Package ‘RBesT’

October 23, 2019

Type Package
Title R Bayesian Evidence Synthesis Tools
Version 1.5-4
Date 2019-10-22
Depends R (>= 3.4.0), Rcpp (>= 0.12.0), methods
Imports assertthat, mvtnorm, Formula, checkmate, rstan (>= 2.19.2), bayesplot (>= 1.4.0), ggplot2, dplyr, stats, utils
LinkingTo StanHeaders (>= 2.19.0), rstan (>= 2.19.2), BH (>= 1.69.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0)
License GPL (>= 3)
LazyData true
Biarch true
NeedsCompilation yes
Suggests rmarkdown, knitr, testthat (>= 2.0.0), foreach, tidyverse, purrr, rstanarm (>= 2.17.2), scales, tools, broom, tidyr, rstantools (>= 2.0.0)
VignetteBuilder knitr
SystemRequirements GNU make, pandoc (>= 1.12.3), pandoc-citeproc
Encoding UTF-8
RoxygenNote 6.1.1
Author Novartis Pharma AG [cph],
  Sebastian Weber [aut, cre],
  Beat Neuenschwander [ctb],
  Heinz Schmidli [ctb],
  Baldur Magnusson [ctb],
Yue Li [ctb],
Satrajit Roychoudhury [ctb],
Trustees of Columbia University [cph] (R/stanmodels.R, configure, configure.win)

Maintainer  Sebastian Weber <sebastian.weber@novartis.com>
Repository  CRAN
Date/Publication  2019-10-23 16:40:02 UTC

R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBesT-package</td>
<td>3</td>
</tr>
<tr>
<td>AS</td>
<td>4</td>
</tr>
<tr>
<td>automixfit</td>
<td>5</td>
</tr>
<tr>
<td>BinaryExactCI</td>
<td>6</td>
</tr>
<tr>
<td>colitis</td>
<td>7</td>
</tr>
<tr>
<td>crohn</td>
<td>7</td>
</tr>
<tr>
<td>decision1S</td>
<td>8</td>
</tr>
<tr>
<td>decision1S_boundary</td>
<td>11</td>
</tr>
<tr>
<td>decision2S</td>
<td>13</td>
</tr>
<tr>
<td>decision2S_boundary</td>
<td>15</td>
</tr>
<tr>
<td>ess</td>
<td>17</td>
</tr>
<tr>
<td>forest_plot</td>
<td>20</td>
</tr>
<tr>
<td>gMAP</td>
<td>22</td>
</tr>
<tr>
<td>likelihood</td>
<td>29</td>
</tr>
<tr>
<td>lodds</td>
<td>30</td>
</tr>
<tr>
<td>mix</td>
<td>31</td>
</tr>
<tr>
<td>mixbeta</td>
<td>33</td>
</tr>
<tr>
<td>mixcombine</td>
<td>35</td>
</tr>
<tr>
<td>mixdiff</td>
<td>36</td>
</tr>
<tr>
<td>mixfit</td>
<td>38</td>
</tr>
<tr>
<td>mixgamma</td>
<td>40</td>
</tr>
<tr>
<td>mixnorm</td>
<td>42</td>
</tr>
<tr>
<td>oc1S</td>
<td>44</td>
</tr>
<tr>
<td>oc2S</td>
<td>47</td>
</tr>
<tr>
<td>plot.EM</td>
<td>49</td>
</tr>
<tr>
<td>plot.gMAP</td>
<td>51</td>
</tr>
<tr>
<td>plot.mix</td>
<td>52</td>
</tr>
<tr>
<td>pos1S</td>
<td>53</td>
</tr>
<tr>
<td>pos2S</td>
<td>55</td>
</tr>
<tr>
<td>postmix</td>
<td>58</td>
</tr>
<tr>
<td>preddist</td>
<td>61</td>
</tr>
<tr>
<td>predict.gMAP</td>
<td>63</td>
</tr>
<tr>
<td>robustify</td>
<td>65</td>
</tr>
<tr>
<td>transplant</td>
<td>67</td>
</tr>
</tbody>
</table>

Index  68
**Description**

The RBesT tools are designed to support in the derivation of parametric informative priors, assess design characteristics and perform analyses. Supported endpoints include normal, binary and Poisson.

**Details**

For introductory material, please refer to the vignettes which include

- Introduction (binary)
- Introduction (normal)
- Customizing RBesT Plots
- Robust MAP, advanced usage

The main function of the package is `gMAP`. See it’s help page for a detailed description of the statistical model.

**Global Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBesT.MC.warmup</td>
<td>2000</td>
<td>MCMC warmup iterations</td>
</tr>
<tr>
<td>RBesT.MC.iter</td>
<td>6000</td>
<td>total MCMC iterations</td>
</tr>
<tr>
<td>RBesT.MC.chains</td>
<td>4</td>
<td>MCMC chains</td>
</tr>
<tr>
<td>RBesT.MC.thin</td>
<td>4</td>
<td>MCMC thinning</td>
</tr>
<tr>
<td>RBesT.MC.control</td>
<td>list(adapt_delta=0.99, stepsize=0.01, max_treedepth=20)</td>
<td>sets control argument for Stan call</td>
</tr>
<tr>
<td>RBesT.MC.ncp</td>
<td>1</td>
<td>parametrization: 0=CP, 1=NCP, 2=Automatic</td>
</tr>
<tr>
<td>RBesT.MC.init</td>
<td>1</td>
<td>range of initial uniform [-1,1] is the default</td>
</tr>
<tr>
<td>RBesT.MC.rescale</td>
<td>TRUE</td>
<td>Automatic rescaling of raw parameters</td>
</tr>
<tr>
<td>RBesT.verbose</td>
<td>FALSE</td>
<td>requests outputs to be more verbose</td>
</tr>
<tr>
<td>RBesT.integrate_args</td>
<td>list(lower=-Inf, upper=Inf, rel.tol=.Machine$double.eps<em>0.25, abs.tol=.Machine$double.eps</em>0.25, subdivisions=1E3)</td>
<td>arguments passed to integrate for integration of densities</td>
</tr>
</tbody>
</table>
Version History

See NEWS file.

References


AS

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AS</td>
<td>Ankylosing Spondylitis.</td>
</tr>
</tbody>
</table>

Description

Data set containing historical information for placebo for a phase II trial of ankylosing spondylitis patients. The primary efficacy endpoint was the percentage of patients with a 20 according to the Assessment of SpondyloArthritis international Society criteria for improvement (ASAS20) at week 6.

Usage

AS

Format

A data frame with 8 rows and 3 variables:

- study study
- n study size
- r number of events

References

Baeten D. et. al, The Lancet, 2013, (382), 9906, p 1705

Examples

```r
## Setting up dummy sampling for fast execution of example
## Please use 4 chains and 20x more warmup & iter in practice
.user_mc_options <- options(RBesT.MC.warmup=50, RBesT.MC.iter=100, 
RBesT.MC.chains=2, RBesT.MC.thin=1)
set.seed(34563)
map_AS <- gMAP(cbind(r, n-r) ~ 1 | study, 
family=binomial, 
data=AS, 
tau.dist="HalfNormal", tau.prior=1, 
beta.prior=2)
## Recover user set sampling defaults
options(.user_mc_options)
```
Description

Fitting a series of mixtures of conjugate distributions to a sample, using Expectation-Maximization (EM). The number of mixture components is specified by the vector Nc. First a Nc[1] component mixture is fitted, then a Nc[2] component mixture, and so on. The mixture providing the best AIC value is then selected.

Usage

automixfit(sample, Nc = seq(1, 4), k = 6, thresh = -Inf, verbose = FALSE, ...)

Arguments

sample Sample to be fitted by a mixture distribution.
Nc Vector of mixture components to try out (default seq(1, 4)).
k Penalty parameter for AIC calculation (default 6)
thresh The procedure stops if the difference of subsequent AIC values is smaller than this threshold (default -Inf). Setting the threshold to 0 stops automixfit once the AIC becomes worse.
verbose Enable verbose logging.
... Further arguments passed to mixfit, including type.

Details

The type argument specifies the distribution of the mixture components, and can be a normal, beta or gamma distribution.

The penalty parameter k is 2 for the standard AIC definition. Collet (2003) suggested to use values in the range from 2 to 6, where larger values of k penalize more complex models. To favor mixtures with fewer components a value of 6 is used as default.

Value

As result the best fitting mixture model is returned, i.e. the model with lowest AIC. All other models are saved in the attribute models.

References

Collet D. Modeling Survival Data in Medical Research. 2003; Chapman and Hall/CRC.
Examples

# random sample of size 1000 from a mixture of 2 beta components
bm <- mixbeta(beta1=c(0.4, 20, 90), beta2=c(0.6, 35, 65))
bmSamp <- rmix(bm, 1000)

# fit with EM mixture models with up to 10 components and stop if
# AIC increases
bmFit <- automixfit(bmSamp, Nc=1:10, thresh=0, type="beta")
bmFit

# advanced usage: find out about all discarded models
bmFitAll <- attr(bmFit, "models")
sapply(bmFitAll, AIC, k=6)

BinaryExactCI

Exact Confidence interval for Binary Proportion

Description

This function calculates the exact confidence interval for a response rate presented by \( n \) and \( r \).

Usage

BinaryExactCI(r, n, alpha = 0.05, drop = TRUE)

Arguments

- **r**: Number of success or responder
- **n**: Sample size
- **alpha**: confidence level
- **drop**: Determines if drop will be called on the result

Details

Confidence intervals are obtained by a procedure first given in Clopper and Pearson (1934). This guarantees that the confidence level is at least \((1-\alpha)\).

Details can be found in the publication listed below.

Value

100 \((1-\alpha)\)% exact confidence interval for given response rate
References
Clopper, C. J. & Pearson, E. S. The use of confidence or fiducial limits illustrated in the case of the binomial. Biometrika 1934.

Examples
BinaryExactCI(3,20,0.05)

colitis

Ulcervative Colitis.

Description
Data set containing historical information for placebo arm of a phase II proof-of-concept trial for the treatment of ulcerative colitis. The primary outcome is remission at week 8 (binary).

Usage
colitis

Format
A data frame with 4 rows and 3 variables:
study  study
n  study size
r  number of events

References

crohn

Crohn’s disease.

Description
Data set containing historical information for placebo arm of relevant studies for the treatment of Crohn’s disease. The primary outcome is change from baseline in Crohn’s Disease Activity Index (CDAI) over a duration of 6 weeks. Standard deviation of change from baseline endpoint is approximately 88.
Usage
crohn

Format
A data frame with 4 rows and 3 variables:

study study
n study size
y mean CDAI change

References

Examples

## Setting up dummy sampling for fast execution of example
## Please use 4 chains and 20x more warmup & iter in practice
.user_mc_options <- options(RBesT.MC.warmup=50, RBesT.MC.iter=100,
                           RBesT.MC.chains=2, RBesT.MC.thin=1)

set.seed(546346)
map_crohn <- gMAP(cbind(y, y.se) ~ 1 | study,
                   family=gaussian,
                   data=transform(crohn, y.se=88/sqrt(n)),
                   weights=n,
                   tau.dist="HalfNormal", tau.prior=44,
                   beta.prior=cbind(0,88))

## Recover user set sampling defaults
options(.user_mc_options)

decision1S

Decision Function for 1 Sample Designs

Description
The function sets up a 1 sample one-sided decision function with an arbitrary number of conditions.

Usage
decision1S(pc = 0.975, qc = 0, lower.tail = TRUE)
oc1Sdecision(pc = 0.975, qc = 0, lower.tail = TRUE)
**decision1S**

**Arguments**
- **pc**: Vector of critical cumulative probabilities.
- **qc**: Vector of respective critical values. Must match the length of **pc**.
- **lower.tail**: Logical; if **TRUE** (default), probabilities are \( P(X \leq x) \), otherwise, \( P(X > x) \).

**Details**

The function creates a one-sided decision function which takes two arguments. The first argument is expected to be a mixture (posterior) distribution. This distribution is tested whether it fulfills all the required threshold conditions specified with the **pc** and **qc** arguments and returns 1 if all conditions are met and 0 otherwise. Hence, for **lower.tail = TRUE** condition \( i \) is equivalent to

\[
P(x \leq q_{c,i}) > p_{c,i}
\]

and the decision function is implemented as indicator function on the basis of the heavy-side step function \( H \) which is 0 for \( x \leq 0 \) and 1 for \( x > 0 \). As all conditions must be met, the final indicator function returns

\[
\Pi_i H_i(P(x \leq q_{c,i}) - p_{c,i}).
\]

When the second argument is set to **TRUE** a distance metric is returned component-wise per defined condition as

\[
D_i = \log(P(p < q_{c,i})) - \log(p_{c,i}).
\]

These indicator functions can be used as input for 1-sample boundary, OC or PoS calculations using **oc1S** or **pos1S**.

**Value**

The function returns a decision function which takes two arguments. The first argument is expected to be a mixture (posterior) distribution which is tested if the specified conditions are met. The logical second argument determines if the function acts as an indicator function or if the function returns the distance from the decision boundary for each condition in log-space, i.e. the distance is 0 at the decision boundary, negative for a 0 decision and positive for a 1 decision.

**Functions**
- **oc1Sdecision**: Deprecated old function name. Please use **decision1S** instead.

**References**


**See Also**

Other design1S: **decision1S_boundary, oc1S, pos1S**
Examples

# see Neuenschwander et al., 2011
# example is for a time-to-event trial evaluating non-inferiority
# using a normal approximation for the log-hazard ratio

# reference scale
s <- 2
theta_ni <- 0.4
theta_a <- 0
alpha <- 0.05
beta <- 0.2
za <- qnorm(1-alpha)
zb <- qnorm(1-beta)
n1 <- round((s * (za + zb)/(theta_ni - theta_a))^2) # n for which design was intended
nL <- 233
c1 <- theta_ni - za * s / sqrt(n1)

# flat prior
flat_prior <- mixnorm(c(1,0,100), sigma=s)

# standard NI design
decA <- decision1S(1 - alpha, theta_ni, lower.tail=TRUE)

# for double criterion with indecision point (mean estimate must be
# lower than this)
theta_c <- c1

# double criterion design
# statistical significance (like NI design)
dec1 <- decision1S(1-alpha, theta_ni, lower.tail=TRUE)
# require mean to be at least as good as theta_c
dec2 <- decision1S(0.5, theta_c, lower.tail=TRUE)
# combination
decComb <- decision1S(c(1-alpha, 0.5), c(theta_ni, theta_c), lower.tail=TRUE)

theta_eval <- c(theta_a, theta_c, theta_ni)

# we can display the decision function definition
decComb

# and use it to decide if a given distribution fulfills all
# criterions defined
# for the prior
decComb(flat_prior)
# or for a possible outcome of the trial
# here with HR of 0.8 for 40 events
decComb(postmix(flat_prior, m=log(0.8), n=40))
Description

Calculates the decision boundary for a 1 sample design. This is the critical value at which the decision function will change from 0 (failure) to 1 (success).

Usage

decision1S_boundary(prior, n, decision, ...)

## S3 method for class 'betaMix'
decision1S_boundary(prior, n, decision, ...)

## S3 method for class 'normMix'
decision1S_boundary(prior, n, decision, sigma,
   eps = 1e-06, ...)

## S3 method for class 'gammaMix'
decision1S_boundary(prior, n, decision, eps = 1e-06,
   ...)

Arguments

prior Prior for analysis.
n Sample size for the experiment.
decision One-sample decision function to use; see decision1S.
... Optional arguments.
sigma The fixed reference scale. If left unspecified, the default reference scale of the prior is assumed.
eps Support of random variables are determined as the interval covering 1-eps probability mass. Defaults to 10^-6.

Details

The specification of the 1 sample design (prior, sample size and decision function, $D(y)$), uniquely defines the decision boundary

$$y_c = \max_y \{ D(y) = 1 \},$$

which is the maximal value of $y$ whenever the decision $D(y)$ function changes its value from 1 to 0 for a decision function with lower.tail=TRUE (otherwise the definition is $y_c = \max_y \{ D(y) = 0 \}$). The decision function may change at most at a single critical value as only one-sided decision
functions are supported. Here, $y$ is defined for binary and Poisson endpoints as the sufficient statistic $y = \sum_{i=1}^{n} y_i$ and for the normal case as the mean $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$.

The convention for the critical value $y_c$ depends on whether a left (lower.tail=TRUE) or right-sided decision function (lower.tail=FALSE) is used. For lower.tail=TRUE the critical value $y_c$ is the largest value for which the decision is 1, $D(y \leq y_c) = 1$, while for lower.tail=FALSE then $D(y > y_c) = 1$ holds. This is aligned with the cumulative density function definition within R (see for example pbinom).

**Value**

Returns the critical value $y_c$.

**Methods (by class)**

- betaMix: Applies for binomial model with a mixture beta prior. The calculations use exact expressions.
- normMix: Applies for the normal model with known standard deviation $\sigma$ and a normal mixture prior for the mean. As a consequence from the assumption of a known standard deviation, the calculation discards sampling uncertainty of the second moment. The function decision1S_boundary has an extra argument eps (defaults to $10^{-6}$). The critical value $y_c$ is searched in the region of probability mass 1-eps for $y$.
- gammaMix: Applies for the Poisson model with a gamma mixture prior for the rate parameter. The function decision1S_boundary takes an extra argument eps (defaults to $10^{-6}$) which determines the region of probability mass 1-eps where the boundary is searched for $y$.

**See Also**

Other design1S: decision1S, oc1S, pos1S

**Examples**

```r
# non-inferiority example using normal approximation of log-hazard ratio, see ?decision1S for all details
s <- 2
flat_prior <- mixnorm(c(1,0,100), sigma=s)
nL <- 233
theta_ni <- 0.4
theta_a <- 0
alpha <- 0.05
beta <- 0.2
za <- qnorm(1-alpha)
zb <- qnorm(1-beta)
n1 <- round((s * (za + zb)/(theta_ni - theta_a))^2 )
theta_c <- theta_ni - za * s / sqrt(n1)

# double criterion design
# statistical significance (like NI design)
dec1 <- decision1S(1-alpha, theta_ni, lower.tail=TRUE)
# require mean to be at least as good as theta_c
```
decision2S <- decision1S(0.5, theta_c, lower.tail=TRUE)
# combination
decComb <- decision1S(c(1-alpha, 0.5), c(theta_ni, theta_c), lower.tail=TRUE)

# critical value of double criterion design
decision1S_boundary(flat_prior, nL, decComb)

# ... is limited by the statistical significance ...
decision1S_boundary(flat_prior, nL, dec1)

# ... or the indecision point (whatever is smaller)
decision1S_boundary(flat_prior, nL, dec2)

description2S

**Decision Function for 2 Sample Designs**

**Description**

The function sets up a 2 sample one-sided decision function with an arbitrary number of conditions on the difference distribution.

**Usage**

```r
decision2S(pc = 0.975, qc = 0, lower.tail = TRUE, 
link = c("identity", "logit", "log"))
```

```r
oc2Sdecision(pc = 0.975, qc = 0, lower.tail = TRUE, 
link = c("identity", "logit", "log"))
```

**Arguments**

- `pc` Vector of critical cumulative probabilities of the difference distribution.
- `qc` Vector of respective critical values of the difference distribution. Must match the length of `pc`.
- `lower.tail` Logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
- `link` Enables application of a link function prior to evaluating the difference distribution. Can take one of the values identity (default), logit or log.

**Details**

This function creates a one-sided decision function on the basis of the difference distribution in a 2 sample situation. To support double criterion designs, see Neuenschwander et al., 2010, an arbitrary number of criterions can be given. The decision function demands that the probability mass below the critical value `qc` of the difference $x_1 - x_2$ is at least `pc`. Hence, for `lower.tail=TRUE` condition $i$ is equivalent to

$$P(x_1 - x_2 \leq q_{c,i}) > p_{c,i}$$
and the decision function is implemented as indicator function using the heavy-side step function \( H \) which is 0 for \( x \leq 0 \) and 1 for \( x > 0 \). As all conditions must be met, the final indicator function returns

\[
\Pi_i H_i(P(x_1 - x_2 \leq q_{c,i}) - p_{c,i}),
\]

which is 1 if all conditions are met and 0 otherwise. For \texttt{lower.tail=FALSE} differences must be greater than the given quantiles \( q_c \).

Note that whenever a link other than identity is requested, then the underlying densities are first transformed using the link function and then the probabilities for the differences are calculated in the transformed space. Hence, for a binary endpoint the default identity link will calculate risk differences, the logit link will lead to decisions based on the differences in logits corresponding to a criterion based on the log-odds. The log link will evaluate ratios instead of absolute differences which could be useful for a binary endpoint or counting rates. The respective critical quantiles \( q_c \) must be given on the transformed scale.

**Value**

The function returns a decision function which takes three arguments. The first and second argument are expected to be mixture (posterior) distributions from which the difference distribution is formed and all conditions are tested. The third argument determines if the function acts as an indicator function or if the function returns the distance from the decision boundary for each condition in log-space. That is, the distance is 0 at the decision boundary, negative for a 0 decision and positive for a 1 decision.

**Functions**

- \texttt{oc2Sdecision}: Deprecated old function name. Please use \texttt{decision2S} instead.

**References**


**See Also**

Other design2S: \texttt{decision2S\_boundary}, \texttt{oc2S}, \texttt{pos2S}

**Examples**

```r
# see Gsponer et al., 2010
priorT <- mixnorm(c(1, 0, 0.001), sigma=88, param="mn")
priorP <- mixnorm(c(1, -49, 20 ), sigma=88, param="mn")
# the success criteria is for delta which are larger than some threshold value which is why we set lower.tail=FALSE
successCrit <- decision2S(c(0.95, 0.5), c(0, 50), FALSE)
# the futility criterion acts in the opposite direction
futilityCrit <- decision2S(c(0.90) , c(40), TRUE)
```
print(successCrit)
print(futilityCrit)

# consider decision for specific outcomes
postP_interim <- postmix(priorP, n=10, m=-50)
postT_interim <- postmix(priorT, n=20, m=-80)
futilityCrit( postP_interim, postT_interim )
successCrit( postP_interim, postT_interim )

# Binary endpoint with double criterion decision on log-odds scale
# 95% certain positive difference and an odds ratio of 2 at least
decL2 <- decision2S(c(0.95, 0.5), c(0, log(2)), lower.tail=FALSE, link="logit")
# 95% certain positive difference and an odds ratio of 3 at least
decL3 <- decision2S(c(0.95, 0.5), c(0, log(3)), lower.tail=FALSE, link="logit")

# data scenario
post1 <- postmix(mixbeta(c(1, 1, 1)), n=40, r=10)
post2 <- postmix(mixbeta(c(1, 1, 1)), n=40, r=18)

# positive outcome and a median odds ratio of at least 2 ...
decL2(post2, post1)
# ... but not more than 3
decL3(post2, post1)

decision2S_boundary  

---

**decision2S_boundary**  

**Description**

The `decision2S_boundary` function defines a 2 sample design (priors, sample sizes, decision function) for the calculation of the decision boundary. A function is returned which calculates the critical value of the first sample $y_1$ as a function of the outcome in the second sample $y_2$. At the decision boundary, the decision function will change between 0 (failure) and 1 (success) for the respective outcomes.

**Usage**

decision2S_boundary(prior1, prior2, n1, n2, decision, ...)

## S3 method for class 'betaMix'
decision2S_boundary(prior1, prior2, n1, n2, decision,
                      eps, ...)

## S3 method for class 'normMix'
decision2S_boundary(prior1, prior2, n1, n2, decision,
                      sigma1, sigma2, eps = 1e-06, Ngrid = 10, ...)

## S3 method for class 'gammaMix'
decision2S_boundary(prior1, prior2, n1, n2, decision, 
eps = 1e-06, ...)
• **normMix**: Applies for the normal model with known standard deviation \( \sigma \) and normal mixture priors for the means. As a consequence from the assumption of a known standard deviation, the calculation discards sampling uncertainty of the second moment. The function has two extra arguments (with defaults): \( \text{eps} (10^{-6}) \) and \( \text{Ngrid} (10) \). The decision boundary is searched in the region of probability mass \( 1-\text{eps} \), respectively for \( y_1 \) and \( y_2 \). The continuous decision function is evaluated at a discrete grid, which is determined by a spacing with \( \delta_2 = \sigma_2/\sqrt{\text{Ngrid}} \). Once the decision boundary is evaluated at the discrete steps, a spline is used to inter-polate the decision boundary at intermediate points.

• **gammaMix**: Applies for the Poisson model with a gamma mixture prior for the rate parameter. The function \( \text{decision2S_boundary} \) takes an extra argument \( \text{eps} \) (defaults to \( 10^{-6} \)) which determines the region of probability mass \( 1-\text{eps} \) where the boundary is searched for \( y_1 \) and \( y_2 \), respectively.

See Also

Other design2S: \texttt{decision2S, oc2S, pos2S}

Examples

```r
# see ?decision2S for details of example
priorT <- mixnorm(c(1, 0, 0.001), sigma=88, param="mn")
priorP <- mixnorm(c(1, -49, 20 ), sigma=88, param="mn")
# the success criteria is for delta which are larger than some
# threshold value which is why we set lower.tail=FALSE
successCrit <- decision2S(c(0.95, 0.5), c(0, 50), FALSE)
# the futility criterion acts in the opposite direction
futilityCrit <- decision2S(c(0.90) , c(40), TRUE)

# success criterion boundary
successBoundary <- decision2S_boundary(priorP, priorT, 10, 20, successCrit)

# futility criterion boundary
futilityBoundary <- decision2S_boundary(priorP, priorT, 10, 20, futilityCrit)

curve(successBoundary(x), -25:25 - 49, xlab="y2", ylab="critical y1")
curve(futilityBoundary(x), lty=2, add=TRUE)

# hence, for mean in sample 2 of 10, the critical value for y1 is
y1c <- futilityBoundary(-10)
# around the critical value the decision for futility changes
futilityCrit(postmix(priorP, m=y1c+1E-3, n=10), postmix(priorT, m=-10, n=20))
futilityCrit(postmix(priorP, m=y1c-1E-3, n=10), postmix(priorT, m=-10, n=20))
```
Description

Calculates the Effective Sample Size (ESS) for a mixture prior. The ESS indicates how many experimental units the prior is roughly equivalent to.

Usage

ess(mix, method = c("elir", "moment", "morita"), ...)

## S3 method for class 'betaMix'

ess(mix, method = c("elir", "moment", "morita"), ..., s = 100)

## S3 method for class 'gammaMix'

ess(mix, method = c("elir", "moment", "morita"), ..., s = 100, eps = 1e-04)

## S3 method for class 'normMix'

ess(mix, method = c("elir", "moment", "morita"), ..., sigma, s = 100)

Arguments

mix Prior (mixture of conjugate distributions).
method Selects the used method. Can be either elir (default), moment or morita.
... Optional arguments applicable to specific methods.
s For morita method large constant to ensure that the prior scaled by this value is vague (default 100); see Morita et al. (2008) for details.
eps Probability mass left out from the numerical integration of the expected information for the Poisson-Gamma case of Morita method (defaults to 1E-4).
sigma reference scale.

Details

The ESS is calculated using either the expected local information ratio (elir) Neuenschwander et al. (submitted), the moments approach or the method by Morita et al. (2008).

The elir approach is the only ESS which fulfills predictive consistency. The predictive consistency of the ESS requires that the ESS of a prior is the same as averaging the posterior ESS after a fixed amount of events over the prior predictive distribution from which the number of forward simulated events is subtracted. The elir approach results in ESS estimates which are neither conservative nor liberal whereas the moments method yields conservative and the morita method liberal results. See the example section for a demonstration of predictive consistency.

For the moments method the mean and standard deviation of the mixture are calculated and then approximated by the conjugate distribution with the same mean and standard deviation. For conjugate distributions, the ESS is well defined. See the examples for a step-wise calculation in the beta mixture case.
The Morita method used here evaluates the mixture prior at the mode instead of the mean as proposed originally by Morita. The method may lead to very optimistic ESS values, especially if the mixture contains many components. The calculation of the Morita approach here follows the approach presented in Neuenschwander B. et al (2019) which avoids the need for a minimization and does not restrict the ESS to be an integer.

**Value**

Returns the ESS of the prior as floating point number.

**Methods (by class)**

- `betaMix`: ESS for beta mixtures.
- `gammaMix`: ESS for gamma mixtures.
- `normMix`: ESS for normal mixtures.

**Supported Conjugate Prior-Likelihood Pairs**

<table>
<thead>
<tr>
<th>Prior/Posterior</th>
<th>Likelihood</th>
<th>Predictive</th>
<th>Summaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beta</td>
<td>Binomial</td>
<td>Beta-Binomial</td>
<td>n, r</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal (fixed $\sigma$)</td>
<td>Normal</td>
<td>n, m, se</td>
</tr>
<tr>
<td>Gamma</td>
<td>Poisson</td>
<td>Gamma-Poisson</td>
<td>n, m</td>
</tr>
<tr>
<td>Gamma</td>
<td>Exponential</td>
<td>Gamma-Exp (not supported)</td>
<td>n, m</td>
</tr>
</tbody>
</table>

**References**


**Examples**

```r
# Conjugate Beta example
a <- 5
b <- 15
prior <- mixbeta(c(1, a, b))

ess(prior)
(a+b)

# Beta mixture example
bmix <- mixbeta(rob=c(0.2, 1, 1), inf=c(0.8, 10, 2))

ess(bmix, "elir")

ess(bmix, "moment")
```
# moments method is equivalent to
# first calculate moments
bmix_sum <- summary(bmix)
# then calculate a and b of a matching beta
ab_matched <- ms2beta(bmix_sum["mean"], bmix_sum["sd"])
# finally take the sum of a and b which are equivalent
# to number of responders/non-responders respectively
round(sum(ab_matched))

ess(bmix, method="morita")

# Predictive consistency of elir
n_forward <- 1E2
bmixPred <- preddist(bmix, n=n_forward)
pred_samp <- rmix(bmixPred, 1E3)
pred_ess <- sapply(pred_samp, function(r) ess(postmix(bmix, r=r, n=n_forward), "elir") )
mean(pred_ess) - n_forward

# Normal mixture example
nmix <- mixnorm(rob=c(0.5, 0, 2), inf=c(0.5, 3, 4), sigma=10)

ess(nmix, "elir")

ess(nmix, "moment")

## the reference scale determines the ESS
sigma(nmix) <- 20
ess(nmix)

# Gamma mixture example
gmix <- mixgamma(rob=c(0.3, 20, 4), inf=c(0.7, 50, 10))

ess(gmix) ## interpreted as appropriate for a Poisson likelihood (default)
likelihood(gmix) <- "exp"

ess(gmix) ## interpreted as appropriate for an exponential likelihood

forest_plot

Description

Creates a forest plot for gMAP analysis objects.
Usage

```r
forest_plot(x, prob = 0.95, est = c("both", "MAP", "Mean", "none"),
            model = c("stratified", "both", "meta"), point_est = c("median",
                                                                  "mean"), size = 1.25, alpha = 0.5)
```

Arguments

- `x`: gMAP object.
- `prob`: confidence interval width and probability mass of credible intervals.
- `est`: can be set to one of both (default), MAP, Mean or none. Controls which model estimates are to be included.
- `model`: controls which estimates are displayed per study. Either stratified (default), both or meta.
- `point_est`: shown point estimate. Either median (default) or mean.
- `size`: controls point and linesize.
- `alpha`: transparency of reference line. Setting alpha=0 suppresses the reference line.

Details

The function creates a forest plot suitable for gMAP analyses. Note that the Meta-Analytic-Predictive prior is included by default in the plot as opposed to only showing the estimated model mean. See the examples below to obtain standard forest plots.

Also note that the plot internally flips the x and y-axis. Therefore, if you want to manipulate the x-axis, you have to give commands affecting the y-axis (see examples).

Value

The function returns a ggplot2 plot object.

Customizing ggplot2 plots

The returned plot is a ggplot2 object. Please refer to the "Customizing Plots" vignette which is part of RBesT documentation for an introduction. For simple modifications (change labels, add reference lines, ...) consider the commands found in bayesplot-helpers. For more advanced customizations please use the ggplot2 package directly. A description of the most common tasks can be found in the R Cookbook and a full reference of available commands can be found at the ggplot2 documentation site.

See Also

gMAP
Examples

```r
# we consider the example AS MAP analysis
e.example(AS)

# default forest plot for a gMAP analysis
forest.plot(map_AS)

# standard forest plot (only stratified estimate and Mean)
forest.plot(map_AS, est=c("Mean"), model="stratified")

# to further customize these plots, first load bayesplot and ggplot2
library(bayesplot)
library(ggplot2)

# to make plots with red colors, big fonts for presentations, suppress
# the x axis label and add another title (with a subtitle)
color_scheme_set("red")
theme_set(theme_default(base_size=16))
forest.plot(map_AS, size=2) +
  yaxis_title(FALSE) +
  ggtitle("Ankylosing Spondylitis Forest Plot",
           subtitle="Control Group Response Rate")

# the defaults are set with
color_scheme_set("blue")
theme_set(theme_default(base_size=12))
```

---

### gMAP

Meta-Analytic-Predictive Analysis for Generalized Linear Models

**Description**

Meta-Analytic-Predictive (MAP) analysis for generalized linear models suitable for normal, binary, or Poisson data. Model specification and overall syntax follows mainly `glm` conventions.

**Usage**

```r
gMAP(formula, family = gaussian, data, weights, offset, tau.strata, 
tau.dist = c("HalfNormal", "TruncNormal", "Uniform", "Gamma", 
"InvGamma", "LogNormal", "TruncCauchy", "Exp", "Fixed"), tau.prior, 
tau.strata.pred = 1, beta.prior, prior_PD = FALSE, 
REdist = c("normal", "t"), t.df = 5, contrasts = NULL, 
iter =getOption("RBesT.MC.iter", 6000), 
warmup =getOption("RBesT.MC.warmup", 2000), 
thin =getOption("RBesT.MC.thin", 4), 
init =getOption("RBesT.MC.init", 1), 
chains =getOption("RBesT.MC.chains", 4), 
cores =getOption("mc.cores", 1L))
```
Arguments

- **formula**: the model formula describing the linear predictor and encoding the grouping; see details
- **family**: the family of distributions defining the statistical model (binomial, gaussian, or poisson)
- **data**: optional data frame containing the variables of the model. If not found in data, the variables are taken from `environment(formula)`.
- **weights**: optional weight vector; see details below.
- **offset**: offset term in statistical model used for Poisson data
- **tau.strata**: sets the exchangability stratum per study. That is, it is expected that each study belongs to a single stratum. Default is to assign all studies to stratum 1. See section differential heterogeneity below.
- **tau.dist**: type of prior distribution for tau; supported priors are HalfNormal (default), TruncNormal, Uniform, Gamma, InvGamma, LogNormal, TruncCauchy, Exp and Fixed.
- **tau.prior**: parameters of prior distribution for tau; see section prior specification below.
- **tau.strata.pred**: the index for the prediction stratum; default is 1.
- **beta.prior**: mean and standard deviation for normal priors of regression coefficients, see section prior specification below.
- **prior_PD**: logical to indicate if the prior predictive distribution should be sampled (no conditioning on the data). Defaults to FALSE.
- **REdist**: type of random effects distribution. Normal (default) or t.
- **t.df**: degrees of freedom if random-effects distribution is t.
- **contrasts**: an optional list; See contrasts.arg from `model.matrix.default`.
- **iter**: number of iterations (including warmup).
warmup  number of warmup iterations.
thin    period of saving samples.
init    positive number to specify uniform range on unconstrained space for random initialization. See \texttt{stan}.
chains  number of Markov chains.
cores   number of cores for parallel sampling of chains.
ex, \texttt{object} \texttt{gMAP} analysis object created by \texttt{gMAP} function
digits  number of displayed significant digits.
probs   defines quantiles to be reported.
...     optional arguments are ignored
type    sets reported scale (\texttt{response} (default) or \texttt{link}).

\textbf{Details}

The meta-analytic-predictive (MAP) approach derives a prior from historical data using a hierarchical model. The statistical model is formulated as a generalized linear mixed model for binary, normal (with fixed $\sigma$) and Poisson endpoints:

$$y_{ih} | \theta_{ih} \sim f(y_{ih} | \theta_{ih})$$

Here, $i = 1, \ldots, N$ is the index for observations, and $h = 1, \ldots, H$ is the index for the grouping (usually studies). The model assumes the linear predictor for a transformed mean as

$$g(\theta_{ih}; x_{ih}, \beta) = x_{ih} \beta + \epsilon_h$$

with $x_{ih}$ being the row vector of $k$ covariates for observation $i$. The variance component is assumed by default normal

$$\epsilon_h \sim N(0, \tau^2), \quad h = 1, \ldots, H$$

Lastly, the Bayesian implementation assumes independent normal priors for the $k$ regression coefficients and a prior for the between-group standard deviation $\tau$ (see \texttt{taudist} for available distributions).

The MAP prior will then be derived from the above model as the conditional distribution of $\theta_\ast$ given the available data and the vector of covariates $x_\ast$ defining the overall intercept

$$\theta_\ast | x_\ast, y.$$

A simple and common case arises for one observation (summary statistic) per trial. For a normal endpoint, the model then simplifies to the standard normal-normal hierarchical model. In the above notation, $i = h = 1, \ldots, H$ and

$$y_h | \theta_h \sim N(\theta_h, s_h^2)$$
\[ \theta_h = \mu + \epsilon_h \]
\[ \epsilon_h \sim N(0, \tau^2) \]

where the more common \( \mu \) is used for the only (intercept) parameter \( \beta_1 \). Since there are no covariates, the MAP prior is simply \( P(\theta_\star | y_1, \ldots, y_H) \).

The hierarchical model is a compromise between the two extreme cases of full pooling (\( \tau = 0 \), full borrowing, no discounting) and no pooling (\( \tau = \infty \), no borrowing, stratification). The information content of the historical data grows with \( H \) (number of historical data items) indefinitely for full pooling whereas no information is gained in a stratified analysis. For a fixed \( \tau \), the maximum effective sample size of the MAP prior is \( n_\infty \) (\( H \to \infty \)), which for a normal endpoint with fixed \( \sigma \) is

\[ n_\infty = \left( \frac{\tau^2}{\sigma^2} \right)^{-1} \]

(Neuenschwander et al., 2010). Hence, the ratio \( \tau / \sigma \) limits the amount of information a MAP prior is equivalent to. This allows for a classification of \( \tau \) values in relation to \( \sigma \), which is crucial to define a prior \( P_\tau \). The following classification is useful in a clinical trial setting:

<table>
<thead>
<tr>
<th>Heterogeneity</th>
<th>( \tau / \sigma )</th>
<th>( n_\infty )</th>
</tr>
</thead>
<tbody>
<tr>
<td>small</td>
<td>0.0625</td>
<td>256</td>
</tr>
<tr>
<td>moderate</td>
<td>0.125</td>
<td>64</td>
</tr>
<tr>
<td>substantial</td>
<td>0.25</td>
<td>16</td>
</tr>
<tr>
<td>large</td>
<td>0.5</td>
<td>4</td>
</tr>
<tr>
<td>very large</td>
<td>1.0</td>
<td>1</td>
</tr>
</tbody>
</table>

The above formula for \( n_\infty \) assumes a known \( \tau \). This is unrealistic as the between-trial heterogeneity parameter is often not well estimable, in particular if the number of trials is small (\( H \) small). The above table helps to specify a prior distribution for \( \tau \) appropriate for the given context which defines the crucial parameter \( \sigma \). For binary and Poisson endpoints, normal approximations can be used to determine \( \sigma \). See examples below for concrete cases.

The design matrix \( X \) is defined by the formula for the linear predictor and is always of the form \( \text{response} \sim \text{predictor} \mid \text{grouping} \), which follows glm conventions. The syntax has been extended to include a specification of the grouping (for example study) factor of the data with a horizontal bar, \( | \). The bar separates the optionally specified grouping level, i.e. in the binary endpoint case \( \text{cbind}(r, n-r) \sim 1 \mid \text{study} \). By default it is assumed that each row corresponds to an individual group (for which an individual parameter is estimated). Specifics for the different endpoints are:

- **normal** family=gaussian assumes an identity link function. The response should be given as matrix with two columns with the first column being the observed mean value \( y_{ih} \) and the second column the standard error \( se_{ih} \) (of the mean). Additionally, it is recommended to specify with the weight argument the number of units which contributed to the (mean) measurement \( y_{ih} \). This information is used to estimate \( \sigma \).

- **binary** family=binomial assumes a logit link function. The response must be given as two-column matrix with number of responders \( r \) (first column) and non-responders \( n-r \) (second column).
Poisson family=poisson assumes a log link function. The response is a vector of counts. The total exposure times can be specified by an offset, which will be linearly added to the linear predictor. The offset can be given as part of the formula, \( y \sim 1 + \text{offset}(\log(\text{exposure})) \) or as the offset argument to gMAP. Note that the exposure unit must be given as log-offset.

Value

The function returns a S3 object of type gMAP. See the methods section below for applicable functions to query the object.

Methods (by generic)

- print: displays a summary of the gMAP analysis.
- fitted: returns the quantiles of the posterior shrinkage estimates for each data item used during the analysis of the given gMAP object.
- coef: returns the quantiles of the predictive distribution. User can choose with type if the result is on the response or the link scale.
- as.matrix: extracts the posterior sample of the model.
- summary: returns the summaries of a gMAP analysis. Output is a gMAPsummary object, which is a list containing
  - tau: posterior summary of the heterogeneity standard deviation
  - beta: posterior summary of the regression coefficients
  - theta.pred: summary of the predictive distribution (given in dependence on the type argument either on response or link scale)
  - theta: posterior summary of the mean estimate (also depends on the type argument)

Differential Discounting

The above model assumes the same between-group standard deviation \( \tau \), which implies that the data are equally relevant. This assumption can be relaxed to more than one \( \tau \). That is,

\[
\epsilon_h \sim N(0, \tau_s(h)^2)
\]

where \( s(h) \) assigns group \( h \) to one of \( S \) between-group heterogeneity strata.

For example, in a situation with two randomized and four observational studies, one may want to assume \( \tau_1 \) (for trials 1 and 2) and \( \tau_2 \) (for trials 3-6) for the between-trial standard deviations of the control means. More heterogeneity (less relevance) for the observational studies can then be expressed by appropriate priors for \( \tau_1 \) and \( \tau_2 \). In this case, \( S = 2 \) and the strata assignments (see tau.strata argument) would be \( s(1) = s(2) = 1, s(3) = \ldots = s(6) = 2 \).

Prior Specification

The prior distribution for the regression coefficients \( \beta \) is normal.

- If a single number is given, then this is used as the standard deviation and the default mean of 0 is used.
• If a vector is given, it must be of the same length as number of covariates defined and is used as standard deviation.
• If a matrix with a single row is given, its first row will be used as mean and the second row will be used as standard deviation for all regression coefficients.
• Lastly, a two-column matrix (mean and standard deviation columns) with as many columns as regression coefficients can be given.

It is recommended to always specify a `beta.prior`. Per default a mean of 0 is set. The standard deviation is set to 2 for the binary case, to $100 \times sd(y)$ for the normal case and to $sd(\log(y + 0.5 + offset))$ for the Poisson case.

For the between-trial heterogeniety $\tau$ prior, a dispersion parameter must always be given for each exchangeability stratum. For the different `tau.prior` distributions, two parameters are needed out of which one is set to a default value if applicable:

<table>
<thead>
<tr>
<th>Prior</th>
<th>a</th>
<th>b</th>
<th>default</th>
</tr>
</thead>
<tbody>
<tr>
<td>HalfNormal</td>
<td>$\mu = 0$</td>
<td>$\sigma$</td>
<td></td>
</tr>
<tr>
<td>TruncNormal</td>
<td>$\mu$</td>
<td>$\sigma$</td>
<td>$\mu = 0$</td>
</tr>
<tr>
<td>Uniform</td>
<td>a</td>
<td>b</td>
<td>$a = 0$</td>
</tr>
<tr>
<td>Gamma</td>
<td>$\alpha$</td>
<td>$\beta$</td>
<td></td>
</tr>
<tr>
<td>InvGamma</td>
<td>$\alpha$</td>
<td>$\beta$</td>
<td></td>
</tr>
<tr>
<td>LogNormal</td>
<td>$\mu_{\log}$</td>
<td>$\sigma_{\log}$</td>
<td></td>
</tr>
<tr>
<td>TruncCauchy</td>
<td>$\mu$</td>
<td>$\sigma$</td>
<td>$\mu = 0$</td>
</tr>
<tr>
<td>Exp</td>
<td>$\beta$</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Fixed</td>
<td>a</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

For a prior distribution with a default location parameter, a vector of length equal to the number of exchangeability strata can be given. Otherwise, a two-column matrix with as many rows as exchangeability strata must be given, except for a single $\tau$ stratum, for which a vector of length two defines the parameters a and b.

**Random seed**

The MAP analysis is performed using Markov-Chain-Monte-Carlo (MCMC) in `rstan`. MCMC is a stochastic algorithm. To obtain exactly reproducible results you must use the `set.seed` function before calling `gMAP`. See `RBesT` overview page for global options on setting further MCMC simulation parameters.

**References**


See Also

plot.gMAP, forest_plot, automixfit, predict.gMAP

Examples

```r
## Setting up dummy sampling for fast execution of example
## Please use 4 chains and 20x more warmup & iter in practice
user_mc_options <- options(RBest.MC.warmup=50, RBest.MC.iter=100,
RBesT.MC.chains=2, RBest.MC.thin=1)

# Binary data example 1

# Mean response rate is ~0.25. For binary endpoints
# a conservative choice for tau is a HalfNormal(0,1) as long as
# the mean response rate is in the range of 0.2 to 0.8. For
# very small or large rates consider the n_infinity approach
# illustrated below.
# for exact reproducible results, the seed must be set
set.seed(34563)
map_AS <- gMAP(cbind(r, n-r) ~ 1 | study,
               family=binomial,
data=AS,
tau.dist="HalfNormal", tau.prior=1,
beta.prior=2)
print(map_AS)

# obtain numerical summaries
map_sum <- summary(map_AS)
print(map_sum)
names(map_sum)
# [1] "tau" "beta" "theta.pred" "theta"
map_sum$theta.pred

# graphical model checks (returns list of ggplot2 plots)
map_checks <- plot(map_AS)
# forest plot with shrinkage estimates
map_checks$forest_model
# density of MAP prior on response scale
map_checks$densityThetaStar
# density of MAP prior on link scale
map_checks$densityThetaStarLink

# obtain shrinkage estimates
fitted(map_AS)

# regression coefficients
coeff(map_AS)

# finally fit MAP prior with parametric mixture
map_mix <- mixfit(map_AS, Nc=2)
```
plot(map_mix)$mix

# optionally select number of components automatically via AIC
map_automix <- automixfit(map_AS)
plot(map_automix)$mix

# Normal example 2, see normal vignette

# Prior considerations

# The general principle to derive a prior for tau can be based on the
# n_infinity concept as discussed in Neuenschwander et al., 2010.
# This assumes a normal approximation which applies for the colitis
# data set as:
p_bar <- mean(with(colitis, r/n))
s <- round(1/sqrt(p_bar * (1-p_bar)), 1)
# s is the approximate sampling standard deviation and a
# conservative prior is tau ~ HalfNormal(0,s/2)
tau_prior_sd <- s/2

# Evaluate HalfNormal prior for tau
tau_cat <- c(pooling=0
  ,small=0.0625
  ,moderate=0.125
  ,substantial=0.25
  ,large=0.5
  ,veryLarge=1
  ,stratified=Inf)
# Interval probabilities (basically saying we are assuming
# heterogeneity to be smaller than very large)
diff(2*pnorm(tau_cat * s, 0, tau_prior_sd))
# Cumulative probabilities as 1-F
1 - 2*(pnorm(tau_cat * s, 0, tau_prior_sd) - 0.5)

## Recover user set sampling defaults
options(.user_mc_options)

---

**likelihood**

*Read and Set Likelihood to the Corresponding Conjugate Prior*

**Description**

Read and set the likelihood distribution corresponding to the conjugate prior distribution.

**Usage**

`likelihood(mix)`
likelihood(mix) <- value

Arguments

mix Prior mixture distribution.
value New likelihood. **Should** only be changed for Gamma priors as these are supported with either Poisson (value="poisson") or Exponential (value="exp") likelihoods.

Details

If the prior and posterior distributions are in the same family, then the prior distribution is called a conjugate prior for the likelihood function.

Supported Conjugate Prior-Likelihood Pairs

<table>
<thead>
<tr>
<th>Prior/Posterior</th>
<th>Likelihood</th>
<th>Predictive</th>
<th>Summaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beta</td>
<td>Binomial</td>
<td>Beta-Binomial</td>
<td>n, r</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal (fixed σ)</td>
<td>Normal</td>
<td>n, m, se</td>
</tr>
<tr>
<td>Gamma</td>
<td>Poisson</td>
<td>Gamma-Poisson</td>
<td>n, m</td>
</tr>
<tr>
<td>Gamma</td>
<td>Exponential</td>
<td>Gamma-Exp (not supported)</td>
<td>n, m</td>
</tr>
</tbody>
</table>

Examples

```r
# Gamma mixture
gmix <- mixgamma(c(0.3, 20, 4), c(0.7, 50, 10))

# read out conjugate partner
likelihood(gmix)

ess(gmix)

# set conjugate partner
likelihood(gmix) <- "exp"

# ... which changes the interpretation of the mixture
ess(gmix)
```

lodds

*Logit (log-odds) and inverse-logit function.*

**Description**

Calculates the logit (log-odds) and inverse-logit.
mix

Usage

logit(mu)

inv_logit(eta)

Arguments

mu A numeric object with probabilities, with values in the in the range [0,1]. Missing values (NAs) are allowed.

eta A numeric object with log-odds values, with values in the range [-Inf,Inf]. Missing values (NAs) are allowed.

Details

Values of mu equal to 0 or 1 will return -Inf or Inf respectively.

Value

A numeric object of the same type as mu and eta containing the logits or inverse logit of the input values. The logit and inverse transformation equates to

\[
\text{logit}(\mu) = \log(\mu/(1 - \mu))
\]

\[
\text{logit}^{-1}(\eta) = \exp(\eta)/(1 + \exp(\eta)).
\]

Examples

logit(0.2)
inv_logit(-1.386)

mix Mixture Distributions

Description

Density, cumulative distribution function, quantile function and random number generation for supported mixture distributions. (d/p/q/r)mix are generic and work with any mixture supported by BesT (see table below).
Usage

dmix(mix, x, log = FALSE)

pmix(mix, q, lower.tail = TRUE, log.p = FALSE)

qmix(mix, p, lower.tail = TRUE, log.p = FALSE)

rmix(mix, n)

## S3 method for class 'mix'
mix[[..., rescale = FALSE]]

Arguments

mix            mixture distribution object
x, q           vector of quantiles
log, log.p     logical; if TRUE (not default), probabilities p are given as log(p)
lower.tail     logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \)
p              vector of probabilities
n              number of observations. If length(n) > 1, the length is taken to be the number required
...            components to subset given mixture.
rescale        logical; if TRUE, mixture weights will be rescaled to sum to 1

Details

A mixture distribution is defined as a linear superposition of \( K \) densities of the same distributional class. The mixture distributions supported have the form

\[
    f(x, w, a, b) = \sum_{k=1}^{K} w_k f_k(x, a_k, b_k).
\]

The \( w_k \) are the mixing coefficients which must sum to 1. Moreover, each density \( f \) is assumed to be parametrized by two parameters such that each component \( k \) is defined by a triplet, \((w_k, a_k, b_k)\).

Individual mixture components can be extracted using the \([[]\) operator, see examples below.

The supported densities are normal, beta and gamma which can be instantiated with \texttt{mixnorm}, \texttt{mixbeta}, or \texttt{mixgamma}, respectively. In addition, the respective predictive distributions are supported. These can be obtained by calling \texttt{preddist} which returns appropriate normal, beta-binomial or Poisson-gamma mixtures.

For convenience a summary function is defined for all mixtures. It returns the mean, standard deviation and the requested quantiles which can be specified with the argument \texttt{probs}.
mixbeta

Value

dmix gives the weighted sum of the densities of each component.

pmix calculates the distribution function by evaluating the weighted sum of each components distribution function.

qmix returns the quantile for the given p by using that the distribution function is monotonous and hence a gradient based minimization scheme can be used to find the matching quantile q.

rmix generates a random sample of size n by first sampling a latent component indicator in the range 1..K for each draw and then the function samples from each component a random draw using the respective sampling function. The rnorm function returns the random draws as numerical vector with an additional attribute ind which gives the sampled component indicator.

Supported Conjugate Prior-Likelihood Pairs

<table>
<thead>
<tr>
<th>Prior/Posterior</th>
<th>Likelihood</th>
<th>Predictive</th>
<th>Summaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beta</td>
<td>Binomial</td>
<td>Beta-Binomial</td>
<td>n, r</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal (fixed σ)</td>
<td>Normal</td>
<td>n, m, se</td>
</tr>
<tr>
<td>Gamma</td>
<td>Poisson</td>
<td>Gamma-Poisson</td>
<td>n, m</td>
</tr>
<tr>
<td>Gamma</td>
<td>Exponential</td>
<td>Gamma-Exp (not supported)</td>
<td>n, m</td>
</tr>
</tbody>
</table>

See Also

plot.mix

Other mixdist: mixbeta, mixcombine, mixgamma, mixnorm, plot.mix

Examples

```r
# a beta mixture
bm <- mixbeta(weak=c(0.2, 2, 10), inf=c(0.4, 10, 100), inf2=c(0.4, 30, 80))

# extract the two most informative components
bm[c(2,3)]
# rescaling needed in order to plot
plot(bm[c(2,3), rescale=TRUE])

summary(bm)
```

---

**mixbeta**  
**Beta Mixture Density**

**Description**

The Beta mixture density and auxiliary functions.
Usage
mixbeta(..., param = c("ab", "ms", "mn"))

ms2beta(m, s, drop = TRUE)

mn2beta(m, n, drop = TRUE)

## S3 method for class 'betaMix'
summary(object, probs = c(0.025, 0.5, 0.975), ...)

## S3 method for class 'betaBinomialMix'
summary(object, probs = c(0.025, 0.5, 0.975), ...

Arguments

... List of mixture components.

param Determines how the parameters in the list are interpreted. See details.

m Vector of means of beta mixture components.

s Vector of standard deviations of beta mixture components.

drop Delete the dimensions of an array which have only one level.

n Vector of number of observations.

object Beta mixture object.

probs Quantiles reported by the summary function.

Details

Each entry in the ... argument list is expected to be a triplet of numbers which defines the weight $w_k$, first and second parameter of the mixture component $k$. A triplet can optionally be named which will be used appropriately.

The first and second parameter can be given in different parametrizations which is set by the param option:

ab Natural parametrization of Beta density (a=shape1 and b=shape2). Default.

ms Mean and standard deviation, $m = a/(a + b)$ and $s = \sqrt{m(1-m)}/(1+n)$, where $n = a + b$ is the number of observations. Note that $s$ must be less than $\sqrt{m(1 - m)}$.

mn Mean and number of observations, $n = a + b$.

Value

mixbeta returns a beta mixture with the specified mixture components. ms2beta and mn2beta return the equivalent natural a and b parametrization given parameters m, s, or n.

See Also

Other mixdist: mixcombine, mixgamma, mixnorm, mix, plot.mix
Examples

```r
# a beta mixture
bm <- mixbeta(rob=c(0.2, 2, 10), inf=c(0.4, 10, 100), inf2=c(0.4, 30, 80))

# mean/standard deviation parametrization
bm2 <- mixbeta(rob=c(0.2, 0.3, 0.2), inf=c(0.8, 0.4, 0.01), param="ms")

# mean/observations parametrization
bm3 <- mixbeta(rob=c(0.2, 0.3, 5), inf=c(0.8, 0.4, 30), param="mn")

# even mixed is possible
bm4 <- mixbeta(rob=c(0.2, mn2beta(0.3, 5)), inf=c(0.8, ms2beta(0.4, 0.1)))

# print methods are defined
bm4
print(bm4)
```

mixcombine

**Combine Mixture Distributions**

**Description**

Combining mixture distributions of the same class to form a new mixture.

**Usage**

```r
mixcombine(..., weight, rescale = TRUE)
```

**Arguments**

- `...` arbitrary number of mixtures with same distributional class. Each component with values of mixture weight and model parameters.
- `weight` relative weight for each component in new mixture distribution. The vector must be of the same length as input mixtures components. The default value gives equal weight to each component.
- `rescale` boolean value indicates if the weights are rescaled to sum to 1.

**Details**

Combines mixtures of the same class of random variable to form a new mixture distribution.

**Value**

A R-object with the new mixture distribution.
mixdiff

Difference of mixture distributions

Description
Density, cumulative distribution function, quantile function and random number generation for the difference of two mixture distributions.

Usage

- `dmixdiff(mix1, mix2, x)`
- `pmixdiff(mix1, mix2, q, lower.tail = TRUE)`
- `qmixdiff(mix1, mix2, p, lower.tail = TRUE)`
- `rmixdiff(mix1, mix2, n)`

Arguments

- `mix1`: first mixture density
- `mix2`: second mixture density
- `x`: vector of values for which density values are computed
- `q`: vector of quantiles for which cumulative probabilities are computed
- `lower.tail`: logical; if TRUE (default), probabilities are P[X <= x], otherwise P[X > x].
- `p`: vector of cumulative probabilities for which quantiles are computed
- `n`: size of random sample
mixdiff

Details

If \( x_1 \sim f_1(x) \) and \( x_2 \sim f_2(x) \), the density of the difference \( x = x_1 - x_2 \) is given by the convolution

\[
f(x) = \int f_1(x) f_2(x - u) \, du = (f_1 \ast f_2)(x).
\]

The cumulative distribution function equates to

\[
F(x) = \int F_1(x + u) f_2(u) \, du.
\]

Both integrals are performed over the full support of the densities and use the numerical integration function `integrate`.

Value

Respective density, quantile, cumulative density or random numbers.

Examples

# 1. Difference between two beta distributions, i.e. Pr( mix1 - mix2 > 0)
mix1 <- mixbeta(c(1, 11, 4))
mix2 <- mixbeta(c(1, 8, 7))

pmixdiff(mix1, mix2, 0, FALSE)

# Interval probability, i.e. Pr( 0.3 > mix1 - mix2 > 0)

pmixdiff(mix1, mix2, 0.3) - pmixdiff(mix1, mix2, 0)

# 2. two distributions, one of them a mixture
m1 <- mixbeta(c(1,30,50))
m2 <- mixbeta(c(0.75,20,50),c(0.25,1,1))

# random sample of difference
set.seed(23434)
rM <- rmixdiff(m1, m2, 1E4)

# histogram of random numbers and exact density
hist(rM,prob=TRUE,new=TRUE,nclass=40)
curve(dmixdiff(m1,m2,x), add=TRUE, n=51)

# threshold probabilities for difference, at 0 and 0.2

pmixdiff(m1, m2, 0)
mean(rM<0)

pmixdiff(m1,m2,0.2)
mean(rM<0.2)

# median of difference
mdn <- qmixdiff(m1, m2, 0.5)
mean(rM<mdn)
# 95%-interval
qmixdiff(m1, m2, c(0.025, 0.975))
quantile(rM, c(0.025, 0.975))

mixfit

Fit of Mixture Densities to Samples

Description
Expectation-Maximization (EM) based fitting of parametric mixture densities to numerical samples. This provides a convenient approach to approximate MCMC samples with a parametric mixture distribution.

Usage
mixfit(sample, type = c("norm", "beta", "gamma"), thin, ...)

# Default S3 method:
mixfit(sample, type = c("norm", "beta", "gamma"), thin,
...

# S3 method for class 'gMAP'
mixfit(sample, type, thin, ...)

# S3 method for class 'gMAPpred'
mixfit(sample, type, thin, ...)

# S3 method for class 'array'
mixfit(sample, type, thin, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sample</td>
<td>Sample to be fitted.</td>
</tr>
<tr>
<td>type</td>
<td>Mixture density to use. Can be either norm, beta or gamma.</td>
</tr>
<tr>
<td>thin</td>
<td>Thinning applied to the sample. See description for default behavior.</td>
</tr>
<tr>
<td>...</td>
<td>Parameters passed to the low-level EM fitting functions. Parameter Nc is mandatory.</td>
</tr>
</tbody>
</table>

Details
Parameters of EM fitting functions

Nc  Number of mixture components. Required parameter.

mix_init  Initial mixture density. If missing (default) then a k-nearest-neighbor algorithm is used to find an initial mixture density.
**Ninit**  Number of data points used for initialization. Defaults to 50.

**verbose**  If set to `TRUE` the function will inform about fitting process

**maxIter**  Maximal number of iterations. Defaults to 500.

**tol**  Defines a convergence criteria as an upper bound for the change in the log-likelihood, i.e. once the derivative (with respect to iterations) of the log-likelihood falls below `tol`, the function declares convergence and stops.

**eps**  Must be a triplet of numbers which set the desired accuracy of the inferred parameters per mixture component. See below for a description of the parameters used during EM. EM is stopped once a running mean of the absolute difference between the last successive `Neps` estimates is below the given `eps` for all parameters. Defaults to 5E-3 for each parameter.

**Neps**  Number of iterations used for the running mean of parameter estimates to test for convergence. Defaults to 5.

By default the EM convergence is declared when the desired accuracy of the parameters has been reached over the last `Neps` estimates. If `tol` and `Neps` is specified, then whatever criterion is met first will stop the EM.

The parameters per component $k$ used internally during fitting are for the different EM procedures:

- **normal**  $\text{logit}(w_k), \mu_k, \log(\sigma_k)$
- **beta**  $\text{logit}(w_k), \text{logit}(\mu_k), \log(n_k)$
- **gamma**  $\text{logit}(w_k), \log(\alpha_k), \log(\beta_k)$

*Note:* Whenever no `mix_init` argument is given, the EM fitting routines assume that the data vector is given in random order. If in the unlikely event that the EM gets caught in a local extremum, then random reordering of the data vector may alleviate the issue.

**Value**

A mixture object according the requested `type` is returned. The object has additional information attached, i.e. the log-likelihood can be queried and diagnostic plots can be generated. See links below.

**Methods (by class)**

- **default**: Performs an EM fit for the given sample. Thinning is applied only if thin is specified.
- **gMAP**: Fits the default predictive distribution from a gMAP analysis. Automatically obtains the predictive distribution of the intercept only case on the response scale mixture from the `gMAP` object. For the binomial case a beta mixture, for the gaussian case a normal mixture and for the Poisson case a gamma mixture will be used. In the gaussian case, the resulting normal mixture will set the reference scale to the estimated sigma in `gMAP` call.
- **gMAPpred**: Fits a mixture density for each prediction from the `gMAP` prediction.
- **array**: Fits a mixture density for an MCMC sample. It is recommended to provide a thinning argument which roughly yields independent draws (i.e. use `acf` to identify a thinning lag with small auto-correlation). The input array is expected to have 3 dimensions which are nested as iterations, chains, and draws.
References

See Also
Other EM: plot.EM

Examples
```r
bmix <- mixbeta(rob = c(0.2, 1, 1), inf = c(0.8, 10, 2))
bsamp <- rmix(bmix, 1000)
bfit <- mixfit(bsamp, type = "beta", Nc = 2)
# diagnostic plots can easily by generated from the EM fit with
bfit.check <- plot(bfit)
names(bfit.check)
# check convergence of parameters
bfit.check$mix
bfit.check$mixens
# obtain the log-likelihood
logLik(bfit)
# or AIC
AIC(bfit)
```

mixgamma

*The Gamma Mixture Distribution*

Description
The gamma mixture density and auxiliary functions.

Usage
```r
mixgamma(..., param = c("ab", "ms", "mn"), likelihood = c("poisson", "exp"))
ms2gamma(m, s, drop = TRUE)
mn2gamma(m, n, likelihood = c("poisson", "exp"), drop = TRUE)
## S3 method for class 'gammaMix'
```
Example usage:

```r
summary(object, probs = c(0.025, 0.5, 0.975), ...)
```

### S3 method for class 'gammaPoissonMix'
```r
summary(object, probs = c(0.025, 0.5, 0.975), ...)
```

#### Arguments

- `...` List of mixture components.
- `param` Determines how the parameters in the list are interpreted. See details.
- `likelihood` Defines with what likelihood the Gamma density is used (Poisson or Exp). Defaults to poisson.
- `m` Vector of means of the Gamma mixture components.
- `s` Vector of standard deviations of the Gamma mixture components.
- `drop` Delete the dimensions of an array which have only one level.
- `n` Vector of sample sizes of the Gamma mixture components.
- `object` Gamma mixture object.
- `probs` Quantiles reported by the `summary` function.

#### Details

Each entry in the `...` argument list is expected to be a triplet of numbers which defines the weight \( w_k \), first and second parameter of the mixture component \( k \). A triplet can optionally be named which will be used appropriately.

The first and second parameter can be given in different parametrizations which is set by the `param` option:

- **ab** Natural parametrization of Gamma density \( a=\text{shape} \text{ and } b=\text{rate} \). Default.
- **ms** Mean and standard deviation, \( m = a/b \text{ and } s = \sqrt{a/b} \).
- **mn** Mean and number of observations. Translation to natural parameter depends on the `likelihood` argument. For a Poisson likelihood \( n = b \text{ (and } a = m \cdot n) \), for an Exp likelihood \( n = a \text{ (and } b = n/m) \).

#### Value

`mixgamma` returns a gamma mixture with the specified mixture components. `ms2gamma` and `mn2gamma` return the equivalent natural \( a \text{ and } b \) parametrization given parameters \( m, s, \text{ or } n \).

#### See Also

Other `mixdist`: `mixbeta, mixcombine, mixnorm, mix, plot.mix`
Examples

# Gamma mixture with robust and informative component
gmix <- mixgamma(rob=c(0.3, 20, 4), inf=c(0.7, 50, 10))

# objects can be printed
gmix
# or explicitly
print(gmix)

# summaries are defined
summary(gmix)

# sub-components may be extracted
# by component number
gmix[2]
# or component name
gmix["inf"]

# alternative mean and standard deviation parametrization
gmsMix <- mixgamma(rob=c(0.5, 8, 0.5), inf=c(0.5, 9, 2), param="ms")

# or mean and number of observations parametrization
gmnMix <- mixgamma(rob=c(0.2, 2, 1), inf=c(0.8, 2, 5), param="mn")

# and mixed parametrizations are also possible
gfmix <- mixgamma(rob1=c(0.15, mn2gamma(2, 1)), rob2=c(0.15, ms2gamma(2, 5)), inf=c(0.7, 50, 10))

mixnorm

Normal Mixture Density

Description

The normal mixture density and auxiliary functions.

Usage

mixnorm(..., sigma, param = c("ms", "mn"))

mn2norm(m, n, sigma, drop = TRUE)

### S3 method for class 'normMix'
summary(object, probs = c(0.025, 0.5, 0.975), ...)

### S3 method for class 'normMix'
sigma(object, ...)

sigma(object) <- value
Arguments

... list of mixture components.
sigma reference scale.
param determines how the parameters in the list are interpreted. See details.
m vector of means
n vector of sample sizes.
drop delete the dimensions of an array which have only one level.
object normal mixture object.
probs quantiles reported by the summary function.
value new value of the reference scale sigma.

Details

Each entry in the ... argument list is expected to be a triplet of numbers which defines the weight \( w_k \), first and second parameter of the mixture component \( k \). A triplet can optionally be named which will be used appropriately.

The first and second parameter can be given in different parametrizations which is set by the param option:

- **ms** Mean and standard deviation. Default.
- **mn** Mean and number of observations. \( n \) determines \( s \) via the relation \( s = \sigma / \sqrt{n} \) with \( \sigma \) being the fixed reference scale.

The reference scale \( \sigma \) is the fixed standard deviation in the one-parameter normal-normal model (observation standard deviation). The function sigma can be used to query the reference scale and may also be used to assign a new reference scale, see examples below. In case the sigma is not specified, the user has to supply sigma as argument to functions which require a reference scale.

Value

Returns a normal mixture with the specified mixture components. mn2norm returns the mean and standard deviation given a mean and sample size parametrization.

Functions

- sigma<-: Allows to assign a new reference scale sigma.

See Also

Other mixdist: mixbeta, mixcombine, mixgamma, mix, plot.mix
Examples

```r
nm <- mixnorm(rob=c(0.2, 0, 2), inf=c(0.8, 2, 2), sigma=5)
print(nm)
summary(nm)
plot(nm)

set.seed(1)
mixSamp <- rmix(nm, 500)
plot(nm, samp=mixSamp)

# support defined by quantiles
qmix(nm, c(0.01, 0.99))

# density function
dmix(nm, seq(-5,5,by=2))

# distribution function
pmix(nm, seq(-5,5,by=2))

# the reference scale can be changed (it determines the ESS)
ess(nm)

sigma(nm) <- 10
ess(nm)
```

---

**oc1S**

*Operating Characteristics for 1 Sample Design*

**Description**

The `oc1S` function defines a 1 sample design (prior, sample size, decision function) for the calculation of the frequency at which the decision is evaluated to 1 conditional on assuming known parameters. A function is returned which performs the actual operating characteristics calculations.

**Usage**

```r
oc1S(prior, n, decision, ...)
```

## S3 method for class 'betaMix'

```r
oc1S(prior, n, decision, ...)
```

## S3 method for class 'normMix'

```r
oc1S(prior, n, decision, sigma, eps = 1e-06, ...)
```

## S3 method for class 'gammaMix'

```r
oc1S(prior, n, decision, eps = 1e-06, ...)
```
**Arguments**

- **prior**: Prior for analysis.
- **n**: Sample size for the experiment.
- **decision**: One-sample decision function to use; see `decision1S`.
- **...**: Optional arguments.
- **sigma**: The fixed reference scale. If left unspecified, the default reference scale of the prior is assumed.
- **eps**: Support of random variables are determined as the interval covering $1-\epsilon$ probability mass. Defaults to $10^{-6}$.

**Details**

The `oc1S` function defines a 1 sample design and returns a function which calculates its operating characteristics. This is the frequency with which the decision function is evaluated to 1 under the assumption of a given true distribution of the data defined by a known parameter $\theta$. The 1 sample design is defined by the prior, the sample size and the decision function, $D(y)$. These uniquely define the decision boundary, see `decision1S_boundary`.

When calling the `oc1S` function, then internally the critical value $y_c$ (using `decision1S_boundary`) is calculated and a function is returns which can be used to calculated the desired frequency which is evaluated as

$$F(y_c|\theta).$$

**Value**

Returns a function with one argument `theta` which calculates the frequency at which the decision function is evaluated to 1 for the defined 1 sample design. Note that the returned function takes vectors arguments.

**Methods (by class)**

- **betaMix**: Applies for binomial model with a mixture beta prior. The calculations use exact expressions.
- **normMix**: Applies for the normal model with known standard deviation $\sigma$ and a normal mixture prior for the mean. As a consequence from the assumption of a known standard deviation, the calculation discards sampling uncertainty of the second moment. The function `oc1S` has an extra argument `eps` (defaults to $10^{-6}$). The critical value $y_c$ is searched in the region of probability mass $1-\epsilon$ for $y$.
- **gammaMix**: Applies for the Poisson model with a gamma mixture prior for the rate parameter. The function `oc1S` takes an extra argument `eps` (defaults to $10^{-6}$) which determines the region of probability mass $1-\epsilon$ where the boundary is searched for $y$.

**See Also**

Other design1S: `decision1S_boundary`, `decision1S`, `pos1S`
Examples

# non-inferiority example using normal approximation of log-hazard
# ratio, see ?decision1S for all details
s <- 2
flat_prior <- mixnorm(c(1,0,100), sigma=s)
nL <- 233
theta_ni <- 0.4
theta_a <- 0
alpha <- 0.05
beta <- 0.2
za <- qnorm(1-alpha)
zb <- qnorm(1-beta)
n1 <- round((s * (za + zb)/(theta_ni - theta_a))^2 )
theta_c <- theta_ni - za * s / sqrt(n1)

# standard NI design
decA <- decision1S(1 - alpha, theta_ni, lower.tail=TRUE)

# double criterion design
# statistical significance (like NI design)
dec1 <- decision1S(1-alpha, theta_ni, lower.tail=TRUE)
# require mean to be at least as good as theta_c
dec2 <- decision1S(0.5, theta_c, lower.tail=TRUE)
# combination
decComb <- decision1S(c(1-alpha, 0.5), c(theta_ni, theta_c), lower.tail=TRUE)

theta_eval <- c(theta_a, theta_c, theta_ni)

# evaluate different designs at two sample sizes
designA_n1 <- oc1S(flat_prior, n1, decA)
designA_nL <- oc1S(flat_prior, nL, decA)
designC_n1 <- oc1S(flat_prior, n1, decComb)
designC_nL <- oc1S(flat_prior, nL, decComb)

# evaluate designs at the key log-HR of positive treatment (HR<1),
# the indecision point and the NI margin

designA_n1(theta_eval)
designA_nL(theta_eval)
designC_n1(theta_eval)
designC_nL(theta_eval)

# to understand further the dual criterion design it is useful to
# evaluate the criterions separately:
# statistical significance criterion to warrant NI...
designC1_nL <- oc1S(flat_prior, nL, dec1)
# ... or the clinically determined indifference point
designC2_nL <- oc1S(flat_prior, nL, dec2)

designC1_nL(theta_eval)
designC2_nL(theta_eval)
 Operating Characteristics for 2 Sample Design

Description

The oc2S function defines a 2 sample design (priors, sample sizes & decision function) for the calculation of operating characteristics. A function is returned which calculates the frequency at which the decision function is evaluated to 1 when assuming known parameters.

Usage

```
oc2S(prior1, prior2, n1, n2, decision, ...)  
## S3 method for class 'betaMix'
oc2S(prior1, prior2, n1, n2, decision, eps, ...)
## S3 method for class 'normMix'
oc2S(prior1, prior2, n1, n2, decision, sigma1, sigma2,  
   eps = 1e-06, Ngrid = 10, ...)
## S3 method for class 'gammaMix'
oc2S(prior1, prior2, n1, n2, decision, eps = 1e-06,  
   ...)```

Arguments

- `prior1`: Prior for sample 1.
- `prior2`: Prior for sample 2.
- `n1, n2`: Sample size of the respective samples. Sample size `n1` must be greater than 0 while sample size `n2` must be greater or equal to 0.
- `decision`: Two-sample decision function to use; see `decision2S`.
- `...`: Optional arguments.
- `eps`: Support of random variables are determined as the interval covering 1-eps probability mass. Defaults to $10^{-6}$.
- `sigma1`: The fixed reference scale of sample 1. If left unspecified, the default reference scale of the prior 1 is assumed.
- `sigma2`: The fixed reference scale of sample 2. If left unspecified, the default reference scale of the prior 2 is assumed.
- `Ngrid`: Determines density of discretization grid on which decision function is evaluated (see below for more details).
Details

The oc2S function defines a 2 sample design and returns a function which calculates its operating characteristics. This is the frequency with which the decision function is evaluated to 1 under the assumption of a given true distribution of the data defined by the known parameter $\theta_1$ and $\theta_2$. The 2 sample design is defined by the priors, the sample sizes and the decision function, $D(y_1, y_2)$. These uniquely define the decision boundary, see decision2S_boundary.

Calling the oc2S function calculates the decision boundary $D_1(y_2)$ (see decision2S_boundary) and returns a function which can be used to calculate the desired frequency which is evaluated as

$$\int f_2(y_2|\theta_2)F_1(D_1(y_2)|\theta_1)dy_2.$$ 

See below for examples and specifics for the supported mixture priors.

Value

Returns a function which when called with two arguments theta1 and theta2 will return the frequencies at which the decision function is evaluated to 1 whenever the data is distributed according to the known parameter values in each sample. Note that the returned function takes vector arguments.

Methods (by class)

- betaMix: Applies for binomial model with a mixture beta prior. The calculations use exact expressions. If the optional argument eps is defined, then an approximate method is used which limits the search for the decision boundary to the region of 1-eps probability mass. This is useful for designs with large sample sizes where an exact approach is very costly to calculate.
- normMix: Applies for the normal model with known standard deviation $\sigma$ and normal mixture priors for the means. As a consequence from the assumption of a known standard deviation, the calculation discards sampling uncertainty of the second moment. The function has two extra arguments (with defaults): eps ($10^{-6}$) and Ngrid (10). The decision boundary is searched in the region of probability mass 1-eps, respectively for $y_1$ and $y_2$. The continuous decision function is evaluated at a discrete grid, which is determined by a spacing with $\delta_2 = \sigma_2 / \sqrt{N_{\text{grid}}}$. Once the decision boundary is evaluated at the discrete steps, a spline is used to interpolate the decision boundary at intermediate points.
- gammaMix: Applies for the Poisson model with a gamma mixture prior for the rate parameter. The function oc2S takes an extra argument eps (defaults to $10^{-6}$) which determines the region of probability mass 1-eps where the boundary is searched for $y_1$ and $y_2$, respectively.

References


See Also

Other design2S: decision2S_boundary, decision2S, pos2S
Examples

```r
# example from Schmidli et al., 2014
dec <- decision2S(0.975, 0, lower.tail=FALSE)

prior_inf <- mixbeta(c(1, 4, 16))
prior_rob <- robustify(prior_inf, weight=0.2, mean=0.5)
prior_uni <- mixbeta(c(1, 1, 1))

N <- 40
N_ctl <- N - 20

# compare designs with different priors
design_uni <- oc2S(prior_uni, prior_uni, N, N_ctl, dec)
design_inf <- oc2S(prior_uni, prior_inf, N, N_ctl, dec)
design_rob <- oc2S(prior_uni, prior_rob, N, N_ctl, dec)

# type I error
curve(design_inf(x,x), 0, 1)
curve(design_uni(x,x), lty=2, add=TRUE)
curve(design_rob(x,x), lty=3, add=TRUE)

# power
curve(design_inf(0.2+x,0.2), 0, 0.5)
curve(design_uni(0.2+x,0.2), lty=2, add=TRUE)
curve(design_rob(0.2+x,0.2), lty=3, add=TRUE)
```

---

`plot.EM` *Diagnostic plots for EM fits*

**Description**

Produce diagnostic plots of EM fits returned from `mixfit`.

**Usage**

```r
## S3 method for class 'EM'
plot(x, size = 1.25, link = c("identity", "logit", "log"), ...
```

**Arguments**

- `x`: EM fit
- `size`: Optional argument passed to ggplot2 routines which control line thickness.
- `link`: Choice of an applied link function. Can take one of the values identity (default), logit or log.
Ignored.

Overlays the fitted mixture density with a histogram and a density plot of the raw sample fitted. Applying a link function can be beneficial, for example a logit (log) link for beta (gamma) mixtures obtained from a Binomial (Poisson) gMAP analysis.

Value

A list of ggplot plots for diagnostics of the EM run. Detailed EM diagnostic plots are included only if the global option RBesT.verbose is set to TRUE. These include plots of the parameters of each component vs the iteration. The plot of the mixture density with a histogram and a density of the fitted sample is always returned.

Customizing ggplot2 plots

The returned plot is a ggplot2 object. Please refer to the "Customizing Plots" vignette which is part of RBesT documentation for an introduction. For simple modifications (change labels, add reference lines, ...) consider the commands found in bayesplot-helpers. For more advanced customizations please use the ggplot2 package directly. A description of the most common tasks can be found in the R Cookbook and a full reference of available commands can be found at the ggplot2 documentation site.

See Also

Other EM: mixfit

Examples

bmix <- mixbeta(rob=c(0.2, 1, 1), inf=c(0.8, 10, 2))
bsamp <- rmix(bmix, 1000)
bfit <- mixfit(bsamp, type="beta", Nc=2)
pl <- plot(bfit)
print(pl$mixdens)
print(pl$mix)

# a number of additional plots are generated in verbose mode
.user_option <- options(RBesT.verbose=TRUE)
pl_all <- plot(bfit)
# recover previous user options
options(.user_option)
names(pl_all)
# [1] "a" "b" "w" "m" "N" "Lm" "1N" "Lw" "l1i" "mixdens" "mix"
Diagnostic plots for gMAP analyses

Description

Diagnostic plots for gMAP analyses

Usage

```r
## S3 method for class 'gMAP'
plot(x, size = NULL, ...)
```

Arguments

- `x`: gMAP object
- `size`: Controls line sizes of traceplots and forest plot.
- `...`: Ignored.

Details

Creates MCMC diagnostics and a forest plot (including model estimates) for a gMAP analysis. For a customized forest plot, please use the dedicated function `forest_plot`.

Value

The function returns a list of `ggplot` objects.

Customizing ggplot2 plots

The returned plot is a `ggplot2` object. Please refer to the "Customizing Plots" vignette which is part of RBesT documentation for an introduction. For simple modifications (change labels, add reference lines, ...) consider the commands found in `bayesplot-helpers`. For more advanced customizations please use the `ggplot2` package directly. A description of the most common tasks can be found in the R Cookbook and a full reference of available commands can be found at the `ggplot2` documentation site.
Plot mixture distributions

Usage

```r
## S3 method for class 'mix'
plot(x, prob = 0.99, fun = dmix, log = FALSE,
     comp = TRUE, size = 1.25, ...)
```

Arguments

- `x`: mixture distribution
- `prob`: defining lower and upper percentile of x-axis. Defaults to the 99% central probability mass.
- `fun`: function to plot which can be any of `dmix`, `qmix` or `pmix`.
- `log`: log argument passed to the function specified in `fun`.
- `comp`: for the density function this can be set to `TRUE` which will display colour-coded each mixture component of the density in addition to the density.
- `size`: controls the linesize in plots.
- `...`: extra arguments passed on to the `qplot` call.

Details

Plot function for mixture distribution objects. It shows the density/quantile/cumulative distribution (corresponds to `d/q/pmix` function) for some specific central probability mass defined by `prob`. By default the x-axis is chosen to show 99% of the probability density mass.

Value

A `ggplot` object is returned.

Customizing ggplot2 plots

The returned plot is a `ggplot2` object. Please refer to the "Customizing Plots" vignette which is part of `RBesT` documentation for an introduction. For simple modifications (change labels, add reference lines, ...) consider the commands found in `bayesplot-helpers`. For more advanced customizations please use the `ggplot2` package directly. A description of the most common tasks can be found in the `R Cookbook` and a full reference of available commands can be found at the `ggplot2` documentation site.

See Also

Other mixdist: `mixbeta`, `mixcombine`, `mixgamma`, `mixnorm`, `mix`
Examples

# beta with two informative components
bm <- mixbeta(inf=c(0.5, 10, 100), inf2=c(0.5, 30, 80))
plot(bm)
plot(bm, fun=pmix)

# for customizations of the plot we need to load ggplot2 first
library(ggplot2)

# show a histogram along with the density
plot(bm) + geom_histogram(data=data.frame(x=rmix(bm, 1000)),
    aes(y=..density..), bins=50, alpha=0.4)

# note: we can also use bayesplot for histogram plots with a density ...
library(bayesplot)
mh <- mcmc_hist(data.frame(x=rmix(bm, 1000)), freq=FALSE) +
    overlay_function(fun=dmix, args=list(mix=bm))
# ...and even add each component
for(k in 1:ncol(bm))
    mh <- mh + overlay_function(fun=dmix, args=list(mix=bm[[k]]), linetype=I(2))
print(mh)

# normal mixture
nm <- mixnorm(rob=c(0.2, 0, 2), inf=c(0.8, 6, 2), sigma=5)
plot(nm)
plot(nm, fun=qmix)

# obtain ggplot2 object and change title
pl <- plot(nm)
pl + ggtitle("Normal 2-Component Mixture")

Description

The pos1S function defines a 1 sample design (prior, sample size, decision function) for the calculation of the frequency at which the decision is evaluated to 1 when assuming a distribution for the parameter. A function is returned which performs the actual operating characteristics calculations.

Usage

pos1S(prior, n, decision, ...)

## S3 method for class 'betaMix'
pos1S(prior, n, decision, ...)
## S3 method for class 'normMix'
pos1S(prior, n, decision, sigma, eps = 1e-06, ...)

## S3 method for class 'gammaMix'
pos1S(prior, n, decision, eps = 1e-06, ...)

### Arguments

- **prior**: Prior for analysis.
- **n**: Sample size for the experiment.
- **decision**: One-sample decision function to use; see `decision1S`.
- **...**: Optional arguments.
- **sigma**: The fixed reference scale. If left unspecified, the default reference scale of the prior is assumed.
- **eps**: Support of random variables are determined as the interval covering 1-eps probability mass. Defaults to $10^{-6}$.

### Details

The `pos1S` function defines a 1 sample design and returns a function which calculates its probability of success. The probability of success is the frequency with which the decision function is evaluated to 1 under the assumption of a given true distribution of the data implied by a distribution of the parameter $\theta$.

Calling the `pos1S` function calculates the critical value $y_c$ and returns a function which can be used to evaluate the PoS for different predictive distributions and is evaluated as

$$\int F(y_c|\theta)p(\theta)d\theta,$$

where $F$ is the distribution function of the sampling distribution and $p(\theta)$ specifies the assumed true distribution of the parameter $\theta$. The distribution $p(\theta)$ is a mixture distribution and given as the `mix` argument to the function.

### Value

Returns a function that takes as single argument `mix`, which is the mixture distribution of the control parameter. Calling this function with a mixture distribution then calculates the PoS.

### Methods (by class)

- **betaMix**: Applies for binomial model with a mixture beta prior. The calculations use exact expressions.
- **normMix**: Applies for the normal model with known standard deviation $\sigma$ and a normal mixture prior for the mean. As a consequence from the assumption of a known standard deviation, the calculation discards sampling uncertainty of the second moment. The function `pos1S` has an extra argument `eps` (defaults to $10^{-6}$). The critical value $y_c$ is searched in the region of probability mass 1-eps for $y$. 

• \texttt{gammaMix}: Applies for the Poisson model with a gamma mixture prior for the rate parameter. The function \texttt{pos1S} takes an extra argument \texttt{eps} (defaults to $10^{-6}$) which determines the region of probability mass $1-\texttt{eps}$ where the boundary is searched for $y$.

\textbf{See Also}

Other design functions: \texttt{decision1S\_boundary}, \texttt{decision1S}, \texttt{oct1S}

\textbf{Examples}

```r
# non-inferiority example using normal approximation of log-hazard ratio, see \texttt{?decision1S} for all details
s <- 2
flat_prior <- mixnorm(c(1,0,100), sigma=s)
nL <- 233
theta_ni <- 0.4
theta_a <- 0
alpha <- 0.05
beta <- 0.2
za <- qnorm(1-alpha)
zb <- qnorm(1-beta)
n1 <- round((s * (za + zb)/(theta_ni - theta_a))^2 )
theta_c <- theta_ni - za * s / sqrt(n1)

# assume we would like to conduct at an interim analysis
# of PoS after having observed 20 events with a HR of 0.8.
# We first need the posterior at the interim ...
post_ia <- postmix(flat_prior, m=log(0.8), n=20)

decComb <- decision1S(c(1-alpha, 0.5), c(theta_ni, theta_c), lower.tail=TRUE)

# ... and we would like to know the PoS for a successful
# trial at the end when observing 10 more events
pos_ia <- pos1S(post_ia, 10, decComb)

# our knowledge at the interim is just the posterior at
# interim such that the PoS is
pos_ia(post_ia)
```

\texttt{pos2S} \hspace{1cm} \textit{Probability of Success for 2 Sample Design}

\textbf{Description}

The \texttt{pos2S} function defines a 2 sample design (priors, sample sizes & decision function) for the calculation of the probability of success. A function is returned which calculates the
frequency at which the decision function is evaluated to 1 when parameters are distributed according to the given distributions.

Usage

\[
\text{pos2S}(\text{prior1}, \text{prior2}, n1, n2, \text{decision}, ...) \\
## S3 method for class 'betaMix'
\text{pos2S}(\text{prior1}, \text{prior2}, n1, n2, \text{decision}, \text{eps}, ...)
## S3 method for class 'normMix'
\text{pos2S}(\text{prior1}, \text{prior2}, n1, n2, \text{decision}, \text{sigma1}, \text{sigma2}, \\
\text{eps} = 1e-06, \text{Ngrid} = 10, ...)
## S3 method for class 'gammaMix'
\text{pos2S}(\text{prior1}, \text{prior2}, n1, n2, \text{decision}, \text{eps} = 1e-06, \\
...)
\]

Arguments

- **prior1**: Prior for sample 1.
- **prior2**: Prior for sample 2.
- **n1, n2**: Sample size of the respective samples. Sample size \(n1\) must be greater than 0 while sample size \(n2\) must be greater or equal to 0.
- **decision**: Two-sample decision function to use; see `decision2S`.
- **...**: Optional arguments.
- **eps**: Support of random variables are determined as the interval covering \(1-\text{eps}\) probability mass. Defaults to \(10^{-6}\).
- **sigma1**: The fixed reference scale of sample 1. If left unspecified, the default reference scale of the prior 1 is assumed.
- **sigma2**: The fixed reference scale of sample 2. If left unspecified, the default reference scale of the prior 2 is assumed.
- **Ngrid**: Determines density of discretization grid on which decision function is evaluated (see below for more details).

Details

The `pos2S` function defines a 2 sample design and returns a function which calculates its probability of success. The probability of success is the frequency with which the decision function is evaluated to 1 under the assumption of a given true distribution of the data implied by a distribution of the parameters \(\theta_1\) and \(\theta_2\).

The calculation is analogous to the operating characteristics `oc2S` with the difference that instead of assuming known (point-wise) true parameter values a distribution is specified for each parameter.

Calling the `pos2S` function calculates the decision boundary \(D_1(y_2)\) and returns a function which can be used to evaluate the PoS for different predictive distributions. It is evaluated as
\[\int \int \int f_2(y_2|\theta_2) \, p(\theta_2) \, F_1(D_1(y_2)|\theta_1) \, p(\theta_1) \, dy_2 \, d\theta_2 \, d\theta_1.\]

where \( F \) is the distribution function of the sampling distribution and \( p(\theta_1) \) and \( p(\theta_2) \) specifies the assumed true distribution of the parameters \( \theta_1 \) and \( \theta_2 \), respectively. Each distribution \( p(\theta_1) \) and \( p(\theta_2) \) is a mixture distribution and given as the mix1 and mix2 argument to the function.

For example, in the binary case an integration of the predictive distribution, the BetaBinomial, instead of the binomial distribution will be performed over the data space wherever the decision function is evaluated to 1. All other aspects of the calculation are as for the 2-sample operating characteristics, see \texttt{oc2S}.

**Value**

Returns a function which when called with two arguments mix1 and mix2 will return the frequencies at which the decision function is evaluated to 1. Each argument is expected to be a mixture distribution representing the assumed true distribution of the parameter in each group.

**Methods (by class)**

- \texttt{betaMix}: Applies for binomial model with a mixture beta prior. The calculations use exact expressions. If the optional argument \( \text{eps} \) is defined, then an approximate method is used which limits the search for the decision boundary to the region of \( 1-\text{eps} \) probability mass. This is useful for designs with large sample sizes where an exact approach is very costly to calculate.

- \texttt{normMix}: Applies for the normal model with known standard deviation \( \sigma \) and normal mixture priors for the means. As a consequence from the assumption of a known standard deviation, the calculation discards sampling uncertainty of the second moment. The function has two extra arguments (with defaults): \( \text{eps} = 10^{-6} \) and \( \text{Ngrid} = 10 \). The decision boundary is searched in the region of probability mass \( 1-\text{eps} \), respectively for \( y_1 \) and \( y_2 \). The continuous decision function is evaluated at a discrete grid, which is determined by a spacing with \( \delta_2 = \sigma_2 / \sqrt{\text{Ngrid}}. \) Once the decision boundary is evaluated at the discrete steps, a spline is used to inter-polate the decision boundary at intermediate points.

- \texttt{gammaMix}: Applies for the Poisson model with a gamma mixture prior for the rate parameter. The function \texttt{pos2S} takes an extra argument \( \text{eps} \) (defaults to \( 10^{-6} \)) which determines the region of probability mass \( 1-\text{eps} \) where the boundary is searched for \( y_1 \) and \( y_2 \), respectively.

**See Also**

Other design2S: \texttt{decision2S} \_\texttt{boundary}, \texttt{decision2S}, \texttt{oc2S}

**Examples**

```r
# see ?decision2S for details of example
priorT <- mixnorm(c(1, 0, 0.001), sigma=88, param="mn")
priorP <- mixnorm(c(1, -49, 20 ), sigma=88, param="mn")
# the success criteria is for delta which are larger than some
```
# threshold value which is why we set lower.tail=FALSE
successCrit <- decision2S(c(0.95, 0.5), c(0, 50), FALSE)

# example interim outcome
postP_interim <- postmix(priorP, n=10, m=-50)
postT_interim <- postmix(priorT, n=20, m=-80)

# assume that mean -50 / -80 were observed at the interim for
# placebo control(n=10) / active treatment(n=20) which gives
# the posteriors
postP_interim
postT_interim

# then the PoS to succeed after another 20/30 patients is
pos_final <- pos2S(postP_interim, postT_interim, 20, 30, successCrit)
pos_final(postP_interim, postT_interim)

---

**postmix**  
*Conjugate Posterior Analysis*

**Description**

Calculates the posterior distribution for data *data* given a prior *priormix*, where the prior is a mixture of conjugate distributions. The posterior is then also a mixture of conjugate distributions.

**Usage**

```r
postmix(priormix, data, ...)
```

## S3 method for class 'betaMix'
```r
postmix(priormix, data, n, r, ...)
```

## S3 method for class 'normMix'
```r
postmix(priormix, data, n, m, se, ...)
```

## S3 method for class 'gammaMix'
```r
postmix(priormix, data, n, m, ...)
```

**Arguments**

- **priormix**: prior (mixture of conjugate distributions).
- **data**: individual data. If the individual data is not given, then summary data has to be provided (see below).
- **...**: includes arguments which depend on the specific case, see description below.
- **n**: sample size.
Details

A conjugate prior-likelihood pair has the convenient property that the posterior is in the same distributional class as the prior. This property also applies to mixtures of conjugate priors. Let

$$p(\theta; w, a, b)$$

denote a conjugate mixture prior density for data

$$y|\theta \sim f(y|\theta),$$

where $f(y|\theta)$ is the likelihood. Then the posterior is again a mixture with each component $k$ equal to the respective posterior of the $k$th prior component and updated weights $w'_k$,

$$p(\theta; w', a', b'|y) = \sum_{k=1}^{K} w'_k p_k(\theta; a'_k, b'_k|y).$$

The weight $w'_k$ for $k$th component is determined by the marginal likelihood of the new data $y$ under the $k$th prior distribution which is given by the predictive distribution of the $k$th component,

$$w'_k \propto w_k \int p_k(\theta; a_k, b_k) f(y|\theta) \, d\theta \equiv w^*_k.$$

The final weight $w'_k$ is then given by appropriate normalization, $w'_k = w'_k / \sum_{k=1}^{K} w^*_k$. In other words, the weight of component $k$ is proportional to the likelihood that data $y$ is generated from the respective component, i.e. the marginal probability; for details, see for example Schmidli et al., 2015.

Note: The prior weights $w_k$ are fixed, but the posterior weights $w'_k \neq w_k$ still change due to the changing normalization.

The data $y$ can either be given as individual data or as summary data (sufficient statistics). See below for details for the implemented conjugate mixture prior densities.

Methods (by class)

- **betaMix**: Calculates the posterior beta mixture distribution. The individual data vector is expected to be a vector of 0 and 1, i.e. a series of Bernoulli experiments. Alternatively, the sufficient statistics $n$ and $r$ can be given, i.e. number of trials and successes, respectively.
- **normMix**: Calculates the posterior normal mixture distribution with the sampling likelihood being a normal with fixed standard deviation. Either an individual data vector data can be given or the sufficient statistics which are the standard error $se$ and sample mean $m$. If the sample size $n$ is used instead of the sample standard error, then the reference scale of the prior is used to calculate the standard error. Should standard error $se$ and sample size $n$ be given, then the reference scale of the prior is updated; however it is recommended to use the command `sigma` to set the reference standard deviation.
• `gammaMix`: Calculates the posterior gamma mixture distribution for Poisson and exponential likelihoods. Only the Poisson case is supported in this version.

## Supported Conjugate Prior-Likelihood Pairs

<table>
<thead>
<tr>
<th>Prior/Posterior</th>
<th>Likelihood</th>
<th>Predictive</th>
<th>Summaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beta</td>
<td>Binomial</td>
<td>Beta-Binomial</td>
<td>n, r</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal (fixed σ)</td>
<td>Normal</td>
<td>n, m, se</td>
</tr>
<tr>
<td>Gamma</td>
<td>Poisson</td>
<td>Gamma-Poisson</td>
<td>n, m</td>
</tr>
<tr>
<td>Gamma</td>
<td>Exponential</td>
<td>Gamma-Exp (not supported)</td>
<td>n, m</td>
</tr>
</tbody>
</table>

## References


## Examples

```r
# binary example with individual data (1=event,0=no event), uniform prior
prior.unif <- mixbeta(c(1,1,1))
data.indiv <- c(1,0,1,0,1)
posterior.indiv <- postmix(prior.unif, data.indiv)
print(posterior.indiv)

# or with summary data (number of events and number of patients)
r <- sum(data.indiv); n <- length(data.indiv)
posterior.sum <- postmix(prior.unif, n=n, r=r)
print(posterior.sum)

# binary example with robust informative prior and conflicting data
prior.rob <- mixbeta(c(0.5,4,10),c(0.5,1,1))
prior.rob <- postmix(prior.rob, n=20, r=18)
print(posterior.rob)

# normal example with individual data
sigma <- 88
prior.mean <- -49
prior.se <- sigma/sqrt(20)
prior <- mixnorm(c(1,prior.mean,prior.se),sigma=sigma)
data.indiv <- c(-46,-227,41,-65,-103,-227,-169,-69,90)
posterior.indiv <- postmix(prior, data.indiv)
# or with summary data (mean and number of patients)
mm <- mean(data.indiv); n <- length(data.indiv)
prior <- mixnorm(c(1,prior.mean,prior.se),sigma=sigma)
data.indiv <- c(-46,-227,41,-65,-103,-227,-169,-69,90)
posterior.indiv <- postmix(prior, m=mm, n=n)
print(posterior.sum)
```
Predictive Distributions for Mixture Distributions

Description

Predictive distribution for mixture of conjugate distributions (beta, normal, gamma).

Usage

preddist(mix, ...)

## S3 method for class 'betaMix'
preddist(mix, n = 1, ...)

## S3 method for class 'normMix'
preddist(mix, n = 1, sigma, ...)

## S3 method for class 'gammaMix'
preddist(mix, n = 1, ...)

Arguments

mix          mixture distribution
...
includes arguments which depend on the specific prior-likelihood pair, see description below.
n          predictive sample size, set by default to 1
sigma       The fixed reference scale of a normal mixture. If left unspecified, the default reference scale of the mixture is assumed.

Details

Given a mixture density (either a posterior or a prior)

\[ p(\theta, w, a, b) \]

and a data likelihood of

\[ y|\theta \sim f(y|\theta), \]

the predictive distribution of a one-dimensional summary \( y_n \) of \( n \) future observations is distributed as

\[ y_n \sim \int p(\theta, w, a, b) f(y_n|\theta) d\theta. \]

This distribution is the marginal distribution of the data under the mixture density. For binary and Poisson data \( y_n = \sum_{i=1}^{n} y_i \) is the sum over future events. For normal data, it is the mean \( \bar{y}_n = \frac{1}{n} \sum_{i=1}^{n} y_i \).
Value

The function returns for a normal, beta or gamma mixture the matching predictive distribution for $y_n$.

Methods (by class)

- **betaMix**: Obtain the matching predictive distribution for a beta distribution, the Beta-Binomial.
- **normMix**: Obtain the matching predictive distribution for a Normal with constant standard deviation. Note that the reference scale of the returned Normal mixture is meaningless as the individual components are updated appropriately.
- **gammaMix**: Obtain the matching predictive distribution for a Gamma. Only Poisson likelihoods are supported.

Supported Conjugate Prior-Likelihood Pairs

<table>
<thead>
<tr>
<th>Prior/Posterior</th>
<th>Likelihood</th>
<th>Predictive</th>
<th>Summaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beta</td>
<td>Binomial</td>
<td>Beta-Binomial</td>
<td>$n, r$</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal (fixed $\sigma$)</td>
<td>Normal</td>
<td>$n, m, se$</td>
</tr>
<tr>
<td>Gamma</td>
<td>Poisson</td>
<td>Gamma-Poisson</td>
<td>$n, m$</td>
</tr>
<tr>
<td>Gamma</td>
<td>Exponential</td>
<td>Gamma-Exp (not supported)</td>
<td>$n, m$</td>
</tr>
</tbody>
</table>

Examples

```r
# Example 1: predictive distribution from uniform prior.
bm <- mixbeta(c(1,1,1))
bmPred <- preddist(bm, n=10)
# predictive probabilities and cumulative predictive probabilities
x <- 0:10
d <- dmix(bmPred, x)
names(d) <- x
barplot(d)
cd <- pmix(bmPred, x)
names(cd) <- x
barplot(cd)
# median
mdn <- qmix(bmPred,0.5)
mdn
# Example 2: 2-comp Beta mixture
bm <- mixbeta( inf=c(0.8,15,50), rob=c(0.2,1,1))
plot(bm)
bmPred <- preddist(bm,n=10)
plot(bmPred)
mdn <- qmix(bmPred,0.5)
```
predict.gMAP

## S3 method for class `gMAP` predict(object, newdata, type = c("response", "link"), probs = c(0.025, 0.5, 0.975), na.action = na.pass, thin, ...)

## S3 method for class `gMAPpred` print(x, digits = 3, ...)

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

## S3 method for class `gMAPpred` as.matrix(x, ...)

### Arguments

- **newdata**: data.frame which must contain the same columns as input into the gMAP analysis. If left out, then a posterior prediction for the fitted data entries from the gMAP object is performed (shrinkage estimates).
- **type**: sets reported scale (`response` (default) or `link`).

### Description

Produces a sample of the predictive distribution.
probs defines quantiles to be reported.
na.action how to handle missings.
thin thinning applied is derived from the gMAP object.
... ignored.
x, object gMAP analysis object for which predictions are performed
digits number of displayed significant digits.

Details

Predictions are made using the $\tau$ prediction stratum of the gMAP object. For details on the syntax, please refer to `predict.glm` and the example below.

See Also

gMAP, predict.glm

Examples

# create a fake data set with a covariate
trans_cov <- transform(transplant, country=cut(1:11, c(0,5,8,Inf), c("CH", "US", "DE")))
set.seed(34246)
map <- gMAP(cbind(r, n-r) ~ 1 + country | study,
data=trans_cov,
tau.dist="HalfNormal",
tau.prior=1,
  # Note on priors: we make the overall intercept weakly-informative
  # and the regression coefficients must have tighter sd as these are
  # deviations in the default contrast parametrization
beta.prior=rbind(c(0,2), c(0,1), c(0,1)),
family=binomial,
  ## ensure fast example runtime
thin=1, chains=1)

# posterior predictive distribution for each input data item (shrinkage estimates)
pred_cov <- predict(map)
pred_cov

# extract sample as matrix
samp <- as.matrix(pred_cov)

# predictive distribution for each input data item (if the input studies were new ones)
pred_cov_pred <- predict(map, trans_cov)
pred_cov_pred

# a summary function returns the results as matrix
summary(pred_cov)

# obtain a prediction for new data with specific covariates
pred_new <- predict(map, data.frame(country="CH", study=12))
robustify

Description
Add a non-informative component to a mixture prior.

Usage
robustify(priormix, weight, mean, n = 1, ...)

## S3 method for class 'betaMix'
robustify(priormix, weight, mean, n = 1, ...)

## S3 method for class 'gammaMix'
robustify(priormix, weight, mean, n = 1, ...)

## S3 method for class 'normMix'
robustify(priormix, weight, mean, n = 1, ..., sigma)

Arguments
- priormix: prior (mixture of conjugate distributions).
- weight: weight given to the non-informative component (0 < weight < 1).
- mean: mean of the non-informative component. It is recommended to set this parameter explicitly.
- n: number of observations the non-informative prior corresponds to, defaults to 1.
- ... optional arguments are ignored.
- sigma: Sampling standard deviation for the case of Normal mixtures.

Details
It is recommended to robustify informative priors derived with gMAP using unit-information priors. This protects against prior-data conflict, see for example Schmidli et al., 2015.

The procedure can be used with beta, gamma and normal mixture priors. A unit-information prior (see Kass and Wasserman, 1995) corresponds to a prior which represents the observation of n=1 at the null hypothesis. As the null is problem dependent we strongly recommend to make use of the mean argument accordingly. See below for the definition of the default mean.

The weights of the mixture priors are rescaled to (1-weight) while the non-informative prior is assigned the weight given.
Value

New mixture with an extra non-informative component named robust.

Methods (by class)

- `betaMix`: The default mean is set to 1/2 which represents no difference between the occurrence rates for one of the two outcomes. As the uniform Beta(1,1) is more appropriate in practical applications, RBesT uses n+1 as the sample size such that the default robust prior is the uniform instead of the Beta(1/2,1/2) which strictly defined would be the unit information prior in this case.

- `gammaMix`: The default mean is set to the mean of the prior mixture. It is strongly recommended to explicitly set the mean to the location of the null hypothesis.

- `normMix`: The default mean is set to the mean of the prior mixture. It is strongly recommended to explicitly set the mean to the location of the null hypothesis, which is very often equal to 0. It is also recommended to explicitly set the sampling standard deviation using the sigma argument.

References


See Also

`mixcombine`

Examples

```r
bmix <- mixbeta(inf1=c(0.2, 8, 3), inf2=c(0.8, 10, 2))
plot(bmix)
rbmix <- robustify(bmix, weight=0.1, mean=0.5)
rbmix
plot(rbmix)

gammix <- mixgamma(inf1=c(0.2, 2, 3), inf2=c(0.8, 2, 5), param="mn")
plot(gammix)
r gammix <- robustify(gammix, weight=0.1, mean=2)
r gammix
plot(r gammix)

nm <- mixnorm(inf1=c(0.2, 0.5, 0.7), inf2=c(0.8, 2, 1), sigma=2)
plot(nm)
rnmix <- robustify(nm, weight=0.1, mean=0, sigma=2)
rnmix
plot(rnmix)
```
Description

Data set containing historical information for standard treatment for a phase IV trial in de novo transplant patients. The primary outcome is treatment failure (binary).

Usage

transplant

Format

A data frame with 4 rows and 3 variables:

- study  study
- n       study size
- r       number of events

References

## Index

*Topic **datasets**
- AS, 4
- colitis, 7
- crohn, 7
- transplant, 67
- mix, 31
- mixbeta, 32–34
- mixcombine, 33, 34, 35, 41, 43
- mixdiff, 36
- mixfit, 5, 38
- mixgamma, 32–34
- mixnorm, 32–34
- m2beta, 33
- m2gamma, 40
- m2norm, 42
- model.matrix.default, 23
- ms2beta, 33
- ms2gamma, 40
- oc1S, 9, 12
- oc1Sdecision, 8
- oc2S, 14, 17, 47
- oc2Sdecision, 13
- pbinom, 12
- plot.EM, 40
- plot.gMAP, 28
- plot.mix, 33
- preddist, 32
- predict.glm, 64
- predict.gMAP, 28
- print.gMAP, 22
- print.gMAPpred, 63
- qmix, 31
- qmixdiff, 36
- qplot, 52

- lodds, 30
- logit, 30
- mix, 31, 34, 41, 43
- mixbeta, 32, 33
- mixcombine, 33
- mixdiff, 36
- mixfit, 5
- mixgamma, 32–34
- mixnorm, 32–34
- m2beta, 33
- m2gamma, 40
- m2norm, 42
- model.matrix.default, 23
- ms2beta, 33
- ms2gamma, 40
- oc1S, 9, 12
- oc1Sdecision, 8
- oc2S, 14, 17, 47
- oc2Sdecision, 13
- pbinom, 12
- plot.EM, 40
- plot.gMAP, 28
- plot.mix, 33
- preddist, 32
- predict.glm, 64
- predict.gMAP, 28
- print.gMAP, 22
- print.gMAPpred, 63
- qmix, 31
- qmixdiff, 36
- qplot, 52
INDEX

RBesT, 27
RBesT (RBesT-package), 3
RBesT-package, 3
rmix (mix), 31
rmixdiff (mixdiff), 36
robustify, 36, 65
rstan, 27

set.seed, 27
sigma, 59
sigma (mixnorm), 42
sigma <- (mixnorm), 42
stan, 24
summary.betaBinomialMix (mixbeta), 33
summary.betaMix (mixbeta), 33
summary.gammaMix (mixgamma), 40
summary.gammaPoissonMix (mixgamma), 40
summary.gMAP (gMAP), 22
summary.gMAPpred (predict.gMAP), 63
summary.normMix (mixnorm), 42

transplant, 67