),
           "AP.DP"  = rep(c(0, 1,-1), c(8,5,9)))

# perform Bayesian meta-regression:
bmr01 <- bmr(es, X=X)

# show default output:
print(bmr01)

# specify contrast matrix:
contrastX <- rbind("TMP-SMX vs. D/P"=c(1,0),
                   "AP vs. D/P"     =c(0,1),
                   "TMP-SMX vs. AP" =c(1,-1))
# show summary including contrast estimates:
summary(bmr01, X.mean=contrastX)
# show forest plot including contrast estimates:
forestplot(bmr01, X.mean=contrastX, xlab="log-OR")


# perform frequentist meta-regression:
fmr01 <- rma(es, mods=X, intercept=FALSE)
print(fmr01)

# compare Bayesian and frequentist results;
# estimated log-OR for "TMP-SMX" vs. "D/P"
```

```
rbind("bayesmeta"=bmr01$summary[c("mean","sd"),"TMP.DP"],
      "rma"      =c(fmr01$beta["TMP.DP",], fmr01$se[1]))

# estimated log-OR for "AP" vs. "D/P"
rbind("bayesmeta"=bmr01$summary[c("mean","sd"),"AP.DP"],
      "rma"      =c(fmr01$beta["AP.DP",], fmr01$se[2]))

# estimated heterogeneity:
rbind("bayesmeta"=bmr01$summary["median","tau"],
      "rma"      =sqrt(fmr01$tau2))

## End(Not run)
```

`Cochran1954` *Fly counts example data*

Description

This data set gives average estimated counts of flies along with standard errors from 7 different observers.

Usage

```
data("Cochran1954")
```

Format

The data frame contains the following columns:

observer	`character`	identifier
mean	`numeric`	mean count
se2	`numeric`	*squared* standard error

Details

Quoting from Cochran (1954), example 3, p.119: "In studies by the U.S. Public Health Service of observers' abilities to count the number of flies which settle momentarily on a grill, each of 7 observers was shown, for a brief period, grills with known numbers of flies impaled on them and asked to estimate the numbers. For a given grill, each observer made 5 independent estimates. The data in table 9 are for a grill which actually contained 161 flies. Estimated variances are based on 4 degrees of freedom each. [...] The only point of interest in estimating the overall mean is to test whether there is any consistent bias among observers in estimating the 161 flies on the grill. Although inspection of table 9 suggests no such bias, the data will serve to illustrate the application of partial weighting."

Source

W.G. Cochran. The combination of estimates from different experiments. *Biometrics*, **10**(1):101-129, 1954.

Examples

```
data("Cochran1954")
## Not run:
# analysis using improper uniform prior
# (may take a few seconds to compute!):
bma <- bayesmeta(y=Cochran1954[,"mean"], sigma=sqrt(Cochran1954[,"se2"]),
                 label=Cochran1954[,"observer"])

# show joint posterior density:
plot(bma, which=2, main="Cochran example")
# show (known) true parameter value:
abline(h=161)

# show forest plot:
forestplot(bma, zero=161)

## End(Not run)
```

---

convolve *Convolution of two probability distributions*

Description

Compute the convolution of two probability distributions, specified through their densities or cumulative distribution functions (CDFs).

Usage

```
convolve(dens1, dens2,
         cdf1=Vectorize(function(x){integrate(dens1,-Inf,x)$value}),
         cdf2=Vectorize(function(x){integrate(dens2,-Inf,x)$value}),
         delta=0.01, epsilon=0.0001)
```

Arguments

`dens1`, `dens2`	the two distributions' probability density functions.
`cdf1`, `cdf2`	the two distributions' cumulative distribution functions.
`delta`, `epsilon`	the parameters specifying the desired accuracy for approximation of the convolution, and with that determining the number of support points being used internally. Smaller values imply greater accuracy and greater computational burden (Roever and Friede, 2017).

Details

The distribution of the *sum* of two (independent) random variables technically results as a *convolution* of their probability distributions. In some cases, the calculation of convolutions may be done analytically; e.g., the sum of two normally distributed random variables again turns out as normally

distributed (with mean and variance resulting as the sums of the original ones). In other cases, convolutions may need to be determined numerically. One way to achieve this is via the *DIRECT* algorithm; the present implementation is the one discussed by Roever and Friede (2017). Accuracy of the computations is determined by the `delta` (maximum divergence δ) and `epsilon` (tail probability ϵ) parameters.

Convolutions here are used within the `funnel()` function (to generate funnel plots), but are often useful more generally. The original probability distributions may be specified via their probability density functions or their cumulative distribution functions (CDFs). The `convolve()` function returns the convolution's density, CDF and quantile function (inverse CDF).

Value

A `list` with elements

`delta`	the δ parameter.
`epsilon`	the ϵ parameter.
`binwidth`	the bin width.
`bins`	the total number of bins.
`support`	a `matrix` containing the support points used internally for the convolution approximation.
`density`	the probability density function.
`cdf`	the cumulative distribution function (CDF).
`quantile`	the quantile function (inverse CDF).

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, T. Friede. Discrete approximation of a mixture distribution via restricted divergence. *Journal of Computational and Graphical Statistics*, **26**(1):217-222, 2017. doi:10.1080/10618600.2016.1276840.

See Also

`bayesmeta`, `funnel`.

Examples

```
## Not run:
#  Skew-normal / logistic example:

dens1 <- function(x, shape=4)
# skew-normal distribution's density
# see also: http://azzalini.stat.unipd.it/SN/Intro
{
  return(2 * dnorm(x) * pnorm(shape * x))
}
```

```
dens2 <- function(x)
# logistic distribution's density
{
  return(dlogis(x, location=0, scale=1))
}

rskewnorm <- function(n, shape=4)
# skew-normal random number generation
# (according to http://azzalini.stat.unipd.it/SN/faq-r.html)
{
  delta <- shape / sqrt(shape^2+1)
  u1 <- rnorm(n); v <- rnorm(n)
  u2 <- delta * u1 + sqrt(1-delta^2) * v
  return(apply(cbind(u1,u2), 1, function(x){ifelse(x[1]>=0, x[2], -x[2])}))
}

# compute convolution:
conv <- convolve(dens1, dens2)

# illustrate convolution:
n <- 100000
x <- rskewnorm(n)
y <- rlogis(n)
z <- x + y

# determine empirical and theoretical quantiles:
p      <- c(0.001,0.01, 0.1, 0.5, 0.9, 0.99, 0.999)
equant <- quantile(z, prob=p)
tquant <- conv$quantile(p)

# show numbers:
print(cbind("p"=p, "empirical"=equant, "theoretical"=tquant))

# draw Q-Q plot:
rg <- range(c(equant, tquant))
plot(rg, rg, type="n", asp=1, main="Q-Q-plot",
     xlab="theoretical quantile", ylab="empirical quantile")
abline(0, 1, col="grey")
points(tquant, equant)

## End(Not run)
```

`CrinsEtAl2014` *Pediatric liver transplant example data*

Description

Numbers of cases (transplant patients) and events (acute rejections, steroid resistant rejections, PTLDs, and deaths) in experimental and control groups of six studies.

Usage

```
data("CrinsEtAl2014")
```

Format

The data frame contains the following columns:

publication	character	publication identifier (first author and publication year)
year	numeric	publication year
randomized	factor	randomization status (y/n)
control.type	factor	type of control group ('concurrent' or 'historical')
comparison	factor	type of comparison ('IL-2RA only', 'delayed CNI', or 'no/low steroids')
IL2RA	factor	type of interleukin-2 receptor antagonist (IL-2RA) ('basiliximab' or 'daclizumab')
CNI	factor	type of calcineurin inhibitor (CNI) ('tacrolimus' or 'cyclosporine A')
MMF	factor	use of mycofenolate mofetil (MMF) (y/n)
followup	numeric	follow-up time in months
treat.AR.events	numeric	number of AR events in experimental group
treat.SRR.events	numeric	number of SRR events in experimental group
treat.PTLD.events	numeric	number of PTLD events in experimental group
treat.deaths	numeric	number of deaths in experimental group
treat.total	numeric	number of cases in experimental group
control.AR.events	numeric	number of AR events in control group
control.SRR.events	numeric	number of SRR events in control group
control.PTLD.events	numeric	number of PTLD events in control group
control.deaths	numeric	number of deaths in control group
control.total	numeric	number of cases in control group

Details

A systematic literature review investigated the evidence on the effect of Interleukin-2 receptor antagonists (IL-2RA) and resulted in six controlled studies reporting acute rejection (AR), steroid-resistant rejection (SRR) and post-transplant lymphoproliferative disorder (PTLD) rates as well as mortality in pediatric liver transplant recipients.

Source

N.D. Crins, C. Roever, A.D. Goralczyk, T. Friede. Interleukin-2 receptor antagonists for pediatric liver transplant recipients: A systematic review and meta-analysis of controlled studies. *Pediatric Transplantation*, **18**(8):839-850, 2014. doi:10.1111/petr.12362.

References

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

T.G. Heffron et al. Pediatric liver transplantation with daclizumab induction therapy. *Transplantation*, **75**(12):2040-2043, 2003. doi:10.1097/01.TP.0000065740.69296.DA.

N.E.M. Gibelli et al. Basiliximab-chimeric anti-IL2-R monoclonal antibody in pediatric liver transplantation: comparative study. *Transplantation Proceedings*, **36**(4):956-957, 2004. doi:10.1016/j.transproceed.2004.04.070.

S. Schuller et al. Daclizumab induction therapy associated with tacrolimus-MMF has better outcome compared with tacrolimus-MMF alone in pediatric living donor liver transplantation. *Transplantation Proceedings*, **37**(2):1151-1152, 2005. doi:10.1016/j.transproceed.2005.01.023.

R. Ganschow et al. Long-term results of basiliximab induction immunosuppression in pediatric liver transplant recipients. *Pediatric Transplantation*, **9**(6):741-745, 2005. doi:10.1111/j.1399-3046.2005.00371.x.

M. Spada et al. Randomized trial of basiliximab induction versus steroid therapy in pediatric liver allograft recipients under tacrolimus immunosuppression. *American Journal of Transplantation*, **6**(8):1913-1921, 2006. doi:10.1111/j.16006143.2006.01406.x.

J.M. Gras et al. Steroid-free, tacrolimus-basiliximab immunosuppression in pediatric liver transplantation: Clinical and pharmacoeconomic study in 50 children. *Liver Transplantation*, **14**(4):469-477, 2008. doi:10.1002/lt.21397.

See Also

GoralczykEtAl2011.

Examples

```
data("CrinsEtAl2014")
## Not run:
# compute effect sizes (log odds ratios) from count data
# (using "metafor" package's "escalc()" function):
require("metafor")
crins.es <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)
print(crins.es)

# analyze using weakly informative half-Cauchy prior for heterogeneity:
crins.ma <- bayesmeta(crins.es, tau.prior=function(t){dhalfcauchy(t,scale=1)})

# show results:
print(crins.ma)
forestplot(crins.ma)
plot(crins.ma)

# show heterogeneity posterior along with prior:
plot(crins.ma, which=4, prior=TRUE)

# perform meta analysis using 2 randomized studies only
# but use 4 non-randomized studies to inform heterogeneity prior:
crins.nrand <- bayesmeta(crins.es[crins.es$randomized=="no",],
                         tau.prior=function(t){dhalfcauchy(t,scale=1)})
crins.rand  <- bayesmeta(crins.es[crins.es$randomized=="yes",],
                         tau.prior=function(t){crins.nrand$dposterior(tau=t)})
```

```
plot(crins.nrand, which=4, prior=TRUE,
     main="non-randomized posterior = randomized prior")
plot(crins.rand, which=4, prior=TRUE, main="randomized posterior")
plot(crins.rand, which=1)

## End(Not run)
```

`dhalflogistic` *Half-logistic distribution.*

Description

Half-logistic density, distribution, and quantile functions, random number generation and expectation and variance.

Usage

```
dhalflogistic(x, scale=1, log=FALSE)
phalflogistic(q, scale=1)
qhalflogistic(p, scale=1)
rhalflogistic(n, scale=1)
ehalflogistic(scale=1)
vhalflogistic(scale=1)
```

Arguments

`x`, `q`	quantile.
`p`	probability.
`n`	number of observations.
`scale`	scale parameter (> 0).
`log`	logical; if `TRUE`, logarithmic density will be returned.

Details

The **half-logistic distribution** is simply a zero-mean logistic distribution that is restricted to take only positive values. If $X \sim \mathrm{logistic}$, then $|sX| \sim \mathrm{halflogistic}(\mathrm{scale} = s)$.

Value

'`dhalflogistic()`' gives the density function, '`phalflogistic()`' gives the cumulative distribution function (CDF), '`qhalflogistic()`' gives the quantile function (inverse CDF), and '`rhalflogistic()`' generates random deviates. The '`ehalflogistic()`' and '`vhalflogistic()`' functions return the corresponding half-logistic distribution's expectation and variance, respectively.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

N.L. Johnson, S. Kotz, N. Balakrishnan. *Continuous univariate distributions*, volume 2, chapter 23.11. Wiley, New York, 2nd edition, 1994.

See Also

dlogis, dhalfnormal, dlomax, drayleigh, TurnerEtAlPrior, RhodesEtAlPrior, bayesmeta.

Examples

```
######################
# illustrate densities:
x <- seq(0,6,le=200)
plot(x, dhalfnormal(x), type="l", col="red", ylim=c(0,1),
     xlab=expression(tau), ylab=expression("probability density "*f(tau)))
lines(x, dhalflogistic(x), col="green3")
lines(x, dhalfcauchy(x), col="blue")
lines(x, dexp(x), col="cyan")
abline(h=0, v=0, col="grey")

# show log-densities (note the differing tail behaviour):
plot(x, dhalfnormal(x), type="l", col="red", ylim=c(0.001,1), log="y",
     xlab=expression(tau), ylab=expression("probability density "*f(tau)))
lines(x, dhalflogistic(x), col="green3")
lines(x, dhalfcauchy(x), col="blue")
lines(x, dexp(x), col="cyan")
abline(v=0, col="grey")
```

---

dhalfnormal *Half-normal, half-Student-t and half-Cauchy distributions.*

Description

Half-normal, half-Student-t and half-Cauchy density, distribution, quantile functions, random number generation, and expectation and variance.

Usage

```
dhalfnormal(x, scale=1, log=FALSE)
phalfnormal(q, scale=1)
qhalfnormal(p, scale=1)
rhalfnormal(n, scale=1)
ehalfnormal(scale=1)
vhalfnormal(scale=1)
```

```
dhalft(x, df, scale=1, log=FALSE)
phalft(q, df, scale=1)
qhalft(p, df, scale=1)
rhalft(n, df, scale=1)
ehalft(df, scale=1)
vhalft(df, scale=1)

dhalfcauchy(x, scale=1, log=FALSE)
phalfcauchy(q, scale=1)
qhalfcauchy(p, scale=1)
rhalfcauchy(n, scale=1)
ehalfcauchy(scale=1)
vhalfcauchy(scale=1)
```

Arguments

x, q	quantile.
p	probability.
n	number of observations.
scale	scale parameter (> 0).
df	degrees-of-freedom parameter (> 0).
log	logical; if TRUE, logarithmic density will be returned.

Details

The **half-normal distribution** is simply a zero-mean normal distribution that is restricted to take only positive values. The *scale* parameter σ here corresponds to the underlying normal distribution's standard deviation: if $X \sim \mathrm{Normal}(0, \sigma^2)$, then $|X| \sim \mathrm{halfNormal}(\mathrm{scale} = \sigma)$. Its mean is $\sigma\sqrt{2/\pi}$, and its variance is $\sigma^2(1 - 2/\pi)$. Analogously, the **half-t distribution** is a truncated Student-t distribution with df degrees-of-freedom, and the **half-Cauchy distribution** is again a special case of the half-t distribution with df=1 degrees of freedom.

Note that (half-) Student-t and Cauchy distributions arise as continuous *mixture distributions* of (half-) normal distributions. If

$$Y|\sigma \sim \mathrm{Normal}(0, \sigma^2)$$

where the standard deviation is $\sigma = \sqrt{k/X}$ and X is drawn from a χ^2-distribution with k degrees of freedom, then the marginal distribution of Y is Student-t with k degrees of freedom.

Value

'dhalfnormal()' gives the density function, 'phalfnormal()' gives the cumulative distribution function (CDF), 'qhalfnormal()' gives the quantile function (inverse CDF), and 'rhalfnormal()' generates random deviates. The 'ehalfnormal()' and 'vhalfnormal()' functions return the corresponding half-normal distribution's expectation and variance, respectively. For the 'dhalft()', 'dhalfcauchy()' and related function it works analogously.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

A. Gelman. Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*, **1**(3):515-534, 2006. doi:10.1214/06BA117A.

F. C. Leone, L. S. Nelson, R. B. Nottingham. The folded normal distribution. *Technometrics*, **3**(4):543-550, 1961. doi:10.2307/1266560.

N. G. Polson, J. G. Scott. On the half-Cauchy prior for a global scale parameter. *Bayesian Analysis*, **7**(4):887-902, 2012. doi:10.1214/12BA730.

S. Psarakis, J. Panaretos. The folded t distribution. *Communications in Statistics - Theory and Methods*, **19**(7):2717-2734, 1990. doi:10.1080/03610929008830342.

See Also

dnorm, dt, dcauchy, dlomax, drayleigh, TurnerEtAlPrior, RhodesEtAlPrior, bayesmeta.

Examples

```
#######################
# illustrate densities:
x <- seq(0,6,le=200)
plot(x, dhalfnormal(x), type="l", col="red", ylim=c(0,1),
     xlab=expression(tau), ylab=expression("probability density "*f(tau)))
lines(x, dhalft(x, df=3), col="green")
lines(x, dhalfcauchy(x), col="blue")
lines(x, dexp(x), col="cyan")
abline(h=0, v=0, col="grey")

# show log-densities (note the differing tail behaviour):
plot(x, dhalfnormal(x), type="l", col="red", ylim=c(0.001,1), log="y",
     xlab=expression(tau), ylab=expression("probability density "*f(tau)))
lines(x, dhalft(x, df=3), col="green")
lines(x, dhalfcauchy(x), col="blue")
lines(x, dexp(x), col="cyan")
abline(v=0, col="grey")
```

---

dinvchi *Inverse-Chi distribution.*

Description

(Scaled) inverse-Chi density, distribution, and quantile functions, random number generation and expectation and variance.

Usage

```
dinvchi(x, df, scale=1, log=FALSE)
pinvchi(q, df, scale=1, lower.tail=TRUE, log.p=FALSE)
qinvchi(p, df, scale=1, lower.tail=TRUE, log.p=FALSE)
rinvchi(n, df, scale=1)
einvchi(df, scale=1)
vinvchi(df, scale=1)
```

Arguments

x, q	quantile.
p	probability.
n	number of observations.
df	degrees-of-freedom parameter (> 0).
scale	scale parameter (> 0).
log	logical; if TRUE, logarithmic density will be returned.
lower.tail	logical; if TRUE (default), probabilities are P(X <= x), otherwise, P(X > x).
log.p	logical; if TRUE, probabilities p are returned as log(p).

Details

The **(scaled) inverse-Chi distribution** is defined as the distribution of the (scaled) inverse of the square root of a Chi-square-distributed random variable. It is a special case of the *square-root inverted-gamma* distribution (with $\alpha = \nu/2$ and $\beta = 1/2$) (Bernardo and Smith; 1994). Its probability density function is given by

$$p(x) \;=\; \frac{2^{(1-\nu/2)}}{s\,\Gamma(\nu/2)} \Big(\frac{s}{x}\Big)^{(\nu+1)} \exp\Big(-\frac{s^2}{2\,x^2}\Big)$$

where ν is the *degrees-of-freedom* and s the *scale* parameter.

Value

'dinvchi()' gives the density function, 'pinvchi()' gives the cumulative distribution function (CDF), 'qinvchi()' gives the quantile function (inverse CDF), and 'rinvchi()' generates random deviates. The 'einvchi()' and 'vinvchi()' functions return the corresponding distribution's expectation and variance, respectively.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

J.M. Bernardo, A.F.M. Smith. *Bayesian theory*, Appendix A.1. Wiley, Chichester, UK, 1994.

See Also

dhalfnormal, dhalft.

Examples

```
################################
# illustrate Chi^2 - connection;
# generate Chi^2-draws:
chi2 <- rchisq(1000, df=10)
# transform:
invchi <- sqrt(1 / chi2)
# show histogram:
hist(invchi, probability=TRUE, col="grey")
# show density for comparison:
x <- seq(0, 1, length=100)
lines(x, dinvchi(x, df=10, scale=1), col="red")
# compare theoretical and empirical moments:
rbind("theoretical" = c("mean" = einvchi(df=10, scale=1),
                        "var"  = vinvchi(df=10, scale=1)),
      "Monte Carlo" = c("mean" = mean(invchi),
                        "var"  = var(invchi)))

#############################################################
# illustrate the normal/Student-t - scale mixture connection;
# specify degrees-of-freedom:
df <- 5
# generate standard normal draws:
z <- rnorm(1000)
# generate random scalings:
sigma <- rinvchi(1000, df=df, scale=sqrt(df))
# multiply to yield Student-t draws:
t <- z * sigma
# check Student-t distribution via a Q-Q-plot:
qqplot(qt(ppoints(length(t)), df=df), t)
abline(0, 1, col="red")
```

---

dlomax *The Lomax distribution.*

Description

Lomax density, distribution and quantile functions, random number generation, and expectation and variance.

Usage

```
dlomax(x, shape=1, scale=1, log=FALSE)
plomax(q, shape=1, scale=1)
qlomax(p, shape=1, scale=1)
```

```
rlomax(n, shape=1, scale=1)
elomax(shape=1, scale=1)
vlomax(shape=1, scale=1)
```

Arguments

x, q	quantile.
p	probability.
n	number of observations.
shape	shape parameter ($\alpha > 0$).
scale	scale parameter ($\lambda > 0$).
log	logical; if TRUE, logarithmic density will be returned.

Details

The Lomax distribution is a heavy-tailed distribution that also is a special case of a *Pareto distribution of the 2nd kind*. The probability density function of a Lomax distributed variable with shape $\alpha > 0$ and scale $\lambda > 0$ is given by

$$p(x) = (\alpha/\lambda)(1 + x/\lambda)^{-(\alpha+1)}.$$

The density function is monotonically decreasing in x. Its mean is $\lambda/(\alpha - 1)$ (for $\alpha > 1$) and its median is $\alpha(2^{1/\alpha} - 1)$. Its variance is finite only for $\alpha > 2$ and equals $(\lambda^2\alpha)/((\alpha - 1)^2(\alpha - 2))$. The cumulative distribution function (CDF) is given by

$$P(x) = 1 - (1 + x/\lambda)^{-\alpha}.$$

The Lomax distribution also arises as a **gamma-exponential mixture**. Suppose that X is a draw from an exponential distribution whose rate θ again is drawn from a gamma distribution with shape a and scale s (so that $\mathrm{E}[\theta] = as$ and $\mathrm{Var}(\theta) = as^2$, or $\mathrm{E}[1/\theta] = \frac{1}{s(a+1)}$ and $\mathrm{Var}(1/\theta) = \frac{1}{s^2(a-1)^2(a-2)}$). Then the marginal distribution of X is Lomax with scale $1/s$ and shape a. Consequently, if the moments of θ are given by $\mathrm{E}[\theta] = \mu$ and $\mathrm{Var}(\theta) = \sigma^2$, then X is Lomax distributed with shape $\alpha = \left(\frac{\mu}{\sigma}\right)^2$ and scale $\lambda = \frac{\mu}{\sigma^2} = \frac{\alpha}{\mu}$. The gamma-exponential connection is also illustrated in an example below.

Value

'dlomax()' gives the density function, 'plomax()' gives the cumulative distribution function (CDF), 'qlomax()' gives the quantile function (inverse CDF), and 'rlomax()' generates random deviates. The 'elomax()' and 'vlomax()' functions return the corresponding Lomax distribution's expectation and variance, respectively.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

N.L. Johnson, S. Kotz, N. Balakrishnan. *Continuous univariate distributions*, volume 1. Wiley, New York, 2nd edition, 1994.

See Also

dexp, dgamma, dhalfnormal, dhalft, dhalfcauchy, drayleigh, TurnerEtAlPrior, RhodesEtAlPrior, bayesmeta.

Examples

```
#######################
# illustrate densities:
x <- seq(0,6,le=200)
plot(x, dexp(x, rate=1), type="l", col="cyan", ylim=c(0,1),
     xlab=expression(tau), ylab=expression("probability density "*f(tau)))
lines(x, dlomax(x), col="orange")
abline(h=0, v=0, col="grey")

# show log-densities (note the differing tail behaviour):
plot(x, dexp(x, rate=1), type="l", col="cyan", ylim=c(0.001,1), log="y",
     xlab=expression(tau), ylab=expression("probability density "*f(tau)))
lines(x, dlomax(x), col="orange")
abline(v=0, col="grey")

#####################################################
# illustrate the gamma-exponential mixture connection;
# specify a number of samples:
N <- 10000
# specify some gamma shape and scale parameters
# (via mixing distribution's moments):
expectation <- 2.0
stdev       <- 1.0
gammashape <- (expectation / stdev)^2
gammascale <- stdev^2 / expectation
print(c("expectation"=expectation, "stdev"=stdev,
        "shape"=gammashape, "scale"=gammascale))
# generate gamma-distributed rates:
lambda <- rgamma(N, shape=gammashape, scale=gammascale)
# generate exponential draws according to gamma-rates:
y <- rexp(N, rate=lambda)
# determine Lomax quantiles accordingly parameterized:
x <- qlomax(ppoints(N), scale=1/gammascale, shape=gammashape)
# compare distributions in a Q-Q-plot:
plot(x, sort(y), log="xy", main="quantile-quantile plot",
     xlab="theoretical quantile", ylab="empirical quantile")
abline(0, 1, col="red")
```

drayleigh *The Rayleigh distribution.*

Description

Rayleigh density, distribution, quantile function, random number generation, and expectation and variance.

Usage

```
drayleigh(x, scale=1, log=FALSE)
prayleigh(q, scale=1)
qrayleigh(p, scale=1)
rrayleigh(n, scale=1)
erayleigh(scale=1)
vrayleigh(scale=1)
```

Arguments

x, q	quantile.
p	probability.
n	number of observations.
scale	scale parameter (> 0).
log	logical; if TRUE, logarithmic density will be returned.

Details

The Rayleigh distribution arises as the distribution of the square root of an exponentially distributed (or χ^2_2-distributed) random variable. If X follows an exponential distribution with rate λ and expectation $1/\lambda$, then $Y = \sqrt{X}$ follows a Rayleigh distribution with scale $\sigma = 1/\sqrt{2\lambda}$ and expectation $\sqrt{\pi/(4\lambda)}$.

Note that the exponential distribution is the *maximum entropy distribution* among distributions supported on the positive real numbers and with a pre-specified expectation; so the Rayleigh distribution gives the corresponding distribution of its square root.

Value

'drayleigh()' gives the density function, 'prayleigh()' gives the cumulative distribution function (CDF), 'qrayleigh()' gives the quantile function (inverse CDF), and 'rrayleigh()' generates random deviates. The 'erayleigh()' and 'vrayleigh()' functions return the corresponding Rayleigh distribution's expectation and variance, respectively.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

N.L. Johnson, S. Kotz, N. Balakrishnan. *Continuous univariate distributions*, volume 1. Wiley, New York, 2nd edition, 1994.

See Also

dexp, dlomax, dhalfnormal, dhalft, dhalfcauchy, TurnerEtAlPrior, RhodesEtAlPrior, bayesmeta.

Examples

```
########################
# illustrate densities:
x <- seq(0,6,le=200)
plot(x, drayleigh(x, scale=0.5), type="l", col="green",
     xlab=expression(tau), ylab=expression("probability density "*f(tau)))
lines(x, drayleigh(x, scale=1/sqrt(2)), col="red")
lines(x, drayleigh(x, scale=1), col="blue")
abline(h=0, v=0, col="grey")

##############################################
# illustrate exponential / Rayleigh connection
# via a quantile-quantile plot (Q-Q-plot):
N <- 10000
exprate <- 5
plot(sort(sqrt(rexp(N, rate=exprate))),
     qrayleigh(ppoints(N), scale=1/sqrt(2*exprate)))
abline(0, 1, col="red")

##############################################
# illustrate Maximum Entropy distributions
# under similar but different constraints:
mu <- 0.5
tau <- seq(0, 4*mu, le=100)
plot(tau, dexp(tau, rate=1/mu), type="l", col="red", ylim=c(0,1/mu),
     xlab=expression(tau), ylab="probability density")
lines(tau, drayleigh(tau, scale=1/sqrt(2*1/mu^2)), col="blue")
abline(h=0, v=0, col="grey")
abline(v=mu, col="darkgrey"); axis(3, at=mu, label=expression(mu))
# explicate constraints:
legend("topright", pch=15, col=c("red","blue"),
       c(expression("Exponential:  E["*tau*"]"==mu),
         expression("Rayleigh:  E["*tau^2*"]"==mu^2)))
```

`ess` *Effective sample size (ESS)*

Description

This function computes the effective sample size (ESS) of a posterior predictive distribution.

Usage

```
  ess(object, ...)
  ## S3 method for class 'bayesmeta'
ess(object, uisd, method=c("elir", "vr", "pr", "mtm.pt"), ...)
```

Arguments

`object`	a `bayesmeta` object.
`uisd`	the *unit infomation standard deviation* (a single numerical value, *or* a `function` of the parameter (μ)).
`method`	a character string specifying the method to be used for ESS computation. By default, the expected local-information-ratio ESS (ESS_{ELIR}) is returned.
`...`	additional arguments

Details

The information conveyed by a prior distribution may often be quantified in terms of an *effective sample size (ESS)*. Meta-analyses are commonly utilized to summarize "historical" information in order to inform a future study, leading to a *meta-analytic-predictive (MAP) prior* (Schmidli et al., 2014). In the context of the normal-normal hierarchical model (NNHM), the MAP prior results as the (posterior) predictive distribution for a "new" study mean θ_{k+1}. This function computes the ESS for the posterior predictive distribution based on a `bayesmeta` object.

Within the NNHM, the notion of an effective sample size requires the specification of a *unit information standard deviation (UISD)* (Roever et al., 2020); see also the '`uisd()`' function's help page. The UISD σ_u here determines the *Fisher information for one information unit*, effectively assuming that a study's sample size n_i and standard error σ_i are related simply as

$$\sigma_i = \frac{\sigma_\mathrm{u}}{\sqrt{n_i}},$$

i.e., the squared standard error is inversely proportional to the sample size. For the (possibly hypothetical) case of a sample size of $n_i = 1$, the standard error then is equal to the UISD σ_u.

Specifying the UISD as a constant is often an approximation, sometimes it is also possible to specify the UISD as a function of the parameter (μ). For example, in case the outcome in the meta-analyses are log-odds, then the UISD varies with the (log-) odds and is given by $2\cosh(\mu/2)$ (see also the example below).

The ESS may be computed or approximated in several ways. Possible choices here are:

- `"elir"`: the *expected local-information-ratio (ELIR)* method (the default),

- "vr": the *variance ratio (VR)* method,
- "pr": the *precision ratio (PR)* method,
- "mtm.pt": the *Morita-Thall-Mueller / Pennello-Thompson (MTM.PM)* method.

For more details on these see also Neuenschwander et al. (2020).

Value

The effective sample size (ESS).

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

B. Neuenschwander, S. Weber, H. Schmidli, A. O'Hagan. Predictively consistent prior effective sample sizes. *Biometrics*, **76**(2):578-587, 2020. doi:10.1111/biom.13252.

H. Schmidli, S. Gsteiger, S. Roychoudhury, A. O'Hagan, D. Spiegelhalter, B. Neuenschwander. Robust meta-analytic-predictive priors in clinical trials with historical control information. *Biometrics*, **70**(4):1023-1032, 2014. doi:10.1111/biom.12242.

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

See Also

bayesmeta, uisd.

Examples

```
# load data set:
data("BaetenEtAl2013")
print(BaetenEtAl2013)

## Not run:
# compute effect sizes (logarithmic odds) from the count data:
as <- escalc(xi=events, ni=total, slab=study,
             measure="PLO", data=BaetenEtAl2013)

# estimate the unit information standard deviation (UISD):
uisd(as, individual=TRUE)
uisd(as)  # = 2.35

# perform meta-analysis
# (using uniform priors for effect and heterogeneity):
bm <- bayesmeta(as)

# show forest plot:
forestplot(bm, zero=NA, xlab="log-odds")
```

```
# compute ESS_ELIR (based on fixed UISD):
ess(bm, uisd=2.35)  # = 45.7 patients

# compute ESS_ELIR based on UISD as a function of the log-odds:
uisdLogOdds <- function(logodds)
{
  return(2 * cosh(logodds / 2))
}

# Note: in the present example, probabilities are
# at approximately 0.25, corresponding to odds of 1:3.
uisdLogOdds(log(1/3))
# The UISD value of 2.31 roughly matches the above empirical figure.

ess(bm, uisd=uisdLogOdds)  # = 43.4 patients

## End(Not run)
```

`forest.bayesmeta` *Generate a forest plot for a* `bayesmeta` *object (based on the* `metafor` *package's plotting functions).*

Description

Generates a forest plot, showing individual estimates along with their 95 percent confidence intervals, resulting effect estimate and prediction interval.

Usage

```
  ## S3 method for class 'bayesmeta'
forest(x, xlab="effect size", refline=0, cex=1,...)
```

Arguments

`x`	a `bayesmeta` object.
`xlab`	title for the x-axis.
`refline`	value at which a vertical 'reference' line should be drawn (default is 0). The line can be suppressed by setting this argument to 'NA'.
`cex`	character and symbol expansion factor.
`...`	other arguments.

Details

Generates a simple forest plot illustrating the underlying data and resulting estimates (effect estimate and prediction interval).

Note

This function requires the **metafor** package to be installed.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Lewis and M. Clarke. Forest plots: trying to see the wood and the trees. *BMJ*, **322**:1479, 2001. doi:10.1136/bmj.322.7300.1479.

R.D. Riley, J.P. Higgins and J.J. Deeks. Interpretation of random effects meta-analyses. *BMJ*, **342**:d549, 2011. doi:10.1136/bmj.d549.

See Also

bayesmeta, forest.default, addpoly, forestplot.bayesmeta

Examples

```
data("CrinsEtAl2014")

## Not run:
# compute effect sizes (log odds ratios) from count data
# (using "metafor" package's "escalc()" function):
require("metafor")
es.crins <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)
# derive a prior distribution for the heterogeneity:
tp.crins <- TurnerEtAlPrior("surgical", "pharma", "placebo / control")
# perform meta-analysis:
ma.crins <- bayesmeta(es.crins, tau.prior=tp.crins$dprior)

########
# plot:
forest(ma.crins, xlab="log odds ratio")

forest(ma.crins, trans=exp, refline=1, xlab="odds ratio")

## End(Not run)
```

`forestplot.bayesmeta` *Generate a forest plot for a* `bayesmeta` *object (based on the* `forestplot` *package's plotting functions).*

Description

Generates a forest plot, showing individual estimates along with their 95 percent confidence intervals, shrinkage intervals, resulting effect estimate and prediction interval.

Usage

```
  ## S3 method for class 'bayesmeta'
forestplot(x, labeltext, exponentiate=FALSE,
           prediction=TRUE, shrinkage=TRUE, heterogeneity=TRUE,
           digits=2, plot=TRUE,
           fn.ci_norm, fn.ci_sum, col, legend=NULL, boxsize, ...)
```

Arguments

`x`	a `bayesmeta` object.
`labeltext`	an (alternative) "`labeltext`" argument which is then handed on to the `forestplot()` function (see the help there). You can use this to change contents or add columns to the displayed table; see the example below.
`exponentiate`	a logical flag indicating whether to exponentiate numbers (effect sizes) in table and plot.
`prediction`	a logical flag indicating whether to show the prediction interval below the mean estimate.
`shrinkage`	a logical flag indicating whether to show shrinkage intervals along with the quoted estimates.
`heterogeneity`	a logical flag indicating whether to quote the heterogeneity estimate and CI (at the bottom left).
`digits`	The number of significant digits to be shown. This is interpreted relative to the standard errors of all estimates.
`plot`	a logical flag indicating whether to actually generate a plot.
`fn.ci_norm`, `fn.ci_sum`, `col`, `legend`, `boxsize`, `...`	other arguments passed on to the **forestplot** package's `forestplot` function (see also the help there).

Details

Generates a forest plot illustrating the underlying data and resulting estimates (effect estimate and prediction interval, as well as shrinkage estimates and intervals).

Value

A list containing the following elements:

`data`	a `matrix` of estimates and CIs.
`shrinkage`	a `matrix` of shrinkage estimates and CIs.
`labeltext`	a `matrix` of table entries.
`forestplot`	result of the call to the '`forestplot()`' function.

Note

This function is based on the **forestplot** package's "`forestplot()`" function.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Lewis and M. Clarke. Forest plots: trying to see the wood and the trees. *BMJ*, **322**:1479, 2001. doi:10.1136/bmj.322.7300.1479.

C. Guddat, U. Grouven, R. Bender and G. Skipka. A note on the graphical presentation of prediction intervals in random-effects meta-analyses. *Systematic Reviews*, **1**(34), 2012. doi:10.1186/2046-4053134.

R.D. Riley, J.P. Higgins and J.J. Deeks. Interpretation of random effects meta-analyses. *BMJ*, **342**:d549, 2011. doi:10.1136/bmj.d549.

See Also

`bayesmeta`, `forestplot`, `forest.bayesmeta`, `plot.bayesmeta`.

Examples

```
# load data:
data("CrinsEtAl2014")

## Not run:
# compute effect sizes (log odds ratios) from count data
# (using "metafor" package's "escalc()" function):
require("metafor")
crins.es <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)
print(crins.es)

# perform meta analysis:
crins.ma <- bayesmeta(crins.es, tau.prior=function(t){dhalfcauchy(t,scale=1)})
```

```
########################
# generate forest plots
require("forestplot")

# default options:
forestplot(crins.ma)

# exponentiate values (shown in table and plot), show vertical line at OR=1:
forestplot(crins.ma, expo=TRUE, zero=1)

# logarithmic x-axis:
forestplot(crins.ma, expo=TRUE, xlog=TRUE)

# omit prediction interval:
forestplot(crins.ma, predict=FALSE)

# omit shrinkage intervals:
forestplot(crins.ma, shrink=FALSE)

# show more decimal places:
forestplot(crins.ma, digits=3)

# change table values:
# (here: add columns for event counts)
fp <- forestplot(crins.ma, expo=TRUE, plot=FALSE)
labtext <- fp$labeltext
labtext <- cbind(labtext[,1],
                 c("treatment",
              paste0(CrinsEtAl2014[,"exp.AR.events"], "/", CrinsEtAl2014[,"exp.total"]),
                   "",""),
                 c("control",
             paste0(CrinsEtAl2014[,"cont.AR.events"], "/", CrinsEtAl2014[,"cont.total"]),
                   "",""),
                 labtext[,2:3])
labtext[1,4] <- "OR"
print(fp$labeltext) # before
print(labtext)      # after
forestplot(crins.ma, labeltext=labtext, expo=TRUE, xlog=TRUE)

# see also the "forestplot" help for more arguments that you may change,
# e.g. the "clip", "xticks", "xlab" and "title" arguments,
# or the "txt_gp" argument for label sizes etc.:
forestplot(crins.ma, clip=c(-4,1), xticks=(-3):0,
           xlab="log-OR", title="pediatric transplantation example",
           txt_gp = fpTxtGp(ticks = gpar(cex=1), xlab = gpar(cex=1)))

## End(Not run)
```

`forestplot.bmr` *Generate a forest plot for a* `bmr` *object (based on the* `forestplot` *package's plotting functions).*

Description

Generates a forest plot, showing individual estimates along with their 95 percent confidence intervals, shrinkage intervals, resulting effect estimates and prediction intervals.

Usage

```
  ## S3 method for class 'bmr'
forestplot(x, X.mean, X.prediction,
           labeltext, exponentiate=FALSE,
           shrinkage=TRUE, heterogeneity=TRUE,
           digits=2, decplaces.X, plot=TRUE,
           fn.ci_norm, fn.ci_sum, col, legend=NULL, boxsize, ...)
```

Arguments

`x`	a `bmr` object.
`X.mean`	a regressor matrix (X) for effect estimates that are to be shown in the plot. By default, a diagonal matrix; set to `NULL` in order to suppress showing summary estimates. The matrix' row names define the labels shown in the plot.
`X.prediction`	an optional regressor matrix (X) for predictions that are to be shown in the plot. The matrix' row names define the labels shown in the plot.
`labeltext`	an (alternative) "`labeltext`" argument which is then handed on to the `forestplot()` function (see the help there). You can use this to change contents or add columns to the displayed table; see also the example below.
`exponentiate`	a logical flag indicating whether to exponentiate numbers (effect sizes) in table and plot.
`shrinkage`	a logical flag indicating whether to show shrinkage intervals along with the quoted estimates.
`heterogeneity`	a logical flag indicating whether to quote the heterogeneity estimate and CI (at the bottom left of the plot).
`digits`	the number of significant digits to be shown. This is interpreted relative to the standard errors of all estimates.
`decplaces.X`	The number of decimal places to be shown for the regressors.
`plot`	a logical flag indicating whether to actually generate a plot.
`fn.ci_norm`, `fn.ci_sum`, `col`, `legend`, `boxsize`, `...`	other arguments passed on to the **forestplot** package's `forestplot` function (see also the help there).

Details

Generates a forest plot illustrating the underlying data and resulting estimates (effect estimates and/or prediction intervals, as well as shrinkage estimates and intervals). For effect estimates and prediction intervals, regressor matrices (X) need to be supplied via the '`X.mean`' or '`X.prediction`' arguments. Effect estimates are shown as diamonds, predictions are shown as horizontal bars.

Value

A list containing the following elements:

`data`	a `matrix` of estimates and CIs.
`X.mean`, `X.prediction`	the '`X.mean`' and '`X.prediction`' arguments.
`shrinkage`	a `matrix` of shrinkage estimates and CIs.
`labeltext`	a `matrix` of table entries.
`forestplot`	result of the call to the '`forestplot()`' function.

Note

This function is based on the **forestplot** package's "`forestplot()`" function.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Lewis and M. Clarke. Forest plots: trying to see the wood and the trees. *BMJ*, **322**:1479, 2001. doi:10.1136/bmj.322.7300.1479.

C. Guddat, U. Grouven, R. Bender and G. Skipka. A note on the graphical presentation of prediction intervals in random-effects meta-analyses. *Systematic Reviews*, **1**(34), 2012. doi:10.1186/2046-4053134.

R.D. Riley, J.P. Higgins and J.J. Deeks. Interpretation of random effects meta-analyses. *BMJ*, **342**:d549, 2011. doi:10.1136/bmj.d549.

See Also

`bayesmeta`, `forestplot`, `forestplot.bayesmeta`, `forestplot.escalc`.

Examples

```
## Not run:
#####################################################################
# perform a meta-analysis using binary ("indicator") covariables:

# load data:
data("CrinsEtAl2014")
# compute effect measures (log-OR):
crins.es <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)
# show data:
crins.es[,c("publication", "IL2RA", "exp.AR.events", "exp.total",
            "cont.AR.events", "cont.total", "yi", "vi")]
# specify regressor matrix (binary indicator variables):
X <- cbind("basiliximab"=as.numeric(crins.es$IL2RA=="basiliximab"),
           "daclizumab" =as.numeric(crins.es$IL2RA=="daclizumab"))
print(X)
# perform meta-analysis:
bmr01 <- bmr(crins.es, X=X,
             tau.prior=function(t){dhalfnormal(t, scale=0.5)})

# show forest plot:
forestplot(bmr01)

# show forest plot including contrast
# (difference between the two groups):
forestplot(bmr01,
           X.mean=rbind("basiliximab"      = c(1, 0),
                        "daclizumab"       = c(0, 1),
                        "group difference" = c(-1, 1)))

#############################################
# perform the meta-analysis using a different
# ("intercept / slope") regressor setup:
X <- cbind("intercept"=1,
           "offset.dac"=as.numeric(crins.es$IL2RA=="daclizumab"))
print(X)
# perform meta-analysis:
bmr02 <- bmr(crins.es, X=X,
             tau.prior=function(t){dhalfnormal(t, scale=0.5)})

# show default forest plot:
forestplot(bmr02)

# show forest plot including both group means and their difference:
forestplot(bmr02,
           X.mean=rbind("basiliximab"      = c(1, 0),
                        "daclizumab"       = c(1, 1),
                        "group difference" = c(0, 1)))
```

```
#################################################################
# a meta analysis using a continuous regressor
# and including prediction:
help("NicholasEtAl2019")
# load data:
data("NicholasEtAl2019")
# compute effect sizes (logarithic odds) from count data:
es <- escalc(measure="PLO",
             xi=patients*(prog.percent/100), ni=patients,
             slab=study, data=NicholasEtAl2019)
# set up regressor matrix:
X <- cbind("intercept2000" = 1, "year" = (es$year-2000))
# perform analysis:
bmr03 <- bmr(es, X=X)
# show forest plot including some mean estimates for the
# years from 1990 to 2018, and a prediction for 2019:
forestplot(bmr03,
           X.mean=rbind("intercept (2000)"  = c(1,   0),
                        "annual change"     = c(0,   1),
                        "change per decade" = c(0,  10),
                        "mean 1990"         = c(1, -10),
                        "mean 2000"         = c(1,   0),
                        "mean 2010"         = c(1,  10),
                        "mean 2018"         = c(1,  18)),
           X.predict=rbind("prediction 2019"   = c(1,  19)),
           xlab="log-odds",
           txt_gp = fpTxtGp(ticks = gpar(cex=1), xlab = gpar(cex=1)))

# the shown summaries and predictions may also be computed "manually";
# mean effect (year 2018), posterior median and 95 percent CI:
bmr03$qpredict(p=0.5, x=c(1, 18))
bmr03$pred.interval(level=0.95, x=c(1, 18))

# prediction (year 2019), posterior median and 95 percent CI:
bmr03$qpredict(p=0.5, x=c(1, 19), mean=FALSE)
bmr03$pred.interval(level=0.95, x=c(1, 19), mean=FALSE)

# means and predictions may also be derived
# using the "summary()" function:
summary(bmr03,
        X.mean=rbind("intercept (2000)"  = c(1,   0),
                     "annual change"     = c(0,   1),
                     "change per decade" = c(0,  10),
                     "mean 1990"         = c(1, -10),
                     "mean 2000"         = c(1,   0),
                     "mean 2010"         = c(1,  10),
                     "mean 2018"         = c(1,  18)),
        X.predict=rbind("prediction 2019"   = c(1,  19)))


#########################################################
# the tabular part of the forest plot may also be changed;
# draw a default plot:
```

```
forestplot(bmr03)
# don't plot, only extract the tabular bits:
fp <- forestplot(bmr03, plot=FALSE)
labtxt <- fp$labeltext
head(labtxt)

# drop two columns:
labtxt <- labtxt[,-c(2,3)]
# add two new columns:
labtxt <- cbind(labtxt[,1],
                c("year", es$year, "",""),
                c("events / total",
                  paste(round(es$patients*(es$prog.percent/100)),
                  "/", es$patients), "",""),
                labtxt[,2:3])
head(labtxt)
# draw new forest plot:
forestplot(bmr03, labeltext=labtxt, xlab="log-odds")

## End(Not run)
```

`forestplot.escalc` *Generate a forest plot for an* `escalc` *object (based on the* `forestplot` *package's plotting functions).*

Description

Generates a forest plot, showing estimates along with their 95 percent confidence intervals.

Usage

```
  ## S3 method for class 'escalc'
forestplot(x, labeltext, exponentiate=FALSE,
           digits=2, plot=TRUE,
           fn.ci_norm, fn.ci_sum, col, boxsize, ...)
```

Arguments

`x`	an `escalc` object.
`labeltext`	an (alternative) "`labeltext`" argument which is then handed on to the `forestplot()` function (see the help there). You can use this to change contents or add columns to the displayed table; see the example below.
`exponentiate`	a logical flag indicating whether to exponentiate numbers (effect sizes) in table and plot.
`digits`	The number of significant digits to be shown. This is interpreted relative to the standard errors of all estimates.
`plot`	a logical flag indicating whether to actually generate a plot.

`fn.ci_norm`, `fn.ci_sum`, `col`, `boxsize`, `...`
: other arguments passed on to the **forestplot** package's `forestplot` function (see also the help there).

Details

Generates a forest plot illustrating the data returned by the "`escalc()`" function (showing the estimates potentially to be meta-analyzed, but without a combined summary estimate).

Note

This function is based on the **forestplot** package's "`forestplot()`" function.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Lewis and M. Clarke. Forest plots: trying to see the wood and the trees. *BMJ*, **322**:1479, 2001. doi:10.1136/bmj.322.7300.1479.

R.D. Riley, J.P. Higgins and J.J. Deeks. Interpretation of random effects meta-analyses. *BMJ*, **342**:d549, 2011. doi:10.1136/bmj.d549.

See Also

`escalc`, `forestplot`, `forestplot.bayesmeta`, `forestplot.bmr`.

Examples

```
# load data:
data("CrinsEtAl2014")

# compute effect sizes (log odds ratios) from count data
# (using "metafor" package's "escalc()" function):
crins.es <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)
print(crins.es)

########################
# generate forest plots;
# with default settings:
forestplot(crins.es)

# exponentiate values (shown in table and plot), show vertical line at OR=1:
forestplot(crins.es, expo=TRUE, zero=1)
```

```
# logarithmic x-axis:
forestplot(crins.es, expo=TRUE, xlog=TRUE)

# show more decimal places:
forestplot(crins.es, digits=3)

# change table values:
# (here: add columns for event counts)
fp <- forestplot(crins.es, expo=TRUE, plot=FALSE)
labtext <- fp$labeltext
labtext <- cbind(labtext[,1],
                 c("treatment",
              paste0(CrinsEtAl2014[,"exp.AR.events"], "/", CrinsEtAl2014[,"exp.total"])),
                 c("control",
             paste0(CrinsEtAl2014[,"cont.AR.events"], "/", CrinsEtAl2014[,"cont.total"])),
                 labtext[,2:3])
labtext[1,4] <- "OR"
print(fp$labeltext) # before
print(labtext)      # after
forestplot(crins.es, labeltext=labtext, expo=TRUE, xlog=TRUE)

# see also the "forestplot" help for more arguments that you may change,
# e.g. the "clip", "xticks", "xlab" and "title" arguments,
# or the "txt_gp" argument for label sizes etc.:
forestplot(crins.es, clip=c(-4,1), xticks=(-3):0,
           xlab="log-OR", title="pediatric transplantation example",
           txt_gp = fpTxtGp(ticks = gpar(cex=1), xlab = gpar(cex=1)))


############################################################
# In case effects and standard errors are computed already
# (and normally one wouldn't need to call "escalc()")
# you can still use "escalc()" to assemble the plot, e.g.:

data("HinksEtAl2010")
print(HinksEtAl2010)

hinks.es <- escalc(yi=log.or, sei=log.or.se,
                   slab=study, measure="OR",
                   data=HinksEtAl2010)

forestplot(hinks.es)
```

`funnel.bayesmeta` *Generate a funnel plot for a* `bayesmeta` *object.*

Description

Generates a funnel plot, showing effect estimates (y_i) vs. their standard errors (σ_i).

Usage

```
  ## S3 method for class 'bayesmeta'
funnel(x, main=deparse(substitute(x)), xlab=expression("effect "*y[i]),
       ylab=expression("standard error "*sigma[i]),
       zero=0.0, FE=FALSE, legend=FE, ...)
```

Arguments

`x`	a `bayesmeta` object.
`main`	main title for the plot.
`xlab`	x-axis title.
`ylab`	y-axis title.
`zero`	value at which a vertical 'reference' line should be drawn (default is 0). The line can be suppressed by setting this argument to '`NA`'.
`FE`	a (`logical`) flag indicating whether the "fixed effect" (FE) funnel (for $\tau = 0$) is supposed to be shown along with the "random effects" (RE) funnel.
`legend`	a (`logical`) flag indicating whether a legend identifying 'RE' and 'FE' funnels is supposed to be shown.
`...`	other arguments passed to the `plot()` function.

Details

Generates a funnel plot of effect estimates (y_i) on the x-axis vs. their associated standard errors (σ_i) on the y-axis (Note that the y-axis is pointing downwards). For many studies (large k) and in the absence of publication (selection) bias, the plot should resemble a (more or less) symmetric "funnel" shape (Sterne *et al*, 2005). Presence of publication bias, i.e., selection bias due to the fact that more dramatic effects may have higher chances of publication than less pronounced (or less controversial) findings, may cause asymmetry in the plot; especially towards the bottom of the plot, studies may then populate a preferred corner.

Besides the k individual studies that are shown as circles, a vertical reference line is shown; its position is determined by the '`zero`' argument. The "funnel" indicated in grey shows the estimated central 95% prediction interval for "new" effect estimates y_i conditional on a particular standard error σ_i, which results from convolving the prediction interval for the *true* value θ_i with a normal distribution with variance σ_i^2. At $\sigma_i = 0$ (top of the funnel), this simply corresponds to the "plain" prediction interval for θ_i. Convolutions are computed via the `convolve()` function, using the algorithm described in Roever and Friede (2017).

By setting the "`FE=TRUE`" argument, one may request a "fixed effect" (FE) funnel along with the "random effects" (RE) funnel that is shown by default. The FE funnel is analogous to the RE funnel, except that it is based on *homogeneity* ($\tau = 0$).

Note

Especially for few studies (small k), the conclusions from a forest plot are usually not very obvious (Sterne *et al*, 2011; Terrin *et al.*, 2005). Publication bias often requires rather large sample sizes to become apparent; funnel plots should hence always be interpreted with caution.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

J.A.C. Sterne, B.J. Becker and M. Egger. The funnel plot. In: H.R. Rothstein, A.J. Sutton and M. Borenstein, eds. *Publication bias in meta-analysis - prevention, assessment and adjustments*. Wiley and Sons, 2005 (Chapter 5). doi:10.1002/0470870168.ch5.

J.A.C. Sterne *et al*. Recommendations for examining and interpreting funnel plot asymmetry in meta-analyses of randomised controlled trials. *BMJ*, **343**:d4002, 2011. doi:10.1136/bmj.d4002.

N. Terrin, C.H. Schmid and J. Lau. In an empirical evaluation of the funnel plot, researchers could not visually identify publication bias. *Journal of Clinical Epidemiology*, **58**(9):894-901, 2005. doi:10.1016/j.jclinepi.2005.01.006.

C. Roever, T. Friede. Discrete approximation of a mixture distribution via restricted divergence. *Journal of Computational and Graphical Statistics*, **26**(1):217-222, 2017. doi:10.1080/10618600.2016.1276840.

See Also

bayesmeta, funnel

Examples

```
data("dat.egger2001", package="metafor")
es <- escalc(measure="OR", ai=ai, n1i=n1i, ci=ci, n2i=n2i,
             slab=study, data=dat.egger2001)
## Not run:
bm <- bayesmeta(es)
print(bm)
forestplot(bm)
funnel(bm, xlab="logarithmic odds ratio", ylab="standard error",
       main="Egger (2001) example data")

## End(Not run)
```

GoralczykEtAl2011 *Liver transplant example data*

Description

Numbers of cases (transplant patients) and events (acute rejections, steroid resistant rejections, and deaths) in experimental and control groups of 19 studies.

Usage

```
data("GoralczykEtAl2011")
```

Format

The data frame contains the following columns:

publication	character	publication identifier (first author and publication year)
year	numeric	publication year
randomized	factor	randomization status (yes / no / not stated)
control.type	factor	type of control group ('concurrent' or 'historical')
comparison	factor	type of comparison ('IL-2RA only', 'delayed CNI', or 'no/low steroids')
IL2RA	factor	type of interleukin-2 receptor antagonist (IL-2RA) ('basiliximab' or 'daclizumab')
CNI	factor	type of calcineurin inhibitor (CNI) ('tacrolimus' or 'cyclosporine A')
MMF	factor	use of mycofenolate mofetil (MMF) (y/n)
followup	numeric	follow-up time in months
treat.AR.events	numeric	number of AR events in experimental group
treat.SRR.events	numeric	number of SRR events in experimental group
treat.deaths	numeric	number of deaths in experimental group
treat.total	numeric	number of cases in experimental group
control.AR.events	numeric	number of AR events in control group
control.SRR.events	numeric	number of SRR events in control group
control.deaths	numeric	number of deaths in control group
control.total	numeric	number of cases in control group

Details

A systematic literature review investigated the evidence on the effect of Interleukin-2 receptor antagonists (IL-2RA) and resulted in 19 controlled studies reporting acute rejection (AR) and steroid-resistant rejection (SRR) rates as well as mortality in adult liver transplant recipients.

Source

A.D. Goralczyk, N. Hauke, N. Bari, T.Y. Tsui, T. Lorf, A. Obed. Interleukin-2 receptor antagonists for liver transplant recipients: A systematic review and meta-analysis of controlled studies. *Hepatology*, **54**(2):541-554, 2011. doi:10.1002/hep.24385.

See Also

CrinsEtAl2014.

Examples

```
data("GoralczykEtAl2011")
## Not run:
# compute effect sizes (log odds ratios) from count data
# (using "metafor" package's "escalc()" function):
require("metafor")
goralczyk.es <- escalc(measure="OR",
                       ai=exp.AR.events,  n1i=exp.total,
                       ci=cont.AR.events, n2i=cont.total,
                       slab=publication, data=GoralczykEtAl2011)
print(goralczyk.es[,c(1,10,12,13,15,16,17)])
```

```
# analyze using weakly informative half-Cauchy prior for heterogeneity:
goralczyk.ma <- bayesmeta(goralczyk.es, tau.prior=function(t){dhalfcauchy(t,scale=1)})

# show summary:
print(goralczyk.ma)

# show forest plot:
forestplot(goralczyk.ma)

## End(Not run)
```

`HinksEtAl2010` *JIA example data*

Description

Log odds ratios indicating association of a genetic variant (CCR5) with juvenile idiopathic arthritis (JIA).

Usage

```
data("HinksEtAl2010")
```

Format

The data frame contains the following columns:

study	`character`	publication identifier
year	`numeric`	publication year
country	`character`	country
or	`numeric`	odds ratio (OR)
or.lower	`numeric`	lower 95 percent confidence bound for OR
or.upper	`numeric`	upper 95 percent confidence bound for OR
log.or	`numeric`	logarithmic OR
log.or.se	`numeric`	standard error of logarithmic OR

Details

Results from a genetic association study (Hinks et al, 2010) were combined with data from two additional studies (Prahalad et al., 2006; Lindner et al., 2007) in order to determine the combined evidence regarding the association of a particular genetic marker (CCR5) with juvenile idiopathic arthritis (JIA).

Source

A. Hinks et al. Association of the CCR5 gene with juvenile idiopathic arthritis. *Genes and Immunity*, **11**(7):584-589, 2010. doi:10.1038/gene.2010.25.

References

S. Prahalad et al. Association of two functional polymorphisms in the CCR5 gene with juvenile rheumatoid arthritis. *Genes and Immunity*, **7**:468-475, 2006. doi:10.1038/sj.gene.6364317.

E. Lindner et al. Lack of association between the chemokine receptor 5 polymorphism CCR5delta32 in rheumatoid arthritis and juvenile idiopathic arthritis. *BMC Medical Genetics*, **8**:33, 2007. doi:10.1186/14712350833.

C. Roever, G. Knapp, T. Friede. Hartung-Knapp-Sidik-Jonkman approach and its modification for random-effects meta-analysis with few studies. *BMC Medical Research Methodology*, 15:99, 2015. doi:10.1186/s1287401500911.

Examples

```
data("HinksEtAl2010")

## Not run:
# perform meta analysis based on weakly informative half-normal prior:
bma01 <- bayesmeta(y      = HinksEtAl2010$log.or,
                   sigma  = HinksEtAl2010$log.or.se,
                   labels = HinksEtAl2010$study,
                   tau.prior = function(t){dhalfnormal(t,scale=1.0)})

# perform meta analysis based on slightly more informative half-normal prior:
bma02 <- bayesmeta(y      = HinksEtAl2010$log.or,
                   sigma  = HinksEtAl2010$log.or.se,
                   labels = HinksEtAl2010$study,
                   tau.prior = function(t){dhalfnormal(t,scale=0.5)})

# show heterogeneity posteriors:
par(mfrow=c(2,1))
plot(bma01, which=4, prior=TRUE, taulim=c(0,1))
plot(bma02, which=4, prior=TRUE, taulim=c(0,1))
par(mfrow=c(1,1))

# show heterogeneity estimates:
rbind("half-normal(1.0)"=bma01$summary[,"tau"],
      "half-normal(0.5)"=bma02$summary[,"tau"])
# show q-profile confidence interval for tau in comparison:
require("metafor")
ma03 <- rma.uni(yi=log.or, sei=log.or.se, slab=study, data=HinksEtAl2010)
confint(ma03)$random["tau",c("ci.lb","ci.ub")]
# show I2 values in the relevant range:
tau <- seq(0, 0.7, by=0.1)
cbind("tau"=tau,
      "I2" =bma01$I2(tau=tau))

# show effect estimates:
round(rbind("half-normal(1.0)" = bma01$summary[,"mu"],
            "half-normal(0.5)" = bma02$summary[,"mu"]), 5)

# show forest plot:
forestplot(bma02)
```

```
# show shrinkage estimates:
bma02$theta

## End(Not run)
```

KarnerEtAl2014 *COPD example data*

Description

Data on several endpoints from a systematic review in *chronic obstructive pulmonary disease (COPD)*.

Usage

```
data("KarnerEtAl2014")
```

Format

The data frame contains the following columns:

study	character	publication identifier (first author and publication year)
year	numeric	publication year
duration	factor	study duration ($<$ 1 year vs. $\geq$ 1 year)
inhaler	factor	type of inhaler investigated ("dry powder" or "soft mist")
baseline.age	numeric	mean age at baseline
baseline.males	numeric	proportion of males among study participants
baseline.fev1	numeric	mean FEV1 at baseline (L)
baseline.fev1pp	numeric	mean FEV1 (percent of predicted) at baseline
baseline.pyr	numeric	mean number of pack-years (smoking history)
tiotropium.total	numeric	total number of patients in the treatment group
tiotropium.exa	numeric	number of patients with $\geq$ 1 exacerbation in the treatment group
tiotropium.sexa	numeric	number of patients with $\geq$ 1 *severe* exacerbation in the treatment group
tiotropium.hospi	numeric	number of patients with $\geq$ 1 hospitalisation (all-cause) in the treatment group
tiotropium.deaths	numeric	number of deaths in the treatment group
tiotropium.sae	numeric	number of patients with $\geq$ 1 serious adverse event (non-fatal) in the treatment group
tiotropium.dropout	numeric	number of withdrawals in the treatment group
placebo.total	numeric	total number of patients in the control group
placebo.exa	numeric	number of patients with $\geq$ 1 exacerbation in the control group
placebo.sexa	numeric	number of patients with $\geq$ 1 *severe* exacerbation in the control group
placebo.hospi	numeric	number of patients with $\geq$ 1 hospitalisation (all-cause) in the control group
placebo.deaths	numeric	number of deaths in the control group
placebo.sae	numeric	number of patients with $\geq$ 1 serious adverse event (non-fatal) in the control group
placebo.dropout	numeric	number of withdrawals in the control group
sgrq.md, **sgrq.se**	numeric	mean difference and associated standard error for *St. George's respiratory questionnaire*
fev1.md, **fev1.se**	numeric	mean difference and associated standard error for *forced expiratory volume in 1 second (*

Details

Chronic obstructive pulmonary disease (COPD) is a chronic and progressive condition characterized by recurrent exacerbation phases. Various treatment options are available, aimed at both providing relief during an acute exacerbation, and at delaying overall disease progression. A common drug used in the management of COPD is *tiotropium*, a long-acting muscarinic antagonist (LAMA), which is administered via an inhaler device.

Karner *et al.* (2014) conducted a systematic review in order to evaluate the evidence on the effects of tiotropium in comparison to placebo. 22 placebo-controlled studies were found, and a range of endpoints and subgroups were considered. The data reproduced here relate to analyses 1.1, 1.9, 1.14, 1.15, 1.19, 1.26, 1.27 and 1.28 in the original investigation. A number of study-level covariables are also provided.

Source

C. Karner, J. Chong, P. Poole. Tiotropium versus placebo for chronic obstructive pulmonary disease. *Cochrane Database of Systematic Reviews*, **7**:CD009285, 2014. doi:10.1002/14651858.CD009285.pub3.

Examples

```
data("KarnerEtAl2014")
## Not run:
# compute effect sizes (log odds ratios) from exacerbation count data
# (using the "metafor" package's "escalc()" function):
karner.exa <- escalc(measure="OR",
                     ai=tiotropium.exa, n1i=tiotropium.total,
                     ci=placebo.exa,    n2i=placebo.total,
                     slab=study, data=KarnerEtAl2014)
# show forest plot:
forestplot(karner.exa, title="exacerbation",
           exponentiate=TRUE, xlog=TRUE,
           xlab="odds ratio")

# derive St. George's Respiratory Questionnaire (SGRQ) effect sizes:
karner.sgrq <- escalc(measure="MD",
                      yi=sgrq.md, sei=sgrq.se,
                      slab=study, data=KarnerEtAl2014,
                      subset=is.finite(KarnerEtAl2014$sgrq.md))
# show forest plot:
forestplot(karner.sgrq, title="SGRQ",
           xlab="mean difference")


## End(Not run)
```

`kldiv` *Kullback-Leibler divergence of two multivariate normal distributions.*

Description

Compute the Kullback-Leiber divergence or *symmetrized* KL-divergence based on means and covariances of two normal distributions.

Usage

```
kldiv(mu1, mu2, sigma1, sigma2, symmetrized=FALSE)
```

Arguments

`mu1`, `mu2`	the two mean vectors.
`sigma1`, `sigma2`	the two covariance matrices.
`symmetrized`	logical; if `TRUE`, the *symmetrized* divergence will be returned.

Details

The Kullback-Leibler divergence (or *relative entropy*) of two probability distributions p and q is defined as the integral

$$D_{\mathrm{KL}}(p \,||\, q) = \int_{\Theta} \log\Big(\frac{p(\theta)}{q(\theta)}\Big)\, p(\theta)\, \mathrm{d}\theta.$$

In the case of two normal distributions with mean and variance parameters given by (μ_1, Σ_1) and (μ_2, Σ_2), respectively, this results as

$$D_{\mathrm{KL}}\big(p(\theta|\mu_1,\Sigma_1) \,||\, p(\theta|\mu_2,\Sigma_2)\big) = \frac{1}{2}\bigg(\mathrm{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_1-\mu_2)'\Sigma_2^{-1}(\mu_1-\mu_2) - d + \log\Big(\frac{\det(\Sigma_2)}{\det(\Sigma_1)}\Big)\bigg)$$

where d is the dimension.

The *symmetrized* divergence simply results as

$$D_{\mathrm{s}}(p \,||\, q) = D_{\mathrm{KL}}(p \,||\, q) + D_{\mathrm{KL}}(q \,||\, p).$$

Value

The divergence ($D_{\mathrm{KL}} \geq 0$ or $D_{\mathrm{s}} \geq 0$).

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

S. Kullback. *Information theory and statistics*. John Wiley and Sons, New York, 1959.

C. Roever, T. Friede. Discrete approximation of a mixture distribution via restricted divergence. *Journal of Computational and Graphical Statistics*, **26**(1):217-222, 2017. doi:10.1080/10618600.2016.1276840.

See Also

`bmr`.

Examples

```
kldiv(mu1=c(0,0), mu2=c(1,1), sigma1=diag(c(2,2)), sigma2=diag(c(3,3)))
```

`NicholasEtAl2019` *Multiple sclerosis disability progression example data*

Description

Proportions of patients with disability progression in the placebo groups of 28 studies.

Usage

```
data("NicholasEtAl2019")
```

Format

The data frame contains the following columns:

study	`character`	publication identifier (first author and publication year)
year	`numeric`	publication year
patients	`numeric`	number of placebo patients
prog.percent	`numeric`	percentage of patients with disability progression

Details

A systematic literature review investigated the characteristics of randomized placebo-controlled trials in multiple sclerosis published between 1988 and 2018 (Nicholas *et al.*, 2019). A number of trends were observed in the trial characteristics over the investigated period; one of these was a decline in the proportion of placebo patients experiencing *disability progression within 24 months* during the course of a study. The data set contains the placebo groups' sizes along with the percentages of progressing patients within that group for 28 studies. The data were originally extracted from tables or Kaplan-Meier curves.

Source

R.S. Nicholas, E. Han, J. Raffel, J. Chataway, T. Friede. Over three decades study populations in progressive multiple sclerosis have become older and more disabled, but have lower on-trial progression rates: A systematic review and meta-analysis of 43 randomised placebo-controlled trials. *Multiple Sclerosis Journal*, **25**(11):1462-1471, 2019. doi:10.1177/1352458518794063.

References

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

Examples

```
# load data:
data("NicholasEtAl2019")

# show data:
head(NicholasEtAl2019)

## Not run:
# compute effect sizes (logarithmic odds) from count data
# (note: effect of potential drop-outs is ignored here):
es <- escalc(measure="PLO",
             xi=patients*(prog.percent/100), ni=patients,
             slab=study, data=NicholasEtAl2019)

# illustrate estimates (log-odds):
forestplot(es, zero=NA, xlab="log(odds)", title="Nicholas et al. (2019) data")

# set up regressor matrix
# (note: "year" variable is re-scaled so that the intercept
# corresponds to the log-odds at year=2000):
X <- cbind("intercept2000" = 1, "year" = (es$year-2000))

# perform analysis:
bmr01 <- bmr(es, X=X)

# show results:
print(bmr01)
plot(bmr01)

# illustrate the data and time trend;
# first derive predictions from the model
# and specify corresponding "regressor matrix":
newx <- cbind(1, (1989:2019)-2000)

# compute credible intervals for the mean:
pred <- cbind("median"=bmr01$qpred(0.5, x=newx),
              bmr01$pred.interval(x=newx))

# compute prediction intervals:
map <- cbind("median"=bmr01$qpred(0.5, x=newx, mean=FALSE),
              bmr01$pred.interval(x=newx, mean=FALSE))

# draw empty plot:
plot(range(newx[,2]), range(map), type="n",
     xlab="publication year - 2000", ylab="log(odds)")

# show the 26 studies' estimates (and 95 percent CIs):
matlines(rbind(es$year, es$year)-2000,
         rbind(es$yi-qnorm(0.975)*sqrt(es$vi), es$yi+qnorm(0.975)*sqrt(es$vi)),
         col=1, lty=1)
points(es$year-2000, es$yi)
```

```
# show trend lines (and 95 percent intervals):
matlines(newx[,2], map, col="blue", lty=c(1,2,2))
matlines(newx[,2], pred, col="red", lty=c(1,2,2))
legend("topright", pch=15, col=c("red","blue"), c("mean","prediction"))

## End(Not run)
```

`normalmixture` *Compute normal mixtures*

Description

This function allows to derive density, distribution function and quantile function of a normal mixture with fixed mean (μ) and random standard deviation (σ).

Usage

```
normalmixture(density,
              cdf = Vectorize(function(x){integrate(density,0,x)$value}),
              mu = 0, delta = 0.01, epsilon = 0.0001,
              rel.tol.integrate = 2^16*.Machine$double.eps,
              abs.tol.integrate = rel.tol.integrate,
              tol.uniroot = rel.tol.integrate)
```

Arguments

`density`	the σ mixing distribution's probability density `function`.
`cdf`	the σ mixing distribution's cumulative distribution `function`.
`mu`	the normal mean (μ).
`delta`, `epsilon`	the parameters specifying the desired accuracy for approximation of the mixing distribution, and with that determining the number of σ support points being used internally. Smaller values imply greater accuracy and greater computational burden (Roever and Friede, 2017).
`rel.tol.integrate`, `abs.tol.integrate`, `tol.uniroot`	the `rel.tol`, `abs.tol` and `tol` 'accuracy' arguments that are passed to the `integrate()` or `uniroot()` functions for internal numerical integration or root finding (see also the help there).

Details

When a normal random variable

$$X|\mu,\sigma \ \sim \ \text{Normal}(\mu,\sigma^2)$$

has a fixed mean μ, but a random standard deviation

$$\sigma|\phi \ \sim \ \text{G}(\phi)$$

following some probability distribution $\mathrm{G}(\phi)$, then the *marginal distribution* of X,

$$X|\mu, \phi$$

is a *mixture distribution* (Lindsay, 1995; Seidel, 2010).

The mixture distribution's probability density function etc. result from integration and often are not available in analytical form. The `normalmixture()` function approximates density, distribution function and quantile function to some pre-set accuracy by a *discrete* mixture of normal distributions based on a finite number of σ values using the 'DIRECT' algorithm (Roever and Friede, 2017).

Either the "`density`" or "`cdf`" argument needs to be supplied. If only "`density`" is given, then the CDF is derived via integration, if only "`cdf`" is supplied, then the density function is not necessary.

In **meta-analysis** applications, mixture distributions arise e.g. in the context of **prior predictive distributions**. Assuming that a study-specific effect θ_i *a priori* is distributed as

$$\theta_i|\mu, \tau \ \sim \ \mathrm{Normal}(\mu, \tau^2)$$

with a prior distribution for the heterogeneity τ,

$$\tau|\phi \ \sim \ \mathrm{G}(\phi)$$

yields a setup completely analogous to the above one.

Since it is sometimes hard to judge what constitutes a sensible heterogeneity prior, it is often useful to inspect the implications of certain settings in terms of the corresponding *prior predictive distribution* of

$$\theta_i|\mu, \phi$$

indicating the *a priori* implied variation between studies due to heterogeneity alone based on a certain prior setup (Spiegelhalter et al., 2004, Sec. 5.7.3). Some examples using different heterogeneity priors are illustrated below.

Value

A `list` containing the following elements:

`delta`, `epsilon`	The supplied design parameters.
`mu`	the normal mean.
`bins`	the number of bins used.
`support`	the matrix containing lower, upper and reference points for each bin and its associated probability.
`density`	the mixture's density `function(x)`.
`cdf`	the mixture's cumulative distribution `function(x)` (CDF).
`quantile`	the mixture's quantile `function(p)` (inverse CDF).
`mixing.density`	the mixing distribution's density `function()` (if supplied).
`mixing.cdf`	the mixing distribution's cumulative distribution `function()`.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

B.G. Lindsay. *Mixture models: theory, geometry and applications*. Vol. 5 of *CBMS Regional Conference Series in Probability and Statistics*, Institute of Mathematical Statistics, Hayward, CA, USA, 1995.

C. Roever, T. Friede. Discrete approximation of a mixture distribution via restricted divergence. *Journal of Computational and Graphical Statistics*, **26**(1):217-222, 2017. doi:10.1080/10618600.2016.1276840.

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

W.E. Seidel. Mixture models. In: M. Lovric (ed.), *International Encyclopedia of Statistical Science*, Springer, Heidelberg, pp. 827-829, 2010.

D.J. Spiegelhalter, K.R. Abrams, J.P.Myles. *Bayesian approaches to clinical trials and health-care evaluation*. Wiley & Sons, 2004.

See Also

bayesmeta.

Examples

```
#####################################################################
# compare half-normal mixing distributions with different scales:
nm05 <- normalmixture(cdf=function(x){phalfnormal(x, scale=0.5)})
nm10 <- normalmixture(cdf=function(x){phalfnormal(x, scale=1.0)})
# (this corresponds to the case of assuming a half-normal prior
# for the heterogeneity tau)

# check the structure of the returned object:
str(nm05)

# show density functions:
# (these would be the marginal (prior predictive) distributions
# of study-specific effects theta[i])
x <- seq(-1, 3, by=0.01)
plot(x, nm05$density(x), type="l", col="blue", ylab="density")
lines(x, nm10$density(x), col="red")
abline(h=0, v=0, col="grey")

# show cumulative distributions:
plot(x, nm05$cdf(x), type="l", col="blue", ylab="CDF")
lines(x, nm10$cdf(x), col="red")
abline(h=0:1, v=0, col="grey")

# determine 5 percent and 95 percent quantiles:
rbind("HN(0.5)"=nm05$quantile(c(0.05,0.95)),
      "HN(1.0)"=nm10$quantile(c(0.05,0.95)))
```

```
#####################################################################
# compare different mixing distributions
# (half-normal, half-Cauchy, exponential and Lomax):
nmHN <- normalmixture(cdf=function(x){phalfnormal(x, scale=0.5)})
nmHC <- normalmixture(cdf=function(x){phalfcauchy(x, scale=0.5)})
nmE  <- normalmixture(cdf=function(x){pexp(x, rate=2)})
nmL  <- normalmixture(cdf=function(x){plomax(x, shape=4, scale=2)})

# show densities (logarithmic y-axis):
x <- seq(-1, 3, by=0.01)
plot(x,  nmHN$density(x), col="green",  type="l", ylab="density", ylim=c(0.005, 6.5), log="y")
lines(x, nmHC$density(x), col="red")
lines(x, nmE$density(x),  col="blue")
lines(x, nmL$density(x),  col="cyan")
abline(v=0, col="grey")

# show CDFs:
plot(x,  nmHN$cdf(x), col="green",  type="l", ylab="CDF", ylim=c(0,1))
lines(x, nmHC$cdf(x), col="red")
lines(x, nmE$cdf(x),  col="blue")
lines(x, nmL$cdf(x),  col="cyan")
abline(h=0:1, v=0, col="grey")
# add "exponential" x-axis at top:
axis(3, at=log(c(0.5,1,2,5,10,20)), lab=c(0.5,1,2,5,10,20))
# show 95 percent quantiles:
abline(h=0.95, col="grey", lty="dashed")
abline(v=nmHN$quantile(0.95), col="green", lty="dashed")
abline(v=nmHC$quantile(0.95), col="red", lty="dashed")
abline(v=nmE$quantile(0.95),  col="blue", lty="dashed")
abline(v=nmL$quantile(0.95),  col="cyan", lty="dashed")
rbind("half-normal(0.5)"=nmHN$quantile(0.95),
      "half-Cauchy(0.5)"=nmHC$quantile(0.95),
      "exponential(2.0)"=nmE$quantile(0.95),
      "Lomax(4,2)"      =nmL$quantile(0.95))


#####################################################################
# a normal mixture distribution example where the solution
# is actually known analytically: the Student-t distribution.
# If  Y|sigma ~ N(0,sigma^2),  where  sigma = sqrt(k/X)
# and  X|k ~ Chi^2(df=k),
# then the marginal  Y|k  is Student-t with k degrees of freedom.

# define CDF of sigma:
CDF <- function(sigma, df){pchisq(df/sigma^2, df=df, lower.tail=FALSE)}

# numerically approximate normal mixture (with k=5 d.f.):
k <- 5
nmT1 <- normalmixture(cdf=function(x){CDF(x, df=k)})
# in addition also try a more accurate approximation:
nmT2 <- normalmixture(cdf=function(x){CDF(x, df=k)}, delta=0.001, epsilon=0.00001)
# check: how many grid points were required?
```

```
nmT1$bins
nmT2$bins

# show true and approximate densities:
x <- seq(-2,10,le=400)
plot(x, dt(x, df=k), type="l")
abline(h=0, v=0, col="grey")
lines(x, nmT1$density(x), col="red", lty="dashed")
lines(x, nmT2$density(x), col="blue", lty="dotted")

# show ratios of true and approximate densities:
plot(x, nmT1$density(x)/dt(x, df=k), col="red",
     type="l", log="y", ylab="density ratio")
abline(h=1, v=0, col="grey")
lines(x, nmT2$density(x)/dt(x, df=k), col="blue")
```

`Peto1980` *Aspirin after myocardial infarction example data*

Description

Numbers of cases (patients) and events (deaths) in treatment and control groups of six studies.

Usage

```
data("Peto1980")
```

Format

The data frame contains the following columns:

publication	`character`	publication reference
study	`character`	study acronym or abbreviation
start, **end**	`integer`	duration of study (calendar years)
age	`numeric`	mean patient age (years)
dose	`numeric`	total daily dose (mg)
followup	`numeric`	follow-up duration (months)
treat.cases	`integer`	number of cases in treatment group
treat.events	`integer`	number of events in treatment group
control.cases	`integer`	number of cases in control group
control.events	`integer`	number of events in control group

Details

Peto (1980) investigated mortality data from six randomized, placebo-controlled clinical trials of aspirin, involving a total of 10,703 post-myocardial infarction patients. Canner (1987) later investigated potential heterogeneity between study characteristics as well as their reported estimates. The included studies' abbreviations are:

UK-1	first United Kingdom trial
CDPA	Coronary Drug Project Aspirin trial
GAMS	German-Austrian Multicentre Study
UK-2	second United Kingdom trial
PARIS	Persantine-Aspirin Reinfarction Study
AMIS	Aspirin Myocardial Infarction Study

Source

P.L. Canner. An overview of six clinical trials of aspirin in coronary heart disease. *Statistics in Medicine*, **6**(3):255-263, 1987. doi:10.1002/sim.4780060310

References

R. Peto. Aspirin after myocardial infarction. *The Lancet*, **315**(8179):1172-1173, 1980. doi:10.1016/S01406736(80)916268.

P.C. Elwood, A.L. Cochrane, M.L.Burr, P.M. Sweetnam, G. Williams, E. Welsby, S.J. Hughes, R. Renton. A randomized controlled trial of acetyl salicylic acid in the secondary prevention of mortality from myocardial infarction. *British Medical Journal*, **1**(5905):436-440, 1974. doi:10.1136/bmj.1.5905.436.

The Coronary Drug Project Research Group. Aspirin in coronary heart disease. *Journal of Chronic Diseases*, **29**(10):625-642, 1976. doi:10.1016/00219681(76)900205.

K. Breddin, D. Loew, K. Lechner, K. Ueberla, E. Walter. Secondary prevention of myocardial infarction: a comparison of acetylsalicylic acid, placebo and phenprocoumon. *Haemostasis*, **9**(6):325-344, 1980. doi:10.1159/000214375.

P.C. Elwood, P.M. Sweetnam. Aspirin and secondary mortality after myocardial infarction. *The Lancet*, **314**(8156):1313-1315, 1979. doi:10.1016/S01406736(79)928083.

Aspirin Myocardial Infarction Study Research Group. A randomized, controlled trial of aspirin in persons recovered from myocardial infarction. *Journal of the American Medical Association*, **243**(7):661-669, 1980. doi:10.1001/jama.1980.03300330019023.

The Persantine-Aspirin Reinfarction Study Research Group. Persantine and aspirin in coronary heart disease. *Circulation*, **62**(3):449-461, 1980. doi:10.1161/01.CIR.62.3.449.

Examples

```
data("Peto1980")
## Not run:
# compute effect sizes (log odds ratios) from count data
# (using "metafor" package's "escalc()" function):
require("metafor")
peto.es <- escalc(measure="OR",
                  ai=treat.events,   n1i=treat.cases,
                  ci=control.events, n2i=control.cases,
                  slab=publication, data=Peto1980)
print(peto.es)

# check sensitivity to different prior choices:
peto.ma01 <- bayesmeta(peto.es)
```

```
peto.ma02 <- bayesmeta(peto.es, tau.prior=function(t){dhalfnormal(t, scale=1)})

par(mfrow=c(2,1))
  plot(peto.ma01, which=4, prior=TRUE, taulim=c(0,1), main="uniform prior")
  plot(peto.ma02, which=4, prior=TRUE, taulim=c(0,1), main="half-normal prior")
par(mfrow=c(1,1))

# compare heterogeneity (tau) estimates:
print(rbind("uniform"    =peto.ma01$summary[,"tau"],
            "half-normal"=peto.ma02$summary[,"tau"]))

# compare effect (mu) estimates:
print(rbind("uniform"    =peto.ma01$summary[,"mu"],
            "half-normal"=peto.ma02$summary[,"mu"]))

summary(peto.ma02)
forestplot(peto.ma02)
plot(peto.ma02)

## End(Not run)
```

`plot.bayesmeta` *Generate summary plots for a* `bayesmeta` *object.*

Description

Generates a forest plot, and joint and marginal posterior density plots for the two parameters of the random-effects meta-analysis model.

Usage

```
  ## S3 method for class 'bayesmeta'
plot(x, main=deparse(substitute(x)),
     which=1:4, prior=FALSE, forest.margin=8,
     mulim=c(NA,NA), taulim=c(NA,NA),
     violin=FALSE, ...)
```

Arguments

`x`	a `bayesmeta` object.
`main`	a `character` string giving the main title for the plot(s).
`which`	an indicator of which plots to generate.
`prior`	an indicator whether to also draw the prior density in marginal posterior density plots.
`forest.margin`	the width of the margin to the left of the forest plot. This may require some manual tweaking so that the study labels fit properly.
`mulim`, `taulim`	(optional) ranges of effect (mu) and heterogeneity (tau) values to be used for plotting.

violin	an indicator whether to draw the forest plot as a "violin plot".
...	other graphical parameters.

Details

Depending on the value of the which argument, one or several plots are generated, namely

1. a forest plot, including a 95% credible interval (diamond) and a 95% prediction interval (rectangle) for the effect μ. The shown intervals for μ are based on posterior medians and shortest credible intervals (from x$summary). If violin=TRUE, the forest plot is plotted as a "violin plot", i.e., via Gaussian densities for the estimates y_i (and their associated uncertainties), and the posterior densities for the effect μ, and for the predictive distribution.
2. a plot of the joint posterior density of heterogeneity (τ) and effect (μ). Red lines trace the contours of constant density corresponding to approximate 2D credible regions (based on a χ^2-approximation to the logarithmic posterior density) as labelled. The credible regions are only an approximation based on a 'well-behaved', unimodal posterior; contour lines are omitted if the posterior mode is not finite. Blue lines show the conditional mean effect μ as a function of the heterogeneity τ (solid line) along with conditional 95% confidence bounds (dashed lines). Green lines indicate marginal medians and shortest 95% credible intervals for τ and μ.
3. the marginal posterior probability density of the effect μ with median and shortest 95% credible interval indicated. Depending on the prior argument, a dashed line showing the prior density is added. Note that for improper priors the scaling is arbitrary and may be inappropriate for the plot.
4. the marginal posterior probability density of the heterogeneity τ with median and shortest 95% credible interval indicated. Depending on the prior argument, a dashed line showing the prior density is added. Note that for improper priors the scaling is arbitrary and may be inappropriate for the plot.

The joint posterior density plot (2) especially highlights the dependence of the effect estimate on the heterogeneity parameter. In a 'conventional' frequentist meta-analysis, one would commonly first estimate the heterogeneity τ, and then fix this value and estimate the effect μ based on the assumption that the heterogeneity estimate was the true value. In the joint density plot, this would correspond to considering vertical "slices" of the parameter space, a slice at $\tau = 0$ for the fixed-effects model, and a slice a a different τ value for the random-effects model, where the blue lines would then indicate the corresponding estimate and confidence interval for μ.

Note that when using the prior=TRUE argument, the added line may end up be outside the plotted range, especially when using improper priors with arbitrary normalisation (consider adding it "manually" instead).

Value

Returns the supplied bayesmeta object (x).

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Guddat, U. Grouven, R. Bender and G. Skipka. A note on the graphical presentation of prediction intervals in random-effects meta-analyses. *Systematic Reviews*, **1**(34), 2012. doi:10.1186/2046-4053134.

R.D. Riley, J.P. Higgins and J.J. Deeks. Interpretation of random effects meta-analyses. *BMJ*, **342**:d549, 2011. doi:10.1136/bmj.d549.

See Also

bayesmeta, forestplot.bayesmeta

Examples

```
## Not run:
# example data by Snedecor and Cochran:
data("SnedecorCochran")

# analyze using a weakly informative prior
# (may take a few seconds to compute!):
ma <- bayesmeta(y=SnedecorCochran[,"mean"], sigma=sqrt(SnedecorCochran[,"var"]),
                label=SnedecorCochran[,"no"],
                mu.prior.mean=50, mu.prior.sd=50,
                tau.prior=function(x){dhalfcauchy(x, scale=10)})

# show some plots:
plot(ma, main="Snedecor/Cochran data", prior=TRUE)
plot(ma, main="Snedecor/Cochran data", which=1, violin=TRUE)

## End(Not run)
```

---

pppvalue *Posterior predictive p-values*

Description

Compute posterior or prior predictive p-values from a bayesmeta object.

Usage

```
pppvalue(x, parameter = "mu", value = 0.0,
         alternative = c("two.sided", "less", "greater"),
         statistic = "median",
         rejection.region,
         n = 10,
         prior = FALSE,
         quietly = FALSE,
         parallel, seed, ...)
```

Arguments

`x`	a `bayesmeta` object.
`parameter`	the parameter to be tested. May be the effect (`"mu"`), the heterogeneity (`"tau"`) or one of the study-specific (θ_i) parameters denoted by their label or their index.
`value`	the (null-) hypothesized value.
`alternative`	the type of alternative hypothesis.
`statistic`	the figure to be used a the 'test statistic', or 'discrepancy variable'. May be chosen as `"t"`, `"Q"` or `"cdf"`, or among the row names of the `bayesmeta` object's '`...$summary`' element. *Or* it may be specified as a `function`. For details, see below.
`rejection.region`	the test statistic's rejection region. May be one of `"upper.tail"`, `"lower.tail"` or `"two.tailed"`. If unspecified, it is set automatically based on the '`alternative`' and '`statistic`' parameters.
`n`	the number of Monte Carlo replications to be generated. The default value is `n=10`, but in practice a substantially larger value should be appropriate.
`prior`	a logical flag to request *prior predictive* (instead of *posterior predictive*) p-values. Prior predictive values are only available for hypotheses concerning the effect (μ) and heterogeneity (τ) parameters.
`quietly`	a logical flag to show (or suppress) output during computation; this may also speed up computations slightly.
`parallel`	the number of parallel processes to utilize. By default, if multiple (k) cores are detected, then k-1 parallel processes are used.
`seed`	(optional) an `integer` random seed value to generate reproducible results.
`...`	further parameters passed to '`statistic`', *if* the '`statistic`' argument was specified as a `function`.

Details

Posterior predictive p-values are Bayesian analogues to 'classical' p-values (Meng, 1994; Gelman, Meng and Stern, 1996; Gelman et al., 2014). The `pppvalue()` function allows to compute these values for one- and two-sided hypotheses concerning the effect (μ) or heterogeneity (τ) parameter, or one of the study-specific effect parameters (θ_i) in a random-effects meta-analysis.

Prior predictive p-values have a similar interpretation, but they have a stronger dependence on the prior specification and are only available when the prior is proper; for a more detailed discussion, see Gelman, Meng and Stern (1996, Sec. 4).

The function may also be used to only generate samples (τ, μ, θ_i, y_i) without having to also derive a statistic or a p-value. In order to achieve that, the '`statistic`' argument may be specified as '`NA`', and generated samples may be recovered from the '`...$replicates`' output element.

p-values from Monte Carlo sampling: The computation is based on Monte Carlo sampling and repeated analysis of re-generated data sets drawn from the parameters' (conditional) posterior predictive (or prior) distribution; so the p-value derivation is somewhat computationally expensive. The p-value eventually is computed based on how far in the tail area the actual data are (in terms of the realized 'test statistic' or 'discrepancy') relative to the Monte-Carlo-sampled distribution.

Accuracy of the computed p-value hence strongly depends on the number of samples (as specified through the 'n' argument) that are generated. Also, (near-) zero p-values need to be interpreted with caution, and in relation to the used Monte Carlo sample size (n).

'Test'-statistics or 'discrepancy variables': The 'statistic' argument determines the statistic to be computed from the data as a measure of deviation from the null hypothesis. If specified as "Q", then Cochran's Q statistic is computed; this is useful for testing for homogeneity ($\tau = 0$). If specified as one of the row names of the 'x$summary' element, then, depending on the type of null hypothesis specified through the 'parameter' argument, the corresponding parameter's posterior quantity is used for the statistic. If specified as "t", then a t-type statistic is computed (the difference between the corresponding parameter's posterior mean and its hypothesized value, divided by the posterior standard deviation). If specified as "cdf", the parameter's marginal posterior cumulative distribution function evaluated a the hypothesized value ('value') is used.

The 'statistic' argument may also be specified as an arbitrary function of the data (y). The function's first argument then needs to be the data (y), additional arguments may be passed as arguments ('...') to the 'pppvalue()' function. See also the examples below.

One- and two-sided hypotheses: Specification of one- or two-sided hypotheses not only has implications for the determination of the p-value from the samples, but also for the sampling process itself. Parameter values are drawn from a subspace according to the null hypothesis, which for a two-sided test is a line, and for a one-sided test is a half-plane. This also implies that one- and two-sided p-values cannot simply be converted into one another.

For example, when specifying pppvalue(..., param="mu", val=0, alt="two.sided"), then first paramater values (τ, μ) are drawn from the conditional posterior distribution $p(\tau, \mu|y, \sigma, \mu = 0)$, and subsequently new data sets are generated based on the parameters. If a one-sided hypothesis is specified, e.g. via pppvalue(..., param="mu", val=0, alt="less"), then parameters are drawn from $p(\tau, \mu|y, \sigma, \mu > 0)$.

For a hypothesis concerning the individual effect parameters θ_i, conditions are imposed on the corresponding θ_i. For example, for a specification of pppvalue(..., param=2, val=0, alt="less"), the hypothesis concerns the i=2nd study's effect paramater θ_2. First a sample is generated from $p(\theta_2|y, \sigma, \theta_2 > 0)$. Then samples of μ and τ are generated by conditioning on the generated θ_2 value, and data y are generated by conditioning on all three.

Unless explicitly specified through the 'rejection.region' argument, the test statistic's "rejection region" (the direction in which extreme statistic values indicate a departure from the null hypothesis) is set based on the 'alternative' and 'statistic' parameters. The eventually used setting can be checked in the output's '...$rejection.region' component.

Computation: When aiming to compute a p-value, it is probably a good idea to first start with a smaller 'n' argument to get a rough idea of the p-value's order of magnitude as well as the computational speed, before going over to a larger, more realistic n value. The implementation is able to utilize multiple processors or cores via the **parallel** package; details may be specified via the 'parallel' argument.

Value

A list of class 'htest' containing the following components:

statistic	the 'test statistic' (or 'discrepancy') value based on the actual data.
parameter	the number (n) of Monte Carlo replications used.

`p.value`	the derived p-value.
`null.value`	the (null-) hypothesized parameter value.
`alternative`	the type of alternative hypothesis.
`method`	a character string indicating what type of test was performed.
`data.name`	the name of the underlying `bayesmeta` object.
`call`	an object of class `call` giving the function call that generated the `htest` object.
`rejection.region`	the test statistic's rejection region.
`replicates`	a `list` containing the replicated parameters (τ, μ, θ_i), data (y_i) and statistic, along with an indicator for those samples constituting the distribution's 'tail area'.
`computation.time`	The computation time (in seconds) used.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

X.-L. Meng. Posterior predictive p-values. *The Annals of Statistics*, **22**(3):1142-1160, 1994. doi:10.1214/aos/1176325622.

A. Gelman, X.-L. Meng, H. Stern. Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica*, **6**(4):733-760, 1996.

A. Gelman, J.B. Carlin, H.S. Stern, D.B. Dunson, A. Vehtari, D.B. Rubin. *Bayesian data analysis*. Chapman & Hall / CRC, Boca Raton, 2014.

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

See Also

`bayesmeta`, `prop.test`.

Examples

```
## Not run:
# perform a meta analysis;
# load data:
data("CrinsEtAl2014")
# compute effect sizes (log odds ratios) from count data
# (using "metafor" package's "escalc()" function):
require("metafor")
crins.srr <- escalc(measure="OR",
                    ai=exp.SRR.events,  n1i=exp.total,
                    ci=cont.SRR.events, n2i=cont.total,
                    slab=publication, data=CrinsEtAl2014, subset=c(1,4,6))
# analyze:
```

```
bma <- bayesmeta(crins.srr, mu.prior.mean=0, mu.prior.sd=4,
                 tau.prior=function(t){dhalfnormal(t, scale=0.5)})

# compute a 2-sided p-value for the effect (mu) parameter
# (note: this may take a while!):
p <- pppvalue(bma, parameter="mu", value=0, n=100)

# show result:
print(p)

# show the test statistic's distribution
# along with its actualized value:
plot(ecdf(p$replicates$statistic[,1]),
     xlim=range(c(p$statistic, p$replicates$statistic[,1])))
abline(v=p$statistic, col="red")

# show the parameter values
# drawn from the (conditional) posterior distribution:
plot(bma, which=2)
abline(h=p$null.value)                                # (the null-hypothesized mu value)
points(p$replicates$tau, p$replicates$mu, col="cyan") # (the samples)

######################################################################
#  Among the 3 studies, only the first (Heffron, 2003) was randomized.
#  One might wonder about this particular study's effect (theta[1])
#  in the light of the additional evidence and compute a one-sided
#  p-value:

p <- pppvalue(bma, parameter="Heffron", value=0, n=100, alternative="less")
print(p)

######################################################################
#  One may also define one's own 'test' statistic to be used.
#  For example, one could utilize the Bayes factor to generate
#  a p-value for the homogeneity (tau=0) hypothesis:

BF <- function(y, sigma)
{
  bm <- bayesmeta(y=y, sigma=sigma,
                  mu.prior.mean=0, mu.prior.sd=4,
                  tau.prior=function(t){dhalfnormal(t, scale=0.5)},
                  interval.type="central")
  # (central intervals are faster to compute;
  #  interval type otherwise is not relevant here)
  return(bm$bayesfactor[1,"tau=0"])
}
# NOTE: the 'bayesmeta()' arguments above should probably match
#       the specifications from the original analysis

p <- pppvalue(bma, parameter="tau", statistic=BF, value=0, n=100,
              alternative="greater", rejection.region="lower.tail",
              sigma=bma$sigma)
print(p)
```

```
#########################################################################
#  If one is only interested in generating samples (and not in test
#  statistics or p-values), one may specify the 'statistic' argument
#  as 'NA'.
#  Note that different 'parameter', 'value' and 'alternative' settings
#  imply different sampling schemes.

p <- pppvalue(bma, parameter="mu", statistic=NA, value=0,
              alternative="less", n=100)

plot(bma, which=2)
abline(h=p$null.value)
points(p$replicates$tau, p$replicates$mu, col="cyan")

## End(Not run)
```

RhodesEtAlPrior *Heterogeneity priors for continuous outcomes (standardized mean differences) as proposed by Rhodes et al. (2015).*

Description

Use the prior specifications proposed in the paper by Rhodes et al., based on an analysis of studies using standardized mean differences (SMD) that were published in the *Cochrane Database of Systematic Reviews*.

Usage

```
RhodesEtAlPrior(outcome=c(NA, "obstetric outcome",
  "resource use and hospital stay / process",
  "internal and external structure-related outcome",
 "general physical health and adverse event and pain and quality of life / functioning",
 paste("signs / symptoms reflecting continuation / end of condition and infection",
       "/ onset of new acute / chronic disease"),
 "mental health outcome", "biological marker", "various subjectively measured outcomes"),
  comparator1=c("pharmacological", "non-pharmacological", "placebo / control"),
  comparator2=c("pharmacological", "non-pharmacological", "placebo / control"),
  area=c("other", "respiratory", "cancer"))
```

Arguments

outcome	The type of outcome investigated (see below for a list of possible values). The default (NA) is the general (marginal) setting, without considering meta-analysis characteristics as covariates.
comparator1	One comparator's type.
comparator2	The other comparator's type.
area	The medical area.

Details

Rhodes et al. conducted an analysis of studies listed in the *Cochrane Database of Systematic Reviews* that were investigating standardized mean differences (SMD) as endpoints. As a result, they proposed empirically motivated log-Student-t prior distributions for the (squared!) heterogeneity parameter τ^2, depending on the particular type of outcome investigated and the type of comparison in question. The underlying t-distribution's location and scale parameters here are internally stored in a 3-dimensional array (named `RhodesEtAlParameters`) and are most conveniently accessed using the `RhodesEtAlPrior()` function.

The `outcome` argument specifies the type of outcome investigated. It may take one of the following values (partial matching is supported):

- `NA`
- `"obstetric outcomes"`
- `"resource use and hospital stay / process"`
- `"internal and external structure-related outcome"`
- `"general physical health and adverse event and pain and quality of life / functioning"`
- `"signs / symptoms reflecting continuation / end of condition and infection / onset of new acute / chronic disease"`
- `"mental health outcome"`
- `"biological marker"`
- `"various subjectively measured outcomes"`.

Specifying "`outcome=NA`" (the default) yields the *marginal* setting, without considering meta-analysis characteristics as covariates.

The `comparator1` and `comparator2` arguments together specify the type of comparison in question. These may take one of the following values (partial matching is supported):

- `"pharmacological"`
- `"non-pharmacological"`
- `"placebo / control"`.

Any combination is allowed for the `comparator1` and `comparator2` arguments, as long as not both arguments are set to `"placebo / control"`. The `area` argument specifies the medical context; possible values are:

- `"respiratory"`
- `"cancer"`
- `"other"` (the default).

Note that the location and scale parameters refer to the logarithmic (*squared*) heterogeneity parameter τ^2, which is modelled using a Student-t distribution with 5 degrees of freedom. When you want to use the prior specifications for τ, the square root, as the parameter (as is necessary when using the `bayesmeta()` function), you need to correct for the square root transformation. Taking the square root is equivalent to dividing by two on the log-scale, so the square root will still be log-Student-t distributed, but with halved location and scale parameters. The relevant transformations are already taken care of when using the resulting `$dprior()`, `$pprior()` and `$qprior()` functions; see also the example below.

Value

a list with elements

`parameters`	the location and scale parameters (corresponding to the logarithmic *squared* heterogeneity parameter τ^2 as well as τ).
`outcome.type`	the corresponding type of outcome.
`comparison.type`	the corresponding type of comparison.
`medical.area`	the medical context.
`dprior`	a `function(tau)` returning the prior density of τ.
`pprior`	a `function(tau)` returning the prior cumulative distribution function (CDF) of τ.
`qprior`	a `function(p)` returning the prior quantile function (inverse CDF) of τ.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

K.M. Rhodes, R.M. Turner, J.P.T. Higgins. Predictive distributions were developed for the extent of heterogeneity in meta-analyses of continuous outcome data. *Journal of Clinical Epidemiology*, **68**(1):52-60, 2015. doi:10.1016/j.jclinepi.2014.08.012.

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

See Also

`TurnerEtAlPrior`.

Examples

```
# determine prior distribution for a specific setting:
RP <- RhodesEtAlPrior("obstetric", "pharma", "placebo")
print(RP$parameters)
str(RP)
# a prior 95 percent interval for tau:
RP$qprior(c(0.025,0.975))

# the general (marginal) setting:
RP <- RhodesEtAlPrior()
print(RP$parameters)
str(RP)
# a prior 95 percent interval for tau:
RP$qprior(c(0.025,0.975))

## Not run:
```

```
# load "metafor" package:
require("metafor")
# load data:
data("dat.normand1999")
# compute effect sizes (standardized mean differences):
es <- escalc(measure="SMD", m1i=m1i, sd1i=sd1i, n1i=n1i,
             m2i=m2i, sd2i=sd2i, n2i=n2i,
             slab=source, data=dat.normand1999)

# derive appropriate prior:
RP <- RhodesEtAlPrior("resource use", "non-pharma", "non-pharma")
# show (central) prior 95 percent interval:
RP$qprior(c(0.025, 0.975))
# show prior 95 percent upper limit:
RP$qprior(0.95)

# perform meta analysis:
bma <- bayesmeta(es, tau.prior=RP$dprior)
# show results:
print(bma)
plot(bma, which=4, prior=TRUE)

## End(Not run)
```

`RobergeEtAl2017` *Aspirin during pregnancy example data*

Description

Numbers of cases (patients) and events (preeclampsia (PE) or fetal growth restriction (FGR)) in experimental and control groups of 45 studies.

Usage

```
data("RobergeEtAl2017")
```

Format

The data frame contains the following columns:

study	`character`	publication identifier (first author and publication year)
year	`numeric`	publication year
N	`numeric`	number of patients
onset	`factor`	treatment onset (up to 16 weeks' gestation or later)
dose	`numeric`	dose (mg/day)
control	`factor`	type of control group
asp.PE.events	`numeric`	number of PE events in aspirin group
asp.PE.total	`numeric`	number of PE cases in aspirin group
cont.PE.events	`numeric`	number of PE events in control group

cont.PE.total	numeric	number of PE cases in control group
asp.FGR.events	numeric	number of FGR events in aspirin group
asp.FGR.total	numeric	number of FGR cases in aspirin group
cont.FGR.events	numeric	number of FGR events in control group
cont.FGR.total	numeric	number of FGR cases in control group

Details

A systematic literature review was performed in order to summarize the evidence on effects of aspirin administered during pregnancy. Of particular interest were occurrences of *preeclampsia (PE)* and *fetal growth restriction (FGR)*. A total of 45 relevant randomized controlled trials (RCTs) were found, out of which 40 reported on PE, and 35 reported on FGR. Besides event rates, the mode of administration (treatment onset (early vs. late) and dose (in mg)) was also recorded for each study.

Source

S. Roberge, K. Nicolaides, S. Demers, J. Hyett, N. Chaillet, E. Bujold. The role of aspirin dose on the prevention of preeclampsia and fetal growth restriction: systematic review and meta-analysis. *American Journal of Obstetrics & Gynecology*, **216**(2):110-120, 2017. doi:10.1016/j.ajog.2016.09.076.

References

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

See Also

bmr, escalc, model.matrix.

Examples

```
# load data:
data("RobergeEtAl2017")
str(RobergeEtAl2017)
head(RobergeEtAl2017)

# compute effect sizes (log odds ratios) from count data
# (using the "metafor" package's "escalc()" function);
# preeclampsia (PE):
es.pe  <- escalc(measure="OR",
                 ai=asp.PE.events,  n1i=asp.PE.total,
                 ci=cont.PE.events, n2i=cont.PE.total,
                 slab=study, data=RobergeEtAl2017,
                 subset=complete.cases(RobergeEtAl2017[,7:10]))
# show forest plot:
forestplot(es.pe, title="preeclampsia (PE)")
# show "bubble plot" (bubble sizes are
# inversely proportional to standard errors):
plot(es.pe$dose, es.pe$yi, cex=1/sqrt(es.pe$vi),
```

```
     col=c("blue","red")[as.numeric(es.pe$onset)],
     xlab="dose (mg)", ylab="log-OR (PE)", main="Roberge et al. (2017)")
legend("topright", col=c("blue","red"), c("early onset", "late onset"), pch=1)

# fetal growth restriction (FGR):
es.fgr <- escalc(measure="OR",
                 ai=asp.FGR.events,  n1i=asp.FGR.total,
                 ci=cont.FGR.events, n2i=cont.FGR.total,
                 slab=study, data=RobergeEtAl2017,
                 subset=complete.cases(RobergeEtAl2017[,11:14]))
# show forest plot:
forestplot(es.fgr, title="fetal growth restriction (FGR)")
# show "bubble plot":
plot(es.fgr$dose, es.fgr$yi, cex=1/sqrt(es.fgr$vi),
     col=c("blue","red")[as.numeric(es.fgr$onset)],
     xlab="dose (mg)", ylab="log-OR (FGR)", main="Roberge et al. (2017)")
legend("topright", col=c("blue","red"), c("early onset", "late onset"), pch=1)

## Not run:
# set up regressor matrix (common intercept and slope):
X01 <- model.matrix(~ dose, data=es.pe)
colnames(X01) <- c("intercept", "slope")
print(X01)

# perform regression:
bmr01 <- bmr(es.pe, X=X01)
bmr01$summary

# set up alternative regressor matrix
# (individual intercepts and slopes for two subgroups):
X02 <- model.matrix(~ -1 + onset + onset:dose, data=es.pe)
colnames(X02) <- c("intEarly", "intLate", "slopeEarly", "slopeLate")
print(X02)

# perform regression:
bmr02 <- bmr(es.pe, X=X02)
bmr02$summary

# derive predictions from the model;
# specify corresponding "regressor matrices":
newx.early <- cbind(1, 0, seq(50, 150, by=5), 0)
newx.late  <- cbind(0, 1, 0, seq(50, 150, by=5))
# (note: columns correspond to "beta" parameters)

# compute predicted medians and 95 percent intervals:
pred.early <- cbind("median"=bmr02$qpred(0.5, x=newx.early),
                    bmr02$pred.interval(x=newx.early))
pred.late <- cbind("median"=bmr02$qpred(0.5, x=newx.late),
                    bmr02$pred.interval(x=newx.late))

# draw "bubble plot":
plot(es.pe$dose, es.pe$yi, cex=1/sqrt(es.pe$vi),
     col=c("blue","red")[as.numeric(es.pe$onset)],
```

```
     xlab="dose (mg)", ylab="log-OR (PE)", main="Roberge et al. (2017)")
legend("topright", col=c("blue","red"), c("early onset", "late onset"), pch=1)
# add predictions to bubble plot:
matlines(newx.early[,3], pred.early, col="blue", lty=c(1,2,2))
matlines(newx.late[,4], pred.late, col="red", lty=c(1,2,2))


## End(Not run)
```

`Rubin1981` *8-schools example data*

Description

SAT coaching experiments in 8 schools.

Usage

```
data("Rubin1981")
```

Format

The data frame contains the following columns:

school	`character`	school identifier
n	`integer`	number of students
effect	`numeric`	effect estimate
stderr	`numeric`	associated standard error

Details

Quoting from Gelman et al. (1997), Sec. 5.5: "A study was performed for the Educational Testing Service to analyze the effects of special coaching programs for SAT-V (Scholastic Aptitude Test-Verbal) in each of eight high schools. The outcome variable in each study was the score on a special administration of the SAT-V, a standardized multiple choice test administered by the Educational Testing Service and used to help colleges make admissions decisions; the scores can vary between 200 and 800, with mean about 500 and standard deviation about 100. The SAT examinations are designed to be resistant to short-term efforts directed specifically toward improving performance on the test; instead they are designed to reflect knowledge acquired and abilities developed over many years of education. Nevertheless, each of the eight schools in this study considered its short-term coaching program to be very successful at increasing SAT scores. Also, there was no prior reason to believe that any of the eight programs was more effective than any other or that some were more similar in effect to each other than to any other."

Source

A. Gelman, J.B. Carlin, H. Stern, and D.B. Rubin. *Bayesian data analysis*. Chapman & Hall / CRC, Boca Raton, 1997.

References

D.B. Rubin. Estimation in parallel randomized experiments. *Journal of Educational Statistics*, **6**(4):377-401, 1981. doi:10.3102/10769986006004377.

A. Gelman. Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*, **1**(3):515-534, 2006. doi:10.1214/06BA117A.

See Also

traceplot.

Examples

```
data("Rubin1981")

## Not run:
# analysis using a weakly informative half-Cauchy prior:
schools <- bayesmeta(y=Rubin1981[,"effect"], sigma=Rubin1981[,"stderr"],
                     labels=Rubin1981[,"school"],
                     tau.prior=function(x){return(dhalfcauchy(x, scale=25))})

# show summary:
summary(schools)

# show shrinkage effect for 8 individual estimates:
schools$theta
traceplot(schools)

## End(Not run)
```

SchmidliEtAl2017 *Historical variance example data*

Description

Estimates of endpoint variances from six studies.

Usage

```
data("SchmidliEtAl2017")
```

Format

The data frame contains the following columns:

study	`character`	study label
N	`integer`	total sample size
stdev	`numeric`	standard deviation estimate
df	`integer`	associated degrees of freedom

Details

Schmidli *et al.* (2017) investigated the use of information on an endpoint's variance from previous ("historical") studies for the design and analysis of a new clinical trial. As an example application, the problem of designing a trial in *wet age-related macular degeneration (AMD)* was considered. Trial design, and in particular considerations regarding the required sample size, hinge on the expected amount of variability in the primary endpoint (here: *visual acuity*, which is measured on a scale ranging from 0 to 100 via an eye test chart).

Historical data from six previous trials are available (Szabo *et al.*; 2015), each trial providing an estimate $\hat{s}_i$ of the endpoint's standard deviation along with the associated number of degrees of freedom ν_i. The standard deviations may then be modelled on the logarithmic scale, where the estimates and their associated standard errors are given by

$$y_i = \log(\hat{s}_i) \quad \text{and} \quad \sigma_i = \sqrt{\frac{1}{2\,\nu_i}}$$

The *unit information standard deviation (UISD)* for a logarithmic standard deviation then is at approximately $\frac{1}{\sqrt{2}}$.

Source

H. Schmidli, B. Neuenschwander, T. Friede. Meta-analytic-predictive use of historical variance data for the design and analysis of clinical trials. *Computational Statistics and Data Analysis*, **113**:100-110, 2017. doi:10.1016/j.csda.2016.08.007.

References

S.M. Szabo, M. Hedegaard, K. Chan, K. Thorlund, R. Christensen, H. Vorum, J.P. Jansen. Ranibizumab vs. aflibercept for wet age-related macular degeneration: network meta-analysis to understand the value of reduced frequency dosing. *Current Medical Research and Opinion*, **31**(11):2031-2042, 2015. doi:10.1185/03007995.2015.1084909.

See Also

uisd, ess.

Examples

```
# load data:
data("SchmidliEtAl2017")

# show data:
SchmidliEtAl2017

## Not run:
# derive log-SDs and their standard errors:
dat <- cbind(SchmidliEtAl2017,
             logstdev    = log(SchmidliEtAl2017$stdev),
             logstdev.se = sqrt(0.5/SchmidliEtAl2017$df))
dat
```

```
# alternatively, use "metafor::escalc()" function:
es <- escalc(measure="SDLN",
             yi=log(stdev), vi=0.5/df, ni=N,
             slab=study, data=SchmidliEtAl2017)
es

# perform meta-analysis of log-stdevs:
bm <- bayesmeta(y=dat$logstdev,
                sigma=dat$logstdev.se,
                label=dat$study,
                tau.prior=function(t){dhalfnormal(t, scale=sqrt(2)/4)})

# or, alternatively:
bm <- bayesmeta(es,
                tau.prior=function(t){dhalfnormal(t, scale=sqrt(2)/4)})

# draw forest plot (see Fig.1):
forestplot(bm, zero=NA,
           xlab="log standard deviation")

# show heterogeneity posterior:
plot(bm, which=4, prior=TRUE)

# posterior of log-stdevs, heterogeneity,
# and predictive distribution:
bm$summary

# prediction (standard deviations):
exp(bm$summary[c(2,5,6),"theta"])
exp(bm$qposterior(theta=c(0.025, 0.25, 0.50, 0.75, 0.975), predict=TRUE))

# compute required sample size (per arm):
power.t.test(n=NULL, delta=8, sd=10.9, power=0.8)
power.t.test(n=NULL, delta=8, sd=14.0, power=0.8)

# check UISD:
uisd(es, indiv=TRUE)
uisd(es)
1 / sqrt(2)

# compute predictive distribution's ESS:
ess(bm, uisd=1/sqrt(2))
# actual total sample size:
sum(dat$N)

# illustrate predictive distribution
# on standard-deviation-scale (Fig.2):
x <- seq(from=5, to=20, length=200)
plot(x, (1/x) * bm$dposterior(theta=log(x), predict=TRUE), type="l",
     xlab=expression("predicted standard deviation "*sigma[k+1]),
     ylab="predictive density")
abline(h=0, col="grey")
```

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

`SidikJonkman2007` *Postoperative complication odds example data*

Description

This data set contains the outcomes from 29 randomized clinical trials comparing the odds of postoperative complications in laparoscopic inguinal hernia repair (LIHR) versus conventional open inguinal hernia repair (OIHR).

Usage

```
data("SidikJonkman2007")
```

Format

The data frame contains the following columns:

id	`character`	identifier used in original publication by Memon et al. (2003)
id.sj	`numeric`	identifier used by Sidik and Jonkman (2007)
year	`numeric`	publication year
lihr.events	`numeric`	number of events under LIHR
lihr.cases	`numeric`	number of cases under LIHR
oihr.events	`numeric`	number of events under OIHR
oihr.cases	`numeric`	number of cases under OIHR

Details

Analysis may be done based on the logarithmic odds ratios:

log(`lihr.events`) - log(`lihr.cases-lihr.events`) - log(`oihr.events`) + log(`oihr.cases-oihr.events`)

and corresponding standard errors:

sqrt(1/`lihr.events` + 1/(`lihr.cases-lihr.events`)) + 1/`oihr.events` + 1/(`oihr.cases-oihr.events`))

(you may also leave these computations to the **metafor** package's `escalc()` function).

The data set was used to compare different estimators for the (squared) heterogeneity τ^2. The values yielded for this data set were (see Tab.1 in Sidik and Jonkman (2007)):

method of moments (MM)	0.429
variance component (VC)	0.841
maximum likelihood (ML)	0.562
restricted ML (REML)	0.598
empirical Bayes (EB)	0.703
model error variance (MV)	0.818
variation of MV (MVvc)	0.747

Source

M.A. Memon, N.J. Cooper, B. Memon, M.I. Memon, and K.R. Abrams. Meta-analysis of randomized clinical trials comparing open and laparoscopic inguinal hernia repair. *British Journal of Surgery*, **90**(12):1479-1492, 2003. doi:10.1002/bjs.4301.

References

K. Sidik and J.N. Jonkman. A comparison of heterogeneity variance estimators in combining results of studies. *Statistics in Medicine*, **26**(9):1964-1981, 2007. doi:10.1002/sim.2688.

Examples

```
data("SidikJonkman2007")
# add log-odds-ratios and corresponding standard errors:
sj <- SidikJonkman2007
sj <- cbind(sj, "log.or"=log(sj[,"lihr.events"])-log(sj[,"lihr.cases"]-sj[,"lihr.events"])
                    -log(sj[,"oihr.events"])+log(sj[,"oihr.cases"]-sj[,"oihr.events"]),
        "log.or.se"=sqrt(1/sj[,"lihr.events"] + 1/(sj[,"lihr.cases"]-sj[,"lihr.events"])
                     + 1/sj[,"oihr.events"] + 1/(sj[,"oihr.cases"]-sj[,"oihr.events"])))

## Not run:
# analysis using weakly informative Cauchy prior
# (may take a few seconds to compute!):
ma <- bayesmeta(y=sj[,"log.or"], sigma=sj[,"log.or.se"], label=sj[,"id.sj"],
                tau.prior=function(t){dhalfcauchy(t,scale=1)})

# show heterogeneity's posterior density:
plot(ma, which=4, main="Sidik/Jonkman example", prior=TRUE)

# show some numbers (mode, median and mean):
abline(v=ma$summary[c("mode","median","mean"),"tau"], col="blue")

# compare with Sidik and Jonkman's estimates:
sj.estimates <- sqrt(c("MM"  = 0.429,   # method of moments estimator
                       "VC"  = 0.841,   # variance component type estimator
                       "ML"  = 0.562,   # maximum likelihood estimator
                       "REML"= 0.598,   # restricted maximum likelihood estimator
                       "EB"  = 0.703,   # empirical Bayes estimator
                       "MV"  = 0.818,   # model error variance estimator
                       "MVvc"= 0.747))  # a variation of the MV estimator
abline(v=sj.estimates, col="red", lty="dashed")

# generate forest plot:
fp <- forestplot(ma, exponentiate=TRUE, plot=FALSE)
# add extra columns for ID and year:
labtext <- fp$labeltext
labtext[1,1] <- "ID 2"
labtext[31:32,1] <- ""
labtext <- cbind(c("ID 1", SidikJonkman2007[,"id"], "mean","prediction"),
                 labtext[,1],
                 c("year", as.character(SidikJonkman2007[,"year"]), "", ""),
                 labtext[,-1])
```

```
# plot:
forestplot(ma, labeltext=labtext, exponentiate=TRUE,
           xlog=TRUE, xlab="odds ratio", xticks=c(0.1,1,10))


## End(Not run)
```

SnedecorCochran *Artificial insemination of cows example data*

Description

This data set gives means and (squared) standard errors of percentages of conceptions obtained from samples for six bulls.

Usage

```
data("SnedecorCochran")
```

Format

The data frame contains the following columns:

no	`character`	identifier
n	`numeric`	sample size
mean	`numeric`	mean
var	`numeric`	variance (squared standard error)

Details

Quoting from Snedecor and Cochran (1967), Sec. 10.18: "In research on artificial insemination of cows, a series of semen samples from a bull are sent out and tested for their ability to produce conceptions. The following data from a larger set kindly supplied by Dr. G. W. Salisbury, show the percentages of conceptions obtained from the samples for six bulls."

Source

J. Hartung, G. Knapp, and B.K. Sinha. *Statistical meta-analysis with applications*. Wiley, Hoboken, NJ, USA, 2008.

References

G.W. Snedecor and W.G. Cochran. *Statistical Methods*. Iowa State University Press, Ames, IA, USA, 6th edition, 1967.

Examples

```
data("SnedecorCochran")

## Not run:
# analyze using uniform prior:
bma1 <- bayesmeta(y=SnedecorCochran[,"mean"],
                  sigma=sqrt(SnedecorCochran[,"var"]),
                  label=SnedecorCochran[,"no"],
                  tau.prior="uniform")

# analyze using Jeffreys prior:
bma2 <- bayesmeta(y=SnedecorCochran[,"mean"],
                  sigma=sqrt(SnedecorCochran[,"var"]),
                  label=SnedecorCochran[,"no"],
                  tau.prior="Jeffreys")

# compare results:
print(bma1)
print(bma2)

forestplot(bma1)
forestplot(bma2)

## End(Not run)
```

---

summary.bmr *Summarizing a bmr object).*

Description

Summarizes a bmr object, and (potentially) computes means and predictions.

Usage

```
  ## S3 method for class 'bmr'
summary(object, X.mean, X.prediction, ...)
```

Arguments

object	a bmr object.
X.mean	a regressor matrix (X) for effect estimates that are to be derived. The matrix' row names define the labels passed on to the results.
X.prediction	an optional regressor matrix (X) for predictions that are to be derived. The matrix' row names define the labels passed on to the results.
...	other arguments.

Details

Prints details of the supplied bmr oject.

Specification of the (optional) "X.mean" or "X.prediction" arguments allows to request computation of mean estimates or predictions corresponding to the supplied regressor matrices. Estimates (mode, median, mean, standard deviation, and 95 percent CI) may be retrieved from the returned object's "mean" or "prediction" elements (see example below).

Value

A list (of class summary.bmr) containing the following elements:

bmr	the supplied bmr object.
call	an object of class call giving the function call that generated the summary.bmr object.
X.mean, X.prediction	the 'X.mean' and 'X.prediction' arguments.
mean, prediction	mean and predictions estimates (mode, median, mean, sd, and 95 percent credible intervals)

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

Examples

```
## Not run:
# perform a meta-analysis using binary ("indicator") covariables;
# load data:
data("CrinsEtAl2014")
# compute effect measures (log-OR):
crins.es <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)

# specify regressor matrix (binary indicator variables):
X <- cbind("basiliximab"=as.numeric(crins.es$IL2RA=="basiliximab"),
           "daclizumab" =as.numeric(crins.es$IL2RA=="daclizumab"))
print(X)

# perform meta-analysis:
bmr01 <- bmr(crins.es, X=X,
             tau.prior=function(t){dhalfnormal(t, scale=0.5)})
```

```
# show summary:
summary(bmr01)

# show summary with additional estimates and predictions:
summary(bmr01,
        X.mean = rbind("basiliximab" = c(1,0),
                       "daclizumab"  = c(0,1),
                       "difference"  = c(-1,1)),
        X.pred = rbind("basiliximab" = c(1,0),
                       "daclizumab"  = c(0,1)))

# compute mean estimates
smry <- summary(bmr01,
                X.mean = rbind("basiliximab" = c(1,0),
                               "daclizumab"  = c(0,1),
                               "difference"  = c(-1,1)))
# show mean estimates:
smry$mean

## End(Not run)
```

`traceplot` *Illustrate conditional means of study-specific estimates as well as overall mean (or other linear combinations) as a function of heterogeneity.*

Description

Generates a trace plot of study-specific (shrinkage) estimates as a function of the heterogeneity (τ), along with conditional estimates of the overall mean or other linear combinations of regression parameters. The heterogeneity's posterior distribution is also shown at the bottom.

Usage

```
  traceplot(x, ...)
  ## S3 method for class 'bayesmeta'
traceplot(x, mulim, taulim, ci=FALSE,
          ylab="effect", prior=FALSE, infinity=FALSE,
          rightmargin=8, col=rainbow(x$k), labcol=col,
          meanlabel="overall mean", meancol="black",
          meanlabcol=meancol, ...)
  ## S3 method for class 'bmr'
traceplot(x, mulim, taulim, ci=FALSE,
          ylab="effect", prior=FALSE, infinity=FALSE,
          rightmargin=8, col=rainbow(x$k), labcol=col,
          X, Xlabels, Xcols="black", Xlabcols=Xcols, ...)
```

Arguments

x	a bayesmeta or bmrobject.
mulim, taulim	(optional) ranges for the effect (mu) and heterogeneity (tau) axes. If only one value is given for taulim, then this is taken as the upper limit, and the lower limit is zero.
ci	a logical flag indicating whether to also show (conditional) confidence intervals.
ylab	a label for the effect (mu) axis.
prior	a logical flag indicating whether to show the (heterogeneity) prior density along with its posterior.
infinity	a logical flag indicating whether add an "infinity" tickmark to the heterogeneity (tau) axis and show the corresponding limiting values.
rightmargin	an additional margin to be added to the right side of the plot, in order to accomodate the estimates' labels. In case study labels still extend beyond the figure margin, try increasing this number.
col	colors to be used for plotting the (k) estimates.
labcol	colors to be used for labelling the (k) estimates.
meanlabel	a label for the overall mean estimate (traceplot.bayesmeta()).
meancol	colour specification for the overall mean estimate (traceplot.bayesmeta()).
meanlabcol	colour specification for the overall mean label (traceplot.bayesmeta()).
X	matrix (or vector) of coefficients defining linear combinations of regression parameters to be shown (traceplot.bmr()).
Xlabels	labels for the linear combinations (traceplot.bmr()).
Xcols	colour specification for the linear combinations (traceplot.bmr()).
Xlabcols	colour specification for the linear combination labels (traceplot.bmr()).
...	other arguments passed on to the plot() function.

Details

For a given heterogeneity (τ) value, the *conditional* posterior distributions of the overall effect (μ) as well as the study-specific parameters (θ_i) are again normal. The conditional normal moments (mean and variance) then vary as functions of the heterogeneity; for large heterogeneity, the shrinkage estimates approach the original data (y_i), while the overall mean approaches an un-weighted overall average. For small heterogeneity, both overall mean as well as study-specific estimates are increasingly *shrunk* towards the inverse-variance-weighted 'common-effect' estimate (Roever, 2020).

This trace plot illustrates the conditional (overall and study-specific) estimates along with the heterogeneity's posterior distribution (density) in a layout similar to that utilized by Rubin (1981).

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, D. Rindskopf, T. Friede. How trace plots help interpret meta-analysis results. *(submitted for publication)*, 2023. https://arxiv.org/abs/2306.17043.

C. Roever. Bayesian random-effects meta-analysis using the bayesmeta R package. *Journal of Statistical Software*, **93**(6):1-51, 2020. doi:10.18637/jss.v093.i06.

C. Roever, T. Friede. Using the bayesmeta R package for Bayesian random-effects meta-regression. *Computer Methods and Programs in Biomedicine*, **299**:107303, 2023. doi:10.1016/j.cmpb.2022.107303.

D.B. Rubin. Estimation in parallel randomized experiments. *Journal of Educational Statistics*, **6**(4):377-401, 1981. doi:10.3102/10769986006004377.

DuMouchel, W. H. (1994). Hierarchical Bayes linear models for meta-analysis. Technical Report 27, National Institute of Statistical Sciences (NISS); Research Triangle Park, NC, USA. https://www.niss.org/research/technical-reports/hierarchical-bayes-linear-models-meta-analysis-1994

See Also

bayesmeta, bmr.

Examples

```
## Not run:
########################
# SAT coaching example;
# load example data:
data("Rubin1981")

# perform meta-analysis:
bma01 <- bayesmeta(y=Rubin1981[,"effect"], sigma=Rubin1981[,"stderr"],
                   labels=Rubin1981[,"school"], tau.prior="uniform")

# show meta-analysis results:
forestplot(bma01)

# show trace plot:
traceplot(bma01)


#################################
# COPD (meta-regression) example;
# load example data,
# compute effect sizes (log-ORs):
data("KarnerEtAl2014")
karner.exa <- escalc(measure="OR",
                     ai=tiotropium.exa, n1i=tiotropium.total,
                     ci=placebo.exa,    n2i=placebo.total,
                     slab=study, data=KarnerEtAl2014)


################################
# perform "plain" meta-analysis:
bma02 <- bayesmeta(karner.exa,
```

```
                    tau.prior=function(t){dhalfnormal(t, scale=0.5)})

traceplot(bma02, ylab="log-OR",
          prior=TRUE, infi=TRUE, taulim=0.53)

forestplot(bma02)


############################
# perform meta-regressions:

# 1st regression;
# specify regressor matrix
# (indicator variables, "short" vs. "long" study duration):
X1 <- cbind("short" = as.numeric(karner.exa$duration == "up to 1 year"),
            "long"  = as.numeric(karner.exa$duration == "1 year or longer"))

# perform meta-regression
# (two group means, common heterogeneity):
bmr01 <- bmr(karner.exa, X=X1,
             tau.prior=function(t){dhalfnormal(t, scale=0.5)})

# show trace plot:
traceplot(bmr01, ylab="log-OR", prior=TRUE,
          taulim=0.53, mulim=c(-1, 0.2),
          X=rbind("short" = c(1,0),
                  "long"  = c(0,1)))

# 2nd regression;
# specify regressor matrix
# (baseline FEV1, an indicator of disease severity):
X2 <- cbind("intercept" = 1,
            "fev1"      = karner.exa$baseline.fev1)

# perform meta-regression
# (linear effect of FEV1 on log-OR):
bmr02 <- bmr(karner.exa, X=X2,
             tau.prior=function(t){dhalfnormal(t, scale=0.5)})

traceplot(bmr02, ylab="log-OR", prior=TRUE,
          taulim=0.53, mulim=c(-1.0, 0.2),
          X=rbind("FEV1 = 1.0"=c(1,1.0),
                  "FEV1 = 1.5"=c(1,1.5),
                  "FEV1 = 2.0"=c(1,2.0)))

## End(Not run)
```

`TurnerEtAlPrior` *(Log-Normal) heterogeneity priors for binary outcomes as proposed by Turner et al. (2015).*

Description

Use the prior specifications proposed in the paper by Turner et al., based on an analysis of studies using binary endpoints that were published in the *Cochrane Database of Systematic Reviews*.

Usage

```
TurnerEtAlPrior(outcome=c(NA, "all-cause mortality", "obstetric outcomes",
 "cause-specific mortality / major morbidity event / composite (mortality or morbidity)",
 "resource use / hospital stay / process", "surgical / device related success / failure",
  "withdrawals / drop-outs", "internal / structure-related outcomes",
  "general physical health indicators", "adverse events",
  "infection / onset of new disease",
  "signs / symptoms reflecting continuation / end of condition", "pain",
  "quality of life / functioning (dichotomized)", "mental health indicators",
  "biological markers (dichotomized)", "subjective outcomes (various)"),
  comparator1=c("pharmacological", "non-pharmacological", "placebo / control"),
  comparator2=c("pharmacological", "non-pharmacological", "placebo / control"))
```

Arguments

`outcome`	The type of outcome investigated (see below for a list of possible values).
`comparator1`	One comparator's type.
`comparator2`	The other comparator's type.

Details

Turner et al. conducted an analysis of studies listed in the *Cochrane Database of Systematic Reviews* that were investigating binary endpoints. As a result, they proposed empirically motivated log-normal prior distributions for the (squared!) heterogeneity parameter τ^2, depending on the particular type of outcome investigated and the type of comparison in question. The log-normal parameters (μ and σ) here are internally stored in a 3-dimensional array (named `TurnerEtAlParameters`) and are most conveniently accessed using the `TurnerEtAlPrior()` function.

The `outcome` argument specifies the type of outcome investigated. It may take one of the following values (partial matching is supported):

- `NA`
- `"all-cause mortality"`
- `"obstetric outcomes"`
- `"cause-specific mortality / major morbidity event / composite (mortality or morbidity)"`
- `"resource use / hospital stay / process"`
- `"surgical / device related success / failure"`
- `"withdrawals / drop-outs"`
- `"internal / structure-related outcomes"`
- `"general physical health indicators"`
- `"adverse events"`

- "infection / onset of new disease"
- "signs / symptoms reflecting continuation / end of condition"
- "pain"
- "quality of life / functioning (dichotomized)"
- "mental health indicators"
- "biological markers (dichotomized)"
- "subjective outcomes (various)"

Specifying "outcome=NA" (the default) yields the *marginal* setting, without considering meta-analysis characteristics as covariates.

The comparator1 and comparator2 arguments together specify the type of comparison in question. These may take one of the following values (partial matching is supported):

- "pharmacological"
- "non-pharmacological"
- "placebo / control"

Any combination is allowed for the comparator1 and comparator2 arguments, as long as not both arguments are set to "placebo / control".

Note that the log-normal prior parameters refer to the (*squared*) heterogeneity parameter τ^2. When you want to use the prior specifications for τ, the square root, as the parameter (as is necessary when using the bayesmeta() function), you need to correct for the square root transformation. Taking the square root is equivalent to dividing by two on the log-scale, so the square root's distribution will still be log-normal, but with halved mean and standard deviation. The relevant transformations are already taken care of when using the resulting $dprior(), $pprior() and $qprior() functions; see also the example below.

Value

a list with elements

parameters	the log-normal parameters (μ and σ, corresponding to the *squared* heterogeneity parameter τ^2 as well as τ).
outcome.type	the corresponding type of outcome.
comparison.type	the corresponding type of comparison.
dprior	a function(tau) returning the prior density of τ.
pprior	a function(tau) returning the prior cumulative distribution function (CDF) of τ.
qprior	a function(p) returning the prior quantile function (inverse CDF) of τ.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

R.M. Turner, D. Jackson, Y. Wei, S.G. Thompson, J.P.T. Higgins. Predictive distributions for between-study heterogeneity and simple methods for their application in Bayesian meta-analysis. *Statistics in Medicine*, **34**(6):984-998, 2015. doi:10.1002/sim.6381.

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

See Also

dlnorm, RhodesEtAlPrior.

Examples

```
# load example data:
data("CrinsEtAl2014")

# determine corresponding prior parameters:
TP <- TurnerEtAlPrior("surgical", "pharma", "placebo / control")
print(TP)
# a prior 95 percent interval for tau:
TP$qprior(c(0.025,0.975))

## Not run:
# compute effect sizes (log odds ratios) from count data
# (using "metafor" package's "escalc()" function):
crins.es <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)
print(crins.es)

# perform meta analysis:
crins.ma01 <- bayesmeta(crins.es, tau.prior=TP$dprior)
# for comparison perform analysis using weakly informative Cauchy prior:
crins.ma02 <- bayesmeta(crins.es, tau.prior=function(t){dhalfcauchy(t,scale=1)})

# show results:
print(crins.ma01)
print(crins.ma02)
# compare estimates; heterogeneity (tau):
rbind("Turner prior"=crins.ma01$summary[,"tau"], "Cauchy prior"=crins.ma02$summary[,"tau"])
# effect (mu):
rbind("Turner prior"=crins.ma01$summary[,"mu"], "Cauchy prior"=crins.ma02$summary[,"mu"])

# illustrate heterogeneity priors and posteriors:
par(mfcol=c(2,2))
  plot(crins.ma01, which=4, prior=TRUE, taulim=c(0,2),
       main="informative log-normal prior")
  plot(crins.ma02, which=4, prior=TRUE, taulim=c(0,2),
       main="weakly informative half-Cauchy prior")
```

```
  plot(crins.ma01, which=3, mulim=c(-3,0),
       main="informative log-normal prior")
  abline(v=0, lty=3)
  plot(crins.ma02, which=3, mulim=c(-3,0),
       main="weakly informative half-Cauchy prior")
  abline(v=0, lty=3)
par(mfrow=c(1,1))

# compare prior and posterior 95 percent upper limits for tau:
TP$qprior(0.95)
crins.ma01$qposterior(0.95)
qhalfcauchy(0.95)
crins.ma02$qposterior(0.95)

## End(Not run)
```

`uisd` *Unit information standard deviation*

Description

This function estimates the unit information standard deviation (UISD) from a given set of standard errors and associated sample sizes.

Usage

```
  uisd(n, ...)
  ## Default S3 method:
uisd(n, sigma, sigma2=sigma^2, labels=NULL, individual=FALSE, ...)
  ## S3 method for class 'escalc'
uisd(n, ...)
```

Arguments

`n`	vector of sample sizes *or* an `escalc` object.
`sigma`	vector of standard errors associated with `n`.
`sigma2`	vector of *squared* standard errors (variances) associated with `n`.
`labels`	(optional) a vector of labels corresponding to `n` and `sigma`.
`individual`	a `logical` flag indicating whether individual (study-specific) UISDs are to be returned.
`...`	other `uisd` arguments.

Details

The *unit information standard deviation (UISD)* reflects the "within-study" variability, which, depending on the effect measure considered, sometimes is a somewhat heuristic notion (Roever et al., 2020). For a single study, presuming that standard errors result as

$$\sigma_i = \frac{\sigma_\mathrm{u}}{\sqrt{n_i}},$$

where σ_u is the within-study (population) standard deviation, the UISD simply results as

$$\sigma_\mathrm{u} = \sqrt{n_i \, \sigma_i^2}.$$

This is often appropriate when assuming an (approximately) normal likelihood.

Assuming a constant σ_u value across studies, this figure then may be estimated by

$$s_\mathrm{u} \; = \; \sqrt{\bar{n} \, \bar{s}_\mathrm{h}^2} \; = \; \sqrt{\frac{\sum_{i=1}^k n_i}{\sum_{i=1}^k \sigma_i^{-2}}},$$

where $\bar{n}$ is the average (arithmetic mean) of the studies' sample sizes, and $\bar{s}_\mathrm{h}^2$ is the harmonic mean of the squared standard errors (variances).

The estimator s_u is motivated via meta-analysis using the normal-normal hierarchical model (NNHM). In the special case of homogeneity (zero heterogeneity, $\tau = 0$), the overall mean estimate has standard error

$$\left(\sum_{i=1}^k \sigma_i^{-2} \right)^{-1/2}.$$

Since this estimate corresponds to *complete pooling*, the standard error may also be expressed via the UISD as

$$\frac{\sigma_\mathrm{u}}{\sqrt{\sum_{i=1}^k n_i}}.$$

Equating both above standard error expressions yields s_u as an estimator of the UISD σ_u (Roever *et al*, 2020).

Value

Either a (single) estimate of the UISD, or, if `individual` was set to '`TRUE`', a (potentially named) vector of UISDs for each individual study.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, R. Bender, S. Dias, C.H. Schmid, H. Schmidli, S. Sturtz, S. Weber, T. Friede. On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, **12**(4):448-474, 2021. doi:10.1002/jrsm.1475.

See Also

escalc.

Examples

```
# load data set:
data("CrinsEtAl2014")

# compute logarithmic odds ratios (log-ORs):
CrinsAR  <- escalc(measure="OR",
                   ai=exp.AR.events,  n1i=exp.total,
                   ci=cont.AR.events, n2i=cont.total,
                   slab=publication, data=CrinsEtAl2014)

# estimate the UISD:
uisd(n     = CrinsAR$exp.total + CrinsAR$cont.total,
     sigma = sqrt(CrinsAR$vi),
     label = CrinsAR$publication)

# for an "escalc" object, one may also apply the function directly:
uisd(CrinsAR)

# compute study-specific UISDs:
uisd(CrinsAR, individual=TRUE)
```

---

weightsplot *Illustrate the posterior mean weights for a* bayesmeta *object.*

Description

Generates a bar plot showing individual estimates' posterior mean weights, either for the overall mean estimate, or for a shrinkage estimate.

Usage

```
  weightsplot(x, ...)
  ## S3 method for class 'bayesmeta'
weightsplot(x, individual=FALSE, ordered=TRUE,
            extramargin=4, priorlabel="prior mean", main, xlim, ...)
```

Arguments

x	a bayesmeta object.
individual	this argument allows to request weights for individual *shrinkage estimates*. If FALSE (the default), weights for the overall mean are returned. Otherwise, it may be an integer number (1,...,k) giving the index, or a character string giving the label.
ordered	a logical flag indicating whether to sort weights by their magnitude.

extramargin	an additional margin to be added to the left side of the plot, in order to accomodate the estimates' labels. The value will be added to the 2nd element of the margin settings given by 'par("mar")'. In case study labels still extend beyond the figure margin, try increasing this number. See also the par() function's help.
priorlabel	the label for the effect prior's weight. Only relevant for proper effect priors.
main	the plot's main title.
xlim	the x-axis range.
...	other arguments passed on to the barplot() function.

Details

The individual estimates' contributions to the overall mean estimate are commonly illustrated in terms of *weights*, as the resulting overall estimate may be expressed as a weighted average of the estimates contributing to the analysis. The notion of "study weights" may also be extended to the Bayesian setting, where these result as *posterior mean weights*. Analogous weights may also be derived for *shrinkage estimates* (Roever and Friede, 2021).

This function generates a simple bar plot illustrating the posterior mean weights. The actual numbers are taken from the bayesmeta object's "$weights" or "$weights.theta" elements.

Author(s)

Christian Roever <christian.roever@med.uni-goettingen.de>

References

C. Roever, T. Friede. Bounds for the weight of external data in shrinkage estimation. *Biometrical Journal*, **65**(5):1131-1143, 2021. doi:10.1002/bimj.202000227.

See Also

bayesmeta.

Examples

```
# load example data:
data("Peto1980")
## Not run:
# compute effect sizes (log odds ratios) from count data:
require("metafor")
peto.es <- escalc(measure="OR",
                  ai=treat.events,   n1i=treat.cases,
                  ci=control.events, n2i=control.cases,
                  slab=publication, data=Peto1980)

# perform meta-analysis:
ma01 <- bayesmeta(peto.es)
# show data and results:
forestplot(ma01)
```

```
# check out weights:
ma01$weights
ma01$weights.theta

# illustrate weights:
weightsplot(ma01)
weightsplot(ma01, ordered=FALSE)
weightsplot(ma01, ordered=FALSE, individual="BrMedJ1974")

## End(Not run)
```

Index