Package 'BayesianHybridDesign'

February 6, 2026

Title Bayesian Hybrid Design and Analysis

Version 0.1.0

Description Implements Bayesian hybrid designs that incorporate historical control data into a current clinical trial. The package uses a dynamic power prior method to determine the degree of borrowing from the historical data, creating a 'hybrid' control arm. This approach is primarily designed for studies with a binary primary endpoint, such as the overall response rate (ORR). Functions are provided for design calibration, sample size calculation, power evaluation, and final analysis. Additionally, it includes functions adapted from the 'SAMprior' package (v1.1.1) by Yang et al. (2023) <https://academic.oup.com/biometrics/article/79/4/2857/7587575> to support the Self-Adapting Mixture (SAM) prior framework for comparison.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.1

Depends R (>= 4.1)

Imports assertthat, checkmate, doParallel, foreach, ggplot2, Metrics, parallel, RBesT

Suggests broom, knitr, purrr, rmarkdown, rstanarm, scales, testthat (>= 3.0.0), tidyr, tools

NeedsCompilation no

Author Philip He [aut, cph], Zhaohua Lu [aut, cre, cph]

Maintainer Zhaohua Lu <zhaohua.lu@gmail.com>

Repository CRAN

Date/Publication 2026-02-06 20:00:24 UTC

Contents

Bayesian.Hybrid.Analysis 2

borrow.wt 4
calibration 5
DPP 7
DPP.analysis 9
EESS 11
exactci 12
explore.power.DPP 13
fisher 15
fisher.bound 16
fisher.power 17
Frequentist.Analysis 18
plotDPP 19
plotPMD 19
power.binom.test 20
power.DPP 21
runCalibratedSAM 23
runSAM 26
SAM_prior 30
SAM_weight 33
Two.Prop.Test.Sample.Size 37

Index **38**

`Bayesian.Hybrid.Analysis`
Statistical Analysis for a Bayesian Hybrid Design

Description

This function performs the statistical analysis for a Bayesian Hybrid Design using a dynamic power prior approach.

Usage

```
Bayesian.Hybrid.Analysis(
  Yt = 20,
  nt = 40,
  Yc = 12,
  nc = 40,
  Ych = 73,
  nche = 40,
  nch = 234,
  sig = 0.9,
  credlev = 0.8,
  a0c = 0.001,
  b0c = 0.001,
  a0t = 0.001,
  b0t = 0.001,
```

```
  delta_threshold = 0.1
)
```

Arguments

Yt	Number of responses in the experimental arm in the current study.
nt	Number of patients in the experimental arm in the current study.
Yc	Number of responses in the control arm in the current study.
nc	Number of patients in the control arm in the current study.
Ych	Number of responses in the control treatment in the historical study.
nche	Equivalent number of patients borrowed from the historical study.
nch	Total number of patients in the historical control.
sig	Significance boundary. The hypothesis is considered significant if the posterior probability $P(p_t > p_c \| data) > sig$.
credlev	Credible interval level (e.g., 0.95 for 95 percent CI).
a0c	Prior alpha parameter for control response rate, $Beta(a_{0c}, b_{0c})$.
b0c	Prior beta parameter for control response rate, $Beta(a_{0c}, b_{0c})$.
a0t	Prior alpha parameter for experimental response rate, $Beta(a_{0t}, b_{0t})$.
b0t	Prior beta parameter for experimental response rate, $Beta(a_{0t}, b_{0t})$.
delta_threshold	Borrowing threshold. Borrowing occurs when $\|p_{c,trial} - p_{c,hist}\| \leq$ delta_threshold.

Value

A list containing the following components:

- prob.pt.gt.pc: Probability of experimental arm having a better posterior response rate than control.
- median_hca: Posterior median response rate for hybrid control.
- CI_hca: Credible interval for median response rate for hybrid control.
- median_c: Posterior median response rate for current study control.
- CI_c: Credible interval for median response rate for current study control.
- median_t: Posterior median response rate for current study experimental arm.
- CI_t: Credible interval for median response rate for current experimental arm.
- delta.m: Posterior median response rate difference (experimental - control) based on hybrid design.
- delta.CI: Credible interval for response rate difference based on hybrid design.
- delta.m_trial: Posterior median response rate difference (experimental - control) based on current study only.
- delta.CI_trial: Credible interval for response rate difference based on current study only.
- conclusion: Statistical inference conclusion string.

Examples

```
# Note: This example relies on the internal package function 'borrow.wt'
Bayesian.Hybrid.Analysis(Yt=18, nt=40, Yc=13, nc=40, Ych=73,
                         nche=40, nch=234)
```

`borrow.wt` *Calculate Borrowing Weights from Historical Data*

Description

Calculates borrowing weights for a hybrid control arm using one of several dynamic borrowing methods.

Usage

```
borrow.wt(
  Yc,
  nc,
  Ych,
  nch,
  nche,
  a0c = 0.001,
  b0c = 0.001,
  delta_threshold = 0.1,
  method = "Empirical Bayes",
  theta = 0.5,
  eta = 1
)
```

Arguments

`Yc`	A scalar integer. The number of responders in the current control arm.
`nc`	A scalar integer. The number of subjects in the current control arm.
`Ych`	A scalar integer. The number of responders in the historical control arm.
`nch`	A scalar integer. The number of subjects in the historical control arm.
`nche`	A scalar. The maximum number of subjects that can be borrowed, used to calculate the global weight `a`.
`a0c`, `b0c`	Numerics. Hyperparameters for the Beta(a0c, b0c) prior on the control response rate. Default to 0.001.
`delta_threshold`	A numeric threshold. Borrowing is only triggered if the absolute difference in observed response rates is less than this value. Default is 0.1.
`method`	A string specifying the dynamic borrowing method. Options include "Empirical Bayes", "Bayesian p", "Generalized BC", "JSD".

theta	A numeric scalar in (0, 1), applicable to the "Generalized BC" method. Default is 0.5.
eta	A numeric scalar, applicable to the "Bayesian p", "Generalized BC", and "JSD" methods. Default is 1.

Details

The function implements the following methods:

- "Empirical Bayes": The weight is determined by maximizing the marginal likelihood of the current data.
- "Bayesian p": Similarity is measured by a Bayesian p-value comparing the posterior distributions.
- "Generalized BC": Uses the Generalized Bhattacharyya Coefficient. The standard BC is a special case with theta = 0.5.
- "JSD": Uses the Jensen-Shannon Divergence to measure similarity.

Value

A list containing three values:

- a: The global borrowing weight, calculated as nche / nch.
- wd: The dynamic borrowing weight, calculated based on the chosen method.
- w: The final overall borrowing weight, which is a product of `a`, `wd`, and an indicator for whether the response rates are sufficiently similar.

Examples

```
borrow.wt(Yc=12, nc=40, Ych=54, nch=180, nche=40, a0c=0.001,
b0c=0.001, delta_threshold=0.1, eta=1)
```

---

calibration *Calibration for Bayesian Hybrid Design*

Description

This function calculates the threshold tau for calibration of Bayesian Hybrid Design. P(pt>pc|hybrid data)>tau is used to determine statistical significance.

Usage

```
calibration(
  nt,
  pc.calib,
  nc,
  pch,
  nche,
  nch,
  alpha = 0.1,
  a0c = 0.001,
  b0c = 0.001,
  a0t = 0.001,
  b0t = 0.001,
  delta_threshold = 0.1,
  method = "Empirical Bayes",
  theta = 0.5,
  eta = 1,
  datamat = NULL,
  w0 = NULL,
  nsim = 10000,
  seed = NULL
)
```

Arguments

nt	A scalar number of patients in experimental arm in current study
pc.calib	A scalar. Response rate for control arm in current study for calibration. Usually, pc.calib = pch.
nc	A scalar number of patients in control arm in current study
pch	A scalar. Response rate for control treatment in historical study
nche	A scalar representing the equivalent number of patients borrowed from historical study
nch	A scalar for total number of patients in historical control
alpha	A scalar. One sided type I error rate.
a0c	A scalar. Hyperprior for control response rate beta(a0c, b0c)
b0c	A scalar. Hyperprior for control response rate beta(a0c, b0c)
a0t	A scalar. Hyperprior for experimental response rate beta(a0t, b0t)
b0t	A scalar. Hyperprior for experimental response rate beta(a0t, b0t)
delta_threshold	A scale threshold parameter. Only if abs(pc (current study) - pch) <= delta_threshold, we borrow from historical control. Default 0.1.
method	A string characters. Method for dynamic borrowing, "Empirical Bayes", "Bayesian p", "Generalized BC", "JSD"
theta	A scalar parameter with a range of (0, 1), and applicable to "Generalized BC". Default 0.5.

eta	A scalar parameter with a range of (0, infty), and applicable to methods "Bayesian p", "Generalized BC", "JSD". Default 1.
datamat	A matrix with dimension nsim * 2 containing pre-simulated data for the experimental arm (column 1) and control arm (column 2), respectively. Default is NULL, and binomial random Monte Carlo samples will be generated in the function.
w0	A scale prior power parameters w. If not specified (default), w_d is calculated by the specified method for dynamic borrowing.
nsim	A scalar. Number of replications to calculate power
seed	A scalar. seed for simulations

Value

The scalar threshold for statistical significance that can control the type I error (1-sided)

Examples

```
tau <- calibration(nt=40, pc.calib=0.3, nc=40,
                   pch=0.3, nche=40, nch=200,
                   alpha = 0.10,
                   a0c=0.001, b0c=0.001, a0t=0.001, b0t=0.001,
                   delta_threshold=0.1,
                   method="Empirical Bayes", theta=0.5, eta=1,
                   nsim = 1000, seed=2000) # nsim reduced for quick example
```

DPP *Bayesian Hybrid Design using Dynamic Power Prior Framework*

Description

This function calculates the sample size and Bayesian hybrid design parameters using dynamic power prior framework.

Usage

```
DPP(
  pt,
  pc,
  pch,
  pc.calib,
  nch,
  nc.range = NULL,
  r,
  q,
  alpha = 0.1,
```

```
  power = 0.8,
  delta_threshold = 0.1,
  method = "Empirical Bayes",
  theta = 0.5,
  eta = 1,
  a0c = 0.001,
  b0c = 0.001,
  a0t = 0.001,
  b0t = 0.001,
  nsim = 1e+05,
  seed = NULL
)
```

Arguments

pt	A scalar. Response rate for experimental arm in current study.
pc	A scalar. Response rate for control arm in current study.
pch	A scalar. Response rate for control treatment in historical study.
pc.calib	A scalar. Response rate for control arm in current study for calibration. Usually, pc.calib = pch.
nch	A scalar. Total number of patients in historical control.
nc.range	A vector with length = 2. Search range for nc. Default is NULL, and the range will be automatically determined.
r	A scalar. Randomization ratio for current study. r=1 means 1:1 and r=2 means 2:1.
q	A scalar. Specification of n_che in terms of times of nc; i.e. n_che = q*nc. Usually, q >= 1 and q <= nch/n_che.
alpha	A scalar. One sided type I error rate.
power	A scalar. Power. Default 0.8.
delta_threshold	Borrow when abs(pc_hat (current study) - pch) <= delta_threshold. Default 0.1.
method	A string characters. Method for dynamic borrowing, "Empirical Bayes", "Bayesian p", "Generalized BC", "JSD". Default "Empirical Bayes".
theta	A scalar parameter with a range of (0, 1), and applicable to method: "Generalized BC". Default 0.5.
eta	A scalar parameter with a range of (0, infty), and applicable to method: "Bayesian p", "Generalized BC", "JSD". "Generalized BC" method requires two parameters theta and eta. Default 1.
a0c	A scalar. Hyperprior for control response rate beta(a0c, b0c). Default 0.001.
b0c	A scalar. Hyperprior for control response rate beta(a0c, b0c). Default 0.001.
a0t	A scalar. Hyperprior for experimental response rate beta(a0t, b0t). Default 0.001.
b0t	A scalar. Hyperprior for experimental response rate beta(a0t, b0t). Default 0.001.
nsim	A scalar. Number of replications to calculate power. Default 100,000.
seed	A scalar. seed for simulations. Default NULL.

Value

An object with values

- alpha: nominal type I error rate
- power: The calculated power by simulation.
- type1err: empirical type I error rate.
- tau: The calibrated threshold for statistical significance.
- nt: sample size for experimental arm
- nc: sample size for control arm
- nche: maximum amount of borrowing in terms of number of subjects
- delta.bound: significance boundary of delta between the study experimental group and study control group
- mean.PMD: mean of posterior mean difference over nsim estimates
- sd.PMD: standard deviation of posterior mean difference over nsim estimates
- mean_pc_hca: a vector of nsim length. Storing the posterior means of pc based on hybrid control for nsim replications.
- mean_pc_c: a vector of nsim length. Storing the posterior means of pc based on study control for nsim replications.

Examples

```
o <- DPP(pt = 0.5, pc = 0.3, pch = 0.3, pc.calib = 0.3, nch = 200, nc.range = NULL,
         r = 2, q = 1, alpha = 0.1, power = 0.8,
         delta_threshold = 0.1,
         method = "Empirical Bayes", theta = 0.5, eta = 1,
         a0c = 0.001, b0c = 0.001, a0t = 0.001, b0t = 0.001,
         nsim = 1000, seed = 2000)
print(o)
```

DPP.analysis *Analysis of a Study Using Bayesian Hybrid Design using Dynamic Power Prior Framework*

Description

This function Perform Analysis for a Study Using Bayesian Hybrid Design using Dynamic Power Prior Framework.

Usage

```
DPP.analysis(
  Yt = 39,
  nt = 60,
  Yc = 13,
  nc = 30,
  Ych = 90,
  nch = 200,
  nche = 30,
  a0c = 0.001,
  b0c = 0.001,
  a0t = 0.001,
  b0t = 0.001,
  delta_threshold = 0.1,
  method = "Empirical Bayes",
  theta = 0.5,
  eta = 1
)
```

Arguments

Yt	A scalar. Response rate for experimental arm in current study.
nt	A scalar. sample size for experimental arm.
Yc	A scalar. Response rate for control arm in current study.
nc	A scalar. sample size for control arm.
Ych	A scalar. Number of responders in historical control.
nch	A scalar. Total number of subjects in historical control.
nche	A scalar. maximum amount of borrowing in terms of equivalent number of subjects.
a0c	A scalar. Hyperprior for control response rate beta(a0c, b0c). Default 0.001.
b0c	A scalar. Hyperprior for control response rate beta(a0c, b0c). Default 0.001.
a0t	A scalar. Hyperprior for experimental response rate beta(a0t, b0t). Default 0.001.
b0t	A scalar. Hyperprior for experimental response rate beta(a0t, b0t). Default 0.001.
delta_threshold	Borrow when abs(pc_hat (current study) - pch) <= delta_threshold#'. Default 0.1.
method	A string characters. Method for dynamic borrowing, "Empirical Bayes", "Bayesian p", "Generalized BC", "JSD". Default "Empirical Bayes".
theta	A scalar parameter with a range of (0, 1), and applicable to method: "Generalized BC". Default 0.5.
eta	A scalar parameter with a range of (0, infty), and applicable to method: "Bayesian p", "Generalized BC", "JSD". "Generalized BC" method requires two parameters theta and eta. Default 1.

Value

An object of class `list` with values:

- w: Borrowing weight.
- phat_pt_larger_pc: Posterior probability P(ORR_trt > ORR_ctrl | data).
- apost_c_trial, bpost_c_trial: Parameters for the posterior Beta distribution of the concurrent control arm response rate.
- apost_c_hca, bpost_c_hca: Parameters for the posterior Beta distribution of the hybrid control arm response rate.
- apost_t, bpost_t: Parameters for the posterior Beta distribution of the experimental arm response rate.
- m.t: Posterior median response rate for the experimental arm.
- m.c: Posterior median response rate for the concurrent control arm.
- m.hca: Posterior median response rate for the hybrid control arm.

Examples

```
o <- DPP.analysis(Yt=39, nt=60, Yc=13, nc=30, Ych=90, nch=200, nche = 30,
a0c= 0.001, b0c= 0.001, a0t= 0.001, b0t= 0.001,
delta_threshold = 0.1, method = "Empirical Bayes",
theta = 0.5, eta = 1)
print(o)
```

EESS *Bayesian Hybrid Design*

Description

This function calculates the expected effective sample size of the DPP.

Usage

```
EESS(
  pc,
  nc,
  pch,
  nche,
  nch,
  a0c = 0.001,
  b0c = 0.001,
  delta_threshold = 0.1,
  method = "Empirical Bayes",
  theta = 0.5,
  eta = 1
)
```

Arguments

pc	A scalar. Response rate for control arm in current study.
nc	A scalar. Number of patients in control arm in current study.
pch	A scalar. Response rate for control treatment in historical study.
nche	A scalar. Equivalent number of patients borrowed from historical study.
nch	A scalar. Total number of patients in historical control.
a0c	A scalar. Hyperprior for control response rate beta(a0c, b0c).
b0c	A scalar. Hyperprior for control response rate beta(a0c, b0c).
delta_threshold	A scalar. Borrow when abs(pc_hat (current study) - pch) <= delta_threshold.
method	A string characters. Method for dynamic borrowing, "Empirical Bayes", "Bayesian p", "Generalized BC", "JSD". Default "Empirical Bayes".
theta	A scalar parameter with a range of (0, 1), and applicable to method: "Generalized BC". Default 0.5.
eta	A scalar parameter with a range of (0, infty), and applicable to method: "Bayesian p", "Generalized BC", "JSD". "Generalized BC" method requires two parameters theta and eta. Default 1.

Value

The expected effective sample size.

Examples

```
EESS(pc=0.3,nc=40,pch=0.3,nche=40,nch=180, a0c=0.001,b0c=0.001,
delta_threshold=0.1,method="Empirical Bayes", theta=0.5, eta=1)
```

---

exactci *Clopper-Pearson Exact Confidence Interval for a Binomial Proportion*

Description

Calculates the two-sided Clopper-Pearson exact confidence interval for a binomial proportion.

Usage

```
exactci(r, n, conflev)
```

Arguments

r	A scalar integer. The number of successes or responses.
n	A scalar integer. The total number of trials or subjects.
conflev	A scalar numeric. The desired confidence level (e.g., 0.95 for a 95% CI).

Value

A numeric vector of length two containing the lower and upper confidence limits.

Examples

```
exactci(r=4, n=20, conflev=0.95)
```

`explore.power.DPP` *Explore Power Across Multiple Scenarios for a Bayesian Hybrid Design*

Description

Evaluates statistical power and other design parameters across a grid of specified settings, using parallel computing for efficiency.

Usage

```
explore.power.DPP(
  method,
  pc,
  nc,
  pc.calib,
  q,
  delta,
  r,
  pch,
  nch,
  alpha = 0.1,
  a0c = 0.001,
  b0c = 0.001,
  a0t = 0.001,
  b0t = 0.001,
  delta_threshold = 0.1,
  theta = 0.5,
  eta = 1,
  nsim = 1e+05,
  seed = NULL,
  ncore = NULL
)
```

Arguments

`method`	A character vector. One or more dynamic borrowing methods to explore (e.g., `"Empirical Bayes"`, `"Bayesian p"`).
`pc`	A numeric vector. Response rates for the current control arm to explore.

nc	A numeric vector of integers. Sample sizes for the current control arm to explore.
pc.calib	A numeric vector. Control arm response rates used for calibrating the type I error threshold, tau.
q	A numeric vector. Ratios of nche / nc to explore.
delta	A scalar numeric. The true difference to be added to pc to determine the experimental arm response rate (pt).
r	A scalar numeric. The randomization ratio of the experimental arm to the control arm (nt / nc).
pch	A scalar numeric. The response rate of the historical control arm.
nch	A scalar integer. The total number of subjects in the historical control arm.
alpha	A scalar numeric. The one-sided Type I error rate.
a0c, b0c	Numerics. Hyperparameters for the Beta(a0c, b0c) prior on the control response rate. Defaults to 0.001.
a0t, b0t	Numerics. Hyperparameters for the Beta(a0t, b0t) prior on the experimental response rate. Defaults to 0.001.
delta_threshold	A scalar numeric. The similarity threshold; borrowing is only triggered if abs(pc_hat - pch_hat) <= delta_threshold. Default is 0.1.
theta	A scalar numeric in (0, 1), applicable to the "Generalized BC" method. Default is 0.5.
eta	A scalar numeric, applicable to the "Bayesian p", "Generalized BC", and "JSD" methods. Default is 1.
nsim	A scalar integer. The number of Monte Carlo simulations for each scenario.
seed	A scalar integer. A seed for the random number generator to ensure reproducibility. Default is NULL.
ncore	An integer. The number of CPU cores for parallel processing. If NULL (the default), it uses one less than the number of detected cores.

Details

This function serves as a wrapper for power.DPP(). It uses expand.grid() to create a full factorial design from the vector-based input parameters (method, pc, nc, pc.calib, q). It then iterates through each scenario to calculate the power, type I error, and other metrics.

Value

A data.frame where each row corresponds to one scenario from the input grid. The columns include the input parameters and the following results:

method The dynamic borrowing method used.

nt, nc, nche Sample sizes for the experimental, current control, and effective historical control arms.

pt, pc, pc.calib, pch Response rates used in the scenario.

q The ratio of nche / nc.

tau The calibrated threshold for statistical significance.

type1err The simulated Type I error rate for the scenario.

power The simulated statistical power for the scenario.

delta.boundary The significance boundary for the difference between groups.

mean.PMD The mean of the posterior mean difference over simulations.

sd.PMD The standard deviation of the posterior mean difference.

See Also

power.DPP for the underlying single-scenario calculation.

Examples

```
# Run with 2 cores as an example; on CRAN, examples should use <= 2 cores.
Res1 <- explore.power.DPP(method=c("Empirical Bayes"),
                          pc=c(0.27, 0.37),
                          nc=23:25,
                          pc.calib = 0.27,
                          q = c(1, 1.5),
                          delta = 0.2,
                          r = 1,
                          pch=0.27, nch=500,
                          alpha=0.1,
                          nsim = 200, # Reduced for a quick example
                          seed=1000,
                          ncore=2)
```

---

fisher *Fisher's Exact Test for a 2x2 Contingency Table*

Description

A wrapper for stats::fisher.test to conveniently compare response rates between two arms (experimental and control).

Usage

```
fisher(Yc, nc, Yt, nt, alternative = "greater")
```

Arguments

Yc	A scalar integer. The number of subjects with a response in the control arm.
nc	A scalar integer. The total number of subjects in the control arm.
Yt	A scalar integer. The number of subjects with a response in the experimental (treatment) arm.
nt	A scalar integer. The total number of subjects in the experimental arm.
alternative	A character string specifying the alternative hypothesis. Must be one of "two.sided" (default), "greater" or "less".

Value

An object of class `htest` as returned by `stats::fisher.test`.

See Also

`fisher.test`

Examples

```
fisher(Yc=12, nc=40, Yt=19, nt=40)
```

`fisher.bound` *Calculate Rejection Boundary for Fisher's Exact Test*

Description

For a given control group response rate and sample sizes, this function finds the smallest number of responders in the experimental arm (`rt`) that achieves statistical significance based on a one-sided Fisher's exact test.

Usage

```
fisher.bound(pc, nc, nt, alpha = 0.1)
```

Arguments

pc	A scalar numeric. The response rate for the control arm.
nc	A scalar integer. The number of subjects in the control arm.
nt	A scalar integer. The number of subjects in the experimental arm.
alpha	A scalar numeric. The one-sided p-value threshold for statistical significance. Default is 0.1.

Value

A list containing the following components:

M The 2x2 contingency table at the boundary.

p The p-value corresponding to the boundary.

rc The number of responders in the control arm, calculated as `round(nc * pc)`.

nc The sample size of the control arm.

rt The smallest number of responders in the experimental arm that achieves a p-value <= `alpha`.

nt The sample size of the experimental arm.

delta The minimum detectable difference in response rates (`rt/nt - rc/nc`).

Examples

```
fisher.bound(pc=0.3, nc=40, nt=40, alpha=0.1)
```

`fisher.power` *Power Calculation by Fisher's Exact Test*

Description

This function calculates the power by simulations using Fisher's exact test.

Usage

```
fisher.power(pt, nt, pc, nc, alpha = 0.1, nsim = 10000, seed = NULL)
```

Arguments

`pt`	A scalar. Probablity of success in experimental arm
`nt`	A scalar. Number of subject in experimental arm
`pc`	A scalar. Probablity of success in control arm
`nc`	A scalar. Number of subject in control arm
`alpha`	A scalar. One sided type I error rate.
`nsim`	A scalar. Number of replications to calculate power. Default 100,000.
`seed`	An integer. A seed for the random number generator to ensure reproducibility. Note: For best practice, it is recommended to call `set.seed()` outside of the function.

Value

A numeric scalar representing the calculated statistical power.

Examples

```
# nsim is reduced for a quick example run
fisher.power(pt = 0.5, nt = 40, pc = 0.3, nc = 40, nsim = 1000, seed = 2000)
```

`Frequentist.Analysis` *Statistical Analysis Using Frequentist Method*

Description

Performs a frequentist analysis of a two-arm study using current data only.

Usage

```
Frequentist.Analysis(Yt = 20, nt = 40, Yc = 12, nc = 40, conflev = 0.8)
```

Arguments

`Yt`	A scalar. The number of responses in the experimental arm.
`nt`	A scalar. The number of subjects in the experimental arm.
`Yc`	A scalar. The number of responses in the control arm.
`nc`	A scalar. The number of subjects in the control arm.
`conflev`	A scalar. The desired confidence level for the confidence intervals.

Value

An object of class `list` with the following values:

- pt: The response rate of the experimental arm.
- pc: The response rate of the control arm.
- delta: The difference in response rates (pt - pc).
- exactCI.c: A vector containing the Clopper-Pearson confidence interval for the control arm.
- exactCI.t: A vector containing the Clopper-Pearson confidence interval for the experimental arm.
- p.fisher: A one-sided p-value from Fisher's exact test, testing the alternative hypothesis that the experimental response rate is greater than the control response rate.

Examples

```
Frequentist.Analysis(Yt=18, nt=40, Yc=13, nc=40, conflev=0.8)
```

plotDPP *Plot a DPP Object*

Description

Plots the posterior distributions of the response rates for the control and experimental arms from a DPP object created by the DPP.analysis() function.

Usage

```
plotDPP(DPP)
```

Arguments

DPP	An object produced by the DPP.analysis() function.

Value

No return value, called for side effects.

Examples

```
o <- DPP.analysis(Yt=39, nt=60, Yc=13, nc=30, Ych=90, nch=200, nche = 30,
                  a0c= 0.001, b0c= 0.001, a0t= 0.001, b0t= 0.001,
                  delta_threshold = 0.1, method = "Empirical Bayes",
                  theta = 0.5, eta = 1)

# Call the function using its defined name 'plotDPP'
plotDPP(DPP = o)
```

plotPMD *Plot Posterior Mean Difference (PMD) Distributions*

Description

Plots the density of the posterior mean difference between the hybrid and concurrent control arms, based on an object from power.DPP(). Can compare one or two such objects on the same plot.

Usage

```
plotPMD(o, o2 = NULL)
```

Arguments

o	An object produced by the power.DPP() function.
o2	(Optional) A second object from power.DPP() to compare on the same plot. Default is NULL.

Value

Invisibly returns a list containing summary statistics. The structure of the list depends on whether o2 is provided.

- If o2 is NULL: A list with a scalar mean_PMD, a scalar sd_PMD, and a numeric vector CI95_PMD.
- If o2 is provided: A list with numeric vectors for mean_PMD and sd_PMD, and a matrix for CI95_PMD, with each row corresponding to an input object.

Examples

```
# nsim reduced for faster example
o1a <- power.DPP(pt=0.65, nt=60, pc=0.45, nc=30, pc.calib=0.45,
                 pch=0.45, nche=60, nch=200, alpha=0.1, nsim=1000)
o1b <- power.DPP(pt=0.65, nt=60, pc=0.45, nc=30, pc.calib=0.45,
                 pch=0.45, nche=30, nch=200, alpha=0.1, nsim=1000)

# Plot a single object
stats1 <- plotPMD(o=o1a)

# Plot two objects for comparison
stats2 <- plotPMD(o=o1a, o2=o1b)
```

---

power.binom.test *Power Calculation for the Exact Binomial Test*

Description

Calculates the power of a one-sample exact binomial test via simulation. This is used to compare a treatment response rate against a fixed historical benchmark.

Usage

```
power.binom.test(
  n = 20,
  p = 0.5,
  p0 = 0.3,
  alpha = 0.05,
  alternative = c("two.sided", "greater", "less"),
  nsim = 1e+05,
  seed = 2025
)
```

Arguments

n	Sample size of the experimental group.
p	Assumed true proportion (response rate) of the experimental treatment.

p0	Historical proportion (null hypothesis) used as a benchmark.
alpha	Type I error rate (significance level).
alternative	Character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less".
nsim	Number of simulated trials; defaults to 100,000 for high precision.
seed	Seed for reproducibility.

Value

A numeric value representing the statistical power.

Examples

```
power.binom.test(n=110, p=0.4, p0=0.128)
```

`power.DPP` *Power Calculation for Bayesian Hybrid Design*

Description

Calculates statistical power and other design parameters for a Bayesian Hybrid Design using a dynamic power prior approach, based on simulations.

Usage

```
power.DPP(
  pt,
  nt,
  pc,
  nc,
  pc.calib,
  pch,
  nche,
  nch,
  alpha = 0.1,
  tau = NULL,
  a0c = 0.001,
  b0c = 0.001,
  a0t = 0.001,
  b0t = 0.001,
  delta_threshold = 0.1,
  method = "Empirical Bayes",
  theta = 0.5,
  eta = 1,
```

```
  datamat = NULL,
  w0 = NULL,
  nsim = 1e+05,
  seed = NULL
)
```

Arguments

pt, pc	Numerics. The response rates for the experimental and control arms, respectively.
nt, nc	Integers. The sample sizes for the experimental and control arms, respectively.
pc.calib	A scalar numeric. The control response rate assumed for calibrating the type I error threshold, tau. Often pc.calib = pch.
pch, nch	Numeric and integer. The response rate and sample size of the historical control arm.
nche	An integer. The effective number of subjects to be borrowed, used for calculating the global borrowing weight.
alpha	A scalar numeric. The one-sided Type I error rate, used for calibration if tau is not provided.
tau	(Optional) A scalar numeric. The pre-calibrated threshold for statistical significance. If NULL, it will be calculated internally.
a0c, b0c, a0t, b0t	Numerics. Hyperparameters for the Beta priors on the response rates.
delta_threshold	A scalar numeric. The similarity threshold for borrowing.
method	A string specifying the dynamic borrowing method.
theta, eta	Numerics. Additional parameters for certain borrowing methods.
datamat	(Optional) A matrix with nsim rows and 2 columns (experimental, control) of pre-simulated response counts.
w0	(Optional) A scalar numeric. A fixed borrowing weight to override the dynamic calculation.
nsim	An integer. The number of simulations to run.
seed	An integer. A seed for the random number generator. Default NULL.

Value

A large list containing the power, the calibrated tau, all input parameters, and detailed simulation results such as:

power The calculated statistical power.

tau The calibrated significance threshold.

pc.PMD, pc.sd.PMD The mean and standard deviation of the posterior mean difference between the hybrid and concurrent controls.

delta.bound The minimum detectable difference in response rates.

phat_pt_larger_pc_all A vector of posterior probabilities `P(pt > pc | data)` for each of the `nsim` simulations.

mean_hca, mean_c Vectors of the posterior means for the hybrid and concurrent control arms for each simulation.

simulated.data A matrix of the simulated response counts used.

w A vector of the final borrowing weights used in each simulation.

... and all input parameters.

Examples

```
o <- power.DPP(pt=0.5, nt=40, pc=0.3, nc=40, pc.calib = 0.3, pch=0.3,
               nche=40, nch=180, alpha=0.1, nsim = 1000, seed=2000) # nsim is reduced
```

`runCalibratedSAM` *Run a Calibrated Simulation Using SAM Priors*

Description

Performs a two-step simulation.

1. **Calibration Step:** It first runs a simulation under the null hypothesis (using `*.calib` parameters) to determine the posterior probability thresholds (quantiles) needed to control the Type I Error Rate at the specified `typeIER.cal` level.
2. **Main Simulation Step:** It then uses these calibrated thresholds to run the main simulation (using parameters `nc`, `nt`, `pc`, `pt`) to evaluate final operating characteristics (e.g., Type I Error or Power) and bias.

Usage

```
runCalibratedSAM(
  nsim = 1000,
  pch,
  delta_threshold,
  nche.c,
  nc.calib,
  nt.calib,
  pc.calib,
  xt.cal = NULL,
  xc.cal = NULL,
  typeIER.cal = 0.1,
  nche,
  nc,
  nt,
  pc,
  pt,
```

```
  xt = NULL,
  xc = NULL,
  nf.prior = RBesT::mixbeta(c(1, 0.001, 0.001)),
  seed.hist = 1000,
  seed.gMAP = 2000,
  seed.SAM = 3000,
  seed.cal = 4000
)
```

Arguments

nsim	The total number of simulation trials to run for both the calibration and main simulation steps.
pch	Historical control response rate (used to generate HistData for the RBesT::gMAP() prior).
delta_threshold	The CSD (Clinically Significant Difference) threshold used for the SAM prior in runSAM().
nche.c	Sample size for the historical control data (HistData).
nc.calib	Control group sample size for the **calibration** step.
nt.calib	Treatment group sample size for the **calibration** step.
pc.calib	Control response rate for the **calibration** step. This is also used as the treatment response rate in this step to simulate the null hypothesis.
xt.cal	(Optional) A vector of pre-simulated treatment outcomes for the **calibration** step. If NULL, data is simulated internally.
xc.cal	(Optional) A vector of pre-simulated control outcomes for the **calibration** step. If NULL, data is simulated internally.
typeIER.cal	The target Type I Error rate to control for during the calibration step. Defaults to 0.1.
nche	Historical control sample size. (Note: This parameter is defined in the function signature but not explicitly used in the function body; nche.c is used for calibration data generation.)
nc	Control group sample size for the **main simulation** step.
nt	Treatment group sample size for the **main simulation** step.
pc	Control response rate for the **main simulation** step.
pt	Treatment response rate for the **main simulation** step. (Note: Set pt = pc for Type I Error, or pt > pc for Power).
xt	(Optional) A vector of pre-simulated treatment outcomes for the **main simulation**. If NULL, data is simulated internally.
xc	(Optional) A vector of pre-simulated control outcomes for the **main simulation**. Examples use RBesT::mixbeta(). If NULL, data is simulated internally.
nf.prior	The non-informative prior to use. Defaults to RBesT::mixbeta(c(1, 0.001, 0.001)).

seed.hist	Seed for generating historical data.
seed.gMAP	Seed for the RBesT::gMAP() function.
seed.SAM	Seed for the main runSAM simulation call.
seed.cal	Seed for the calibration runSAM call.

Details

This function wraps the core runSAM() function by adding a calibration layer. It derives historical data priors using RBesT::gMAP() and RBesT::automixfit() based on nche.c and pch.

The calibration step (runSAM with ...calib parameters) finds the (1 - typeIER.cal) quantiles for the posterior distributions (SAM, rMAP, Non-info) under the null.

The main simulation step (runSAM with main parameters) then calculates the proportion of simulations where the posterior probability exceeds these calibrated thresholds. This proportion represents the final Type I Error or Power.

Value

A list with the following components:

Sim_Result A numeric vector (length 3) with the final simulation result (Type I Error or Power) for the "SAMprior", "MAP", and "Noninfo" methods.

pc.PMD Posterior Mean Difference (Bias) for the three methods.

pc.PSDD Posterior SD of the Difference (Bias) for the three methods.

pc.PM A matrix (3 x nsim) of posterior means for the control rate for each simulation iteration.

Calibration_Thresholds A matrix (1x3) containing the decision thresholds (quantiles) determined during the calibration step for the SAM, rMAP, and Non-info methods.

See Also

runSAM, gMAP, automixfit, mixbeta, decision2S

Examples

```
# Example uses functions from the RBesT package
library(RBesT)

sim_power_pc30 <- runCalibratedSAM(
   nsim = 100,
   pch = 0.30,
   delta_threshold = 0.1,
   nche.c = 180,
   nc.calib = 45,
   nt.calib = 45,
   pc.calib = 0.3,
   typeIER.cal = 0.10,
   nche = 180,
   nc = 45,
   nt = 45,
```

```
   pc = 0.30,
   pt = 0.50
)

print(sim_power_pc30$Sim_Result)
```

---

runSAM *Generating Operating Characteristics of SAM Priors*

Description

This function is modified based on the get_OC function from the SAMprior R package version 1.1.1.

Usage

```
runSAM(
  if.prior,
  theta.h,
  method.w,
  prior.odds,
  nf.prior,
  delta,
  n,
  n.t,
  decision,
  ntrial,
  if.MAP,
  weight,
  theta,
  theta.t,
  datamat = NULL,
  ...
)

## S3 method for class 'betaMix'
runSAM(
  if.prior,
  theta.h,
  method.w,
  prior.odds,
  nf.prior,
  delta,
  n,
  n.t,
```

```
  decision,
  ntrial,
  if.MAP,
  weight,
  theta,
  theta.t,
  datamat = NULL,
  ...
)

## S3 method for class 'normMix'
runSAM(
  if.prior,
  theta.h,
  method.w,
  prior.odds,
  nf.prior,
  delta,
  n,
  n.t,
  decision,
  ntrial,
  if.MAP,
  weight,
  theta,
  theta.t,
  datamat = NULL,
  ...,
  sigma
)
```

Arguments

`if.prior`	Informative prior constructed from historical data, represented (approximately) as a mixture of conjugate distributions.
`theta.h`	Estimate of the treatment effect based on historical data. If missing, the default value is set to be the posterior mean estimate from `if.prior`.
`method.w`	Methods used to determine the mixture weight for SAM priors. The default method is LRT (Likelihood Ratio Test), the alternative option can be PPR (Posterior Probability Ratio). See `SAM_weight` for more details.
`prior.odds`	The prior probability of H_0 being true compared to the prior probability of H_1 being true using PPR method. The default value is 1. See `SAM_weight` for more details.
`nf.prior`	Non-informative prior used for constructing the SAM prior and robust MAP prior.
`delta`	Clinically significant difference used for the SAM prior.
`n`	Sample size for the control arm.

n.t	Sample size for the treatment arm.
decision	Decision rule to compare the treatment with the control; see decision2S.
ntrial	Number of trials simulated.
if.MAP	Whether to simulate the operating characteristics of the robust MAP prior for comparison, the default value is FALSE.
weight	Weight assigned to the informative prior component ($0 \leq$ weight ≤ 1) for the robust MAP prior, the default value is 0.5.
theta	A vector of the response rate (binary endpoints) or mean (continuous endpoints) for the control arm.
theta.t	A vector of the response rate (binary endpoints) or mean (continuous endpoints) for the treatment arm.
datamat	A ntrial * 2 matrix of simulated binomial samples, first column is study treament group treatment, second column is study control group
...	Additional parameters for continuous endpoints.
sigma	Variance to simulate the continuous endpoint under normality assumption.

Details

This function is modified based on the get_OC from the SAMprior R package version 1.1.1. The modification is made in order that the function can take simulated data (the datamat argument) from outside the function rather than simulating data within the funciton We used the same data for all methods for comparison in order to get a better comparison.

The runSAM function is designed to generate the operating characteristics of SAM priors (*Yang, et al., 2023*), including the relative bias, relative mean squared error, and type I error and power under a two-arm comparative trial design. As an option, the operating characteristic of robust MAP priors (*Schmidli, et al., 2014*) can also be generated for comparison.

The runSAM function is designed to generate the operating characteristics of SAM priors, including the relative bias, relative mean squared error, and type I error, and power under a two-arm comparative trial design. As an option, the operating characteristics of robust MAP priors (*Schmidli, et al., 2014*) can also be generated for comparison.

The relative bias is defined as the difference between the bias of a method and the bias of using a non-informative prior. The relative mean squared error is the difference between the mean squared error (MSE) of a method and the MES of using a non-informative prior.

To evaluate type I error and power, the determination of whether the treatment is superior to the control is calculated based on function decision2S.

Value

Returns a list result - dataframe that contains the relative bias, relative MSE, type I error, and power for both SAM priors, as well as robust MAP priors. Additionally, the mixture weight of the SAM prior is also displayed.

simulated.data - a matrix of two columns, first treatment, second control post_theta_t_list - list of all replication, posterior distribution of treatment group post_theta_c_list - list of all replication, posterior distribution of control group, non informative prior post_theta_c_SAM_list - list of all replication, posterior distribution of control group, SAM prior post_theta_c_MAP_list - list of all replication, posterior distribution of control group, MAP prior

Methods (by class)

- `runSAM(betaMix)`: The function is designed to generate the operating characteristics of SAM priors for binary endpoints.
- `runSAM(normMix)`: The function is designed to generate the operating characteristics of SAM priors for continuous endpoints.

References

Yang P, Zhao Y, Nie L, Vallejo J, Yuan Y. SAM: Self-adapting mixture prior to dynamically borrow information from historical data in clinical trials. *Biometrics* 2023; 00, 1–12. https://doi.org/10.1111/biom.13927

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

Examples

```
set.seed(123)
## Example of a binary endpoint
## Consider a randomized comparative trial designed to borrow information
## from historical data on the control. We assumed a non-informative prior
## beta(1, 1) and an informative prior beta(30, 50) after incorporating
## the historical data. The treatment is regarded as superior to the control
## if Pr(RR.t > RR.c | data) > 0.95, where RR.t and RR.c are response rates
##  of the treatment and control, respectively. The operating characteristics
##  were assessed under the scenarios of (RR.c, RR.t) = (0.3, 0.36) and (0.3, 0.56).
## OC <- runSAM(## Informative prior constructed based on historical data
##              if.prior = mixbeta(c(1, 30, 50)),
##              ## Non-informative prior used for constructing the SAM prior
##              nf.prior = mixbeta(c(1,1,1)),
##              delta    = 0.2,  ## Clinically significant difference
##              n = 35,          ## Sample size for the control arm
##              n.t = 70,        ## Sample size for the treatment arm
##              ## Decision rule to compare the whether treatment is superior
##              ## than the control
##              decision = decision2S(0.95, 0, lower.tail=FALSE),
##              ntrial   = 1000,  ## Number of trials simulated
##              ## Weight assigned to the informative component for MAP prior
##              weight = 0.5,
##              ## A vector of response rate for the control arm
##              theta    = c(0.3, 0.36),
##              ## A vector of response rate for the treatment arm
##              theta.t  = c(0.3, 0.56))
## OC

## Example of continuous endpoint
## Consider a randomized comparative trial designed to borrow information
## from historical data on the control. We assumed a non-informative prior
## N(0, 1e4) and an informative prior N(0.5, 2) after incorporating
## the historical data. The treatment is regarded as superior to the control
## if Pr(mean.t > mean.c | data) > 0.95, where mean.t and mean.c are mean
##  of the treatment and control, respectively. The operating characteristics
```

```
##  were assessed under the scenarios of (mean.c, mean.t) = (0.1, 0.1) and
## (0.5, 1.0).
sigma      <- 2
prior.mean <- 0.5
prior.se   <- sigma/sqrt(100)
## OC <- runSAM(## Informative prior constructed based on historical data
##              if.prior = mixnorm(c(1, prior.mean, prior.se)),
##              ## Non-informative prior used for constructing the SAM prior
##              nf.prior = mixnorm(c(1, 0, 1e4)),
##              delta    = 0.2 * sigma,  ## Clinically significant difference
##              n = 100,                 ## Sample size for the control arm
##              n.t = 200,               ## Sample size for the treatment arm
##              ## Decision rule to compare the whether treatment is superior
##              ## than the control
##              decision = decision2S(0.95, 0, lower.tail=FALSE),
##              ntrial   = 1000,  ## Number of trials simulated
##              ## A vector of mean for the control arm
##              theta    = c(0.1, 0.5),
##              ## A vector of mean for the treatment arm
##              theta.t  = c(0.1, 1.0),
##              sigma = sigma)
## OC
```

SAM_prior *Calculating SAM priors*

Description

This function is exactly from the SAMprior R package version 1.1.1. It is included to support the runSAM function in this package.

Usage

```
SAM_prior(if.prior, nf.prior, weight, ...)

## S3 method for class 'betaMix'
SAM_prior(if.prior, nf.prior, weight, ...)

## S3 method for class 'gammaMix'
SAM_prior(if.prior, nf.prior, weight, ...)

## S3 method for class 'normMix'
SAM_prior(if.prior, nf.prior, weight, ..., sigma)
```

Arguments

if.prior	Informative prior constructed from historical data, represented (approximately) as a mixture of conjugate distributions.

nf.prior	Non-informative prior used for the mixture.
weight	Weight assigned to the informative prior component ($0 \leq$ weight ≤ 1), which should be determined by SAM_weight function.
...	Additional parameters required for different endpoints.
sigma	Variance used for constructing the non-informative prior for continuous endpoints.

Details

The SAM_prior function is designed to display the SAM prior, given the informative prior (constructed from historical data), non-informative prior, and the mixture weight calculated using SAM_weight function (*Yang, et al., 2023*).

SAM prior is constructed by mixing an informative prior $\pi_1(\theta)$, constructed based on historical data, with a non-informative prior $\pi_0(\theta)$ using the mixture weight w determined by SAM_weight function to achieve the degree of prior-data conflict (*Schmidli et al., 2015, Yang et al., 2023*).

Let θ and θ_h denote the treatment effects associated with the current arm data D and historical data D_h, respectively. Let δ denote the clinically significant difference such that if $|\theta_h - \theta| \geq \delta$, then θ_h is regarded as clinically distinct from θ, and it is therefore inappropriate to borrow any information from D_h. Consider two hypotheses:

$$H_0 : \theta = \theta_h,\ H_1 : \theta = \theta_h + \delta \ or\ \theta = \theta_h - \delta.$$

H_0 represents that D_h and D are consistent (i.e., no prior-data conflict) and thus information borrowing is desirable, whereas H_1 represents that the treatment effect of D differs from D_h to such a degree that no information should be borrowed.

The SAM prior uses the likelihood ratio test (LRT) statistics R to quantify the degree of prior-data conflict and determine the extent of information borrowing.

$$R = P(D|H_0, \theta_h)/P(D|H_1, \theta_h) = P(D|\theta = \theta_h)/\max(P(D|\theta = \theta_h + \delta), P(D|\theta = \theta_h - \delta)),$$

where $P(D|\cdot)$ denotes the likelihood function. An alternative Bayesian choice is the posterior probability ratio (PPR):

$$R = P(D|H_0, \theta_h)/P(D|H_1, \theta_h) = P(H_0)/P(H_1) \times BF,$$

where $P(H_0)$ and $P(H_1)$ is the prior probabilities of H_0 and H_1 being true. BF is the Bayes Factor that in this case is the same as the LRT.

The SAM prior, denoted as $\pi_{sam}(\theta)$, is then defined as a mixture of an informative prior $\pi_1(\theta)$, constructed based on D_h and a non-informative prior $\pi_0(\theta)$:

$$\pi_{sam}(\theta) = w\pi_1(\theta) + (1 - w)\pi_0(\theta),$$

where the mixture weight w is calculated as:

$$w = R/(1 + R).$$

As the level of prior-data conflict increases, the likelihood ratio R decreases, resulting in a decrease in the weight w assigned to the informative prior and thus a decrease in information borrowing. As a result, $\pi_{sam}(\theta)$ is data-driven and has the ability to self-adapt the information borrowing based on the degree of prior-data conflict.

Value

Displays the SAM prior as a mixture of an informative prior (constructed based on the historical data) and a non-informative prior.

Methods (by class)

- `SAM_prior(betaMix)`: The function calculates the SAM prior for beta mixture distribution. The default `nf.prior` is set to be `mixbeta(c(1,1,1))` which represents a uniform prior `Beta(1,1)`.
- `SAM_prior(gammaMix)`: The function calculates the SAM prior for gamma mixture distribution. The default `nf.prior` is set to be `mixgamma(c(1,0.001,0.001))` which represents a vague gamma prior `Gamma(0.001,0.001)`.
- `SAM_prior(normMix)`: The function calculates the SAM prior for normal mixture distribution. The default `nf.prior` is set to be `mixnorm(c(1,summary(if.prior)['mean'], sigma))` which represents a unit-information prior.

References

Yang P, Zhao Y, Nie L, Vallejo J, Yuan Y. SAM: Self-adapting mixture prior to dynamically borrow information from historical data in clinical trials. *Biometrics* 2023; 00, 1–12. https://doi.org/10.1111/biom.13927

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

See Also

SAM_weight

Examples

```
set.seed(123)
## Examples for binary endpoints
## Suppose that the informative prior constructed based on historical data is
## beta(40, 60)
prior.historical <- RBesT::mixbeta(c(1, 40, 60))
## Data of the control arm
data.control     <- stats::rbinom(60, size = 1, prob = 0.42)
## Calculate the mixture weight of the SAM prior
wSAM <- SAM_weight(if.prior = prior.historical,
                   delta = 0.15,      ## Clinically significant difference
                   data = data.control  ## Control arm data
                   )
## Assume beta(1,1) as the non-informative prior used for mixture
nf.prior  <- RBesT::mixbeta(nf.prior = c(1,1,1))
```

```
## Generate the SAM prior
SAM.prior <- SAM_prior(if.prior = prior.historical, ## Informative prior
                       nf.prior = nf.prior,         ## Non-informative prior
                       weight = wSAM                ## Mixture weight of the SAM prior
                       )
graphics::plot(SAM.prior)

## Examples for continuous endpoints
## Suppose that the informative prior constructed based on historical data is
## N(0, 3)
sigma      <- 3
prior.mean <- 0
prior.se   <- sigma/sqrt(100)
prior.historical <- RBesT::mixnorm(c(1, prior.mean, prior.se), sigma = sigma)
## Data of the control arm
data.control <- stats::rnorm(80, mean = 0, sd = sigma)
## Calculate the mixture weight of the SAM prior
wSAM <- SAM_weight(if.prior = prior.historical,
                   delta = 0.2 * sigma,   ## Clinically significant difference
                   data = data.control     ## Control arm data
                   )
## Assume unit-information prior N(0,3) as the non-informative prior used
## for the mixture
nf.prior         <- RBesT::mixnorm(nf.prior = c(1,prior.mean, sigma),
                                   sigma = sigma)
## Generate the SAM prior
SAM.prior <- SAM_prior(if.prior = prior.historical, ## Informative prior
                       nf.prior = nf.prior,         ## Non-informative prior
                       weight = wSAM                ## Mixture weight of the SAM prior
                       )
graphics::plot(SAM.prior)
```

`SAM_weight` *Calculating Mixture Weight of SAM Priors*

Description

This function is exactly from the SAMprior R package version 1.1.1. It is included to support the runSAM function in this package.

Usage

```
SAM_weight(if.prior, theta.h, method.w, prior.odds, data, delta, ...)

## S3 method for class 'betaMix'
SAM_weight(if.prior, theta.h, method.w, prior.odds, data, delta, n, r, ...)

## S3 method for class 'normMix'
```

```
SAM_weight(
  if.prior,
  theta.h,
  method.w,
  prior.odds,
  data,
  delta,
  m,
  n,
  sigma,
  ...
)

## S3 method for class 'gammaMix'
SAM_weight(if.prior, theta.h, method.w, prior.odds, data, delta, u, w, ...)
```

Arguments

`if.prior`	Informative prior constructed based on historical data, represented (approximately) as a mixture of conjugate distributions.
`theta.h`	Estimate of the treatment effect based on historical data. If missing, the default value is set to be the posterior mean estimate from `if.prior`.
`method.w`	Methods used to determine the mixture weight for SAM priors. The default method is "LRT" (Likelihood Ratio Test), the alternative option is "PPR" (Posterior Probability Ratio). See Details section for more information.
`prior.odds`	The prior probability of H_0 being true compared to the prior probability of H_1 being true using PPR method. The default value is 1. See Details section for more information.
`data`	Data of the control arm from the current trial, see Methods section for more details.
`delta`	Clinically significant difference used for the SAM prior.
`...`	Additional parameters required for different endpoints.
`n`	Number of subjects in the control arm for continuous endpoint.
`r`	Number of responses in the control arm for binary endpoint.
`m`	Mean estimate in the control arm for continuous endpoint.
`sigma`	Standard deviation in the control arm for continuous endpoint.
`u`	Number of events in the control arm for time-to-event endpoint.
`w`	Total observed time in the control arm for time-to-event endpoint.

Details

The `SAM_weight` function is designed to calculate the mixture weight of the SAM priors according to the degree of prior-data conflicts (*Yang, et al., 2023*).

SAM prior is constructed by mixing an informative prior $\pi_1(\theta)$, constructed based on historical data, with a non-informative prior $\pi_0(\theta)$ using the mixture weight w determined by `SAM_weight` function to achieve the degree of prior-data conflict (*Schmidli et al., 2015, Yang et al., 2023*).

Let θ and θ_h denote the treatment effects associated with the current arm data D and historical data D_h, respectively. Let δ denote the clinically significant difference such that if $|\theta_h - \theta| \geq \delta$, then θ_h is regarded as clinically distinct from θ, and it is therefore inappropriate to borrow any information from D_h. Consider two hypotheses:

$$H_0 : \theta = \theta_h,\ H_1 : \theta = \theta_h + \delta \ or\ \theta = \theta_h - \delta.$$

H_0 represents that D_h and D are consistent (i.e., no prior-data conflict) and thus information borrowing is desirable, whereas H_1 represents that the treatment effect of D differs from D_h to such a degree that no information should be borrowed.

The SAM prior uses the likelihood ratio test (LRT) statistics R to quantify the degree of prior-data conflict and determine the extent of information borrowing.

$$R = P(D|H_0, \theta_h)/P(D|H_1, \theta_h) = P(D|\theta = \theta_h)/\max(P(D|\theta = \theta_h + \delta), P(D|\theta = \theta_h - \delta)),$$

where $P(D|\cdot)$ denotes the likelihood function. An alternative Bayesian choice is the posterior probability ratio (PPR):

$$R = P(D|H_0, \theta_h)/P(D|H_1, \theta_h) = P(H_0)/P(H_1) \times BF,$$

where $P(H_0)$ and $P(H_1)$ is the prior probabilities of H_0 and H_1 being true. BF is the Bayes Factor that in this case is the same as the LRT.

The SAM prior, denoted as $\pi_{sam}(\theta)$, is then defined as a mixture of an informative prior $\pi_1(\theta)$, constructed based on D_h and a non-informative prior $\pi_0(\theta)$:

$$\pi_{sam}(\theta) = w\pi_1(\theta) + (1 - w)\pi_0(\theta),$$

where the mixture weight w is calculated as:

$$w = R/(1 + R).$$

As the level of prior-data conflict increases, the likelihood ratio R decreases, resulting in a decrease in the weight w assigned to the informative prior and thus a decrease in information borrowing. As a result, $\pi_{sam}(\theta)$ is data-driven and has the ability to self-adapt the information borrowing based on the degree of prior-data conflict.

Value

The mixture weight of the SAM priors.

Methods (by class)

- `SAM_weight(betaMix)`: The function calculates the mixture weight of SAM priors for beta mixture distribution. The input `data` can be patient-level data (i.e., a vector of 0 and 1 representing the response status of each patient) or summary statistics (i.e., the number of patients and the number of responses).

- `SAM_weight(normMix)`: The function calculates the mixture weight of SAM priors for normal mixture distribution. The input `data` should be a vector of patient-level observations. The input `data` can be patient-level data (i.e., a vector of continuous response of each patient) or summary statistics (i.e., the mean estimate, number of subjects, and the standard deviation in the control arm).
- `SAM_weight(gammaMix)`: The function calculates the mixture weight of SAM priors for gamma mixture distribution. The input `data` can be patient-level data (i.e., a matrix with the first row as the censoring indicator and the second row recording the observed time) or summary statistics (i.e., the number of uncensored observations `u` and total observed time `w`).

References

Yang P, Zhao Y, Nie L, Vallejo J, Yuan Y. SAM: Self-adapting mixture prior to dynamically borrow information from historical data in clinical trials. *Biometrics* 2023; 00, 1–12. https://doi.org/10.1111/biom.13927

Examples

```
set.seed(123)
## Examples for binary endpoints
## Example 1: no prior-data conflict
## Suppose that the informative prior is beta(40, 60)
prior.historical <- RBesT::mixbeta(c(1, 40, 60))
## Data of control arm
data.control     <- stats::rbinom(60, size = 1, prob = 0.42)
## Calculate the mixture weight of the SAM prior
wSAM <- SAM_weight(if.prior = prior.historical,
                   delta = 0.15,      ## Clinically significant difference
                   data = data.control  ## Control arm data
                   )
print(wSAM)

## Example 2: with prior-data conflict (12 responses in 60 patients)
wSAM <- SAM_weight(if.prior = prior.historical,
                   delta = 0.15,
                   method.w = 'PPR',
                   prior.odds = 1/9,
                   n = 60,
                   r = 12
                   )
print(wSAM)

## Examples for continuous endpoints
## Example 1: no prior-data conflict
## Suppose the informative prior is N(0, 3)
sigma      <- 3
prior.mean <- 0
prior.se   <- sigma/sqrt(100)
prior.historical <- RBesT::mixnorm(c(1, prior.mean, prior.se), sigma = sigma)
## Data of the control arm
data.control <- stats::rnorm(80, mean = 0, sd = sigma)
wSAM <- SAM_weight(if.prior = prior.historical,
                   delta = 0.3 * sigma,
```

```
                    data = data.control
                    )
print(wSAM)
```

`Two.Prop.Test.Sample.Size`
Sample Size for Comparing Two Proportions

Description

Calculates the required sample size for a two-group comparison of proportions based on the method of Casagrande, Pike, and Smith (1978).

Usage

```
Two.Prop.Test.Sample.Size(p1, p2, alpha, beta, r)
```

Arguments

`p1`	A scalar. The anticipated proportion in Arm 1 (e.g., control).
`p2`	A scalar. The anticipated proportion in Arm 2 (e.g., experimental).
`alpha`	A scalar. The one-sided Type I error rate.
`beta`	A scalar. The Type II error rate (1 - power).
`r`	A scalar. The randomization ratio of Arm 2 to Arm 1 (n_2/n_1).

Value

A numeric vector of length two:

- The required sample size for Arm 1 (n_1).
- The total required sample size ($n_1 + n_2$).

See Also

https://www2.ccrb.cuhk.edu.hk/stat/proportion/Casagrande.htm

Examples

```
Two.Prop.Test.Sample.Size(p1=0.27, p2=0.47, alpha = 0.10, beta=0.2, r=2)
```

Index

automixfit, *25*

Bayesian.Hybrid.Analysis, 2
borrow.wt, 4

calibration, 5

decision2S, *25*, *28*
DPP, 7
DPP.analysis, 9

EESS, 11
exactci, 12
explore.power.DPP, 13

fisher, 15
fisher.bound, 16
fisher.power, 17
fisher.test, *16*
Frequentist.Analysis, 18

gMAP, *25*

mixbeta, *25*

plotDPP, 19
plotPMD, 19
power.binom.test, 20
power.DPP, *15*, 21

RBesT::automixfit(), *25*
RBesT::gMAP(), *24*, *25*
RBesT::mixbeta(), *24*
runCalibratedSAM, 23
runSAM, *25*, 26
runSAM(), *24*, *25*

SAM_prior, 30
SAM_weight, *27*, *31*, *32*, 33, *34*

Two.Prop.Test.Sample.Size, 37