Package 'BayesianQDM'

May 6, 2026

Type Package

Title Bayesian Quantitative Decision-Making Framework for Binary and Continuous Endpoints

Version 0.1.0

Description Provides comprehensive methods to calculate posterior probabilities, posterior predictive probabilities, and Go/NoGo/Gray decision probabilities for quantitative decision-making under a Bayesian paradigm in clinical trials. The package supports both single and two-endpoint analyses for binary and continuous outcomes, with controlled, uncontrolled, and external designs. For single continuous endpoints, three calculation methods are available: numerical integration (NI), Monte Carlo simulation (MC), and Moment-Matching approximation (MM). For two continuous endpoints, a bivariate Normal-Inverse-Wishart conjugate model is implemented with MC and MM methods. For two binary endpoints, a Dirichlet-multinomial model is implemented. External designs incorporate historical data through power priors using exact conjugate representations (Normal-Inverse-Chi-squared for single continuous, Normal-Inverse-Wishart for two continuous, and Dirichlet for binary endpoints), enabling closed-form posterior computation without Markov chain Monte Carlo (MCMC) sampling. This approach significantly reduces computational burden while preserving complete Bayesian rigor. The package also provides grid-search functions to find optimal Go and NoGo thresholds that satisfy user-specified operating characteristic criteria for all supported endpoint types and study designs. S3 print() and plot() methods are provided for all decision probability classes, enabling formatted display and visualisation of Go/NoGo/Gray operating characteristics across treatment scenarios.
See Kang, Yamaguchi, and Han (2026) <doi:10.1080/10543406.2026.2655410> for the methodological framework.

License GPL (>= 2)

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

Imports ggplot2 (>= 3.4.0), gridExtra, mvtnorm, stats

Suggests testthat (>= 3.0.0), knitr, rmarkdown, dplyr, tidyr, purrr, spelling

VignetteBuilder knitr

URL https://gosukehommaEX.github.io/BayesianQDM/, https://github.com/gosukehommaEX/BayesianQDM

BugReports https://github.com/gosukehommaEX/BayesianQDM/issues

NeedsCompilation no

Author Gosuke Homma [aut, cre], Yusuke Yamaguchi [aut]

Maintainer Gosuke Homma <my.name.is.gosuke@gmail.com>

Repository CRAN

Date/Publication 2026-04-21 18:12:38 UTC

Contents

`allmultinom` *Enumerate All Multinomial Count Vectors for a Bivariate Binary Outcome*

Description

Generates the complete set of non-negative integer vectors $(x_{00}, x_{01}, x_{10}, x_{11})$ satisfying $x_{00} + x_{01} + x_{10} + x_{11} = n_j$, where n_j is the sample size of group $j \in \{t, c\}$. Each row of the returned matrix corresponds to one possible realisation of the aggregated bivariate binary counts. This enumeration is a prerequisite for exact operating characteristic assessment in `pbayesdecisionprob2bin`.

Usage

```
allmultinom(n)
```

Arguments

`n`	A single non-negative integer giving the sample size of the group (n_j).

Details

The number of non-negative integer solutions to $x_{00} + x_{01} + x_{10} + x_{11} = n_j$ is the stars-and-bars count

$$\binom{n_j + 3}{3} = \frac{(n_j + 1)(n_j + 2)(n_j + 3)}{6}.$$

For example, $n_j = 10$ yields 286 rows and $n_j = 20$ yields 1771 rows. The matrix is pre-allocated to avoid repeated memory reallocation, making the function efficient for the sample sizes typical in rare-disease proof-of-concept studies.

This function is an internal computational building block used by `pbayesdecisionprob2bin` to enumerate the sample space over which multinomial probabilities and decision indicators are summed when computing exact operating characteristics.

Value

An integer matrix with $\binom{n_j+3}{3}$ rows and 4 columns named `x00`, `x01`, `x10`, `x11`. Each row is a distinct non-negative integer solution to $x_{00} + x_{01} + x_{10} + x_{11} = n_j$. Rows are ordered by `x00` (ascending), then `x10` (ascending), then `x01` (ascending).

Examples

```
# Example 1: n = 2 (smallest non-trivial case)
allmultinom(2)

# Example 2: n = 10 (typical rare-disease PoC group size)
mat <- allmultinom(10)
nrow(mat)              # Should be choose(13, 3) = 286
all(rowSums(mat) == 10)  # Every row must sum to n

# Example 3: n = 0 (edge case - only the all-zero row)
allmultinom(0)

# Example 4: Verify column names
colnames(allmultinom(5))

# Example 5: Row counts match the stars-and-bars formula
n <- 15L
mat <- allmultinom(n)
nrow(mat) == choose(n + 3L, 3L)  # Should be TRUE
```

`getgamma1bin` *Find Optimal Go/NoGo Thresholds for a Single Binary Endpoint*

Description

Computes the optimal Go threshold γ_{go} and NoGo threshold γ_{nogo} for a single binary endpoint by searching over a grid of candidate values. The two thresholds are calibrated independently under separate scenarios:

- γ_{go} is the **smallest** value in `gamma_grid` such that the marginal Go probability $\Pr(g_{\text{Go}} \geq \gamma_{\text{go}})$ is strictly less than `target_go` under the Go-calibration scenario (`pi_t_go`, `pi_c_go`); typically the Null scenario.
- γ_{nogo} is the **smallest** value in `gamma_grid` such that the marginal NoGo probability $\Pr(g_{\text{NoGo}} \geq \gamma_{\text{nogo}})$ is strictly less than `target_nogo` under the NoGo-calibration scenario (`pi_t_nogo`, `pi_c_nogo`); typically the Alternative scenario.

Here $g_{\text{Go}} = P(\theta > \theta_{\text{TV}} \mid y_t, y_c)$ and $g_{\text{NoGo}} = P(\theta \leq \theta_{\text{MAV}} \mid y_t, y_c)$ for `prob = 'posterior'`, consistent with the decision rule in `pbayesdecisionprob1bin`.

Usage

```
getgamma1bin(
  prob = "posterior",
  design = "controlled",
  theta_TV = NULL,
  theta_MAV = NULL,
  theta_NULL = NULL,
```

```
  pi_t_go,
  pi_c_go = NULL,
  pi_t_nogo,
  pi_c_nogo = NULL,
  target_go,
  target_nogo,
  n_t,
  n_c,
  a_t,
  a_c,
  b_t,
  b_c,
  z = NULL,
  m_t = NULL,
  m_c = NULL,
  ne_t = NULL,
  ne_c = NULL,
  ye_t = NULL,
  ye_c = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  gamma_grid = seq(0.01, 0.99, by = 0.01)
)
```

Arguments

prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
theta_TV	A numeric scalar in (-1, 1) giving the Target Value (TV) threshold for the treatment effect. Required when prob = 'posterior'; set to NULL otherwise.
theta_MAV	A numeric scalar in (-1, 1) giving the Minimum Acceptable Value (MAV) threshold. Must satisfy theta_TV > theta_MAV. Required when prob = 'posterior'; set to NULL otherwise.
theta_NULL	A numeric scalar in (-1, 1) giving the null hypothesis threshold used for the predictive probability. Required when prob = 'predictive'; set to NULL otherwise.
pi_t_go	A numeric scalar in (0, 1) giving the true treatment response rate under the Go-calibration scenario (typically Null).
pi_c_go	A numeric scalar in (0, 1) giving the true control response rate under the Go-calibration scenario. Set to NULL for design = 'uncontrolled'.
pi_t_nogo	A numeric scalar in (0, 1) giving the true treatment response rate under the NoGo-calibration scenario (typically Alternative).
pi_c_nogo	A numeric scalar in (0, 1) giving the true control response rate under the NoGo-calibration scenario. Set to NULL for design = 'uncontrolled'.

`target_go` A numeric scalar in `(0, 1)` giving the upper bound on the marginal Go probability under the Go-calibration scenario. The optimal γ_{go} is the smallest grid value satisfying $\Pr(\mathrm{Go}) <$ `target_go`.

`target_nogo` A numeric scalar in `(0, 1)` giving the upper bound on the marginal NoGo probability under the NoGo-calibration scenario. The optimal γ_{nogo} is the largest grid value satisfying $\Pr(\mathrm{NoGo}) <$ `target_nogo`.

`n_t` A positive integer giving the number of patients in the treatment group in the PoC trial.

`n_c` A positive integer giving the number of patients in the control group in the PoC trial.

`a_t` A positive numeric scalar giving the first shape parameter of the Beta prior for the treatment group.

`a_c` A positive numeric scalar giving the first shape parameter of the Beta prior for the control group.

`b_t` A positive numeric scalar giving the second shape parameter of the Beta prior for the treatment group.

`b_c` A positive numeric scalar giving the second shape parameter of the Beta prior for the control group.

`z` A non-negative integer giving the hypothetical number of responders in the control group. Required when `design = 'uncontrolled'`; set to `NULL` otherwise.

`m_t` A positive integer giving the future sample size for the treatment group. Required when `prob = 'predictive'`; set to `NULL` otherwise.

`m_c` A positive integer giving the future sample size for the control group. Required when `prob = 'predictive'`; set to `NULL` otherwise.

`ne_t` A positive integer giving the number of patients in the treatment group of the external data set. Required when `design = 'external'`; set to `NULL` otherwise.

`ne_c` A positive integer giving the number of patients in the control group of the external data set. Required when `design = 'external'`; set to `NULL` otherwise.

`ye_t` A non-negative integer giving the number of responders in the treatment group of the external data set. Required when `design = 'external'`; set to `NULL` otherwise.

`ye_c` A non-negative integer giving the number of responders in the control group of the external data set. Required when `design = 'external'`; set to `NULL` otherwise.

`alpha0e_t` A numeric scalar in `(0, 1]` giving the power prior weight for the treatment group. Required when `design = 'external'`; set to `NULL` otherwise.

`alpha0e_c` A numeric scalar in `(0, 1]` giving the power prior weight for the control group. Required when `design = 'external'`; set to `NULL` otherwise.

`gamma_grid` A numeric vector of candidate threshold values in `(0, 1)` to search over. Defaults to `seq(0.01, 0.99, by = 0.01)`.

Details

The function uses a two-stage precompute-then-sweep strategy:

1. **Precomputation**: All possible outcome pairs (y_t, y_c) are enumerated. For each pair, pbayespostpred1bin computes g_{Go} (`lower.tail` = FALSE at `theta_TV`) and g_{NoGo} (`lower.tail` = TRUE at `theta_MAV`). This step is independent of γ.
2. **Gamma sweep**: Marginal probabilities are computed as weighted sums of binary indicators over the grid: $\Pr(\mathrm{Go})$ uses `w_go` (weights under `pi_t_go`, `pi_c_go`) and the indicator $g_{\mathrm{Go}} \geq \gamma$; $\Pr(\mathrm{NoGo})$ uses `w_nogo` (weights under `pi_t_nogo`, `pi_c_nogo`) and the indicator $g_{\mathrm{NoGo}} \geq \gamma$.

Both $\Pr(\mathrm{Go})$ and $\Pr(\mathrm{NoGo})$ are monotone non-increasing functions of γ. The optimal γ_{go} is the *smallest* grid value crossing below `target_go`. The optimal γ_{nogo} is also the *smallest* grid value crossing below `target_nogo`: a smaller γ_{nogo} makes NoGo harder to trigger (more permissive), so this is the least restrictive threshold that still controls the false NoGo rate.

Value

A list of class `getgamma1bin` with the following elements:

gamma_go Optimal Go threshold: the smallest value in `gamma_grid` for which $\Pr(\mathrm{Go}) <$ `target_go` under the Go-calibration scenario. `NA` if no such value exists.

gamma_nogo Optimal NoGo threshold: the smallest value in `gamma_grid` for which $\Pr(\mathrm{NoGo}) <$ `target_nogo` under the NoGo-calibration scenario. `NA` if no such value exists.

PrGo_opt Marginal $\Pr(g_{\mathrm{Go}} \geq \gamma_{\mathrm{go}})$ at the optimal γ_{go} under the Go-calibration scenario. `NA` if `gamma_go` is `NA`.

PrNoGo_opt Marginal $\Pr(g_{\mathrm{NoGo}} \geq \gamma_{\mathrm{nogo}})$ at the optimal γ_{nogo} under the NoGo-calibration scenario. `NA` if `gamma_nogo` is `NA`.

target_go The value of `target_go` supplied by the user.

target_nogo The value of `target_nogo` supplied by the user.

grid_results A data frame with columns `gamma_grid`, `PrGo_grid` (marginal Go probability under the Go-calibration scenario), and `PrNoGo_grid` (marginal NoGo probability under the NoGo-calibration scenario).

Examples

```
# Example 1: Controlled design, posterior probability
# gamma_go  : smallest gamma s.t. Pr(Go)   < 0.05 under Null (pi_t = pi_c = 0.15)
# gamma_nogo: largest  gamma s.t. Pr(NoGo) < 0.20 under Alt  (pi_t = 0.35, pi_c = 0.15)
getgamma1bin(
  prob = 'posterior', design = 'controlled',
  theta_TV = 0.20, theta_MAV = 0.05, theta_NULL = NULL,
  pi_t_go = 0.15, pi_c_go = 0.15,
  pi_t_nogo = 0.35, pi_c_nogo = 0.15,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 12L, n_c = 12L,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = NULL, m_t = NULL, m_c = NULL,
```

```
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL
)

# Example 2: Uncontrolled design, posterior probability
getgamma1bin(
  prob = 'posterior', design = 'uncontrolled',
  theta_TV = 0.20, theta_MAV = 0.05, theta_NULL = NULL,
  pi_t_go = 0.15, pi_c_go = NULL,
  pi_t_nogo = 0.35, pi_c_nogo = NULL,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 12L, n_c = 12L,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = 3L, m_t = NULL, m_c = NULL,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL
)

# Example 3: External design, posterior probability
getgamma1bin(
  prob = 'posterior', design = 'external',
  theta_TV = 0.20, theta_MAV = 0.05, theta_NULL = NULL,
  pi_t_go = 0.15, pi_c_go = 0.15,
  pi_t_nogo = 0.35, pi_c_nogo = 0.15,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 12L, n_c = 12L,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = NULL, m_t = NULL, m_c = NULL,
  ne_t = 15L, ne_c = 15L, ye_t = 6L, ye_c = 4L,
  alpha0e_t = 0.5, alpha0e_c = 0.5
)

# Example 4: Controlled design, predictive probability
getgamma1bin(
  prob = 'predictive', design = 'controlled',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 0.10,
  pi_t_go = 0.15, pi_c_go = 0.15,
  pi_t_nogo = 0.35, pi_c_nogo = 0.15,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 12L, n_c = 12L,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = NULL, m_t = 30L, m_c = 30L,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL
)

# Example 5: Uncontrolled design, predictive probability
getgamma1bin(
  prob = 'predictive', design = 'uncontrolled',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 0.10,
  pi_t_go = 0.15, pi_c_go = NULL,
  pi_t_nogo = 0.35, pi_c_nogo = NULL,
  target_go = 0.05, target_nogo = 0.20,
```

```
  n_t = 12L, n_c = 12L,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = 3L, m_t = 30L, m_c = 30L,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL
)

# Example 6: External design, predictive probability
getgamma1bin(
  prob = 'predictive', design = 'external',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 0.10,
  pi_t_go = 0.15, pi_c_go = 0.15,
  pi_t_nogo = 0.35, pi_c_nogo = 0.15,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 12L, n_c = 12L,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = NULL, m_t = 30L, m_c = 30L,
  ne_t = 15L, ne_c = 15L, ye_t = 6L, ye_c = 4L,
  alpha0e_t = 0.5, alpha0e_c = 0.5
)
```

`getgamma1cont` *Find Optimal Go/NoGo Thresholds for a Single Continuous Endpoint*

Description

Computes the optimal Go threshold γ_{go} and NoGo threshold γ_{nogo} for a single continuous endpoint by searching over a grid of candidate values. The two thresholds are calibrated independently under separate scenarios:

- γ_{go} is the **smallest** value in `gamma_grid` such that the marginal Go probability $\Pr(g_{\text{Go}} \geq \gamma_{\text{go}})$ is strictly less than `target_go` under the Go-calibration scenario (`mu_t_go`, `mu_c_go`, `sigma_t_go`, `sigma_c_go`); typically the Null scenario.
- γ_{nogo} is the **smallest** value in `gamma_grid` such that the marginal NoGo probability $\Pr(g_{\text{NoGo}} \geq \gamma_{\text{nogo}})$ is strictly less than `target_nogo` under the NoGo-calibration scenario (`mu_t_nogo`, `mu_c_nogo`, `sigma_t_nogo`, `sigma_c_nogo`); typically the Alternative scenario.

Here $g_{\text{Go}} = P(\theta > \theta_{\text{TV}} \mid \bar{y}_t, s_t, \bar{y}_c, s_c)$ and $g_{\text{NoGo}} = P(\theta \leq \theta_{\text{MAV}} \mid \bar{y}_t, s_t, \bar{y}_c, s_c)$ for `prob = 'posterior'`, consistent with the decision rule in `pbayesdecisionprob1cont`.

Usage

```
getgamma1cont(
  nsim,
  prob = "posterior",
  design = "controlled",
  prior = "vague",
  CalcMethod = "NI",
```

```
  theta_TV = NULL,
  theta_MAV = NULL,
  theta_NULL = NULL,
  nMC = NULL,
  mu_t_go,
  mu_c_go = NULL,
  sigma_t_go,
  sigma_c_go = NULL,
  mu_t_nogo,
  mu_c_nogo = NULL,
  sigma_t_nogo,
  sigma_c_nogo = NULL,
  target_go,
  target_nogo,
  n_t,
  n_c = NULL,
  m_t = NULL,
  m_c = NULL,
  kappa0_t = NULL,
  kappa0_c = NULL,
  nu0_t = NULL,
  nu0_c = NULL,
  mu0_t = NULL,
  mu0_c = NULL,
  sigma0_t = NULL,
  sigma0_c = NULL,
  r = NULL,
  ne_t = NULL,
  ne_c = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  bar_ye_t = NULL,
  bar_ye_c = NULL,
  se_t = NULL,
  se_c = NULL,
  gamma_grid = seq(0.01, 0.99, by = 0.01),
  seed
)
```

Arguments

nsim	A positive integer giving the number of Monte Carlo simulation replicates used to approximate the operating characteristics.
prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
prior	A character string specifying the prior distribution. Must be 'vague' or 'N-Inv-Chisq'.

CalcMethod	A character string specifying the computation method for pbayespostpred1cont. Must be 'NI', 'MC', or 'MM'.
theta_TV	A numeric scalar giving the Target Value (TV) threshold for the treatment effect. Required when prob = 'posterior'; set to NULL otherwise.
theta_MAV	A numeric scalar giving the Minimum Acceptable Value (MAV) threshold. Must satisfy theta_TV > theta_MAV. Required when prob = 'posterior'; set to NULL otherwise.
theta_NULL	A numeric scalar giving the null hypothesis threshold for the predictive probability. Required when prob = 'predictive'; set to NULL otherwise.
nMC	A positive integer giving the number of inner Monte Carlo draws passed to pbayespostpred1cont. Required when CalcMethod = 'MC'; set to NULL otherwise.
mu_t_go	A numeric scalar giving the true mean for the treatment group under the Go-calibration scenario (typically Null).
mu_c_go	A numeric scalar giving the true mean for the control group under the Go-calibration scenario. Set to NULL for design = 'uncontrolled'.
sigma_t_go	A positive numeric scalar giving the true standard deviation for the treatment group under the Go-calibration scenario.
sigma_c_go	A positive numeric scalar giving the true standard deviation for the control group under the Go-calibration scenario. Set to NULL for design = 'uncontrolled'.
mu_t_nogo	A numeric scalar giving the true mean for the treatment group under the NoGo-calibration scenario (typically Alternative).
mu_c_nogo	A numeric scalar giving the true mean for the control group under the NoGo-calibration scenario. Set to NULL for design = 'uncontrolled'.
sigma_t_nogo	A positive numeric scalar giving the true standard deviation for the treatment group under the NoGo-calibration scenario.
sigma_c_nogo	A positive numeric scalar giving the true standard deviation for the control group under the NoGo-calibration scenario. Set to NULL for design = 'uncontrolled'.
target_go	A numeric scalar in (0, 1) giving the upper bound on the marginal Go probability under the Go-calibration scenario. The optimal γ_{go} is the smallest grid value satisfying $\Pr(\mathrm{Go}) <$ target_go.
target_nogo	A numeric scalar in (0, 1) giving the upper bound on the marginal NoGo probability under the NoGo-calibration scenario. The optimal γ_{nogo} is the smallest grid value satisfying $\Pr(\mathrm{NoGo}) <$ target_nogo.
n_t	A positive integer giving the number of patients in the treatment group in the PoC trial.
n_c	A positive integer giving the number of patients in the control group in the PoC trial. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.
m_t	A positive integer giving the future sample size for the treatment group. Required when prob = 'predictive'; set to NULL otherwise.
m_c	A positive integer giving the future sample size for the control group. Required when prob = 'predictive'; set to NULL otherwise.

`kappa0_t` A positive numeric scalar giving the prior precision parameter for the treatment group under the N-Inv-Chi-squared prior. Required when `prior = 'N-Inv-Chisq'`; set to `NULL` otherwise.

`kappa0_c` A positive numeric scalar giving the prior precision parameter for the control group. Required when `prior = 'N-Inv-Chisq'` and `design != 'uncontrolled'`; set to `NULL` otherwise.

`nu0_t` A positive numeric scalar giving the prior degrees of freedom for the treatment group under the N-Inv-Chi-squared prior. Required when `prior = 'N-Inv-Chisq'`; set to `NULL` otherwise.

`nu0_c` A positive numeric scalar giving the prior degrees of freedom for the control group. Required when `prior = 'N-Inv-Chisq'` and `design != 'uncontrolled'`; set to `NULL` otherwise.

`mu0_t` A numeric scalar giving the prior mean for the treatment group under the N-Inv-Chi-squared prior. Required when `prior = 'N-Inv-Chisq'`; set to `NULL` otherwise.

`mu0_c` A numeric scalar giving the prior mean for the control group (or the fixed hypothetical control mean for uncontrolled design). Required when `prior = 'N-Inv-Chisq'`; set to `NULL` otherwise.

`sigma0_t` A positive numeric scalar giving the prior scale parameter for the treatment group under the N-Inv-Chi-squared prior. Required when `prior = 'N-Inv-Chisq'`; set to `NULL` otherwise.

`sigma0_c` A positive numeric scalar giving the prior scale parameter for the control group. Required when `prior = 'N-Inv-Chisq'` and `design != 'uncontrolled'`; set to `NULL` otherwise.

`r` A positive numeric scalar giving the variance ratio σ_c^2/σ_t^2 used for uncontrolled design. Required when `design = 'uncontrolled'`; set to `NULL` otherwise.

`ne_t` A positive integer giving the number of patients in the treatment group of the external data set. Required when `design = 'external'` and external treatment data are available; otherwise set to `NULL`.

`ne_c` A positive integer giving the number of patients in the control group of the external data set. Required when `design = 'external'` and external control data are available; otherwise set to `NULL`.

`alpha0e_t` A numeric scalar in `(0, 1]` giving the power prior weight for external treatment group data. Required when `design = 'external'` and `ne_t` is non-NULL; set to `NULL` otherwise.

`alpha0e_c` A numeric scalar in `(0, 1]` giving the power prior weight for external control group data. Required when `design = 'external'` and `ne_c` is non-NULL; set to `NULL` otherwise.

`bar_ye_t` A numeric scalar giving the sample mean of the external treatment group data. Required when `ne_t` is non-NULL; set to `NULL` otherwise.

`bar_ye_c` A numeric scalar giving the sample mean of the external control group data. Required when `ne_c` is non-NULL; set to `NULL` otherwise.

`se_t` A positive numeric scalar giving the sample standard deviation of the external treatment group data. Required when `ne_t` is non-NULL; set to `NULL` otherwise.

se_c	A positive numeric scalar giving the sample standard deviation of the external control group data. Required when ne_c is non-NULL; set to NULL otherwise.
gamma_grid	A numeric vector of candidate threshold values in (0, 1) to search over. Defaults to seq(0.01, 0.99, by = 0.01).
seed	A numeric scalar for reproducible random number generation. The Go-calibration simulation uses seed and the NoGo-calibration simulation uses seed + 1 to ensure independence between the two scenarios.

Details

The function uses a two-stage simulate-then-sweep strategy:

1. **Simulation**: nsim datasets are generated independently for each calibration scenario. For the Go-calibration scenario, standardised residuals are shifted by mu_t_go and mu_c_go; for the NoGo-calibration scenario, by mu_t_nogo and mu_c_nogo. pbayespostpred1cont is called once per scenario to obtain g_{Go} (lower.tail = FALSE at theta_TV) and g_{NoGo} (lower.tail = TRUE at theta_MAV).
2. **Gamma sweep**: Marginal probabilities are estimated as the proportion of simulated datasets satisfying the respective indicator: $g_{\mathrm{Go}} \geq \gamma$ for PrGo_grid, and $g_{\mathrm{NoGo}} \geq \gamma$ for PrNoGo_grid. Both are monotone non-increasing in γ.

The optimal γ_{go} and γ_{nogo} are each the *smallest* grid value crossing below the respective target probability.

Value

A list of class getgamma1cont with the following elements:

gamma_go Optimal Go threshold: the smallest value in gamma_grid for which $\Pr(\mathrm{Go}) <$ target_go under the Go-calibration scenario. NA if no such value exists.

gamma_nogo Optimal NoGo threshold: the smallest value in gamma_grid for which $\Pr(\mathrm{NoGo}) <$ target_nogo under the NoGo-calibration scenario. NA if no such value exists.

PrGo_opt Marginal $\Pr(g_{\mathrm{Go}} \geq \gamma_{\mathrm{go}})$ at the optimal γ_{go} under the Go-calibration scenario. NA if gamma_go is NA.

PrNoGo_opt Marginal $\Pr(g_{\mathrm{NoGo}} \geq \gamma_{\mathrm{nogo}})$ at the optimal γ_{nogo} under the NoGo-calibration scenario. NA if gamma_nogo is NA.

target_go The value of target_go supplied by the user.

target_nogo The value of target_nogo supplied by the user.

grid_results A data frame with columns gamma_grid, PrGo_grid (marginal Go probability under the Go-calibration scenario), and PrNoGo_grid (marginal NoGo probability under the NoGo-calibration scenario).

Examples

```
# Example 1: Controlled design, vague prior, posterior probability
# gamma_go  : smallest gamma s.t. Pr(Go)   < 0.05 under Null (mu_t = mu_c = 1.0)
# gamma_nogo: smallest gamma s.t. Pr(NoGo) < 0.20 under Alt  (mu_t = 2.5, mu_c = 1.0)
getgamma1cont(
```

```
  nsim = 1000L, prob = 'posterior', design = 'controlled',
  prior = 'vague', CalcMethod = 'MM',
  theta_TV = 1.5, theta_MAV = 0.0, theta_NULL = NULL, nMC = NULL,
  mu_t_go = 1.0, mu_c_go = 1.0, sigma_t_go = 2.0, sigma_c_go = 2.0,
  mu_t_nogo = 2.5, mu_c_nogo = 1.0, sigma_t_nogo = 2.0, sigma_c_nogo = 2.0,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 15L, n_c = 15L, m_t = NULL, m_c = NULL,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, mu0_c = NULL, sigma0_t = NULL, sigma0_c = NULL,
  r = NULL, ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  gamma_grid = seq(0.01, 0.99, by = 0.01), seed = 1L
)

# Example 2: Uncontrolled design, N-Inv-Chisq prior, posterior probability
getgamma1cont(
  nsim = 1000L, prob = 'posterior', design = 'uncontrolled',
  prior = 'N-Inv-Chisq', CalcMethod = 'NI',
  theta_TV = 1.0, theta_MAV = 0.0, theta_NULL = NULL, nMC = NULL,
  mu_t_go = 1.5, mu_c_go = NULL, sigma_t_go = 1.5, sigma_c_go = NULL,
  mu_t_nogo = 3.0, mu_c_nogo = NULL, sigma_t_nogo = 1.5, sigma_c_nogo = NULL,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 20L, n_c = NULL, m_t = NULL, m_c = NULL,
  kappa0_t = 2, kappa0_c = NULL, nu0_t = 5, nu0_c = NULL,
  mu0_t = 3.0, mu0_c = 1.5, sigma0_t = 1.5, sigma0_c = NULL,
  r = 1.0, ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  gamma_grid = seq(0.01, 0.99, by = 0.01), seed = 2L
)

# Example 3: External design, vague prior, posterior probability
getgamma1cont(
  nsim = 1000L, prob = 'posterior', design = 'external',
  prior = 'vague', CalcMethod = 'MM',
  theta_TV = 1.0, theta_MAV = 0.0, theta_NULL = NULL, nMC = NULL,
  mu_t_go = 1.0, mu_c_go = 1.0, sigma_t_go = 1.5, sigma_c_go = 1.5,
  mu_t_nogo = 2.5, mu_c_nogo = 1.0, sigma_t_nogo = 1.5, sigma_c_nogo = 1.5,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 15L, n_c = 15L, m_t = NULL, m_c = NULL,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, mu0_c = NULL, sigma0_t = NULL, sigma0_c = NULL,
  r = NULL, ne_t = NULL, ne_c = 20L, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = 0.0, se_t = NULL, se_c = 1.5,
  gamma_grid = seq(0.01, 0.99, by = 0.01), seed = 4L
)

# Example 4: Controlled design, vague prior, predictive probability
getgamma1cont(
  nsim = 1000L, prob = 'predictive', design = 'controlled',
  prior = 'vague', CalcMethod = 'MM',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 1.0, nMC = NULL,
  mu_t_go = 1.0, mu_c_go = 1.0, sigma_t_go = 2.0, sigma_c_go = 2.0,
  mu_t_nogo = 2.5, mu_c_nogo = 1.0, sigma_t_nogo = 2.0, sigma_c_nogo = 2.0,
```

```
  target_go = 0.05, target_nogo = 0.20,
  n_t = 15L, n_c = 15L, m_t = 50L, m_c = 50L,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, mu0_c = NULL, sigma0_t = NULL, sigma0_c = NULL,
  r = NULL, ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  gamma_grid = seq(0.01, 0.99, by = 0.01), seed = 3L
)

# Example 5: Uncontrolled design, vague prior, predictive probability
getgamma1cont(
  nsim = 1000L, prob = 'predictive', design = 'uncontrolled',
  prior = 'vague', CalcMethod = 'MM',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 1.0, nMC = NULL,
  mu_t_go = 1.0, mu_c_go = NULL, sigma_t_go = 2.0, sigma_c_go = NULL,
  mu_t_nogo = 2.5, mu_c_nogo = NULL, sigma_t_nogo = 2.0, sigma_c_nogo = NULL,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 15L, n_c = NULL, m_t = 50L, m_c = 50L,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, mu0_c = 0.0, sigma0_t = NULL, sigma0_c = NULL,
  r = 1.0, ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  gamma_grid = seq(0.01, 0.99, by = 0.01), seed = 5L
)

# Example 6: External design, vague prior, predictive probability
getgamma1cont(
  nsim = 1000L, prob = 'predictive', design = 'external',
  prior = 'vague', CalcMethod = 'MM',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 1.0, nMC = NULL,
  mu_t_go = 1.0, mu_c_go = 1.0, sigma_t_go = 1.5, sigma_c_go = 1.5,
  mu_t_nogo = 2.5, mu_c_nogo = 1.0, sigma_t_nogo = 1.5, sigma_c_nogo = 1.5,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 15L, n_c = 15L, m_t = 50L, m_c = 50L,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, mu0_c = NULL, sigma0_t = NULL, sigma0_c = NULL,
  r = NULL, ne_t = NULL, ne_c = 20L, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = 0.0, se_t = NULL, se_c = 1.5,
  gamma_grid = seq(0.01, 0.99, by = 0.01), seed = 6L
)
```

`getgamma2bin` *Find Optimal Go/NoGo Thresholds for Two Binary Endpoints*

Description

Computes the optimal Go threshold γ_{go} and NoGo threshold γ_{nogo} for two binary endpoints by searching over a two-dimensional grid of candidate value pairs. The two thresholds are calibrated independently under separate scenarios:

- γ_{go} is the **smallest** value in `gamma_go_grid` such that the worst-case marginal Go probability over all γ_{nogo} in `gamma_nogo_grid` is strictly less than `target_go` under the Go-calibration scenario (`pi_t1_go`, `pi_t2_go`, `rho_t_go`, `pi_c1_go`, `pi_c2_go`, `rho_c_go`); typically the Null scenario.
- γ_{nogo} is the **smallest** value in `gamma_nogo_grid` such that the worst-case marginal NoGo probability over all γ_{go} in `gamma_go_grid` is strictly less than `target_nogo` under the NoGo-calibration scenario (`pi_t1_nogo`, `pi_t2_nogo`, `rho_t_nogo`, `pi_c1_nogo`, `pi_c2_nogo`, `rho_c_nogo`); typically the Alternative scenario.

Usage

```
getgamma2bin(
  prob = "posterior",
  design = "controlled",
  GoRegions,
  NoGoRegions,
  pi_t1_go,
  pi_t2_go,
  rho_t_go,
  pi_c1_go = NULL,
  pi_c2_go = NULL,
  rho_c_go = NULL,
  pi_t1_nogo,
  pi_t2_nogo,
  rho_t_nogo,
  pi_c1_nogo = NULL,
  pi_c2_nogo = NULL,
  rho_c_nogo = NULL,
  target_go,
  target_nogo,
  n_t,
  n_c,
  a_t_00,
  a_t_01,
  a_t_10,
  a_t_11,
  a_c_00,
  a_c_01,
  a_c_10,
  a_c_11,
  theta_TV1 = NULL,
  theta_MAV1 = NULL,
  theta_TV2 = NULL,
  theta_MAV2 = NULL,
  theta_NULL1 = NULL,
  theta_NULL2 = NULL,
  m_t = NULL,
  m_c = NULL,
```

```
  z00 = NULL,
  z01 = NULL,
  z10 = NULL,
  z11 = NULL,
  xe_t_00 = NULL,
  xe_t_01 = NULL,
  xe_t_10 = NULL,
  xe_t_11 = NULL,
  xe_c_00 = NULL,
  xe_c_01 = NULL,
  xe_c_10 = NULL,
  xe_c_11 = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  nMC = 1000L,
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01)
)
```

Arguments

prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
GoRegions	An integer vector of region indices (subset of 1:9 for prob = 'posterior' or 1:4 for prob = 'predictive') that constitute the Go region.
NoGoRegions	An integer vector of region indices that constitute the NoGo region. Must be disjoint from GoRegions.
pi_t1_go	A numeric scalar in (0, 1) giving the true treatment response probability for Endpoint 1 under the Go-calibration scenario (typically Null).
pi_t2_go	A numeric scalar in (0, 1) giving the true treatment response probability for Endpoint 2 under the Go-calibration scenario.
rho_t_go	A numeric scalar giving the within-group correlation in the treatment group under the Go-calibration scenario.
pi_c1_go	A numeric scalar in (0, 1) giving the true control response probability for Endpoint 1 under the Go-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.
pi_c2_go	A numeric scalar in (0, 1) giving the true control response probability for Endpoint 2 under the Go-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.
rho_c_go	A numeric scalar giving the within-group correlation in the control group under the Go-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.
pi_t1_nogo	A numeric scalar in (0, 1) giving the true treatment response probability for Endpoint 1 under the NoGo-calibration scenario (typically Alternative).

pi_t2_nogo — A numeric scalar in (0, 1) giving the true treatment response probability for Endpoint 2 under the NoGo-calibration scenario.

rho_t_nogo — A numeric scalar giving the within-group correlation in the treatment group under the NoGo-calibration scenario.

pi_c1_nogo — A numeric scalar in (0, 1) giving the true control response probability for Endpoint 1 under the NoGo-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.

pi_c2_nogo — A numeric scalar in (0, 1) giving the true control response probability for Endpoint 2 under the NoGo-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.

rho_c_nogo — A numeric scalar giving the within-group correlation in the control group under the NoGo-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.

target_go — A numeric scalar in (0, 1) giving the upper bound on the worst-case marginal Go probability under the Go-calibration scenario. The optimal γ_{go} is the smallest grid value satisfying the constraint.

target_nogo — A numeric scalar in (0, 1) giving the upper bound on the worst-case marginal NoGo probability under the NoGo-calibration scenario. The optimal γ_{nogo} is the smallest grid value satisfying the constraint.

n_t — A positive integer giving the number of patients in the treatment group in the PoC trial.

n_c — A positive integer giving the number of patients in the control group in the PoC trial.

a_t_00 — A positive numeric scalar giving the Dirichlet prior parameter for the (0,0) response pattern in the treatment group.

a_t_01 — A positive numeric scalar; see a_t_00.

a_t_10 — A positive numeric scalar; see a_t_00.

a_t_11 — A positive numeric scalar; see a_t_00.

a_c_00 — A positive numeric scalar giving the Dirichlet prior parameter for the (0,0) response pattern in the control group.

a_c_01 — A positive numeric scalar; see a_c_00.

a_c_10 — A positive numeric scalar; see a_c_00.

a_c_11 — A positive numeric scalar; see a_c_00.

theta_TV1 — A numeric scalar giving the TV threshold for Endpoint 1. Required when prob = 'posterior'; otherwise set to NULL.

theta_MAV1 — A numeric scalar giving the MAV threshold for Endpoint 1. Required when prob = 'posterior'; otherwise set to NULL.

theta_TV2 — A numeric scalar giving the TV threshold for Endpoint 2. Required when prob = 'posterior'; otherwise set to NULL.

theta_MAV2 — A numeric scalar giving the MAV threshold for Endpoint 2. Required when prob = 'posterior'; otherwise set to NULL.

theta_NULL1 — A numeric scalar giving the null hypothesis threshold for Endpoint 1. Required when prob = 'predictive'; otherwise set to NULL.

theta_NULL2 — A numeric scalar giving the null hypothesis threshold for Endpoint 2. Required when prob = 'predictive'; otherwise set to NULL.

m_t — A positive integer giving the future sample size for the treatment group. Required when prob = 'predictive'; set to NULL otherwise.

m_c — A positive integer giving the future sample size for the control group. Required when prob = 'predictive'; set to NULL otherwise.

z00 — A non-negative integer giving the hypothetical control count for pattern (0,0). Required when design = 'uncontrolled'; otherwise set to NULL.

z01 — A non-negative integer; see z00.

z10 — A non-negative integer; see z00.

z11 — A non-negative integer; see z00.

xe_t_00 — A non-negative integer giving the external treatment group count for pattern (0,0). Required when design = 'external' and external treatment data are used; otherwise NULL.

xe_t_01 — A non-negative integer; see xe_t_00.

xe_t_10 — A non-negative integer; see xe_t_00.

xe_t_11 — A non-negative integer; see xe_t_00.

xe_c_00 — A non-negative integer giving the external control group count for pattern (0,0). Required when design = 'external' and external control data are used; otherwise NULL.

xe_c_01 — A non-negative integer; see xe_c_00.

xe_c_10 — A non-negative integer; see xe_c_00.

xe_c_11 — A non-negative integer; see xe_c_00.

alpha0e_t — A numeric scalar in (0, 1] giving the power prior weight for external treatment group data. Required when external treatment data are used; otherwise NULL.

alpha0e_c — A numeric scalar in (0, 1] giving the power prior weight for external control group data. Required when external control data are used; otherwise NULL.

nMC — A positive integer giving the number of Dirichlet draws used to evaluate region probabilities for each count combination in Stage 1. Default is 1000L.

gamma_go_grid — A numeric vector of candidate Go threshold values in (0, 1) to search over. Defaults to seq(0.01, 0.99, by = 0.01).

gamma_nogo_grid — A numeric vector of candidate NoGo threshold values in (0, 1) to search over. Defaults to seq(0.01, 0.99, by = 0.01).

Details

The function uses the same two-stage precompute-then-sweep strategy as pbayesdecisionprob2bin.

Stage 1 (precomputation): pbayespostpred2bin is called for every possible multinomial outcome combination (x_t, x_c) enumerated by allmultinom. The resulting region probability vector is summed over GoRegions and NoGoRegions to obtain $\hat{g}_{Go,ij}$ and $\hat{g}_{NoGo,ij}$. These are independent of the calibration scenario; only the multinomial weights differ.

Stage 2 (gamma sweep): For each pair $(\gamma_{\mathrm{go}}, \gamma_{\mathrm{nogo}})$ in the two-dimensional grid, operating characteristics are computed separately under each calibration scenario using the respective multinomial weights:

$$\Pr(\mathrm{Go}) = \sum_{i,j} w_{ij}^{(\mathrm{go})} \mathbf{1}[\hat{g}_{Go,ij} \geq \gamma_{\mathrm{go}},\ \hat{g}_{NoGo,ij} < \gamma_{\mathrm{nogo}}]$$

$$\Pr(\mathrm{NoGo}) = \sum_{i,j} w_{ij}^{(\mathrm{nogo})} \mathbf{1}[\hat{g}_{NoGo,ij} \geq \gamma_{\mathrm{nogo}},\ \hat{g}_{Go,ij} < \gamma_{\mathrm{go}}]$$

Stage 3 (optimal threshold selection): For each candidate γ_{go}, the worst-case $\Pr(\mathrm{Go})$ over all γ_{nogo} in `gamma_nogo_grid` is computed; the optimal γ_{go} is the *smallest* grid value for which this worst-case probability is less than `target_go`. Analogously, the optimal γ_{nogo} is the *smallest* grid value for which the worst-case $\Pr(\mathrm{NoGo})$ is less than `target_nogo`.

Value

A list of class `getgamma2bin` with the following elements:

gamma_go Optimal Go threshold: the smallest value in `gamma_go_grid` for which the marginal $\Pr(\mathrm{Go}) <$ `target_go` under the Go-calibration scenario. `NA` if no such value exists.

gamma_nogo Optimal NoGo threshold: the smallest value in `gamma_nogo_grid` for which the marginal $\Pr(\mathrm{NoGo}) <$ `target_nogo` under the NoGo-calibration scenario. `NA` if no such value exists.

PrGo_opt Marginal $\Pr(\mathrm{Go})$ at `gamma_go` under the Go-calibration scenario. `NA` if `gamma_go` is `NA`.

PrNoGo_opt Marginal $\Pr(\mathrm{NoGo})$ at `gamma_nogo` under the NoGo-calibration scenario. `NA` if `gamma_nogo` is `NA`.

target_go The value of `target_go` supplied by the user.

target_nogo The value of `target_nogo` supplied by the user.

grid_results A data frame with columns `gamma_grid`, `PrGo_grid` (marginal Go probability under the Go-calibration scenario), and `PrNoGo_grid` (marginal NoGo probability under the NoGo-calibration scenario).

Examples

```
# Example 1: Controlled design, posterior probability
# gamma_go  : smallest gamma_go   s.t. max_{gamma_nogo} Pr(Go)   < 0.05 under Null
# gamma_nogo: smallest gamma_nogo s.t. max_{gamma_go}   Pr(NoGo) < 0.20 under Alt

getgamma2bin(
  prob = 'posterior', design = 'controlled',
  GoRegions = 1L, NoGoRegions = 9L,
  pi_t1_go = 0.15, pi_t2_go = 0.20, rho_t_go = 0.0,
  pi_c1_go = 0.15, pi_c2_go = 0.20, rho_c_go = 0.0,
  pi_t1_nogo = 0.35, pi_t2_nogo = 0.40, rho_t_nogo = 0.0,
  pi_c1_nogo = 0.15, pi_c2_nogo = 0.20, rho_c_nogo = 0.0,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 7L, n_c = 7L,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
```

```
  theta_TV1   = 0.15, theta_MAV1 = 0.10,
  theta_TV2   = 0.15, theta_MAV2 = 0.10,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  m_t = NULL, m_c = NULL,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL,
  nMC = 100L,
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01)
)


# Example 2: Uncontrolled design, posterior probability

getgamma2bin(
  prob = 'posterior', design = 'uncontrolled',
  GoRegions = 1L, NoGoRegions = 9L,
  pi_t1_go = 0.15, pi_t2_go = 0.20, rho_t_go = 0.0,
  pi_c1_go = NULL, pi_c2_go = NULL, rho_c_go = NULL,
  pi_t1_nogo = 0.35, pi_t2_nogo = 0.40, rho_t_nogo = 0.0,
  pi_c1_nogo = NULL, pi_c2_nogo = NULL, rho_c_nogo = NULL,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 7L, n_c = 7L,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  theta_TV1   = 0.15, theta_MAV1 = 0.10,
  theta_TV2   = 0.15, theta_MAV2 = 0.10,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  m_t = NULL, m_c = NULL,
  z00 = 3L, z01 = 2L, z10 = 3L, z11 = 2L,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL,
  nMC = 100L,
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01)
)


# Example 3: External design, posterior probability

getgamma2bin(
  prob = 'posterior', design = 'external',
  GoRegions = 1L, NoGoRegions = 9L,
  pi_t1_go = 0.15, pi_t2_go = 0.20, rho_t_go = 0.0,
  pi_c1_go = 0.15, pi_c2_go = 0.20, rho_c_go = 0.0,
  pi_t1_nogo = 0.35, pi_t2_nogo = 0.40, rho_t_nogo = 0.0,
  pi_c1_nogo = 0.15, pi_c2_nogo = 0.20, rho_c_nogo = 0.0,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 7L, n_c = 7L,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
```

```
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  theta_TV1   = 0.15, theta_MAV1 = 0.10,
  theta_TV2   = 0.15, theta_MAV2 = 0.10,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  m_t = NULL, m_c = NULL,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = 3L, xe_c_01 = 2L, xe_c_10 = 3L, xe_c_11 = 2L,
  alpha0e_t = NULL, alpha0e_c = 0.5,
  nMC = 100L,
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01)
)


# Example 4: Controlled design, predictive probability

getgamma2bin(
  prob = 'predictive', design = 'controlled',
  GoRegions = 1L, NoGoRegions = 4L,
  pi_t1_go = 0.15, pi_t2_go = 0.20, rho_t_go = 0.0,
  pi_c1_go = 0.15, pi_c2_go = 0.20, rho_c_go = 0.0,
  pi_t1_nogo = 0.35, pi_t2_nogo = 0.40, rho_t_nogo = 0.0,
  pi_c1_nogo = 0.15, pi_c2_nogo = 0.20, rho_c_nogo = 0.0,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 7L, n_c = 7L,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  theta_TV1   = NULL, theta_MAV1 = NULL,
  theta_TV2   = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.10, theta_NULL2 = 0.10,
  m_t = 5L, m_c = 5L,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL,
  nMC = 100L,
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01)
)


# Example 5: Uncontrolled design, predictive probability

getgamma2bin(
  prob = 'predictive', design = 'uncontrolled',
  GoRegions = 1L, NoGoRegions = 4L,
  pi_t1_go = 0.15, pi_t2_go = 0.20, rho_t_go = 0.0,
  pi_c1_go = NULL, pi_c2_go = NULL, rho_c_go = NULL,
  pi_t1_nogo = 0.35, pi_t2_nogo = 0.40, rho_t_nogo = 0.0,
  pi_c1_nogo = NULL, pi_c2_nogo = NULL, rho_c_nogo = NULL,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 7L, n_c = 7L,
```

```
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  theta_TV1   = NULL, theta_MAV1 = NULL,
  theta_TV2   = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.10, theta_NULL2 = 0.10,
  m_t = 5L, m_c = 5L,
  z00 = 3L, z01 = 2L, z10 = 3L, z11 = 2L,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL,
  nMC = 100L,
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01)
)


# Example 6: External design, predictive probability

getgamma2bin(
  prob = 'predictive', design = 'external',
  GoRegions = 1L, NoGoRegions = 4L,
  pi_t1_go = 0.15, pi_t2_go = 0.20, rho_t_go = 0.0,
  pi_c1_go = 0.15, pi_c2_go = 0.20, rho_c_go = 0.0,
  pi_t1_nogo = 0.35, pi_t2_nogo = 0.40, rho_t_nogo = 0.0,
  pi_c1_nogo = 0.15, pi_c2_nogo = 0.20, rho_c_nogo = 0.0,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 7L, n_c = 7L,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  theta_TV1   = NULL, theta_MAV1 = NULL,
  theta_TV2   = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.10, theta_NULL2 = 0.10,
  m_t = 5L, m_c = 5L,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = 3L, xe_t_01 = 2L, xe_t_10 = 3L, xe_t_11 = 2L,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = 0.5, alpha0e_c = NULL,
  nMC = 100L,
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01)
)
```

`getgamma2cont` *Find Optimal Go/NoGo Thresholds for Two Continuous Endpoints*

Description

Computes the optimal Go threshold γ_{go} and NoGo threshold γ_{nogo} for two continuous endpoints by searching independently over candidate threshold grids. The two thresholds are calibrated

marginally under separate scenarios:

- γ_{go} is the **smallest** value in `gamma_go_grid` such that the marginal Go probability is strictly less than `target_go` under the Go-calibration scenario (`mu_t_go`, `Sigma_t_go`, `mu_c_go`, `Sigma_c_go`); typically the Null scenario.
- γ_{nogo} is the **smallest** value in `gamma_nogo_grid` such that the marginal NoGo probability is strictly less than `target_nogo` under the NoGo-calibration scenario (`mu_t_nogo`, `Sigma_t_nogo`, `mu_c_nogo`, `Sigma_c_nogo`); typically the Alternative scenario.

Usage

```
getgamma2cont(
  nsim = 10000L,
  prob = "posterior",
  design = "controlled",
  prior = "vague",
  GoRegions,
  NoGoRegions,
  mu_t_go,
  Sigma_t_go,
  mu_c_go = NULL,
  Sigma_c_go = NULL,
  mu_t_nogo,
  Sigma_t_nogo,
  mu_c_nogo = NULL,
  Sigma_c_nogo = NULL,
  target_go,
  target_nogo,
  n_t,
  n_c = NULL,
  theta_TV1 = NULL,
  theta_MAV1 = NULL,
  theta_TV2 = NULL,
  theta_MAV2 = NULL,
  theta_NULL1 = NULL,
  theta_NULL2 = NULL,
  m_t = NULL,
  m_c = NULL,
  kappa0_t = NULL,
  nu0_t = NULL,
  mu0_t = NULL,
  Lambda0_t = NULL,
  kappa0_c = NULL,
  nu0_c = NULL,
  mu0_c = NULL,
  Lambda0_c = NULL,
  r = NULL,
  ne_t = NULL,
  ne_c = NULL,
```

```
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  bar_ye_t = NULL,
  bar_ye_c = NULL,
  se_t = NULL,
  se_c = NULL,
  nMC = NULL,
  CalcMethod = "MC",
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01),
  seed
)
```

Arguments

nsim	A positive integer giving the number of Monte Carlo datasets to simulate per calibration scenario. Default is 10000L.
prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
prior	A character string specifying the prior distribution. Must be 'vague' or 'N-Inv-Wishart'.
GoRegions	An integer vector of region indices (subset of 1:9) that constitute the Go region. The 9 regions are defined by the cross-classification of the two treatment effects (θ_1, θ_2) relative to (TV1, MAV1) and (TV2, MAV2): region $k = (z_1 - 1) \times 3 + z_2$ where $z_i = 1$ if $\theta_i > TV_i$, $z_i = 2$ if $MAV_i < \theta_i \leq TV_i$, $z_i = 3$ if $\theta_i \leq MAV_i$.
NoGoRegions	An integer vector of region indices (subset of 1:9) that constitute the NoGo region. Must be disjoint from GoRegions.
mu_t_go	A length-2 numeric vector giving the true bivariate mean for the treatment group under the Go-calibration scenario (typically Null).
Sigma_t_go	A 2x2 positive-definite numeric matrix giving the true within-group covariance in the treatment group under the Go-calibration scenario.
mu_c_go	A length-2 numeric vector giving the true bivariate mean for the control group under the Go-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.
Sigma_c_go	A 2x2 positive-definite numeric matrix giving the true within-group covariance in the control group under the Go-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.
mu_t_nogo	A length-2 numeric vector giving the true bivariate mean for the treatment group under the NoGo-calibration scenario (typically Alternative).
Sigma_t_nogo	A 2x2 positive-definite numeric matrix giving the true within-group covariance in the treatment group under the NoGo-calibration scenario.
mu_c_nogo	A length-2 numeric vector giving the true bivariate mean for the control group under the NoGo-calibration scenario. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.

`Sigma_c_nogo` A 2x2 positive-definite numeric matrix giving the true within-group covariance in the control group under the NoGo-calibration scenario. Required for `design = 'controlled'` or `'external'`; set to `NULL` for `design = 'uncontrolled'`.

`target_go` A numeric scalar in `(0, 1)` giving the upper bound on the marginal Go probability under the Go-calibration scenario. The optimal γ_{go} is the smallest grid value satisfying the constraint.

`target_nogo` A numeric scalar in `(0, 1)` giving the upper bound on the marginal NoGo probability under the NoGo-calibration scenario. The optimal γ_{nogo} is the smallest grid value satisfying the constraint.

`n_t` A positive integer giving the number of patients in the treatment group in the PoC trial.

`n_c` A positive integer giving the number of patients in the control group in the PoC trial. Set to `NULL` for `design = 'uncontrolled'`.

`theta_TV1` A numeric scalar giving the TV threshold for Endpoint 1. Required when `prob = 'posterior'`; otherwise set to `NULL`.

`theta_MAV1` A numeric scalar giving the MAV threshold for Endpoint 1. Required when `prob = 'posterior'`; otherwise set to `NULL`.

`theta_TV2` A numeric scalar giving the TV threshold for Endpoint 2. Required when `prob = 'posterior'`; otherwise set to `NULL`.

`theta_MAV2` A numeric scalar giving the MAV threshold for Endpoint 2. Required when `prob = 'posterior'`; otherwise set to `NULL`.

`theta_NULL1` A numeric scalar giving the null hypothesis threshold for Endpoint 1. Required when `prob = 'predictive'`; otherwise set to `NULL`.

`theta_NULL2` A numeric scalar giving the null hypothesis threshold for Endpoint 2. Required when `prob = 'predictive'`; otherwise set to `NULL`.

`m_t` A positive integer giving the future sample size for the treatment group. Required when `prob = 'predictive'`; set to `NULL` otherwise.

`m_c` A positive integer giving the future sample size for the control group. Required when `prob = 'predictive'`; set to `NULL` otherwise.

`kappa0_t` Positive numeric scalar. NIW prior hyperparameter κ_{01} for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise `NULL`.

`nu0_t` Positive numeric scalar. NIW prior degrees of freedom ν_{01} for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise `NULL`.

`mu0_t` Length-2 numeric vector. NIW prior mean μ_{01} for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise `NULL`.

`Lambda0_t` A 2x2 positive-definite numeric matrix. NIW prior scale matrix Λ_{01} for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise `NULL`.

`kappa0_c` Positive numeric scalar; see `kappa0_t`. For the control group.

`nu0_c` Positive numeric scalar; see `nu0_t`. For the control group.

`mu0_c` Length-2 numeric vector; see `mu0_t`. For the control group. May be required for the vague prior uncontrolled design; see `pbayesdecisionprob2cont`.

`Lambda0_c` A 2x2 matrix; see `Lambda0_t`. For the control group.

r	A positive numeric scalar giving the power prior weight for the control group when design = 'uncontrolled' and prior = 'vague'. Otherwise NULL.
ne_t	A positive integer giving the external treatment sample size. Required when design = 'external' and external treatment data are used; otherwise NULL.
ne_c	A positive integer giving the external control sample size. Required when design = 'external' and external control data are used; otherwise NULL.
alpha0e_t	A numeric scalar in (0, 1] giving the power prior weight for external treatment data. Required when external treatment data are used; otherwise NULL.
alpha0e_c	A numeric scalar in (0, 1] giving the power prior weight for external control data. Required when external control data are used; otherwise NULL.
bar_ye_t	A length-2 numeric vector. External treatment sample mean. Required when external treatment data are used; otherwise NULL.
bar_ye_c	A length-2 numeric vector. External control sample mean. Required when external control data are used; otherwise NULL.
se_t	A 2x2 numeric matrix. External treatment sum-of-squares matrix. Required when external treatment data are used; otherwise NULL.
se_c	A 2x2 numeric matrix. External control sum-of-squares matrix. Required when external control data are used; otherwise NULL.
nMC	A positive integer giving the number of Monte Carlo draws passed to pbayespostpred2cont. Required when CalcMethod = 'MC'. May be set to NULL when CalcMethod = 'MM' and $\nu_k > 4$; if CalcMethod = 'MM' but $\nu_k \leq 4$ causes a fallback to MC, nMC must be a positive integer. Default is NULL.
CalcMethod	A character string specifying the computation method passed to pbayespostpred2cont. Must be 'MC' (default) or 'MM'.
gamma_go_grid	A numeric vector of candidate Go threshold values in (0, 1) to search over. Defaults to seq(0.01, 0.99, by = 0.01).
gamma_nogo_grid	A numeric vector of candidate NoGo threshold values in (0, 1) to search over. Defaults to seq(0.01, 0.99, by = 0.01).
seed	A numeric scalar for reproducible random number generation. The Go-calibration simulation uses seed and the NoGo-calibration simulation uses seed + 1 to ensure independence between the two scenarios.

Details

The function uses a two-stage simulate-then-sweep strategy:

Stage 1 (simulation and precomputation): nsim bivariate datasets are generated independently for each calibration scenario. For the Go-calibration scenario, datasets are drawn from $N_2(\mu_{t,\text{go}}, \Sigma_{t,\text{go}})$ (and $N_2(\mu_{c,\text{go}}, \Sigma_{c,\text{go}})$ for controlled/external designs); for the NoGo-calibration scenario, the corresponding _nogo parameters are used. pbayespostpred2cont is called once per scenario in vectorised mode to return an $nsim \times 9$ matrix of region probabilities. The probabilities are summed over GoRegions (for the Go scenario) and NoGoRegions (for the NoGo scenario) to obtain $\hat{g}_{Go,i}$ and $\hat{g}_{NoGo,i}$, independent of the decision thresholds.

Stage 2 (gamma sweep): For each pair $(\gamma_{\text{go}}, \gamma_{\text{nogo}})$ in the two-dimensional grid, operating characteristics are computed separately under each calibration scenario:

$$\Pr(\text{Go}) = \frac{1}{n_{\text{sim}}} \sum_{i=1}^{n_{\text{sim}}} \mathbf{1}[\hat{g}_{Go,i} \geq \gamma_{\text{go}},\ \hat{g}_{NoGo,i} < \gamma_{\text{nogo}}]$$

$$\Pr(\text{NoGo}) = \frac{1}{n_{\text{sim}}} \sum_{i=1}^{n_{\text{sim}}} \mathbf{1}[\hat{g}_{NoGo,i} \geq \gamma_{\text{nogo}},\ \hat{g}_{Go,i} < \gamma_{\text{go}}]$$

Stage 3 (optimal threshold selection): For each candidate γ_{go}, the worst-case $\Pr(\text{Go})$ over all γ_{nogo} in `gamma_nogo_grid` is computed; the optimal γ_{go} is the *smallest* grid value for which this worst-case probability is less than `target_go`. Analogously, the optimal γ_{nogo} is the *smallest* grid value for which the worst-case $\Pr(\text{NoGo})$ is less than `target_nogo`.

Value

A list of class `getgamma2cont` with the following elements:

gamma_go Optimal Go threshold: the smallest value in `gamma_go_grid` for which the marginal $\Pr(\text{Go}) <$ `target_go` under the Go-calibration scenario. `NA` if no such value exists.

gamma_nogo Optimal NoGo threshold: the smallest value in `gamma_nogo_grid` for which the marginal $\Pr(\text{NoGo}) <$ `target_nogo` under the NoGo-calibration scenario. `NA` if no such value exists.

PrGo_opt Marginal $\Pr(\text{Go})$ at `gamma_go` under the Go-calibration scenario. `NA` if `gamma_go` is `NA`.

PrNoGo_opt Marginal $\Pr(\text{NoGo})$ at `gamma_nogo` under the NoGo-calibration scenario. `NA` if `gamma_nogo` is `NA`.

target_go The value of `target_go` supplied by the user.

target_nogo The value of `target_nogo` supplied by the user.

grid_results A data frame with columns `gamma_grid`, `PrGo_grid` (marginal Go probability under the Go-calibration scenario), and `PrNoGo_grid` (marginal NoGo probability under the NoGo-calibration scenario).

Examples

```
# Example 1: Controlled design, posterior probability, vague prior
# gamma_go  : smallest gamma_go   s.t. max_{gamma_nogo} Pr(Go)   < 0.05 under Null
# gamma_nogo: smallest gamma_nogo s.t. max_{gamma_go}   Pr(NoGo) < 0.20 under Alt

Sigma_null <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
Sigma_alt  <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
getgamma2cont(
  nsim = 1000L, prob = 'posterior', design = 'controlled',
  prior = 'vague',
  GoRegions = 1L, NoGoRegions = 9L,
  mu_t_go = c(-5.0, 0.0), Sigma_t_go = Sigma_null,
  mu_c_go = c(-10.0, -1.0), Sigma_c_go = Sigma_null,
  mu_t_nogo = c(5.0, 1.0), Sigma_t_nogo = Sigma_alt,
  mu_c_nogo = c(-10.0, -1.0), Sigma_c_nogo = Sigma_alt,
```

```
  target_go = 0.05, target_nogo = 0.20,
  n_t = 30L, n_c = 30L,
  theta_TV1 = 10.0, theta_MAV1 = 5.0,
  theta_TV2 = 2.0,  theta_MAV2 = 1.0,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  m_t = NULL, m_c = NULL,
  kappa0_t = NULL, nu0_t = NULL, mu0_t = NULL, Lambda0_t = NULL,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = NULL, Lambda0_c = NULL,
  r = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 500L, CalcMethod = 'MC',
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01),
  seed = 1L
)


# Example 2: Uncontrolled design, posterior probability, vague prior

Sigma_null <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
Sigma_alt  <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
getgamma2cont(
  nsim = 1000L, prob = 'posterior', design = 'uncontrolled',
  prior = 'vague',
  GoRegions = 1L, NoGoRegions = 9L,
  mu_t_go = c(-5.0, 0.0), Sigma_t_go = Sigma_null,
  mu_c_go = NULL, Sigma_c_go = NULL,
  mu_t_nogo = c(5.0, 1.0), Sigma_t_nogo = Sigma_alt,
  mu_c_nogo = NULL, Sigma_c_nogo = NULL,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 30L, n_c = NULL,
  theta_TV1 = 10.0, theta_MAV1 = 5.0,
  theta_TV2 = 2.0,  theta_MAV2 = 1.0,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  m_t = NULL, m_c = NULL,
  kappa0_t = NULL, nu0_t = NULL, mu0_t = NULL, Lambda0_t = NULL,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = c(-10.0, -1.0), Lambda0_c = NULL,
  r = 1.0,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = NULL, CalcMethod = 'MM',
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01),
  seed = 1L
)


# Example 3: External design (control only), posterior probability, NIW prior

Sigma  <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
Lambda <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
se_c   <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
```

```
getgamma2cont(
  nsim = 1000L, prob = 'posterior', design = 'external',
  prior = 'N-Inv-Wishart',
  GoRegions = 1L, NoGoRegions = 9L,
  mu_t_go = c(-5.0, 0.0), Sigma_t_go = Sigma,
  mu_c_go = c(-10.0, -1.0), Sigma_c_go = Sigma,
  mu_t_nogo = c(5.0, 1.0), Sigma_t_nogo = Sigma,
  mu_c_nogo = c(-10.0, -1.0), Sigma_c_nogo = Sigma,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 30L, n_c = 30L,
  theta_TV1 = 10.0, theta_MAV1 = 5.0,
  theta_TV2 = 2.0,  theta_MAV2 = 1.0,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  m_t = NULL, m_c = NULL,
  kappa0_t = 0.1, nu0_t = 4.0, mu0_t = c(0.0, 1.0),  Lambda0_t = Lambda,
  kappa0_c = 0.1, nu0_c = 4.0, mu0_c = c(-10.0, -1.0), Lambda0_c = Lambda,
  r = NULL,
  ne_t = NULL, ne_c = 10L, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = c(-10.0, -1.0), se_t = NULL, se_c = se_c,
  nMC = 500L, CalcMethod = 'MC',
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01),
  seed = 1L
)


# Example 4: Controlled design, predictive probability, vague prior

Sigma_null <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
Sigma_alt  <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
getgamma2cont(
  nsim = 1000L, prob = 'predictive', design = 'controlled',
  prior = 'vague',
  GoRegions = 1L, NoGoRegions = 4L,
  mu_t_go = c(-5.0, 0.0), Sigma_t_go = Sigma_null,
  mu_c_go = c(-10.0, -1.0), Sigma_c_go = Sigma_null,
  mu_t_nogo = c(5.0, 1.0), Sigma_t_nogo = Sigma_alt,
  mu_c_nogo = c(-10.0, -1.0), Sigma_c_nogo = Sigma_alt,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 30L, n_c = 30L,
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 5.0, theta_NULL2 = 1.0,
  m_t = 100L, m_c = 100L,
  kappa0_t = NULL, nu0_t = NULL, mu0_t = NULL, Lambda0_t = NULL,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = NULL, Lambda0_c = NULL,
  r = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 500L, CalcMethod = 'MC',
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01),
  seed = 1L
```

```
)


# Example 5: Uncontrolled design, predictive probability, vague prior

Sigma_null <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
Sigma_alt  <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
getgamma2cont(
  nsim = 1000L, prob = 'predictive', design = 'uncontrolled',
  prior = 'vague',
  GoRegions = 1L, NoGoRegions = 4L,
  mu_t_go = c(-5.0, 0.0), Sigma_t_go = Sigma_null,
  mu_c_go = NULL, Sigma_c_go = NULL,
  mu_t_nogo = c(5.0, 1.0), Sigma_t_nogo = Sigma_alt,
  mu_c_nogo = NULL, Sigma_c_nogo = NULL,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 30L, n_c = NULL,
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 5.0, theta_NULL2 = 1.0,
  m_t = 100L, m_c = 100L,
  kappa0_t = NULL, nu0_t = NULL, mu0_t = NULL, Lambda0_t = NULL,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = c(-10.0, -1.0), Lambda0_c = NULL,
  r = 1.0,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 500L, CalcMethod = 'MC',
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01),
  seed = 1L
)


# Example 6: External design (control only), predictive probability, NIW prior

Sigma  <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
Lambda <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
se_c   <- matrix(c(6400.0, 15.0, 15.0, 36.0), 2, 2)
getgamma2cont(
  nsim = 1000L, prob = 'predictive', design = 'external',
  prior = 'N-Inv-Wishart',
  GoRegions = 1L, NoGoRegions = 4L,
  mu_t_go = c(-5.0, 0.0), Sigma_t_go = Sigma,
  mu_c_go = c(-10.0, -1.0), Sigma_c_go = Sigma,
  mu_t_nogo = c(5.0, 1.0), Sigma_t_nogo = Sigma,
  mu_c_nogo = c(-10.0, -1.0), Sigma_c_nogo = Sigma,
  target_go = 0.05, target_nogo = 0.20,
  n_t = 30L, n_c = 30L,
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 5.0, theta_NULL2 = 1.0,
  m_t = 100L, m_c = 100L,
  kappa0_t = 0.1, nu0_t = 4.0, mu0_t = c(0.0, 1.0),  Lambda0_t = Lambda,
```

```
  kappa0_c = 0.1, nu0_c = 4.0, mu0_c = c(-10.0, -1.0), Lambda0_c = Lambda,
  r = NULL,
  ne_t = NULL, ne_c = 10L, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = c(-10.0, -1.0), se_t = NULL, se_c = se_c,
  nMC = 500L, CalcMethod = 'MC',
  gamma_go_grid = seq(0.01, 0.99, by = 0.01),
  gamma_nogo_grid = seq(0.01, 0.99, by = 0.01),
  seed = 1L
)
```

`getjointbin` *Compute Joint Bivariate Binary Probabilities from Marginal Rates and Correlation*

Description

Converts the parameterisation $(\pi_{j1}, \pi_{j2}, \rho_j)$ into the four-cell joint probability vector $(p_{j,00}, p_{j,01}, p_{j,10}, p_{j,11})$ for a bivariate binary outcome in group $j \in \{t, c\}$. The conversion uses the standard moment-matching identity for the Pearson correlation of two Bernoulli variables, and checks that the requested correlation is feasible for the supplied marginal rates.

Usage

```
getjointbin(pi1, pi2, rho, tol = 1e-10)
```

Arguments

`pi1`	A single numeric value in `(0, 1)` giving the marginal response probability for Endpoint 1 (π_{j1}) in group j.
`pi2`	A single numeric value in `(0, 1)` giving the marginal response probability for Endpoint 2 (π_{j2}) in group j.
`rho`	A single numeric value giving the Pearson correlation (ρ_j) between Endpoint 1 and Endpoint 2 in group j. Must lie within the feasible range $[\rho_{\min}, \rho_{\max}]$ determined by `pi1` and `pi2` (see Details). Use `rho = 0` for independence.
`tol`	A single positive numeric value specifying the tolerance used when checking whether `rho` lies within its feasible range and whether the resulting probabilities are non-negative. Default is `1e-10`.

Details

For a bivariate binary outcome (Y_{i1}, Y_{i2}) of patient i ($i = 1, \ldots, n_j$) in group j with marginal success probabilities $\pi_{j1} = \Pr(Y_{i1} = 1)$ and $\pi_{j2} = \Pr(Y_{i2} = 1)$, the Pearson correlation is

$$\rho_j = \frac{p_{j,11} - \pi_{j1}\pi_{j2}}{\sqrt{\pi_{j1}(1-\pi_{j1})\,\pi_{j2}(1-\pi_{j2})}}.$$

Solving for $p_{j,11}$ gives

$$p_{j,11} = \rho_j \sqrt{\pi_{j1}(1-\pi_{j1})\,\pi_{j2}(1-\pi_{j2})} + \pi_{j1}\pi_{j2},$$

from which the remaining probabilities follow:

$$p_{j,10} = \pi_{j1} - p_{j,11}, \quad p_{j,01} = \pi_{j2} - p_{j,11}, \quad p_{j,00} = 1 - p_{j,10} - p_{j,01} - p_{j,11}.$$

For all four cell probabilities to lie in `[0, 1]`, the correlation must satisfy

$$\rho_{\min} = \frac{\max(0,\ \pi_{j1}+\pi_{j2}-1) - \pi_{j1}\pi_{j2}}{\sqrt{\pi_{j1}(1-\pi_{j1})\,\pi_{j2}(1-\pi_{j2})}} \ \le\ \rho_j\ \le\ \frac{\min(\pi_{j1},\pi_{j2}) - \pi_{j1}\pi_{j2}}{\sqrt{\pi_{j1}(1-\pi_{j1})\,\pi_{j2}(1-\pi_{j2})}} = \rho_{\max}.$$

The function raises an error if `rho` falls outside this range (subject to `tol`).

Value

A named numeric vector of length 4 with elements `p00`, `p01`, `p10`, `p11`, where `p_lm` = `Pr(Endpoint 1 = l, Endpoint 2 = m)` for $l, m \in \{0, 1\}$. All elements are non-negative and sum to 1.

Examples

```
# Example 1: Independent endpoints (rho = 0)
getjointbin(pi1 = 0.3, pi2 = 0.4, rho = 0.0)

# Example 2: Positive correlation
getjointbin(pi1 = 0.3, pi2 = 0.4, rho = 0.3)

# Example 3: Negative correlation
getjointbin(pi1 = 0.3, pi2 = 0.4, rho = -0.2)

# Example 4: Verify cell probabilities sum to 1
p <- getjointbin(pi1 = 0.25, pi2 = 0.35, rho = 0.1)
sum(p)  # Should be 1

# Example 5: Verify marginal recovery
p <- getjointbin(pi1 = 0.25, pi2 = 0.35, rho = 0.1)
p["p10"] + p["p11"]  # Should equal pi1 = 0.25
p["p01"] + p["p11"]  # Should equal pi2 = 0.35
```

`pbayesdecisionprob1bin` *Go/NoGo/Gray Decision Probabilities for a Clinical Trial with a Single Binary Endpoint*

Description

Evaluates operating characteristics (Go, NoGo, Gray probabilities) for binary-outcome clinical trials under the Bayesian framework by enumerating all possible trial outcomes. The function supports controlled, uncontrolled, and external designs.

Usage

```
pbayesdecisionprob1bin(
  prob = "posterior",
  design = "controlled",
  theta_TV = NULL,
  theta_MAV = NULL,
  theta_NULL = NULL,
  gamma_go,
  gamma_nogo,
  pi_t,
  pi_c = NULL,
  n_t,
  n_c,
  a_t,
  a_c,
  b_t,
  b_c,
  z = NULL,
  m_t = NULL,
  m_c = NULL,
  ne_t = NULL,
  ne_c = NULL,
  ye_t = NULL,
  ye_c = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  error_if_Miss = TRUE,
  Gray_inc_Miss = FALSE
)
```

Arguments

prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
theta_TV	A numeric scalar giving the target value (TV) threshold used for the Go decision when prob = 'posterior'. Set to NULL when prob = 'predictive'.
theta_MAV	A numeric scalar giving the minimum acceptable value (MAV) threshold used for the NoGo decision when prob = 'posterior'. Must satisfy theta_TV > theta_MAV. Set to NULL when prob = 'predictive'.
theta_NULL	A numeric scalar giving the null hypothesis threshold used for both Go and NoGo decisions when prob = 'predictive'. Set to NULL when prob = 'posterior'.
gamma_go	A numeric scalar in (0, 1) giving the minimum posterior or predictive probability required for a Go decision.
gamma_nogo	A numeric scalar in (0, 1) giving the minimum posterior or predictive probability required for a NoGo decision. No ordering constraint on gamma_go and

gamma_nogo is imposed, though their combination determines the frequency of Miss outcomes.

pi_t — A numeric value or vector giving the true response probability(s) for the treatment group used to evaluate operating characteristics. Each element must be in (0, 1).

pi_c — A numeric value or vector giving the true response probability(s) for the control group. For design = 'uncontrolled', this parameter is not used in calculations but must be supplied; it is excluded from the output. When supplied as a vector, must have the same length as pi_t.

n_t — A positive integer giving the number of patients in the treatment group in the proof-of-concept (PoC) trial.

n_c — A positive integer giving the number of patients in the control group in the PoC trial. For design = 'uncontrolled', this is the hypothetical control sample size (required for consistency with other designs).

a_t — A positive numeric scalar giving the first shape parameter (alpha) of the prior Beta distribution for the treatment group.

a_c — A positive numeric scalar giving the first shape parameter (alpha) of the prior Beta distribution for the control group.

b_t — A positive numeric scalar giving the second shape parameter (beta) of the prior Beta distribution for the treatment group.

b_c — A positive numeric scalar giving the second shape parameter (beta) of the prior Beta distribution for the control group.

z — A non-negative integer giving the hypothetical number of responders in the control group. Required when design = 'uncontrolled'; otherwise set to NULL. When used, y_c should be NULL.

m_t — A positive integer giving the number of patients in the treatment group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

m_c — A positive integer giving the number of patients in the control group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

ne_t — A positive integer giving the number of patients in the treatment group of the external data set. Required when design = 'external' and external treatment data are available; otherwise set to NULL.

ne_c — A positive integer giving the number of patients in the control group of the external data set. Required when design = 'external' and external control data are available; otherwise set to NULL.

ye_t — A non-negative integer giving the number of responders in the treatment group of the external data set. Required when design = 'external'; otherwise set to NULL.

ye_c — A non-negative integer giving the number of responders in the control group of the external data set. Required when design = 'external'; otherwise set to NULL.

alpha0e_t — A numeric scalar in (0, 1] giving the power prior weight for the treatment group. Required when design = 'external'; otherwise NULL.

`alpha0e_c` A numeric scalar in `(0, 1]` giving the power prior weight for the control group. Required when `design = 'external'`; otherwise `NULL`.

`error_if_Miss` A logical scalar; if `TRUE` (default), the function stops with an error if Miss probability > 0, prompting reconsideration of thresholds.

`Gray_inc_Miss` A logical scalar; if `TRUE`, Miss probability is added to Gray probability. If `FALSE` (default), Miss is reported separately. Active only when `error_if_Miss = FALSE`.

Details

Operating characteristics are computed by exact enumeration:

1. All possible outcome pairs (y_t, y_c) with $y_t \in \{0, \ldots, n_t\}$ and $y_c \in \{0, \ldots, n_c\}$ (or fixed at z for uncontrolled) are evaluated.
2. For each pair, `pbayespostpred1bin` computes the posterior or predictive probability at both thresholds (TV/MAV or NULL).
3. Outcomes are classified into Go, NoGo, Miss, or Gray:
   - **Go**: $P(\text{Go}) \geq \gamma_1$ AND $P(\text{NoGo}) < \gamma_2$
   - **NoGo**: $P(\text{Go}) < \gamma_1$ AND $P(\text{NoGo}) \geq \gamma_2$
   - **Miss**: both Go and NoGo criteria met simultaneously
   - **Gray**: neither Go nor NoGo criteria met
4. Each outcome is weighted by its binomial probability under the true rates.

Value

A data frame with one row per `pi_t` scenario and columns:

pi_t True treatment response probability.

pi_c True control response probability (omitted for uncontrolled design).

Go Probability of making a Go decision.

Gray Probability of making a Gray (inconclusive) decision.

NoGo Probability of making a NoGo decision.

Miss (Optional) Probability where Go and NoGo criteria are simultaneously met. Included when `error_if_Miss = FALSE` and `Gray_inc_Miss = FALSE`.

The returned object has S3 class `pbayesdecisionprob1bin` with an associated `print` method.

Examples

```
# Example 1: Controlled design with posterior probability
pbayesdecisionprob1bin(
  prob = 'posterior', design = 'controlled',
  theta_TV = 0.4, theta_MAV = 0.2, theta_NULL = NULL,
  gamma_go = 0.8, gamma_nogo = 0.2,
  pi_t = c(0.2, 0.4, 0.6, 0.8), pi_c = rep(0.2, 4),
  n_t = 12, n_c = 12,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
```

```
  z = NULL, m_t = NULL, m_c = NULL,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE
)

# Example 2: Uncontrolled design with hypothetical control
pbayesdecisionprob1bin(
  prob = 'posterior', design = 'uncontrolled',
  theta_TV = 0.30, theta_MAV = 0.15, theta_NULL = NULL,
  gamma_go = 0.75, gamma_nogo = 0.25,
  pi_t = c(0.3, 0.5, 0.7), pi_c = NULL,
  n_t = 15, n_c = 15,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = 5, m_t = NULL, m_c = NULL,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE
)

# Example 3: External design with 50 percent power prior borrowing
pbayesdecisionprob1bin(
  prob = 'posterior', design = 'external',
  theta_TV = 0.4, theta_MAV = 0.2, theta_NULL = NULL,
  gamma_go = 0.8, gamma_nogo = 0.2,
  pi_t = c(0.2, 0.4, 0.6, 0.8), pi_c = rep(0.2, 4),
  n_t = 12, n_c = 12,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = NULL, m_t = NULL, m_c = NULL,
  ne_t = 15, ne_c = 15, ye_t = 6, ye_c = 4, alpha0e_t = 0.5, alpha0e_c = 0.5,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE
)

# Example 4: Posterior predictive probability for controlled design
pbayesdecisionprob1bin(
  prob = 'predictive', design = 'controlled',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 0,
  gamma_go = 0.9, gamma_nogo = 0.3,
  pi_t = c(0.2, 0.4, 0.6, 0.8), pi_c = rep(0.2, 4),
  n_t = 12, n_c = 12,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = NULL, m_t = 30, m_c = 30,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE
)

# Example 5: Uncontrolled design with posterior predictive probability
pbayesdecisionprob1bin(
  prob = 'predictive', design = 'uncontrolled',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 0,
  gamma_go = 0.75, gamma_nogo = 0.25,
  pi_t = c(0.3, 0.5, 0.7), pi_c = NULL,
  n_t = 15, n_c = 15,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = 5, m_t = 30, m_c = 30,
```

```
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE
)

# Example 6: External design with posterior predictive probability
pbayesdecisionprob1bin(
  prob = 'predictive', design = 'external',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 0,
  gamma_go = 0.9, gamma_nogo = 0.3,
  pi_t = c(0.2, 0.4, 0.6, 0.8), pi_c = rep(0.2, 4),
  n_t = 12, n_c = 12,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  z = NULL, m_t = 30, m_c = 30,
  ne_t = 15, ne_c = 15, ye_t = 6, ye_c = 4, alpha0e_t = 0.5, alpha0e_c = 0.5,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE
)
```

`pbayesdecisionprob1cont` *Bayesian Go/NoGo/Gray Decision Probabilities for Single Continuous Endpoint*

Description

Evaluates Go/NoGo/Gray decision probabilities for a single continuous endpoint via Monte Carlo simulation. Supports controlled (parallel control), uncontrolled (single-arm with informative priors), and external (power prior borrowing) designs with both posterior and predictive probability approaches.

Usage

```
pbayesdecisionprob1cont(
  nsim,
  prob,
  design,
  prior,
  CalcMethod,
  theta_TV = NULL,
  theta_MAV = NULL,
  theta_NULL = NULL,
  nMC = NULL,
  gamma_go,
  gamma_nogo,
  n_t,
  n_c = NULL,
  m_t = NULL,
  m_c = NULL,
```

```
  kappa0_t = NULL,
  kappa0_c = NULL,
  nu0_t = NULL,
  nu0_c = NULL,
  mu0_t = NULL,
  mu0_c = NULL,
  sigma0_t = NULL,
  sigma0_c = NULL,
  mu_t,
  mu_c = NULL,
  sigma_t,
  sigma_c = NULL,
  r = NULL,
  ne_t = NULL,
  ne_c = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  bar_ye_t = NULL,
  bar_ye_c = NULL,
  se_t = NULL,
  se_c = NULL,
  error_if_Miss = TRUE,
  Gray_inc_Miss = FALSE,
  seed
)
```

Arguments

nsim	A positive integer specifying the number of Monte Carlo simulation replicates.
prob	A character string specifying the probability type: 'posterior' or 'predictive'.
design	A character string specifying the trial design: 'controlled', 'uncontrolled', or 'external'.
prior	A character string specifying the prior distribution: 'vague' or 'N-Inv-Chisq'.
CalcMethod	A character string specifying the computation method: 'NI' (Numerical Integration), 'MC' (Monte Carlo), or 'MM' (Moment Matching).
theta_TV	A numeric value representing the target value (TV) threshold for the Go decision. Required if prob = 'posterior'; otherwise NULL. Default is NULL.
theta_MAV	A numeric value representing the minimum acceptable value (MAV) threshold for the NoGo decision. Required if prob = 'posterior'; otherwise NULL. Default is NULL.
theta_NULL	A numeric value representing the null hypothesis threshold. Required if prob = 'predictive'; otherwise NULL. Default is NULL.
nMC	A positive integer specifying the number of Monte Carlo draws for computing posterior probabilities. Required if CalcMethod = 'MC'; otherwise NULL.
gamma_go	A numeric scalar in (0, 1) giving the minimum posterior or predictive probability required for a Go decision.

gamma_nogo — A numeric scalar in (0, 1) giving the minimum posterior or predictive probability required for a NoGo decision. No ordering constraint on gamma_go and gamma_nogo is imposed, though their combination determines the frequency of Miss outcomes.

n_t — A positive integer giving the number of patients in the treatment group in the proof-of-concept (PoC) trial.

n_c — A positive integer giving the number of patients in the control group in the proof-of-concept (PoC) trial. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.

m_t — A positive integer representing the future sample size for the treatment group. Required if prob = 'predictive'; otherwise NULL. Default is NULL.

m_c — A positive integer representing the future sample size for the control group. Required if prob = 'predictive'; otherwise NULL. Default is NULL.

kappa0_t — A positive numeric value representing the prior hyperparameter kappa for the treatment group. Required if prior = 'N-Inv-Chisq'; otherwise NULL. Default is NULL.

kappa0_c — A positive numeric value representing the prior hyperparameter kappa for the control group. Required if prior = 'N-Inv-Chisq' and design %in% c('controlled', 'external'); otherwise NULL. Default is NULL.

nu0_t — A positive numeric value representing the prior hyperparameter nu for the treatment group. Required if prior = 'N-Inv-Chisq'; otherwise NULL. Default is NULL.

nu0_c — A positive numeric value representing the prior hyperparameter nu for the control group. Required if prior = 'N-Inv-Chisq' and design %in% c('controlled', 'external'); otherwise NULL. Default is NULL.

mu0_t — A numeric value representing the prior mean for the treatment group. Required if prior = 'N-Inv-Chisq'; otherwise NULL. Default is NULL.

mu0_c — A numeric value representing the prior mean for the control group. For design = 'uncontrolled', this represents the hypothetical control mean. Required if prior = 'N-Inv-Chisq' and design %in% c('controlled', 'external'), or if design = 'uncontrolled'; otherwise NULL. Default is NULL.

sigma0_t — A positive numeric value representing the prior standard deviation for the treatment group. Required if prior = 'N-Inv-Chisq'; otherwise NULL. Default is NULL.

sigma0_c — A positive numeric value representing the prior standard deviation for the control group. Required if prior = 'N-Inv-Chisq' and design %in% c('controlled', 'external'); otherwise NULL. Default is NULL.

mu_t — A numeric value representing the true mean for the treatment group in the simulation.

mu_c — A numeric value representing the true mean for the control group in the simulation. For uncontrolled design, this represents the historical control mean. Set to NULL if design = 'uncontrolled'.

sigma_t — A positive numeric value representing the true standard deviation for the treatment group in the simulation.

sigma_c	A positive numeric value representing the true standard deviation for the control group in the simulation. For uncontrolled design, this represents the historical control standard deviation. Set to NULL if design = 'uncontrolled'.
r	A positive numeric value representing the variance scaling factor that allows the scale of hypothetical control to be different from treatment. Specifically, sd.control = sqrt(r) * sd.treatment. Required if design = 'uncontrolled'. When r = 1, the control and treatment have the same variance scale.
ne_t	A positive integer representing the number of patients in the treatment group for the external data. Required if design = 'external' and external treatment data are available.
ne_c	A positive integer representing the number of patients in the control group for the external data. Required if design = 'external' and external control data are available.
alpha0e_t	A numeric value in (0, 1] representing the power prior scale parameter for the treatment group. Controls the degree of borrowing from external treatment data: 0 = no borrowing, 1 = full borrowing. Required if ne_t is specified.
alpha0e_c	A numeric value in (0, 1] representing the power prior scale parameter for the control group. Controls the degree of borrowing from external control data: 0 = no borrowing, 1 = full borrowing. Required if ne_c is specified.
bar_ye_t	A numeric value representing the sample mean of the external data for the treatment group. Required if ne_t is specified.
bar_ye_c	A numeric value representing the sample mean of the external data for the control group. Required if ne_c is specified.
se_t	A positive numeric value representing the sample standard deviation of the external data for the treatment group. Required if ne_t is specified.
se_c	A positive numeric value representing the sample standard deviation of the external data for the control group. Required if ne_c is specified.
error_if_Miss	A logical value; if TRUE (default), the function stops with an error when positive Miss probability is obtained, indicating poorly chosen thresholds. If FALSE, the function proceeds and reports Miss probability based on Gray_inc_Miss setting.
Gray_inc_Miss	A logical value; if TRUE, Miss probability is included in Gray probability (Miss is not reported separately). If FALSE (default), Miss probability is reported as a separate category. This parameter is only active when error_if_Miss = FALSE.
seed	A numeric value representing the seed number for reproducible random number generation.

Details

The function performs Monte Carlo simulation to evaluate operating characteristics by:

- Generating random trial data based on specified true parameters
- Computing posterior or predictive probabilities for each simulated trial
- Classifying each trial as Go, NoGo, or Gray based on decision thresholds

Posterior parameter calculations (mu.t1, mu.t2, sd.t1, sd.t2, nu.t1, nu.t2) are fully vectorized over the nsim simulated datasets. pbayespostpred1cont is called twice (once per threshold), with each call receiving vectors of length nsim and returning a vector of nsim probabilities - no inner loop over simulation replicates is required.

For external designs, power priors are incorporated using exact conjugate representation:

- Power priors for normal data are mathematically equivalent to Normal-Inverse-Chi-squared distributions
- This enables closed-form computation without MCMC sampling
- Alpha parameters control the degree of borrowing (0 = no borrowing, 1 = full borrowing)

Decision rules:

- **Go**: gGo >= gamma_go and gNoGo < gamma_nogo
- **NoGo**: gGo < gamma_go and gNoGo >= gamma_nogo
- **Gray**: neither Go nor NoGo criterion is met
- **Miss**: both Go and NoGo criteria are met simultaneously

Handling Miss probability:

- When `error_if_Miss = TRUE` (default): Function stops with error if Miss probability > 0, prompting reconsideration of thresholds
- When `error_if_Miss = FALSE` and `Gray_inc_Miss = TRUE`: Miss probability is added to Gray probability
- When `error_if_Miss = FALSE` and `Gray_inc_Miss = FALSE`: Miss probability is reported as a separate category

The function can be used for:

- **Controlled design**: Two-arm randomized trial
- **Uncontrolled design**: Single-arm trial with informative priors (historical control)
- **External design**: Incorporating historical data through power priors

Value

A data frame containing the true means for both groups and the Go, NoGo, and Gray probabilities. When `error_if_Miss = FALSE` and `Gray_inc_Miss = FALSE`, Miss probability is also included as a separate column. For uncontrolled design, only mu_t is included (not mu_c).

Examples

```
# Example 1: Controlled design with vague prior and NI method
# (default: error_if_Miss = TRUE, Gray_inc_Miss = FALSE)
pbayesdecisionprob1cont(
  nsim = 100, prob = 'posterior', design = 'controlled', prior = 'vague', CalcMethod = 'NI',
   theta_TV = 1.5, theta_MAV = -0.5, theta_NULL = NULL,
   nMC = NULL, gamma_go = 0.7, gamma_nogo = 0.2,
   n_t = 15, n_c = 15, m_t = NULL, m_c = NULL,
   kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
```

```
  mu0_t = NULL, mu0_c = NULL, sigma0_t = NULL, sigma0_c = NULL,
  mu_t = 3, mu_c = 1, sigma_t = 1.2, sigma_c = 1.1,
  r = NULL, ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 2
)

# Example 2: Uncontrolled design with informative prior
pbayesdecisionprob1cont(
 nsim = 100, prob = 'posterior', design = 'uncontrolled', prior = 'N-Inv-Chisq', CalcMethod = 'NI',
  theta_TV = 1.0, theta_MAV = 0.0, theta_NULL = NULL,
  nMC = NULL, gamma_go = 0.8, gamma_nogo = 0.2,
  n_t = 20, n_c = NULL, m_t = NULL, m_c = NULL,
  kappa0_t = 2, kappa0_c = NULL, nu0_t = 5, nu0_c = NULL,
  mu0_t = 3.0, mu0_c = 1.5, sigma0_t = 1.5, sigma0_c = NULL,
  mu_t = 3.5, mu_c = NULL, sigma_t = 1.3, sigma_c = NULL,
  r = 1, ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 3
)

# Example 3: External design with control data using MM approximation
pbayesdecisionprob1cont(
  nsim = 100, prob = 'posterior', design = 'external', prior = 'vague', CalcMethod = 'MM',
  theta_TV = 1.0, theta_MAV = 0.0, theta_NULL = NULL,
  nMC = NULL, gamma_go = 0.8, gamma_nogo = 0.2,
  n_t = 12, n_c = 12, m_t = NULL, m_c = NULL,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, mu0_c = NULL, sigma0_t = NULL, sigma0_c = NULL,
  mu_t = 2, mu_c = 0, sigma_t = 1, sigma_c = 1,
  r = NULL, ne_t = NULL, ne_c = 20, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = 0, se_t = NULL, se_c = 1,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 4
)

# Example 4: Controlled design with predictive probability
pbayesdecisionprob1cont(
 nsim = 100, prob = 'predictive', design = 'controlled', prior = 'N-Inv-Chisq', CalcMethod = 'NI',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 2.0,
  nMC = NULL, gamma_go = 0.75, gamma_nogo = 0.35,
  n_t = 15, n_c = 15, m_t = 50, m_c = 50,
  kappa0_t = 3, kappa0_c = 3, nu0_t = 4, nu0_c = 4,
  mu0_t = 3.5, mu0_c = 1.5, sigma0_t = 1.5, sigma0_c = 1.5,
  mu_t = 3.2, mu_c = 1.3, sigma_t = 1.4, sigma_c = 1.2,
  r = NULL, ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 5
)

# Example 5: Uncontrolled design with predictive probability
pbayesdecisionprob1cont(
 nsim = 100, prob = 'predictive', design = 'uncontrolled', prior = 'vague', CalcMethod = 'NI',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 1.0,
```

```
  nMC = NULL, gamma_go = 0.75, gamma_nogo = 0.35,
  n_t = 20, n_c = NULL, m_t = 40, m_c = 40,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, mu0_c = 1.5, sigma0_t = NULL, sigma0_c = NULL,
  mu_t = 3, mu_c = NULL, sigma_t = 1.3, sigma_c = NULL,
  r = 1, ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 9
)

# Example 6: External design with predictive probability using MC method
pbayesdecisionprob1cont(
  nsim = 100, prob = 'predictive', design = 'external', prior = 'vague', CalcMethod = 'MC',
  theta_TV = NULL, theta_MAV = NULL, theta_NULL = 1.5,
  nMC = 5000, gamma_go = 0.7, gamma_nogo = 0.4,
  n_t = 12, n_c = 12, m_t = 30, m_c = 30,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, mu0_c = NULL, sigma0_t = NULL, sigma0_c = NULL,
  mu_t = 2.5, mu_c = 1.0, sigma_t = 1.3, sigma_c = 1.1,
  r = NULL, ne_t = 15, ne_c = 18, alpha0e_t = 0.6, alpha0e_c = 0.7,
  bar_ye_t = 2.3, bar_ye_c = 0.9, se_t = 1.2, se_c = 1.0,
  error_if_Miss = FALSE, Gray_inc_Miss = FALSE, seed = 6
)
```

`pbayesdecisionprob2bin` *Go/NoGo/Gray Decision Probabilities for a Clinical Trial with Two Binary Endpoints*

Description

Evaluates operating characteristics (Go, NoGo, Gray probabilities) for clinical trials with two binary endpoints under the Bayesian framework. The function supports controlled, uncontrolled, and external designs, and uses both posterior probability and posterior predictive probability criteria.

Usage

```
pbayesdecisionprob2bin(
  nsim = 10000L,
  prob = "posterior",
  design = "controlled",
  GoRegions,
  NoGoRegions,
  gamma_go,
  gamma_nogo,
  pi_t1,
  pi_t2,
  rho_t,
```

```
  pi_c1 = NULL,
  pi_c2 = NULL,
  rho_c = NULL,
  n_t,
  n_c = NULL,
  a_t_00,
  a_t_01,
  a_t_10,
  a_t_11,
  a_c_00,
  a_c_01,
  a_c_10,
  a_c_11,
  m_t = NULL,
  m_c = NULL,
  theta_TV1 = NULL,
  theta_MAV1 = NULL,
  theta_TV2 = NULL,
  theta_MAV2 = NULL,
  theta_NULL1 = NULL,
  theta_NULL2 = NULL,
  z00 = NULL,
  z01 = NULL,
  z10 = NULL,
  z11 = NULL,
  xe_t_00 = NULL,
  xe_t_01 = NULL,
  xe_t_10 = NULL,
  xe_t_11 = NULL,
  xe_c_00 = NULL,
  xe_c_01 = NULL,
  xe_c_10 = NULL,
  xe_c_11 = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  nMC = 10000L,
  CalcMethod = "Exact",
  error_if_Miss = TRUE,
  Gray_inc_Miss = FALSE
)
```

Arguments

nsim	A positive integer giving the number of PoC count vectors sampled via rmultinom per arm per scenario when CalcMethod = 'MC' (outer loop). The sampled vectors are deduplicated into K_t and K_c unique vectors ($K_t, K_c \ll$ nsim); Dirichlet sampling is then performed only for these unique vectors using nMC draws each. Ignored when CalcMethod = 'Exact'. Default is 10000.

prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
GoRegions	An integer vector specifying which of the nine posterior regions (R1–R9) or four predictive regions (R1–R4) constitute a Go decision. For prob = 'posterior', valid values are integers in 1–9; for prob = 'predictive', in 1–4. A common choice is GoRegions = 1 (both endpoints exceed TV or NULL for posterior/predictive, respectively).
NoGoRegions	An integer vector specifying which regions constitute a NoGo decision. A common choice is NoGoRegions = 9 (both endpoints below MAV) for posterior, or NoGoRegions = 4 for predictive. Must be disjoint from GoRegions.
gamma_go	A numeric scalar in (0, 1). Go threshold: a Go decision is made if $P(\mathrm{GoRegions}) \geq \gamma_1$.
gamma_nogo	A numeric scalar in (0, 1). NoGo threshold: a NoGo decision is made if $P(\mathrm{NoGoRegions}) \geq \gamma_2$. No ordering constraint on gamma_go and gamma_nogo is imposed; their combination determines the frequency of Miss outcomes.
pi_t1	A numeric vector of true treatment response probabilities for Endpoint 1. Each element must be in (0, 1).
pi_t2	A numeric vector of true treatment response probabilities for Endpoint 2. Must have the same length as pi_t1.
rho_t	A numeric vector of true within-treatment-arm correlations between Endpoint 1 and Endpoint 2. Must have the same length as pi_t1. If any element is outside the feasible range for the corresponding (pi_t1, pi_t2) pair, that scenario returns NA for all decision probabilities instead of stopping with an error.
pi_c1	A numeric vector of true control response probabilities for Endpoint 1. For design = 'uncontrolled', this parameter is not used in probability calculations but must still be supplied; it is retained in the output for reference. Must have the same length as pi_t1.
pi_c2	A numeric vector of true control response probabilities for Endpoint 2. For design = 'uncontrolled', see note for pi_c1. Must have the same length as pi_t1.
rho_c	A numeric vector of true within-control-arm correlations. For design = 'uncontrolled', not used in probability calculations. Must have the same length as pi_t1. If any element is outside the feasible range for the corresponding (pi_c1, pi_c2) pair, that scenario returns NA for all decision probabilities.
n_t	A positive integer giving the number of patients in the treatment group in the proof-of-concept (PoC) trial.
n_c	A positive integer giving the number of patients in the control group in the PoC trial. For design = 'uncontrolled', this is the hypothetical control sample size (required for consistency with other designs).
a_t_00	A positive numeric scalar giving the Dirichlet prior parameter for the (0, 0) response pattern in the treatment group.
a_t_01	A positive numeric scalar giving the Dirichlet prior parameter for the (0, 1) response pattern in the treatment group.

a_t_10 A positive numeric scalar giving the Dirichlet prior parameter for the (1, 0) response pattern in the treatment group.

a_t_11 A positive numeric scalar giving the Dirichlet prior parameter for the (1, 1) response pattern in the treatment group.

a_c_00 A positive numeric scalar giving the Dirichlet prior parameter for the (0, 0) response pattern in the control group. For design = 'uncontrolled', serves as a hyperparameter of the hypothetical control distribution.

a_c_01 A positive numeric scalar giving the Dirichlet prior parameter for the (0, 1) response pattern in the control group. For design = 'uncontrolled', serves as a hyperparameter of the hypothetical control distribution.

a_c_10 A positive numeric scalar giving the Dirichlet prior parameter for the (1, 0) response pattern in the control group. For design = 'uncontrolled', serves as a hyperparameter of the hypothetical control distribution.

a_c_11 A positive numeric scalar giving the Dirichlet prior parameter for the (1, 1) response pattern in the control group. For design = 'uncontrolled', serves as a hyperparameter of the hypothetical control distribution.

m_t A positive integer giving the number of patients in the treatment group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

m_c A positive integer giving the number of patients in the control group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

theta_TV1 A numeric scalar giving the target value (TV) threshold for Endpoint 1. Required when prob = 'posterior'; must satisfy theta_TV1 > theta_MAV1. Set to NULL when prob = 'predictive'.

theta_MAV1 A numeric scalar giving the minimum acceptable value (MAV) threshold for Endpoint 1. Required when prob = 'posterior'; must satisfy theta_TV1 > theta_MAV1. Set to NULL when prob = 'predictive'.

theta_TV2 A numeric scalar giving the target value (TV) threshold for Endpoint 2. Required when prob = 'posterior'; must satisfy theta_TV2 > theta_MAV2. Set to NULL when prob = 'predictive'.

theta_MAV2 A numeric scalar giving the minimum acceptable value (MAV) threshold for Endpoint 2. Required when prob = 'posterior'; must satisfy theta_TV2 > theta_MAV2. Set to NULL when prob = 'predictive'.

theta_NULL1 A numeric scalar giving the null hypothesis threshold for Endpoint 1. Required when prob = 'predictive'; set to NULL when prob = 'posterior'.

theta_NULL2 A numeric scalar giving the null hypothesis threshold for Endpoint 2. Required when prob = 'predictive'; set to NULL when prob = 'posterior'.

z00 A non-negative integer giving the hypothetical control count for pattern (0, 0). Required when design = 'uncontrolled'; otherwise set to NULL.

z01 A non-negative integer giving the hypothetical control count for pattern (0, 1). Required when design = 'uncontrolled'; otherwise set to NULL.

z10 A non-negative integer giving the hypothetical control count for pattern (1, 0). Required when design = 'uncontrolled'; otherwise set to NULL.

z11 A non-negative integer giving the hypothetical control count for pattern (1, 1). Required when design = 'uncontrolled'; otherwise set to NULL.

`xe_t_00` A non-negative integer giving the external treatment group count for pattern `(0, 0)`. Required when `design = 'external'` and external treatment data are used; otherwise `NULL`.

`xe_t_01` A non-negative integer giving the external treatment group count for pattern `(0, 1)`. Required for external treatment data; otherwise `NULL`.

`xe_t_10` A non-negative integer giving the external treatment group count for pattern `(1, 0)`. Required for external treatment data; otherwise `NULL`.

`xe_t_11` A non-negative integer giving the external treatment group count for pattern `(1, 1)`. Required for external treatment data; otherwise `NULL`.

`xe_c_00` A non-negative integer giving the external control group count for pattern `(0, 0)`. Required when `design = 'external'` and external control data are used; otherwise `NULL`.

`xe_c_01` A non-negative integer giving the external control group count for pattern `(0, 1)`. Required for external control data; otherwise `NULL`.

`xe_c_10` A non-negative integer giving the external control group count for pattern `(1, 0)`. Required for external control data; otherwise `NULL`.

`xe_c_11` A non-negative integer giving the external control group count for pattern `(1, 1)`. Required for external control data; otherwise `NULL`.

`alpha0e_t` A numeric scalar in `(0, 1]` giving the power prior weight for the treatment group. Required when external treatment data are used; otherwise `NULL`.

`alpha0e_c` A numeric scalar in `(0, 1]` giving the power prior weight for the control group. Required when external control data are used; otherwise `NULL`.

`nMC` A positive integer giving the number of Dirichlet draws used to evaluate the decision probability for each count combination in Stage 1. Used by both `CalcMethod = 'Exact'` and `CalcMethod = 'MC'` (inner loop). Default is `10000`.

`CalcMethod` A character string specifying the computation method. Must be `'Exact'` (default) or `'MC'`. `'Exact'` uses full enumeration of all possible multinomial count combinations (two-stage approach described in Details). `'MC'` samples `nsim` (x_t, x_c) pairs via `rmultinom`, deduplicates them into K unique pairs ($K \ll$ `nsim`), calls `pbayespostpred2bin` for each unique pair to obtain Go/NoGo probabilities, and weights the decisions by the observed pair frequencies. This reuses the same validated probability logic as the Exact method, avoiding any duplication of computation. The `'MC'` method trades some Monte Carlo variance for substantially reduced computation time when `n_t` and/or `n_c` are large.

`error_if_Miss` A logical scalar; if `TRUE` (default), the function stops with an error if the Miss probability is positive, prompting reconsideration of the thresholds.

`Gray_inc_Miss` A logical scalar; if `TRUE`, the Miss probability is added to Gray. If `FALSE` (default), Miss is reported separately. Active only when `error_if_Miss = FALSE`.

Details

Computation proceeds in two stages for efficiency. In Stage 1, posterior or predictive probabilities are precomputed for every possible multinomial count combination (x_t, x_c). In Stage 2, operating characteristics for each scenario are obtained by weighting the Stage 1 decisions by their multinomial probabilities under the true parameters, avoiding repeated Monte Carlo sampling per scenario.

Two-stage computation.

Stage 1 (precomputation): All possible multinomial count combinations x_t are enumerated using `allmultinom`. For `design = 'controlled'` or `'external'`, all combinations x_c are also enumerated and the Go/NoGo probability matrix `PrGo[i, j]` has dimensions $n_t \times n_c$. For `design = 'uncontrolled'`, the control distribution is fixed as $\mathrm{Dir}(\alpha_{2,**} + z_{**})$ and does not depend on any observed control counts; the probability matrix therefore has dimensions $n_t \times 1$, eliminating the $\binom{n_2+3}{3}$-row control enumeration and the associated Gamma sampling. This stage is independent of the true scenario parameters.

Stage 2 (scenario evaluation): For each scenario $(\pi_{t1}, \pi_{t2}, \rho_t, \pi_{c1}, \pi_{c2}, \rho_c)$, `getjointbin` converts the marginal rates and correlation into the four-cell probability vector p_k. The multinomial probability mass function weights the Stage 1 decisions:

$$\Pr(\mathrm{Go} \mid \Theta) = \sum_{i,j} \mathbf{1}(\mathrm{Go}_{ij}) \cdot P(x_{t,i} \mid n_1, p_t) \cdot P(x_{c,j} \mid n_2, p_c),$$

where $\mathbf{1}(\mathrm{Go}_{ij})$ is the Go indicator from Stage 1. Stage 2 requires no additional Monte Carlo and is fast even for large scenario grids.

Decision categories.

- **Go**: sum of Go region probabilities $\geq \gamma_1$ AND sum of NoGo region probabilities $< \gamma_2$.
- **NoGo**: sum of Go region probabilities $< \gamma_1$ AND sum of NoGo region probabilities $\geq \gamma_2$.
- **Miss**: both Go and NoGo criteria satisfied simultaneously.
- **Gray**: neither Go nor NoGo criterion satisfied.

Value

A data frame with one row per scenario and columns:

pi_t1 True treatment response probability for Endpoint 1.

pi_t2 True treatment response probability for Endpoint 2.

rho_t True within-arm correlation in the treatment arm.

pi_c1 True control response probability for Endpoint 1 (omitted when `design = 'uncontrolled'`).

pi_c2 True control response probability for Endpoint 2 (omitted when `design = 'uncontrolled'`).

rho_c True within-arm correlation in the control arm (omitted when `design = 'uncontrolled'`).

Go Probability of making a Go decision.

Gray Probability of making a Gray (inconclusive) decision.

NoGo Probability of making a NoGo decision.

Miss Probability where Go and NoGo criteria are simultaneously met. Included when `error_if_Miss = FALSE` and `Gray_inc_Miss = FALSE`.

The returned object has S3 class `pbayesdecisionprob2bin` with an associated `print` method.

Examples

```
# Example 1: Posterior probability, controlled design (rho > 0)
# Explores the impact of positive within-arm endpoint correlation.
pbayesdecisionprob2bin(
  prob        = 'posterior',
  design      = 'controlled',
  GoRegions   = 1L,
  NoGoRegions = 9L,
  gamma_go    = 0.52,
  gamma_nogo  = 0.52,
  pi_t1       = c(0.15, 0.25, 0.30, 0.40),
  pi_t2       = c(0.20, 0.30, 0.35, 0.45),
  rho_t       = rep(0.6, 4),
  pi_c1       = rep(0.15, 4),
  pi_c2       = rep(0.20, 4),
  rho_c       = rep(0.6, 4),
  n_t = 10, n_c = 10,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  m_t = NULL, m_c = NULL,
  theta_TV1   = 0.15, theta_MAV1 = 0.10,
  theta_TV2   = 0.15, theta_MAV2 = 0.10,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL,
  nMC = 100,
  error_if_Miss = FALSE, Gray_inc_Miss = FALSE
)

# Example 2: Posterior probability, uncontrolled design
# Hypothetical control specified via pseudo-counts.
pbayesdecisionprob2bin(
  prob        = 'posterior',
  design      = 'uncontrolled',
  GoRegions   = 1L,
  NoGoRegions = 9L,
  gamma_go    = 0.27,
  gamma_nogo  = 0.36,
  pi_t1       = c(0.15, 0.30, 0.40),
  pi_t2       = c(0.20, 0.35, 0.45),
  rho_t       = rep(0.2, 3),
  pi_c1       = NULL,
  pi_c2       = NULL,
  rho_c       = NULL,
  n_t = 10, n_c = NULL,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  m_t = NULL, m_c = NULL,
  theta_TV1   = 0.15, theta_MAV1 = 0.10,
  theta_TV2   = 0.15, theta_MAV2 = 0.10,
```

```
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  z00 = 2L, z01 = 1L, z10 = 2L, z11 = 1L,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL,
  nMC = 100,
  error_if_Miss = FALSE, Gray_inc_Miss = FALSE
)

# Example 3: Posterior probability, external control design
# External control data incorporated via power prior (alpha0e_c = 0.5).
pbayesdecisionprob2bin(
  prob        = 'posterior',
  design      = 'external',
  GoRegions   = 1L,
  NoGoRegions = 9L,
  gamma_go    = 0.27,
  gamma_nogo  = 0.36,
  pi_t1       = c(0.15, 0.25, 0.30, 0.40),
  pi_t2       = c(0.20, 0.30, 0.35, 0.45),
  rho_t       = rep(0.0, 4),
  pi_c1       = rep(0.15, 4),
  pi_c2       = rep(0.20, 4),
  rho_c       = rep(0.0, 4),
  n_t = 10, n_c = 10,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  m_t = NULL, m_c = NULL,
  theta_TV1   = 0.15, theta_MAV1 = 0.10,
  theta_TV2   = 0.15, theta_MAV2 = 0.10,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = 4L,  xe_c_01 = 2L,  xe_c_10 = 3L,  xe_c_11 = 1L,
  alpha0e_t = NULL, alpha0e_c = 0.5,
  nMC = 100,
  error_if_Miss = FALSE, Gray_inc_Miss = FALSE
)

# Example 4: Predictive probability, controlled design
# Future trial has m_t = m_c = 30 patients per arm.
pbayesdecisionprob2bin(
  prob        = 'predictive',
  design      = 'controlled',
  GoRegions   = 1L,
  NoGoRegions = 4L,
  gamma_go    = 0.60,
  gamma_nogo  = 0.80,
  pi_t1       = c(0.15, 0.25, 0.30, 0.40),
  pi_t2       = c(0.20, 0.30, 0.35, 0.45),
  rho_t       = rep(0.3, 4),
  pi_c1       = rep(0.15, 4),
  pi_c2       = rep(0.20, 4),
```

```
  rho_c       = rep(0.3, 4),
  n_t = 10, n_c = 10,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  m_t = 30L, m_c = 30L,
  theta_TV1   = NULL, theta_MAV1 = NULL,
  theta_TV2   = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.10, theta_NULL2 = 0.10,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL,
  nMC = 100,
  error_if_Miss = FALSE, Gray_inc_Miss = FALSE
)

# Example 5: Predictive probability, uncontrolled design
# Hypothetical control specified via pseudo-counts; future trial m_t = m_c = 30.
pbayesdecisionprob2bin(
  prob        = 'predictive',
  design      = 'uncontrolled',
  GoRegions   = 1L,
  NoGoRegions = 4L,
  gamma_go    = 0.60,
  gamma_nogo  = 0.80,
  pi_t1       = c(0.15, 0.30, 0.40),
  pi_t2       = c(0.20, 0.35, 0.45),
  rho_t       = rep(0.7, 3),
  pi_c1       = rep(0.15, 3),
  pi_c2       = rep(0.20, 3),
  rho_c       = rep(0.7, 3),
  n_t = 10, n_c = 10,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  m_t = 30L, m_c = 30L,
  theta_TV1   = NULL, theta_MAV1 = NULL,
  theta_TV2   = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.10, theta_NULL2 = 0.10,
  z00 = 2L, z01 = 1L, z10 = 2L, z11 = 1L,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL,
  nMC = 100,
  error_if_Miss = FALSE, Gray_inc_Miss = FALSE
)

# Example 6: Predictive probability, external treatment design
# External treatment data incorporated via power prior (alpha0e_t = 0.5).
pbayesdecisionprob2bin(
  prob        = 'predictive',
  design      = 'external',
  GoRegions   = 1L,
  NoGoRegions = 4L,
```

```
  gamma_go    = 0.60,
  gamma_nogo  = 0.80,
  pi_t1       = c(0.15, 0.25, 0.30, 0.40),
  pi_t2       = c(0.20, 0.30, 0.35, 0.45),
  rho_t       = rep(0.5, 4),
  pi_c1       = rep(0.15, 4),
  pi_c2       = rep(0.20, 4),
  rho_c       = rep(0.5, 4),
  n_t = 10, n_c = 10,
  a_t_00 = 0.25, a_t_01 = 0.25, a_t_10 = 0.25, a_t_11 = 0.25,
  a_c_00 = 0.25, a_c_01 = 0.25, a_c_10 = 0.25, a_c_11 = 0.25,
  m_t = 30L, m_c = 30L,
  theta_TV1   = NULL, theta_MAV1 = NULL,
  theta_TV2   = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.10, theta_NULL2 = 0.10,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = 3L, xe_t_01 = 2L, xe_t_10 = 3L, xe_t_11 = 2L,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = 0.5, alpha0e_c = NULL,
  nMC = 100,
  error_if_Miss = FALSE, Gray_inc_Miss = FALSE
)
```

`pbayesdecisionprob2cont` *Go/NoGo/Gray Decision Probabilities for Two Continuous Endpoints*

Description

Computes the operating characteristics (Go, NoGo, Gray, and optionally Miss probabilities) for a two-continuous-endpoint Bayesian Go/NoGo decision framework by Monte Carlo simulation over treatment scenarios.

Usage

```
pbayesdecisionprob2cont(
  nsim,
  prob,
  design,
  prior,
  GoRegions,
  NoGoRegions,
  gamma_go,
  gamma_nogo,
  theta_TV1 = NULL,
  theta_MAV1 = NULL,
  theta_TV2 = NULL,
  theta_MAV2 = NULL,
```

```
  theta_NULL1 = NULL,
  theta_NULL2 = NULL,
  n_t,
  n_c = NULL,
  m_t = NULL,
  m_c = NULL,
  mu_t,
  Sigma_t,
  mu_c = NULL,
  Sigma_c = NULL,
  kappa0_t = NULL,
  nu0_t = NULL,
  mu0_t = NULL,
  Lambda0_t = NULL,
  kappa0_c = NULL,
  nu0_c = NULL,
  mu0_c = NULL,
  Lambda0_c = NULL,
  r = NULL,
  ne_t = NULL,
  ne_c = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  bar_ye_t = NULL,
  bar_ye_c = NULL,
  se_t = NULL,
  se_c = NULL,
  nMC = NULL,
  CalcMethod = "MC",
  error_if_Miss = TRUE,
  Gray_inc_Miss = FALSE,
  seed
)
```

Arguments

nsim	A positive integer. Number of simulated datasets per scenario.
prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
prior	A character string specifying the prior distribution. Must be 'vague' or 'N-Inv-Wishart'.
GoRegions	An integer vector specifying which of the nine posterior regions (R1–R9) or four predictive regions (R1–R4) constitute a Go decision. For prob = 'posterior', valid values are integers in 1–9; for prob = 'predictive', in 1–4. A common choice is GoRegions = 1 (both endpoints exceed TV or NULL for posterior/predictive, respectively).

NoGoRegions — An integer vector specifying which regions constitute a NoGo decision. A common choice is NoGoRegions = 9 (both endpoints below MAV) for posterior, or NoGoRegions = 4 for predictive. Must be disjoint from GoRegions.

gamma_go — A numeric scalar in (0, 1). Go threshold: a Go decision is made if $P(\text{GoRegions}) \geq \gamma_1$.

gamma_nogo — A numeric scalar in (0, 1). NoGo threshold: a NoGo decision is made if $P(\text{NoGoRegions}) \geq \gamma_2$. No ordering constraint on gamma_go and gamma_nogo is imposed; their combination determines the frequency of Miss outcomes.

theta_TV1 — A numeric scalar giving the target value (TV) threshold for Endpoint 1. Required when prob = 'posterior'; must satisfy theta_TV1 > theta_MAV1. Set to NULL when prob = 'predictive'.

theta_MAV1 — A numeric scalar giving the minimum acceptable value (MAV) threshold for Endpoint 1. Required when prob = 'posterior'; must satisfy theta_TV1 > theta_MAV1. Set to NULL when prob = 'predictive'.

theta_TV2 — A numeric scalar giving the target value (TV) threshold for Endpoint 2. Required when prob = 'posterior'; must satisfy theta_TV2 > theta_MAV2. Set to NULL when prob = 'predictive'.

theta_MAV2 — A numeric scalar giving the minimum acceptable value (MAV) threshold for Endpoint 2. Required when prob = 'posterior'; must satisfy theta_TV2 > theta_MAV2. Set to NULL when prob = 'predictive'.

theta_NULL1 — A numeric scalar giving the null hypothesis threshold for Endpoint 1. Required when prob = 'predictive'; set to NULL when prob = 'posterior'.

theta_NULL2 — A numeric scalar giving the null hypothesis threshold for Endpoint 2. Required when prob = 'predictive'; set to NULL when prob = 'posterior'.

n_t — A positive integer giving the number of patients in the treatment group in the proof-of-concept (PoC) trial.

n_c — A positive integer giving the number of patients in the control group in the PoC trial. For design = 'uncontrolled', this is the hypothetical control sample size (required for consistency with other designs).

m_t — A positive integer giving the number of patients in the treatment group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

m_c — A positive integer giving the number of patients in the control group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

mu_t — Numeric matrix with 2 columns. Each row gives the true treatment mean vector for one scenario. A length-2 vector is coerced to a 1-row matrix.

Sigma_t — A 2x2 positive-definite matrix. True treatment covariance.

mu_c — Numeric matrix with 2 columns or a length-2 vector. True control (or hypothetical control) mean vector(s).

Sigma_c — A 2x2 positive-definite matrix. True control covariance.

kappa0_t — A positive numeric scalar giving the NIW prior concentration parameter for the treatment group. Required when prior = 'N-Inv-Wishart'; otherwise set to NULL.

`nu0_t` A numeric scalar giving the NIW prior degrees of freedom for the treatment group. Must be greater than 3. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`mu0_t` A numeric vector of length 2 giving the NIW prior mean for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`Lambda0_t` A 2x2 positive-definite numeric matrix giving the NIW prior scale matrix for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`kappa0_c` A positive numeric scalar giving the NIW prior concentration parameter for the control group. Required when `prior = 'N-Inv-Wishart'` and `design != 'uncontrolled'`; otherwise set to `NULL`.

`nu0_c` A numeric scalar giving the NIW prior degrees of freedom for the control group. Must be greater than 3. Required when `prior = 'N-Inv-Wishart'` and `design != 'uncontrolled'`; otherwise set to `NULL`.

`mu0_c` A numeric vector of length 2 giving the NIW prior mean for the control group, or the hypothetical control location when `design = 'uncontrolled'`. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`Lambda0_c` A 2x2 positive-definite numeric matrix giving the NIW prior scale matrix for the control group. Required when `prior = 'N-Inv-Wishart'` and `design != 'uncontrolled'`; otherwise set to `NULL`.

`r` A positive numeric scalar giving the variance scaling factor for the hypothetical control distribution. Required when `design = 'uncontrolled'`; otherwise set to `NULL`.

`ne_t` A positive integer giving the external treatment group sample size. Required when `design = 'external'` and external treatment data are used; otherwise set to `NULL`.

`ne_c` A positive integer giving the external control group sample size. Required when `design = 'external'` and external control data are used; otherwise set to `NULL`.

`alpha0e_t` A numeric scalar in `(0, 1]` giving the power prior weight for the external treatment data. Required when external treatment data are used; otherwise set to `NULL`.

`alpha0e_c` A numeric scalar in `(0, 1]` giving the power prior weight for the external control data. Required when external control data are used; otherwise set to `NULL`.

`bar_ye_t` A numeric vector of length 2 giving the external treatment group sample mean. Required when external treatment data are used; otherwise set to `NULL`.

`bar_ye_c` A numeric vector of length 2 giving the external control group sample mean. Required when external control data are used; otherwise set to `NULL`.

`se_t` A 2x2 numeric matrix giving the external treatment group sum-of-squares matrix. Required when external treatment data are used; otherwise set to `NULL`.

`se_c` A 2x2 numeric matrix giving the external control group sum-of-squares matrix. Required when external control data are used; otherwise set to `NULL`.

`nMC` A positive integer giving the number of Monte Carlo draws used to estimate region probabilities. Default is `10000`. Required when `CalcMethod = 'MC'`. May be set to `NULL` when `CalcMethod = 'MM'` and $\nu_k > 4$ (the MM method

	uses `mvtnorm::pmvt` analytically); if `CalcMethod = 'MM'` but $\nu_k \leq 4$ causes a fallback to MC, `nMC` must be a positive integer.
`CalcMethod`	A character string specifying the computation method. Must be `'MC'` (Monte Carlo, default) or `'MM'` (Moment-Matching via `mvtnorm::pmvt`). When `CalcMethod = 'MM'` and $\nu_k \leq 4$, a warning is issued and the function falls back to `CalcMethod = 'MC'`.
`error_if_Miss`	Logical. If `TRUE` (default), the function stops with an error when positive Miss probability is obtained. If `FALSE`, Miss probability is handled according to `Gray_inc_Miss`.
`Gray_inc_Miss`	Logical. If `TRUE`, Miss probability is included in Gray probability. If `FALSE` (default), Miss probability is reported as a separate column. Active only when `error_if_Miss = FALSE`.
`seed`	A single numeric value. Seed for reproducible random number generation.

Details

The function follows the same structure as `pbayesdecisionprob1cont`:

1. For each scenario s, `nsim` datasets are simulated by generating treatment (and control) observations from $N_2(\mu_k^{(s)}, \Sigma_k)$. To minimise overhead, raw standardised residuals are generated *once* (scenario- invariant) and shifted by the scenario mean.
2. All `nsim` simulated sufficient statistics $(\bar{y}_{1,i}, S_{1,i})$ (and $(\bar{y}_{2,i}, S_{2,i})$ for controlled/external designs) are passed to `pbayespostpred2cont` in a *single vectorised call*, returning an $nsim \times n_{\text{regions}}$ matrix of region probabilities.
3. Go/NoGo/Miss probabilities are obtained as the column means of indicator matrices derived from the region probability matrix.

Value

A data frame of class `c("pbayesdecisionprob2cont", "data.frame")` with columns for the scenario parameters and Go, NoGo, Gray (and optionally Miss) probabilities. All input parameters are attached as attributes.

Examples

```
# Example 1: Controlled design, posterior probability, vague prior
Sigma <- matrix(c(4.0, 0.8, 0.8, 1.0), 2, 2)
pbayesdecisionprob2cont(
  nsim = 100L, prob = 'posterior', design = 'controlled',
  prior = 'vague',
  GoRegions = 1L, NoGoRegions = 9L,
  gamma_go = 0.8, gamma_nogo = 0.8,
  theta_TV1 = 1.5, theta_MAV1 = 0.5,
  theta_TV2 = 1.0, theta_MAV2 = 0.3,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  n_t = 20L, n_c = 20L, m_t = NULL, m_c = NULL,
  mu_t = rbind(c(1.0, 0.5), c(2.5, 1.5), c(4.0, 2.5)),
  Sigma_t = Sigma,
  mu_c = rbind(c(0.0, 0.0), c(0.0, 0.0), c(0.0, 0.0)),
```

```
  Sigma_c = Sigma,
  kappa0_t = NULL, nu0_t = NULL, mu0_t = NULL, Lambda0_t = NULL,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = NULL, Lambda0_c = NULL,
  r = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 500L, CalcMethod = 'MC',
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 1L
)

# Example 2: Uncontrolled design, posterior probability, NIW prior
Sigma <- matrix(c(4.0, 0.8, 0.8, 1.0), 2, 2)
L0    <- matrix(c(8.0, 0.0, 0.0, 2.0), 2, 2)
pbayesdecisionprob2cont(
  nsim = 100L, prob = 'posterior', design = 'uncontrolled',
  prior = 'N-Inv-Wishart',
  GoRegions = 1L, NoGoRegions = 9L,
  gamma_go = 0.8, gamma_nogo = 0.8,
  theta_TV1 = 1.5, theta_MAV1 = 0.5,
  theta_TV2 = 1.0, theta_MAV2 = 0.3,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  n_t = 20L, n_c = NULL, m_t = NULL, m_c = NULL,
  mu_t = rbind(c(1.0, 0.5), c(2.5, 1.5), c(4.0, 2.5)),
  Sigma_t = Sigma,
  mu_c = rbind(c(0.0, 0.0), c(0.0, 0.0), c(0.0, 0.0)),
  Sigma_c = Sigma,
  kappa0_t = 2.0, nu0_t = 5.0, mu0_t = c(2.0, 1.0), Lambda0_t = L0,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = c(0.0, 0.0), Lambda0_c = NULL,
  r = 1.0,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 500L, CalcMethod = 'MC',
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 3L
)

# Example 3: External design (control only), posterior probability, NIW prior
Sigma  <- matrix(c(4.0, 0.8, 0.8, 1.0), 2, 2)
L0     <- matrix(c(8.0, 0.0, 0.0, 2.0), 2, 2)
se_mat <- matrix(c(7.0, 1.2, 1.2, 1.8), 2, 2)
pbayesdecisionprob2cont(
  nsim = 100L, prob = 'posterior', design = 'external',
  prior = 'N-Inv-Wishart',
  GoRegions = 1L, NoGoRegions = 9L,
  gamma_go = 0.8, gamma_nogo = 0.8,
  theta_TV1 = 1.5, theta_MAV1 = 0.5,
  theta_TV2 = 1.0, theta_MAV2 = 0.3,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  n_t = 20L, n_c = 20L, m_t = NULL, m_c = NULL,
  mu_t = rbind(c(1.0, 0.5), c(2.5, 1.5), c(4.0, 2.5)),
  Sigma_t = Sigma,
  mu_c = rbind(c(0.0, 0.0), c(0.0, 0.0), c(0.0, 0.0)),
  Sigma_c = Sigma,
  kappa0_t = 2.0, nu0_t = 5.0, mu0_t = c(2.0, 1.0), Lambda0_t = L0,
```

```
  kappa0_c = 2.0, nu0_c = 5.0, mu0_c = c(0.0, 0.0), Lambda0_c = L0,
  r = NULL,
  ne_t = NULL, ne_c = 15L, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = c(0.2, 0.1), se_t = NULL, se_c = se_mat,
  nMC = 500L, CalcMethod = 'MC',
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 5L
)

# Example 4: Controlled design, predictive probability, NIW prior
Sigma <- matrix(c(4.0, 0.8, 0.8, 1.0), 2, 2)
L0    <- matrix(c(8.0, 0.0, 0.0, 2.0), 2, 2)
pbayesdecisionprob2cont(
  nsim = 100L, prob = 'predictive', design = 'controlled',
  prior = 'N-Inv-Wishart',
  GoRegions = 1L, NoGoRegions = 4L,
  gamma_go = 0.8, gamma_nogo = 0.8,
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.5, theta_NULL2 = 0.3,
  n_t = 20L, n_c = 20L, m_t = 60L, m_c = 60L,
  mu_t = rbind(c(1.0, 0.5), c(2.5, 1.5), c(4.0, 2.5)),
  Sigma_t = Sigma,
  mu_c = rbind(c(0.0, 0.0), c(0.0, 0.0), c(0.0, 0.0)),
  Sigma_c = Sigma,
  kappa0_t = 2.0, nu0_t = 5.0, mu0_t = c(2.0, 1.0), Lambda0_t = L0,
  kappa0_c = 2.0, nu0_c = 5.0, mu0_c = c(0.0, 0.0), Lambda0_c = L0,
  r = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 500L, CalcMethod = 'MC',
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 4L
)

# Example 5: Uncontrolled design, predictive probability, vague prior
Sigma <- matrix(c(4.0, 0.8, 0.8, 1.0), 2, 2)
pbayesdecisionprob2cont(
  nsim = 100L, prob = 'predictive', design = 'uncontrolled',
  prior = 'vague',
  GoRegions = 1L, NoGoRegions = 4L,
  gamma_go = 0.8, gamma_nogo = 0.8,
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.5, theta_NULL2 = 0.3,
  n_t = 20L, n_c = NULL, m_t = 60L, m_c = 60L,
  mu_t = rbind(c(1.0, 0.5), c(2.5, 1.5), c(4.0, 2.5)),
  Sigma_t = Sigma,
  mu_c = NULL,
  Sigma_c = NULL,
  kappa0_t = NULL, nu0_t = NULL, mu0_t = NULL, Lambda0_t = NULL,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = c(0.0, 0.0), Lambda0_c = NULL,
  r = 1.0,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
```

```
  nMC = 500L, CalcMethod = 'MC',
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 8L
)

# Example 6: External design (control only), predictive probability, NIW prior
Sigma  <- matrix(c(4.0, 0.8, 0.8, 1.0), 2, 2)
L0     <- matrix(c(8.0, 0.0, 0.0, 2.0), 2, 2)
se_mat <- matrix(c(7.0, 1.2, 1.2, 1.8), 2, 2)
pbayesdecisionprob2cont(
  nsim = 100L, prob = 'predictive', design = 'external',
  prior = 'N-Inv-Wishart',
  GoRegions = 1L, NoGoRegions = 4L,
  gamma_go = 0.8, gamma_nogo = 0.8,
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.5, theta_NULL2 = 0.3,
  n_t = 20L, n_c = 20L, m_t = 60L, m_c = 60L,
  mu_t = rbind(c(1.0, 0.5), c(2.5, 1.5), c(4.0, 2.5)),
  Sigma_t = Sigma,
  mu_c = rbind(c(0.0, 0.0), c(0.0, 0.0), c(0.0, 0.0)),
  Sigma_c = Sigma,
  kappa0_t = 2.0, nu0_t = 5.0, mu0_t = c(2.0, 1.0), Lambda0_t = L0,
  kappa0_c = 2.0, nu0_c = 5.0, mu0_c = c(0.0, 0.0), Lambda0_c = L0,
  r = NULL,
  ne_t = NULL, ne_c = 15L, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = c(0.2, 0.1), se_t = NULL, se_c = se_mat,
  nMC = 500L, CalcMethod = 'MC',
  error_if_Miss = TRUE, Gray_inc_Miss = FALSE, seed = 9L
)
```

`pbayespostpred1bin` *Bayesian Posterior or Posterior Predictive Probability for a Single Binary Endpoint*

Description

Computes the Bayesian posterior probability or posterior predictive probability for binary-outcome clinical trials under a beta-binomial conjugate model. The function supports controlled, uncontrolled, and external designs, with optional incorporation of external data through power priors. Vector inputs for `y_t` and `y_c` are supported for efficient batch processing (e.g., across all possible trial outcomes in `pbayesdecisionprob1bin`).

Usage

```
pbayespostpred1bin(
  prob = "posterior",
  design = "controlled",
  theta0,
```

```
  n_t,
  n_c,
  y_t,
  y_c = NULL,
  a_t,
  a_c,
  b_t,
  b_c,
  m_t = NULL,
  m_c = NULL,
  z = NULL,
  ne_t = NULL,
  ne_c = NULL,
  ye_t = NULL,
  ye_c = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  lower.tail = TRUE
)
```

Arguments

prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
theta0	A numeric scalar in (-1, 1) giving the threshold for the treatment effect (difference in response rates).
n_t	A positive integer giving the number of patients in the treatment group in the proof-of-concept (PoC) trial.
n_c	A positive integer giving the number of patients in the control group in the PoC trial. For design = 'uncontrolled', this is the hypothetical control sample size (required for consistency with other designs).
y_t	A non-negative integer or integer vector giving the observed number of responders in the treatment group (must satisfy 0 <= y_t <= n_t).
y_c	A non-negative integer or integer vector giving the number of responders in the control group (must satisfy 0 <= y_c <= n_c). Set to NULL for design = 'uncontrolled' and use z instead. When provided as a vector, must have the same length as y_t.
a_t	A positive numeric scalar giving the first shape parameter (alpha) of the prior Beta distribution for the treatment group.
a_c	A positive numeric scalar giving the first shape parameter (alpha) of the prior Beta distribution for the control group.
b_t	A positive numeric scalar giving the second shape parameter (beta) of the prior Beta distribution for the treatment group.

b_c	A positive numeric scalar giving the second shape parameter (beta) of the prior Beta distribution for the control group.
m_t	A positive integer giving the number of patients in the treatment group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.
m_c	A positive integer giving the number of patients in the control group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.
z	A non-negative integer giving the hypothetical number of responders in the control group. Required when design = 'uncontrolled'; otherwise set to NULL. When used, y_c should be NULL.
ne_t	A positive integer giving the number of patients in the treatment group of the external data set. Required when design = 'external' and external treatment data are available; otherwise set to NULL.
ne_c	A positive integer giving the number of patients in the control group of the external data set. Required when design = 'external' and external control data are available; otherwise set to NULL.
ye_t	A non-negative integer giving the number of responders in the treatment group of the external data set. Required when design = 'external'; otherwise set to NULL.
ye_c	A non-negative integer giving the number of responders in the control group of the external data set. Required when design = 'external'; otherwise set to NULL.
alpha0e_t	A numeric scalar in (0, 1] giving the power prior weight for the external treatment data. Required when external treatment data are used; otherwise set to NULL.
alpha0e_c	A numeric scalar in (0, 1] giving the power prior weight for the external control data. Required when external control data are used; otherwise set to NULL.
lower.tail	A logical scalar; if TRUE (default), the function returns $P(\text{effect} \leq \theta_0)$, otherwise $P(\text{effect} > \theta_0)$.

Details

Posterior shape parameters are computed from the beta-binomial conjugate model:

- Prior: $\pi_j \sim \text{Beta}(a_j, b_j)$.
- Posterior: $\pi_j \mid y_j \sim \text{Beta}(a_j + y_j, b_j + n_j - y_j)$.

For design = 'external', external data are incorporated through the power prior, which inflates the prior by the weighted external sufficient statistics:

$$(\alpha_j^*, \beta_j^*) = (a_j + ae_j \cdot ye_j, b_j + ae_j \cdot (ne_j - ye_j)).$$

For design = 'uncontrolled', the hypothetical control value z is used in place of y_c, and n_c acts as the hypothetical control sample size.

The final probability is obtained by calling pbetadiff for prob = 'posterior' or pbetabinomdiff for prob = 'predictive', applied element-wise via mapply.

Value

A numeric scalar or vector in `[0, 1]`. When `y_t` and `y_c` are vectors of length n, a vector of length n is returned.

Examples

```
# Example 1: Controlled design - posterior probability
pbayespostpred1bin(
  prob = 'posterior', design = 'controlled', theta0 = 0.15,
  n_t = 12, n_c = 15, y_t = 7, y_c = 5,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  m_t = NULL, m_c = NULL, z = NULL,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL, lower.tail = FALSE
)

# Example 2: Uncontrolled design - posterior probability
pbayespostpred1bin(
  prob = 'posterior', design = 'uncontrolled', theta0 = 0.20,
  n_t = 20, n_c = 20, y_t = 12, y_c = NULL,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  m_t = NULL, m_c = NULL, z = 3,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL, lower.tail = FALSE
)

# Example 3: External design - posterior probability
pbayespostpred1bin(
  prob = 'posterior', design = 'external', theta0 = 0.15,
  n_t = 12, n_c = 15, y_t = 7, y_c = 9,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  m_t = NULL, m_c = NULL, z = NULL,
  ne_t = 12, ne_c = 12, ye_t = 6, ye_c = 6, alpha0e_t = 0.5, alpha0e_c = 0.5,
  lower.tail = FALSE
)

# Example 4: Controlled design - posterior predictive probability
pbayespostpred1bin(
  prob = 'predictive', design = 'controlled', theta0 = 0.10,
  n_t = 12, n_c = 15, y_t = 7, y_c = 5,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  m_t = 30, m_c = 30, z = NULL,
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL, lower.tail = FALSE
)

# Example 5: Uncontrolled design - posterior predictive probability
pbayespostpred1bin(
  prob = 'predictive', design = 'uncontrolled', theta0 = 0.20,
  n_t = 20, n_c = 20, y_t = 12, y_c = NULL,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  m_t = 30, m_c = 30, z = 3,
```

```
  ne_t = NULL, ne_c = NULL, ye_t = NULL, ye_c = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL, lower.tail = FALSE
)

# Example 6: External design - posterior predictive probability
pbayespostpred1bin(
  prob = 'predictive', design = 'external', theta0 = 0.15,
  n_t = 12, n_c = 15, y_t = 7, y_c = 9,
  a_t = 0.5, a_c = 0.5, b_t = 0.5, b_c = 0.5,
  m_t = 30, m_c = 30, z = NULL,
  ne_t = 12, ne_c = 12, ye_t = 6, ye_c = 6, alpha0e_t = 0.5, alpha0e_c = 0.5,
  lower.tail = FALSE
)
```

`pbayespostpred1cont` *Bayesian Posterior or Posterior Predictive Probability for a Single Continuous Endpoint*

Description

Computes the Bayesian posterior probability or posterior predictive probability for continuous-outcome clinical trials under a Normal-Inverse-Chi-squared (or vague Jeffreys) conjugate model. The function supports controlled, uncontrolled, and external designs, with optional incorporation of external data through power priors. Vector inputs for `bar_y_t`, `s_t`, `bar_y_c`, and `s_c` are supported for efficient batch processing (e.g., across simulation replicates in `pbayesdecisionprob1cont`).

Usage

```
pbayespostpred1cont(
  prob = "posterior",
  design = "controlled",
  prior = "vague",
  CalcMethod = "NI",
  theta0,
  nMC = NULL,
  n_t,
  n_c = NULL,
  m_t = NULL,
  m_c = NULL,
  kappa0_t = NULL,
  kappa0_c = NULL,
  nu0_t = NULL,
  nu0_c = NULL,
  mu0_t = NULL,
  mu0_c = NULL,
  sigma0_t = NULL,
  sigma0_c = NULL,
```

```
  bar_y_t,
  bar_y_c = NULL,
  s_t,
  s_c = NULL,
  r = NULL,
  ne_t = NULL,
  ne_c = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  bar_ye_t = NULL,
  bar_ye_c = NULL,
  se_t = NULL,
  se_c = NULL,
  lower.tail = TRUE
)
```

Arguments

prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
prior	A character string specifying the prior type. Must be 'vague' (Jeffreys) or 'N-Inv-Chisq' (Normal-Inverse-Chi-squared conjugate).
CalcMethod	A character string specifying the calculation method. Must be 'NI' (numerical integration), 'MC' (Monte Carlo), or 'MM' (Moment-Matching approximation). For large nsim or multi-scenario use, 'MM' is strongly recommended; see Details.
theta0	A numeric scalar giving the threshold for the treatment effect (difference in means).
nMC	A positive integer giving the number of Monte Carlo draws. Required when CalcMethod = 'MC'; otherwise NULL.
n_t	A positive integer giving the number of patients in the treatment group in the proof-of-concept (PoC) trial.
n_c	A positive integer giving the number of patients in the control group in the PoC trial. Required for design = 'controlled' or 'external'; set to NULL for design = 'uncontrolled'.
m_t	A positive integer giving the number of patients in the treatment group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.
m_c	A positive integer giving the number of patients in the control group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.
kappa0_t	A positive numeric scalar giving the prior precision parameter for the treatment group. Required when prior = 'N-Inv-Chisq'; otherwise NULL.
kappa0_c	A positive numeric scalar giving the prior precision parameter for the control group. Required when prior = 'N-Inv-Chisq' and design = 'controlled'; otherwise NULL.

`nu0_t` A positive numeric scalar giving the prior degrees of freedom for the treatment group. Required when `prior = 'N-Inv-Chisq'`; otherwise `NULL`.

`nu0_c` A positive numeric scalar giving the prior degrees of freedom for the control group. Required when `prior = 'N-Inv-Chisq'` and `design = 'controlled'`; otherwise `NULL`.

`mu0_t` A numeric scalar giving the prior mean for the treatment group. Required when `prior = 'N-Inv-Chisq'`; otherwise `NULL`.

`mu0_c` A numeric scalar giving the prior mean for the control group (`design = 'controlled'` with `prior = 'N-Inv-Chisq'`) or the hypothetical control mean (`design = 'uncontrolled'`). Required for uncontrolled design; otherwise `NULL`.

`sigma0_t` A positive numeric scalar giving the prior scale for the treatment group. Required when `prior = 'N-Inv-Chisq'`; otherwise `NULL`.

`sigma0_c` A positive numeric scalar giving the prior scale for the control group. Required when `prior = 'N-Inv-Chisq'` and `design = 'controlled'`; otherwise `NULL`.

`bar_y_t` A numeric scalar or vector giving the sample mean for the treatment group. When a vector of length `nsim` is supplied, all posterior parameters are computed simultaneously for all replicates.

`bar_y_c` A numeric scalar or vector giving the sample mean for the control group. Required for `design = 'controlled'` or `'external'`; set to `NULL` for uncontrolled design.

`s_t` A positive numeric scalar or vector giving the sample standard deviation for the treatment group.

`s_c` A positive numeric scalar or vector giving the sample standard deviation for the control group. Required for `design = 'controlled'` or `'external'`; otherwise `NULL`.

`r` A positive numeric scalar giving the variance scaling factor for the hypothetical control. Required for `design = 'uncontrolled'`; otherwise `NULL`. The hypothetical control scale is $\mathrm{sd.control} = \sqrt{r} \cdot \mathrm{sd.treatment}$.

`ne_t` A positive integer giving the number of patients in the treatment group of the external data set. Required when `design = 'external'` and external treatment data are available; otherwise set to `NULL`.

`ne_c` A positive integer giving the number of patients in the control group of the external data set. Required when `design = 'external'` and external control data are available; otherwise set to `NULL`.

`alpha0e_t` A numeric scalar in `(0, 1]` giving the power prior weight for the external treatment data. Required when external treatment data are used; otherwise set to `NULL`.

`alpha0e_c` A numeric scalar in `(0, 1]` giving the power prior weight for the external control data. Required when external control data are used; otherwise set to `NULL`.

`bar_ye_t` A numeric scalar giving the external treatment group sample mean. Required when `ne_t` is provided; otherwise `NULL`.

`bar_ye_c` A numeric scalar giving the external control group sample mean. Required when `ne_c` is provided; otherwise `NULL`.

se_t	A positive numeric scalar giving the external treatment group sample SD. Required when ne_t is provided; otherwise NULL.
se_c	A positive numeric scalar giving the external control group sample SD. Required when ne_c is provided; otherwise NULL.
lower.tail	A logical scalar; if TRUE (default), the function returns $P(\text{effect} \leq \theta_0)$, otherwise $P(\text{effect} > \theta_0)$.

Details

Under the Normal-Inverse-Chi-squared (N-Inv-ChiSq) conjugate model, the marginal posterior/predictive distribution of the group mean follows a non-standardised t-distribution, parameterised by a location μ_j, a scale σ_j, and degrees of freedom ν_j that depend on the design and probability type. The final probability is computed by one of three methods:

- NI: exact numerical integration via ptdiff_NI.
- MC: Monte Carlo simulation via ptdiff_MC.
- MM: Moment-Matching approximation via ptdiff_MM.

Performance note: CalcMethod = 'NI' calls integrate() once per element of the input vector, which is prohibitively slow for large vectors (e.g., nsim x n_scenarios). CalcMethod = 'MC' creates an nMC x n matrix internally, consuming $O(nMC \cdot n)$ memory. For large-scale simulation use CalcMethod = 'MM', which is fully vectorised and exact in the normal limit.

Value

A numeric scalar or vector in [0, 1]. When the input data parameters (bar_y_t, etc.) are vectors of length n, a vector of length n is returned.

Examples

```
# Example 1: Controlled design - posterior probability with N-Inv-Chisq
pbayespostpred1cont(
  prob = 'posterior', design = 'controlled', prior = 'N-Inv-Chisq',
  CalcMethod = 'NI', theta0 = 2, n_t = 12, n_c = 12,
  kappa0_t = 5, kappa0_c = 5, nu0_t = 5, nu0_c = 5,
  mu0_t = 5, mu0_c = 5, sigma0_t = sqrt(5), sigma0_c = sqrt(5),
  bar_y_t = 2, bar_y_c = 0, s_t = 1, s_c = 1,
  m_t = NULL, m_c = NULL, r = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  lower.tail = FALSE
)

# Example 2: Uncontrolled design - posterior probability with vague prior
pbayespostpred1cont(
  prob = 'posterior', design = 'uncontrolled', prior = 'vague',
  CalcMethod = 'MM', theta0 = 1.5, n_t = 15, n_c = NULL,
  bar_y_t = 3.5, bar_y_c = NULL, s_t = 1.2, s_c = NULL,
  mu0_c = 1.5, r = 1.2,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
```

```
  mu0_t = NULL, sigma0_t = NULL, sigma0_c = NULL,
  m_t = NULL, m_c = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  lower.tail = FALSE
)

# Example 3: External design - posterior probability
pbayespostpred1cont(
  prob = 'posterior', design = 'external', prior = 'N-Inv-Chisq',
  CalcMethod = 'MM', theta0 = 2, n_t = 12, n_c = 12,
  kappa0_t = 5, kappa0_c = 5, nu0_t = 5, nu0_c = 5,
  mu0_t = 5, mu0_c = 5, sigma0_t = sqrt(5), sigma0_c = sqrt(5),
  bar_y_t = 2.5, bar_y_c = 1, s_t = 1.1, s_c = 0.9,
  m_t = NULL, m_c = NULL, r = NULL,
  ne_t = 10, ne_c = 10, alpha0e_t = 0.5, alpha0e_c = 0.5,
  bar_ye_t = 2, bar_ye_c = 0.5, se_t = 1, se_c = 0.8,
  lower.tail = FALSE
)

# Example 4: Controlled design - posterior predictive probability
pbayespostpred1cont(
  prob = 'predictive', design = 'controlled', prior = 'N-Inv-Chisq',
  CalcMethod = 'MM', theta0 = 1, n_t = 12, n_c = 12,
  kappa0_t = 5, kappa0_c = 5, nu0_t = 5, nu0_c = 5,
  mu0_t = 5, mu0_c = 5, sigma0_t = sqrt(5), sigma0_c = sqrt(5),
  bar_y_t = 2, bar_y_c = 0, s_t = 1, s_c = 1,
  m_t = 20, m_c = 20, r = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  lower.tail = FALSE
)

# Example 5: Uncontrolled design - posterior predictive probability
pbayespostpred1cont(
  prob = 'predictive', design = 'uncontrolled', prior = 'vague',
  CalcMethod = 'MM', theta0 = 1.5, n_t = 15, n_c = NULL,
  bar_y_t = 3.5, bar_y_c = NULL, s_t = 1.2, s_c = NULL,
  mu0_c = 1.5, r = 1.2,
  kappa0_t = NULL, kappa0_c = NULL, nu0_t = NULL, nu0_c = NULL,
  mu0_t = NULL, sigma0_t = NULL, sigma0_c = NULL,
  m_t = 20, m_c = 20,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  lower.tail = FALSE
)

# Example 6: External design - posterior predictive probability
pbayespostpred1cont(
  prob = 'predictive', design = 'external', prior = 'N-Inv-Chisq',
  CalcMethod = 'MM', theta0 = 1, n_t = 12, n_c = 12,
  kappa0_t = 5, kappa0_c = 5, nu0_t = 5, nu0_c = 5,
  mu0_t = 5, mu0_c = 5, sigma0_t = sqrt(5), sigma0_c = sqrt(5),
```

```
  bar_y_t = 2.5, bar_y_c = 1, s_t = 1.1, s_c = 0.9,
  m_t = 20, m_c = 20, r = NULL,
  ne_t = 10, ne_c = 10, alpha0e_t = 0.5, alpha0e_c = 0.5,
  bar_ye_t = 2, bar_ye_c = 0.5, se_t = 1, se_c = 0.8,
  lower.tail = FALSE
)
```

`pbayespostpred2bin` *Bayesian Posterior or Posterior Predictive Probability for Two Binary Endpoints*

Description

Computes the Bayesian posterior probability or posterior predictive probability for clinical trials with two binary endpoints under a Dirichlet-multinomial conjugate model. The function returns probabilities for nine decision regions (posterior) or four decision regions (predictive) defined by target values (TV) and minimum acceptable values (MAV) for both endpoints. Three study designs are supported: controlled, uncontrolled (hypothetical control), and external (power prior).

Usage

```
pbayespostpred2bin(
  prob = "posterior",
  design = "controlled",
  theta_TV1 = NULL,
  theta_MAV1 = NULL,
  theta_TV2 = NULL,
  theta_MAV2 = NULL,
  theta_NULL1 = NULL,
  theta_NULL2 = NULL,
  x_t_00,
  x_t_01,
  x_t_10,
  x_t_11,
  x_c_00 = NULL,
  x_c_01 = NULL,
  x_c_10 = NULL,
  x_c_11 = NULL,
  a_t_00,
  a_t_01,
  a_t_10,
  a_t_11,
  a_c_00,
  a_c_01,
  a_c_10,
  a_c_11,
```

```
  m_t = NULL,
  m_c = NULL,
  z00 = NULL,
  z01 = NULL,
  z10 = NULL,
  z11 = NULL,
  xe_t_00 = NULL,
  xe_t_01 = NULL,
  xe_t_10 = NULL,
  xe_t_11 = NULL,
  xe_c_00 = NULL,
  xe_c_01 = NULL,
  xe_c_10 = NULL,
  xe_c_11 = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  nMC = 10000L
)
```

Arguments

prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
theta_TV1	A numeric scalar giving the target value (TV) threshold for Endpoint 1. Required when prob = 'posterior'; must satisfy theta_TV1 > theta_MAV1. Set to NULL when prob = 'predictive'.
theta_MAV1	A numeric scalar giving the minimum acceptable value (MAV) threshold for Endpoint 1. Required when prob = 'posterior'; must satisfy theta_TV1 > theta_MAV1. Set to NULL when prob = 'predictive'.
theta_TV2	A numeric scalar giving the target value (TV) threshold for Endpoint 2. Required when prob = 'posterior'; must satisfy theta_TV2 > theta_MAV2. Set to NULL when prob = 'predictive'.
theta_MAV2	A numeric scalar giving the minimum acceptable value (MAV) threshold for Endpoint 2. Required when prob = 'posterior'; must satisfy theta_TV2 > theta_MAV2. Set to NULL when prob = 'predictive'.
theta_NULL1	A numeric scalar giving the null hypothesis threshold for Endpoint 1. Required when prob = 'predictive'; set to NULL when prob = 'posterior'.
theta_NULL2	A numeric scalar giving the null hypothesis threshold for Endpoint 2. Required when prob = 'predictive'; set to NULL when prob = 'posterior'.
x_t_00	A non-negative integer giving the count of (0, 0) responses in the treatment group (Endpoint 1 = 0, Endpoint 2 = 0).
x_t_01	A non-negative integer giving the count of (0, 1) responses in the treatment group.

x_t_10 — A non-negative integer giving the count of (1, 0) responses in the treatment group.

x_t_11 — A non-negative integer giving the count of (1, 1) responses in the treatment group.

x_c_00 — A non-negative integer giving the count of (0, 0) responses in the control group. Not used when design = 'uncontrolled'; set to NULL in that case.

x_c_01 — A non-negative integer giving the count of (0, 1) responses in the control group. Not used when design = 'uncontrolled'; set to NULL in that case.

x_c_10 — A non-negative integer giving the count of (1, 0) responses in the control group. Not used when design = 'uncontrolled'; set to NULL in that case.

x_c_11 — A non-negative integer giving the count of (1, 1) responses in the control group. Not used when design = 'uncontrolled'; set to NULL in that case.

a_t_00 — A positive numeric scalar giving the Dirichlet prior parameter for the (0, 0) response pattern in the treatment group.

a_t_01 — A positive numeric scalar giving the Dirichlet prior parameter for the (0, 1) response pattern in the treatment group.

a_t_10 — A positive numeric scalar giving the Dirichlet prior parameter for the (1, 0) response pattern in the treatment group.

a_t_11 — A positive numeric scalar giving the Dirichlet prior parameter for the (1, 1) response pattern in the treatment group.

a_c_00 — A positive numeric scalar giving the Dirichlet prior parameter for the (0, 0) response pattern in the control group. For design = 'uncontrolled', serves as a hyperparameter of the hypothetical control distribution.

a_c_01 — A positive numeric scalar giving the Dirichlet prior parameter for the (0, 1) response pattern in the control group. For design = 'uncontrolled', serves as a hyperparameter of the hypothetical control distribution.

a_c_10 — A positive numeric scalar giving the Dirichlet prior parameter for the (1, 0) response pattern in the control group. For design = 'uncontrolled', serves as a hyperparameter of the hypothetical control distribution.

a_c_11 — A positive numeric scalar giving the Dirichlet prior parameter for the (1, 1) response pattern in the control group. For design = 'uncontrolled', serves as a hyperparameter of the hypothetical control distribution.

m_t — A positive integer giving the number of patients in the treatment group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

m_c — A positive integer giving the number of patients in the control group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

z00 — A non-negative integer giving the hypothetical control count for pattern (0, 0). Required when design = 'uncontrolled'; otherwise set to NULL.

z01 — A non-negative integer giving the hypothetical control count for pattern (0, 1). Required when design = 'uncontrolled'; otherwise set to NULL.

z10 — A non-negative integer giving the hypothetical control count for pattern (1, 0). Required when design = 'uncontrolled'; otherwise set to NULL.

`z11` A non-negative integer giving the hypothetical control count for pattern `(1, 1)`. Required when `design = 'uncontrolled'`; otherwise set to `NULL`.

`xe_t_00` A non-negative integer giving the external treatment group count for pattern `(0, 0)`. Required when `design = 'external'` and external treatment data are used; otherwise `NULL`.

`xe_t_01` A non-negative integer giving the external treatment group count for pattern `(0, 1)`. Required for external treatment data; otherwise `NULL`.

`xe_t_10` A non-negative integer giving the external treatment group count for pattern `(1, 0)`. Required for external treatment data; otherwise `NULL`.

`xe_t_11` A non-negative integer giving the external treatment group count for pattern `(1, 1)`. Required for external treatment data; otherwise `NULL`.

`xe_c_00` A non-negative integer giving the external control group count for pattern `(0, 0)`. Required when `design = 'external'` and external control data are used; otherwise `NULL`.

`xe_c_01` A non-negative integer giving the external control group count for pattern `(0, 1)`. Required for external control data; otherwise `NULL`.

`xe_c_10` A non-negative integer giving the external control group count for pattern `(1, 0)`. Required for external control data; otherwise `NULL`.

`xe_c_11` A non-negative integer giving the external control group count for pattern `(1, 1)`. Required for external control data; otherwise `NULL`.

`alpha0e_t` A numeric scalar in `(0, 1]` giving the power prior weight for the external treatment data. Required when external treatment data are used; otherwise set to `NULL`.

`alpha0e_c` A numeric scalar in `(0, 1]` giving the power prior weight for the external control data. Required when external control data are used; otherwise set to `NULL`.

`nMC` A positive integer giving the number of Monte Carlo draws used to estimate region probabilities. Default is `10000`. Larger values reduce Monte Carlo error at the cost of computation time.

Details

Model. The four response categories are ordered as `(0, 0)`, `(0, 1)`, `(1, 0)`, `(1, 1)`. For each group j, the observed count vector follows a multinomial distribution, and a conjugate Dirichlet prior is placed on the cell probability vector p_j:

$$p_j \sim \mathrm{Dir}(\alpha_{j,00}, \alpha_{j,01}, \alpha_{j,10}, \alpha_{j,11}).$$

The posterior is

$$p_j \mid x_j \sim \mathrm{Dir}(\alpha_{j,00} + x_{j,00}, \alpha_{j,01} + x_{j,01}, \alpha_{j,10} + x_{j,10}, \alpha_{j,11} + x_{j,11}).$$

Marginal response rates are $\pi_{j1} = p_{j,10} + p_{j,11}$ (Endpoint 1) and $\pi_{j2} = p_{j,01} + p_{j,11}$ (Endpoint 2). Treatment effects are $\theta_1 = \pi_{t1} - \pi_{c1}$ and $\theta_2 = \pi_{t2} - \pi_{c2}$.

Uncontrolled design. When `design = 'uncontrolled'`, the hypothetical control distribution is specified as a Dirichlet distribution directly via the prior hyperparameters and hypothetical control counts `z00`, `z01`, `z10`, `z11`.

External design. When `design = 'external'`, the power prior augments the Dirichlet prior parameters.

Value

A named numeric vector of region probabilities. For `prob = 'posterior'`: length 9, named `R1`, ..., `R9`, corresponding to regions defined by TV and MAV thresholds for both endpoints (row-major order: Endpoint 1 varies slowest). For `prob = 'predictive'`: length 4, named `R1`, ..., `R4`. All elements are non-negative and sum to 1.

Examples

```
# Example 1: Controlled design - posterior probability
pbayespostpred2bin(
  prob = 'posterior', design = 'controlled',
  theta_TV1 = 0.15, theta_MAV1 = 0.05, theta_TV2 = 0.10, theta_MAV2 = 0.0,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  x_t_00 = 2, x_t_01 = 3, x_t_10 = 5, x_t_11 = 2,
  x_c_00 = 3, x_c_01 = 2, x_c_10 = 4, x_c_11 = 1,
  a_t_00 = 1, a_t_01 = 1, a_t_10 = 1, a_t_11 = 1,
  a_c_00 = 1, a_c_01 = 1, a_c_10 = 1, a_c_11 = 1,
  m_t = NULL, m_c = NULL,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL
)

# Example 2: Uncontrolled design - posterior probability
pbayespostpred2bin(
  prob = 'posterior', design = 'uncontrolled',
  theta_TV1 = 0.15, theta_MAV1 = 0.05, theta_TV2 = 0.10, theta_MAV2 = 0.0,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  x_t_00 = 2, x_t_01 = 3, x_t_10 = 5, x_t_11 = 2,
  x_c_00 = NULL, x_c_01 = NULL, x_c_10 = NULL, x_c_11 = NULL,
  a_t_00 = 1, a_t_01 = 1, a_t_10 = 1, a_t_11 = 1,
  a_c_00 = 1, a_c_01 = 1, a_c_10 = 1, a_c_11 = 1,
  m_t = NULL, m_c = NULL,
  z00 = 1, z01 = 2, z10 = 2, z11 = 1,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL
)

# Example 3: External design - posterior probability
pbayespostpred2bin(
  prob = 'posterior', design = 'external',
  theta_TV1 = 0.15, theta_MAV1 = 0.05, theta_TV2 = 0.10, theta_MAV2 = 0.0,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  x_t_00 = 2, x_t_01 = 3, x_t_10 = 5, x_t_11 = 2,
  x_c_00 = 3, x_c_01 = 2, x_c_10 = 4, x_c_11 = 1,
  a_t_00 = 1, a_t_01 = 1, a_t_10 = 1, a_t_11 = 1,
  a_c_00 = 1, a_c_01 = 1, a_c_10 = 1, a_c_11 = 1,
  m_t = NULL, m_c = NULL,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = 2, xe_t_01 = 2, xe_t_10 = 3, xe_t_11 = 1,
```

```
  xe_c_00 = 2, xe_c_01 = 1, xe_c_10 = 2, xe_c_11 = 1,
  alpha0e_t = 0.5, alpha0e_c = 0.5
)

# Example 4: Controlled design - posterior predictive probability
pbayespostpred2bin(
  prob = 'predictive', design = 'controlled',
  theta_TV1 = NULL, theta_MAV1 = NULL, theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.05, theta_NULL2 = 0.0,
  x_t_00 = 2, x_t_01 = 3, x_t_10 = 5, x_t_11 = 2,
  x_c_00 = 3, x_c_01 = 2, x_c_10 = 4, x_c_11 = 1,
  a_t_00 = 1, a_t_01 = 1, a_t_10 = 1, a_t_11 = 1,
  a_c_00 = 1, a_c_01 = 1, a_c_10 = 1, a_c_11 = 1,
  m_t = 20, m_c = 20,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL
)

# Example 5: Uncontrolled design - posterior predictive probability
pbayespostpred2bin(
  prob = 'predictive', design = 'uncontrolled',
  theta_TV1 = NULL, theta_MAV1 = NULL, theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.05, theta_NULL2 = 0.0,
  x_t_00 = 2, x_t_01 = 3, x_t_10 = 5, x_t_11 = 2,
  x_c_00 = NULL, x_c_01 = NULL, x_c_10 = NULL, x_c_11 = NULL,
  a_t_00 = 1, a_t_01 = 1, a_t_10 = 1, a_t_11 = 1,
  a_c_00 = 1, a_c_01 = 1, a_c_10 = 1, a_c_11 = 1,
  m_t = 20, m_c = 20,
  z00 = 1, z01 = 2, z10 = 2, z11 = 1,
  xe_t_00 = NULL, xe_t_01 = NULL, xe_t_10 = NULL, xe_t_11 = NULL,
  xe_c_00 = NULL, xe_c_01 = NULL, xe_c_10 = NULL, xe_c_11 = NULL,
  alpha0e_t = NULL, alpha0e_c = NULL
)

# Example 6: External design - posterior predictive probability
pbayespostpred2bin(
  prob = 'predictive', design = 'external',
  theta_TV1 = NULL, theta_MAV1 = NULL, theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.05, theta_NULL2 = 0.0,
  x_t_00 = 2, x_t_01 = 3, x_t_10 = 5, x_t_11 = 2,
  x_c_00 = 3, x_c_01 = 2, x_c_10 = 4, x_c_11 = 1,
  a_t_00 = 1, a_t_01 = 1, a_t_10 = 1, a_t_11 = 1,
  a_c_00 = 1, a_c_01 = 1, a_c_10 = 1, a_c_11 = 1,
  m_t = 20, m_c = 20,
  z00 = NULL, z01 = NULL, z10 = NULL, z11 = NULL,
  xe_t_00 = 2, xe_t_01 = 2, xe_t_10 = 3, xe_t_11 = 1,
  xe_c_00 = 2, xe_c_01 = 1, xe_c_10 = 2, xe_c_11 = 1,
  alpha0e_t = 0.5, alpha0e_c = 0.5
)
```

pbayespostpred2cont *Region Probabilities for Two Continuous Endpoints*

Description

Computes the posterior or posterior predictive probability that the bivariate treatment effect $\theta = \mu_t - \mu_c$ falls in each of the 9 (posterior) or 4 (predictive) rectangular regions defined by the decision thresholds.

Usage

```
pbayespostpred2cont(
  prob,
  design,
  prior,
  theta_TV1 = NULL,
  theta_MAV1 = NULL,
  theta_TV2 = NULL,
  theta_MAV2 = NULL,
  theta_NULL1 = NULL,
  theta_NULL2 = NULL,
  n_t,
  n_c = NULL,
  ybar_t,
  S_t,
  ybar_c = NULL,
  S_c = NULL,
  m_t = NULL,
  m_c = NULL,
  kappa0_t = NULL,
  nu0_t = NULL,
  mu0_t = NULL,
  Lambda0_t = NULL,
  kappa0_c = NULL,
  nu0_c = NULL,
  mu0_c = NULL,
  Lambda0_c = NULL,
  r = NULL,
  ne_t = NULL,
  ne_c = NULL,
  alpha0e_t = NULL,
  alpha0e_c = NULL,
  bar_ye_t = NULL,
  bar_ye_c = NULL,
  se_t = NULL,
  se_c = NULL,
  nMC = 10000L,
```

```
  CalcMethod = "MC"
)
```

Arguments

prob	A character string specifying the probability type. Must be 'posterior' or 'predictive'.
design	A character string specifying the trial design. Must be 'controlled', 'uncontrolled', or 'external'.
prior	A character string specifying the prior distribution. Must be 'vague' or 'N-Inv-Wishart'.
theta_TV1	A numeric scalar giving the target value (TV) threshold for Endpoint 1. Required when prob = 'posterior'; must satisfy theta_TV1 > theta_MAV1. Set to NULL when prob = 'predictive'.
theta_MAV1	A numeric scalar giving the minimum acceptable value (MAV) threshold for Endpoint 1. Required when prob = 'posterior'; must satisfy theta_TV1 > theta_MAV1. Set to NULL when prob = 'predictive'.
theta_TV2	A numeric scalar giving the target value (TV) threshold for Endpoint 2. Required when prob = 'posterior'; must satisfy theta_TV2 > theta_MAV2. Set to NULL when prob = 'predictive'.
theta_MAV2	A numeric scalar giving the minimum acceptable value (MAV) threshold for Endpoint 2. Required when prob = 'posterior'; must satisfy theta_TV2 > theta_MAV2. Set to NULL when prob = 'predictive'.
theta_NULL1	A numeric scalar giving the null hypothesis threshold for Endpoint 1. Required when prob = 'predictive'; set to NULL when prob = 'posterior'.
theta_NULL2	A numeric scalar giving the null hypothesis threshold for Endpoint 2. Required when prob = 'predictive'; set to NULL when prob = 'posterior'.
n_t	A positive integer giving the number of patients in the treatment group in the proof-of-concept (PoC) trial.
n_c	A positive integer giving the number of patients in the control group in the PoC trial. For design = 'uncontrolled', this is the hypothetical control sample size (required for consistency with other designs).
ybar_t	A numeric vector of length 2 **or** a numeric matrix with 2 columns giving the sample mean(s) for the treatment group. When a matrix with N rows is supplied, the function computes probabilities for all N observations simultaneously and returns an $N \times n_{\text{regions}}$ matrix.
S_t	A 2x2 numeric matrix giving the sum-of-squares matrix for the treatment group (single observation), **or** a list of N such matrices (vectorised call). Must be consistent with ybar_t: if ybar_t is a matrix, S_t must be a list of the same length.
ybar_c	A numeric vector of length 2, a numeric matrix with 2 columns, or NULL giving the sample mean(s) for the control group. Not used when design = 'uncontrolled'.
S_c	A 2x2 numeric matrix, a list of 2x2 matrices, or NULL giving the sum-of-squares matrix/matrices for the control group. Not used when design = 'uncontrolled'.
m_t	A positive integer giving the number of patients in the treatment group for the future trial. Required when prob = 'predictive'; otherwise set to NULL.

`m_c` A positive integer giving the number of patients in the control group for the future trial. Required when `prob = 'predictive'`; otherwise set to `NULL`.

`kappa0_t` A positive numeric scalar giving the NIW prior concentration parameter for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`nu0_t` A numeric scalar giving the NIW prior degrees of freedom for the treatment group. Must be greater than 3. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`mu0_t` A numeric vector of length 2 giving the NIW prior mean for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`Lambda0_t` A 2x2 positive-definite numeric matrix giving the NIW prior scale matrix for the treatment group. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`kappa0_c` A positive numeric scalar giving the NIW prior concentration parameter for the control group. Required when `prior = 'N-Inv-Wishart'` and `design != 'uncontrolled'`; otherwise set to `NULL`.

`nu0_c` A numeric scalar giving the NIW prior degrees of freedom for the control group. Must be greater than 3. Required when `prior = 'N-Inv-Wishart'` and `design != 'uncontrolled'`; otherwise set to `NULL`.

`mu0_c` A numeric vector of length 2 giving the NIW prior mean for the control group, or the hypothetical control location when `design = 'uncontrolled'`. Required when `prior = 'N-Inv-Wishart'`; otherwise set to `NULL`.

`Lambda0_c` A 2x2 positive-definite numeric matrix giving the NIW prior scale matrix for the control group. Required when `prior = 'N-Inv-Wishart'` and `design != 'uncontrolled'`; otherwise set to `NULL`.

`r` A positive numeric scalar giving the variance scaling factor for the hypothetical control distribution. Required when `design = 'uncontrolled'`; otherwise set to `NULL`.

`ne_t` A positive integer giving the external treatment group sample size. Required when `design = 'external'` and external treatment data are used; otherwise set to `NULL`.

`ne_c` A positive integer giving the external control group sample size. Required when `design = 'external'` and external control data are used; otherwise set to `NULL`.

`alpha0e_t` A numeric scalar in `(0, 1]` giving the power prior weight for the external treatment data. Required when external treatment data are used; otherwise set to `NULL`.

`alpha0e_c` A numeric scalar in `(0, 1]` giving the power prior weight for the external control data. Required when external control data are used; otherwise set to `NULL`.

`bar_ye_t` A numeric vector of length 2 giving the external treatment group sample mean. Required when external treatment data are used; otherwise set to `NULL`.

`bar_ye_c` A numeric vector of length 2 giving the external control group sample mean. Required when external control data are used; otherwise set to `NULL`.

`se_t` A 2x2 numeric matrix giving the external treatment group sum-of-squares matrix. Required when external treatment data are used; otherwise set to `NULL`.

`se_c` A 2x2 numeric matrix giving the external control group sum-of-squares matrix. Required when external control data are used; otherwise set to `NULL`.

`nMC` A positive integer giving the number of Monte Carlo draws used to estimate region probabilities. Default is `10000`. Required when `CalcMethod = 'MC'`. May be set to `NULL` when `CalcMethod = 'MM'` and $\nu_k > 4$ (the MM method uses `mvtnorm::pmvt` analytically); if `CalcMethod = 'MM'` but $\nu_k \leq 4$ causes a fallback to MC, `nMC` must be a positive integer.

`CalcMethod` A character string specifying the computation method. Must be `'MC'` (Monte Carlo, default) or `'MM'` (Moment-Matching via `mvtnorm::pmvt`). When `CalcMethod = 'MM'` and $\nu_k \leq 4$, a warning is issued and the function falls back to `CalcMethod = 'MC'`.

Details

Model. Both endpoints follow a bivariate Normal distribution $y_{k,j} \sim N_2(\mu_k, \Sigma_k)$ for group $k \in \{t, c\}$. The treatment effect is $\theta = \mu_t - \mu_c$.

Posterior distribution (vague prior).

$$\mu_k | Y_k \sim t_{n_k-2}\left(\bar{y}_k,\ \frac{S_k}{n_k(n_k-2)}\right)$$

Posterior distribution (NIW prior).

$$\mu_k | Y_k \sim t_{\nu_{nk}-1}\left(\mu_{nk},\ \frac{\Lambda_{nk}}{\kappa_{nk}(\nu_{nk}-1)}\right)$$

with updated hyperparameters $\kappa_{nk} = \kappa_{0k} + n_k$, $\nu_{nk} = \nu_{0k} + n_k$, $\mu_{nk} = (\kappa_{0k}\mu_{0k} + n_k\bar{y}_k)/\kappa_{nk}$, and $\Lambda_{nk} = \Lambda_{0k} + S_k + \kappa_{0k}n_k(\bar{y}_k - \mu_{0k})(\bar{y}_k - \mu_{0k})^T/\kappa_{nk}$.

Predictive distribution. The scale matrix of a single future observation is inflated by $(1 + n_k)/n_k$ (vague) or $(1 + \kappa_{nk})/\kappa_{nk}$ (NIW) relative to the posterior. The mean of m_k future observations has scale divided by m_k.

Posterior probability regions (prob = 'posterior'). Row-major 3x3 grid; Endpoint 1 varies slowest:

- R1: $\theta_1 > TV_1$ AND $\theta_2 > TV_2$
- R2: $\theta_1 > TV_1$ AND $TV_2 \geq \theta_2 > MAV_2$
- R3: $\theta_1 > TV_1$ AND $\theta_2 \leq MAV_2$
- R4: $TV_1 \geq \theta_1 > MAV_1$ AND $\theta_2 > TV_2$
- R5: $TV_1 \geq \theta_1 > MAV_1$ AND $TV_2 \geq \theta_2 > MAV_2$
- R6: $TV_1 \geq \theta_1 > MAV_1$ AND $\theta_2 \leq MAV_2$
- R7: $\theta_1 \leq MAV_1$ AND $\theta_2 > TV_2$
- R8: $\theta_1 \leq MAV_1$ AND $TV_2 \geq \theta_2 > MAV_2$
- R9: $\theta_1 \leq MAV_1$ AND $\theta_2 \leq MAV_2$

Predictive probability regions (prob = 'predictive'). Row-major 2x2 grid; Endpoint 1 varies slowest:

- R1: $\tilde{\theta}_1 > \theta_{\mathrm{NULL1}}$ AND $\tilde{\theta}_2 > \theta_{\mathrm{NULL2}}$
- R2: $\tilde{\theta}_1 > \theta_{\mathrm{NULL1}}$ AND $\tilde{\theta}_2 \leq \theta_{\mathrm{NULL2}}$
- R3: $\tilde{\theta}_1 \leq \theta_{\mathrm{NULL1}}$ AND $\tilde{\theta}_2 > \theta_{\mathrm{NULL2}}$
- R4: $\tilde{\theta}_1 \leq \theta_{\mathrm{NULL1}}$ AND $\tilde{\theta}_2 \leq \theta_{\mathrm{NULL2}}$

Uncontrolled design. Under NIW prior: $\mu_c \sim t_{\nu_{nt}-1}(\mu_{0c},\ r\Lambda_{nt}/[\kappa_{nt}(\nu_{nt}-1)])$. The parameter `r` allows the variance scale of the hypothetical control to differ from the treatment group.

External design. Incorporated via the power prior with NIW conjugate representation. The effective posterior hyperparameters are obtained by constructing the power prior from external data with weight a_0, then updating with current PoC data (see Conjugate_power_prior.pdf, Theorem 5).

Vectorised usage. When `ybar_t` is supplied as an $N \times 2$ matrix and `S_t` as a list of N scatter matrices, the function computes region probabilities for all N observations in a single call, returning an $N \times n_{\mathrm{regions}}$ matrix. For `CalcMethod = 'MC'`, standard normal and chi-squared variates are pre-generated once (size `nMC`) and reused across all observations, with only the Cholesky factor of the replicate-specific scale matrix recomputed per observation. For `CalcMethod = 'MM'`, the moment-matching parameters are computed per observation (since they depend on the replicate-specific V_k) and `mvtnorm::pmvt` is called once per region per observation.

Value

When `ybar_t` is a length-2 vector (single observation): a named numeric vector of length 9 (`R1`–`R9`) for `prob = 'posterior'` or length 4 (`R1`–`R4`) for `prob = 'predictive'`. All elements are non-negative and sum to 1.

```
When \code{ybar_t} is an \eqn{N \times 2} matrix (vectorised call):
a numeric matrix with \eqn{N} rows and 9 (or 4) columns, with
column names \code{R1}--\code{R9} (or \code{R1}--\code{R4}). Each
row sums to 1.
```

Examples

```
# Example 1: Controlled design - posterior probability, vague prior
S_t <- matrix(c(18.0, 3.6, 3.6, 9.0), 2, 2)
S_c <- matrix(c(16.0, 2.8, 2.8, 8.5), 2, 2)
pbayespostpred2cont(
  prob = 'posterior', design = 'controlled', prior = 'vague',
  theta_TV1 = 1.5, theta_MAV1 = 0.5,
  theta_TV2 = 1.0, theta_MAV2 = 0.3,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  n_t = 12L, n_c = 12L,
  ybar_t = c(3.5, 2.1), S_t = S_t,
  ybar_c = c(1.8, 1.0), S_c = S_c,
  m_t = NULL, m_c = NULL,
  kappa0_t = NULL, nu0_t = NULL, mu0_t = NULL, Lambda0_t = NULL,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = NULL, Lambda0_c = NULL,
  r = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 1000L
```

```
)

# Example 2: Uncontrolled design - posterior probability, NIW prior
S_t <- matrix(c(18.0, 3.6, 3.6, 9.0), 2, 2)
L0  <- matrix(c(20.0, 0.0, 0.0, 10.0), 2, 2)
pbayespostpred2cont(
  prob = 'posterior', design = 'uncontrolled', prior = 'N-Inv-Wishart',
  theta_TV1 = 1.5, theta_MAV1 = 0.5,
  theta_TV2 = 1.0, theta_MAV2 = 0.3,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  n_t = 12L, n_c = NULL,
  ybar_t = c(3.5, 2.1), S_t = S_t,
  ybar_c = NULL, S_c = NULL,
  m_t = NULL, m_c = NULL,
  kappa0_t = 2.0, nu0_t = 5.0, mu0_t = c(2.0, 1.0), Lambda0_t = L0,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = c(1.0, 0.5), Lambda0_c = NULL,
  r = 1.0,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 1000L
)

# Example 3: External design - posterior probability, NIW prior
S_t  <- matrix(c(18.0, 3.6, 3.6, 9.0), 2, 2)
S_c  <- matrix(c(16.0, 2.8, 2.8, 8.5), 2, 2)
L0   <- matrix(c(20.0, 0.0, 0.0, 10.0), 2, 2)
se_c <- matrix(c(15.0, 2.5, 2.5, 7.5), 2, 2)
pbayespostpred2cont(
  prob = 'posterior', design = 'external', prior = 'N-Inv-Wishart',
  theta_TV1 = 1.5, theta_MAV1 = 0.5,
  theta_TV2 = 1.0, theta_MAV2 = 0.3,
  theta_NULL1 = NULL, theta_NULL2 = NULL,
  n_t = 12L, n_c = 12L,
  ybar_t = c(3.5, 2.1), S_t = S_t,
  ybar_c = c(1.8, 1.0), S_c = S_c,
  m_t = NULL, m_c = NULL,
  kappa0_t = 2.0, nu0_t = 5.0, mu0_t = c(2.0, 1.0), Lambda0_t = L0,
  kappa0_c = 2.0, nu0_c = 5.0, mu0_c = c(1.0, 0.5), Lambda0_c = L0,
  r = NULL,
  ne_t = NULL, ne_c = 10L, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = c(1.5, 0.8), se_t = NULL, se_c = se_c,
  nMC = 1000L
)

# Example 4: Controlled design - posterior predictive probability, vague prior
S_t <- matrix(c(18.0, 3.6, 3.6, 9.0), 2, 2)
S_c <- matrix(c(16.0, 2.8, 2.8, 8.5), 2, 2)
pbayespostpred2cont(
  prob = 'predictive', design = 'controlled', prior = 'vague',
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.5, theta_NULL2 = 0.3,
  n_t = 12L, n_c = 12L,
```

```
  ybar_t = c(3.5, 2.1), S_t = S_t,
  ybar_c = c(1.8, 1.0), S_c = S_c,
  m_t = 30L, m_c = 30L,
  kappa0_t = NULL, nu0_t = NULL, mu0_t = NULL, Lambda0_t = NULL,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = NULL, Lambda0_c = NULL,
  r = NULL,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 1000L
)

# Example 5: Uncontrolled design - posterior predictive probability, NIW prior
S_t <- matrix(c(18.0, 3.6, 3.6, 9.0), 2, 2)
L0  <- matrix(c(20.0, 0.0, 0.0, 10.0), 2, 2)
pbayespostpred2cont(
  prob = 'predictive', design = 'uncontrolled', prior = 'N-Inv-Wishart',
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.5, theta_NULL2 = 0.3,
  n_t = 12L, n_c = NULL,
  ybar_t = c(3.5, 2.1), S_t = S_t,
  ybar_c = NULL, S_c = NULL,
  m_t = 30L, m_c = 30L,
  kappa0_t = 2.0, nu0_t = 5.0, mu0_t = c(2.0, 1.0), Lambda0_t = L0,
  kappa0_c = NULL, nu0_c = NULL, mu0_c = c(1.0, 0.5), Lambda0_c = NULL,
  r = 1.0,
  ne_t = NULL, ne_c = NULL, alpha0e_t = NULL, alpha0e_c = NULL,
  bar_ye_t = NULL, bar_ye_c = NULL, se_t = NULL, se_c = NULL,
  nMC = 1000L
)

# Example 6: External design - posterior predictive probability, NIW prior
S_t  <- matrix(c(18.0, 3.6, 3.6, 9.0), 2, 2)
S_c  <- matrix(c(16.0, 2.8, 2.8, 8.5), 2, 2)
L0   <- matrix(c(20.0, 0.0, 0.0, 10.0), 2, 2)
se_c <- matrix(c(15.0, 2.5, 2.5, 7.5), 2, 2)
pbayespostpred2cont(
  prob = 'predictive', design = 'external', prior = 'N-Inv-Wishart',
  theta_TV1 = NULL, theta_MAV1 = NULL,
  theta_TV2 = NULL, theta_MAV2 = NULL,
  theta_NULL1 = 0.5, theta_NULL2 = 0.3,
  n_t = 12L, n_c = 12L,
  ybar_t = c(3.5, 2.1), S_t = S_t,
  ybar_c = c(1.8, 1.0), S_c = S_c,
  m_t = 30L, m_c = 30L,
  kappa0_t = 2.0, nu0_t = 5.0, mu0_t = c(2.0, 1.0), Lambda0_t = L0,
  kappa0_c = 2.0, nu0_c = 5.0, mu0_c = c(1.0, 0.5), Lambda0_c = L0,
  r = NULL,
  ne_t = NULL, ne_c = 10L, alpha0e_t = NULL, alpha0e_c = 0.5,
  bar_ye_t = NULL, bar_ye_c = c(1.5, 0.8), se_t = NULL, se_c = se_c,
  nMC = 1000L
)
```

`pbetabinomdiff` *Cumulative Distribution Function of the Difference Between Two Independent Beta-Binomial Proportions*

Description

Calculates the cumulative distribution function (CDF) of the difference between two independent Beta-Binomial proportions by exact enumeration. Specifically, computes $P((Y_t/m_t) - (Y_c/m_c) \leq q)$ or $P((Y_t/m_t) - (Y_c/m_c) > q)$, where $Y_j \sim \text{BetaBinomial}(m_j, \alpha_j, \beta_j)$ for $j \in \{t, c\}$.

Usage

```
pbetabinomdiff(
  q,
  m_t,
  m_c,
  alpha_t,
  alpha_c,
  beta_t,
  beta_c,
  lower.tail = TRUE
)
```

Arguments

`q`	A numeric scalar representing the quantile threshold for the difference in proportions.
`m_t`	A positive integer giving the number of future patients in the treatment group.
`m_c`	A positive integer giving the number of future patients in the control group.
`alpha_t`	A positive numeric scalar giving the first shape parameter of the Beta mixing distribution for the treatment group.
`alpha_c`	A positive numeric scalar giving the first shape parameter of the Beta mixing distribution for the control group.
`beta_t`	A positive numeric scalar giving the second shape parameter of the Beta mixing distribution for the treatment group.
`beta_c`	A positive numeric scalar giving the second shape parameter of the Beta mixing distribution for the control group.
`lower.tail`	A logical scalar; if `TRUE` (default), the function returns $P((Y_t/m_t) - (Y_c/m_c) \leq q)$, otherwise $P((Y_t/m_t) - (Y_c/m_c) > q)$.

Details

The probability mass function of $Y_j \sim \text{BetaBinomial}(m_j, \alpha_j, \beta_j)$ is:

$$P(Y_j = k) = \binom{m_j}{k} \frac{B(k + \alpha_j,\ m_j - k + \beta_j)}{B(\alpha_j, \beta_j)}, \quad k = 0, \ldots, m_j$$

where $B(\cdot, \cdot)$ is the Beta function.

The exact CDF is obtained by enumerating all $(m_t + 1)(m_c + 1)$ outcome combinations and summing the joint probabilities for which the proportion difference satisfies the specified condition. Computation time therefore grows quadratically in m_t and m_c; for large future sample sizes consider a normal approximation.

The Beta-Binomial distribution arises when the success probability in a Binomial model follows a Beta prior, making it appropriate for posterior predictive calculations in Bayesian binary-endpoint trials.

Value

A numeric scalar in `[0, 1]`.

Examples

```
# P((Y_t/12) - (Y_c/12) > 0.2) with symmetric Beta(0.5, 0.5) priors
pbetabinomdiff(0.2, 12, 12, 0.5, 0.5, 0.5, 0.5, lower.tail = FALSE)

# P((Y_t/20) - (Y_c/15) > 0.1) with different future sample sizes
pbetabinomdiff(0.1, 20, 15, 1, 1, 1, 1, lower.tail = FALSE)

# P((Y_t/10) - (Y_c/10) > 0) with informative priors
pbetabinomdiff(0, 10, 10, 2, 3, 3, 2, lower.tail = FALSE)

# Lower tail: P((Y_t/15) - (Y_c/15) <= 0.05) with vague priors
pbetabinomdiff(0.05, 15, 15, 1, 1, 1, 1, lower.tail = TRUE)
```

---

pbetadiff *Cumulative Distribution Function of the Difference Between Two Independent Beta Variables*

Description

Calculates the cumulative distribution function (CDF) of the difference between two independent Beta-distributed random variables. Specifically, computes $P(\pi_t - \pi_c \leq q)$ or $P(\pi_t - \pi_c > q)$, where $\pi_j \sim \text{Beta}(\alpha_j, \beta_j)$ for $j \in \{t, c\}$.

Usage

```
pbetadiff(q, alpha_t, alpha_c, beta_t, beta_c, lower.tail = TRUE)
```

Arguments

`q`	A numeric scalar in `[-1, 1]` representing the quantile threshold for the difference in proportions.
`alpha_t`	A positive numeric scalar giving the first shape parameter of the Beta distribution for the treatment group.

alpha_c	A positive numeric scalar giving the first shape parameter of the Beta distribution for the control group.
beta_t	A positive numeric scalar giving the second shape parameter of the Beta distribution for the treatment group.
beta_c	A positive numeric scalar giving the second shape parameter of the Beta distribution for the control group.
lower.tail	A logical scalar; if TRUE (default), the function returns $P(\pi_t - \pi_c \leq q)$, otherwise $P(\pi_t - \pi_c > q)$.

Details

The upper-tail probability is obtained via the convolution formula:

$$P(\pi_t - \pi_c > q) = \int_0^1 F_{\mathrm{Beta}(\alpha_c,\beta_c)}(x - q)\, f_{\mathrm{Beta}(\alpha_t,\beta_t)}(x)\, dx$$

where $f_{\mathrm{Beta}(\alpha_t,\beta_t)}$ is the density of π_t and $F_{\mathrm{Beta}(\alpha_c,\beta_c)}$ is the CDF of π_c. Boundary cases are handled automatically by pbeta (0 for $x - q \leq 0$, 1 for $x - q \geq 1$), so integrating over [0, 1] is safe.

This single-integral convolution replaces an equivalent double-integral formulation based on Appell's F1 hypergeometric function, yielding the same result with lower computational cost because both pbeta and dbeta are implemented in compiled C code.

Value

A numeric scalar in [0, 1].

Examples

```
# P(pi_t - pi_c > 0.2) with symmetric Beta(0.5, 0.5) priors
pbetadiff(0.2, 0.5, 0.5, 0.5, 0.5, lower.tail = FALSE)

# P(pi_t - pi_c > -0.1) with informative priors
pbetadiff(-0.1, 2, 1, 3, 4, lower.tail = FALSE)

# P(pi_t - pi_c > 0) with equal priors -- should be approximately 0.5
pbetadiff(0, 1, 1, 1, 1, lower.tail = FALSE)

# Lower tail: P(pi_t - pi_c <= 0.1) with symmetric priors
pbetadiff(0.1, 2, 2, 2, 2, lower.tail = TRUE)
```

`plot.getgamma1bin` *Plot Method for getgamma1bin Objects*

Description

Displays calibration curves of marginal Go and NoGo probabilities against the threshold grid γ for results returned by `getgamma1bin`.

Usage

```
## S3 method for class 'getgamma1bin'
plot(
  x,
  title = NULL,
  col_go = "#658D1B",
  col_nogo = "#D91E49",
  base_size = 28,
  ...
)
```

Arguments

`x`	An object of class `getgamma1bin`.
`title`	A character string for the plot title. Defaults to `NULL` (no title displayed).
`col_go`	A character string specifying the colour for the Go curve. Default is `"#658D1B"`.
`col_nogo`	A character string specifying the colour for the NoGo curve. Default is `"#D91E49"`.
`base_size`	A positive numeric scalar specifying the base font size (in points) passed to `theme_bw()`. Default is `28`.
`...`	Further arguments passed to or from other methods (ignored).

Details

The x-axis represents candidate threshold values $\gamma \in (0, 1)$. The y-axis represents the marginal probability:

- **Go curve**: $\Pr(g_{\mathrm{Go}} \geq \gamma)$ evaluated under the Go-calibration scenario (typically the Null scenario).
- **NoGo curve**: $\Pr(g_{\mathrm{NoGo}} \geq \gamma)$ evaluated under the NoGo-calibration scenario (typically the Alternative scenario).

Horizontal reference lines are drawn at `target_go` and `target_nogo`. Filled circles (`geom_point`) mark the optimal thresholds $(\gamma_{\mathrm{go}}, \Pr(\mathrm{Go})_{\mathrm{opt}})$ and $(\gamma_{\mathrm{nogo}}, \Pr(\mathrm{NoGo})_{\mathrm{opt}})$, with their values shown in the legend. If either optimal threshold is `NA`, the corresponding point is omitted.

Value

Invisibly returns a `ggplot` object.

`plot.getgamma1cont` *Plot Method for getgamma1cont Objects*

Description

Displays calibration curves of marginal Go and NoGo probabilities against the threshold grid γ for results returned by `getgamma1cont`.

Usage

```
## S3 method for class 'getgamma1cont'
plot(
  x,
  title = NULL,
  col_go = "#658D1B",
  col_nogo = "#D91E49",
  base_size = 28,
  ...
)
```

Arguments

`x`	An object of class `getgamma1cont`.
`title`	A character string for the plot title. Defaults to `NULL` (no title displayed).
`col_go`	A character string specifying the colour for the Go curve. Default is `"#658D1B"`.
`col_nogo`	A character string specifying the colour for the NoGo curve. Default is `"#D91E49"`.
`base_size`	A positive numeric scalar specifying the base font size (in points) passed to `theme_bw()`. Default is `28`.
`...`	Further arguments passed to or from other methods (ignored).

Details

The x-axis represents candidate threshold values $\gamma \in (0, 1)$. The y-axis represents the marginal probability:

- **Go curve**: $\Pr(g_{\mathrm{Go}} \geq \gamma)$ evaluated under the Go-calibration scenario (typically the Null scenario).
- **NoGo curve**: $\Pr(g_{\mathrm{NoGo}} \geq \gamma)$ evaluated under the NoGo-calibration scenario (typically the Alternative scenario).

Horizontal reference lines are drawn at `target_go` and `target_nogo`. Filled circles (`geom_point`) mark the optimal thresholds $(\gamma_{\mathrm{go}}, \Pr(\mathrm{Go})_{\mathrm{opt}})$ and $(\gamma_{\mathrm{nogo}}, \Pr(\mathrm{NoGo})_{\mathrm{opt}})$, with their values shown in the legend. If either optimal threshold is `NA`, the corresponding point is omitted.

Value

Invisibly returns a `ggplot` object.

`plot.getgamma2bin` *Plot Method for getgamma2bin Objects*

Description

Displays calibration curves of marginal Go and NoGo probabilities against the threshold grid γ for results returned by `getgamma2bin`.

Usage

```
## S3 method for class 'getgamma2bin'
plot(
  x,
  title = NULL,
  col_go = "#658D1B",
  col_nogo = "#D91E49",
  base_size = 28,
  ...
)
```

Arguments

`x`	An object of class `getgamma2bin`.
`title`	A character string for the plot title. Defaults to `NULL` (no title displayed).
`col_go`	A character string specifying the colour for the Go curve. Default is `"#658D1B"`.
`col_nogo`	A character string specifying the colour for the NoGo curve. Default is `"#D91E49"`.
`base_size`	A positive numeric scalar specifying the base font size (in points) passed to `theme_bw()`. Default is `28`.
`...`	Further arguments passed to or from other methods (ignored).

Details

The x-axis represents candidate threshold values $\gamma \in (0, 1)$. The y-axis represents the marginal probability:

- **Go curve**: $\Pr(g_{\mathrm{Go}} \geq \gamma)$ evaluated under the Go-calibration scenario (typically the Null scenario).
- **NoGo curve**: $\Pr(g_{\mathrm{NoGo}} \geq \gamma)$ evaluated under the NoGo-calibration scenario (typically the Alternative scenario).

Horizontal reference lines are drawn at `target_go` and `target_nogo`. Filled circles (`geom_point`) mark the optimal thresholds $(\gamma_{\mathrm{go}}, \Pr(\mathrm{Go})_{\mathrm{opt}})$ and $(\gamma_{\mathrm{nogo}}, \Pr(\mathrm{NoGo})_{\mathrm{opt}})$, with their values shown in the legend. If either optimal threshold is `NA`, the corresponding point is omitted.

Value

Invisibly returns a `ggplot` object.

`plot.getgamma2cont` *Plot Method for getgamma2cont Objects*

Description

Displays calibration curves of marginal Go and NoGo probabilities against the threshold grid γ for results returned by `getgamma2cont`.

Usage

```
## S3 method for class 'getgamma2cont'
plot(
  x,
  title = NULL,
  col_go = "#658D1B",
  col_nogo = "#D91E49",
  base_size = 28,
  ...
)
```

Arguments

`x`	An object of class `getgamma2cont`.
`title`	A character string for the plot title. Defaults to `NULL` (no title displayed).
`col_go`	A character string specifying the colour for the Go curve. Default is `"#658D1B"`.
`col_nogo`	A character string specifying the colour for the NoGo curve. Default is `"#D91E49"`.
`base_size`	A positive numeric scalar specifying the base font size (in points) passed to `theme_bw()`. Default is `28`.
`...`	Further arguments passed to or from other methods (ignored).

Details

The x-axis represents candidate threshold values $\gamma \in (0, 1)$. The y-axis represents the marginal probability:

- **Go curve**: $\Pr(g_{\mathrm{Go}} \geq \gamma)$ evaluated under the Go-calibration scenario (typically the Null scenario).
- **NoGo curve**: $\Pr(g_{\mathrm{NoGo}} \geq \gamma)$ evaluated under the NoGo-calibration scenario (typically the Alternative scenario).

Horizontal reference lines are drawn at `target_go` and `target_nogo`. Filled circles (`geom_point`) mark the optimal thresholds $(\gamma_{\mathrm{go}}, \Pr(\mathrm{Go})_{\mathrm{opt}})$ and $(\gamma_{\mathrm{nogo}}, \Pr(\mathrm{NoGo})_{\mathrm{opt}})$, with their values shown in the legend. If either optimal threshold is `NA`, the corresponding point is omitted.

Value

Invisibly returns a `ggplot` object.

plot.pbayesdecisionprob1bin *Plot Method for pbayesdecisionprob1bin Objects*

Description

Displays an operating characteristics curve of Go/NoGo/Gray decision probabilities against the true treatment effect for binary endpoint results returned by pbayesdecisionprob1bin.

Usage

```
## S3 method for class 'pbayesdecisionprob1bin'
plot(
  x,
  title = NULL,
  xlab = NULL,
  col_go = "#658D1B",
  col_nogo = "#D91E49",
  col_gray = "#939597",
  base_size = 28,
  ...
)
```

Arguments

x	An object of class pbayesdecisionprob1bin.
title	A character string for the plot title. Defaults to NULL (no title displayed).
xlab	A character string or expression for the x-axis label. Defaults to NULL, which auto-generates a label based on design.
col_go	A character string specifying the colour for the Go curve. Default is "#658D1B".
col_nogo	A character string specifying the colour for the NoGo curve. Default is "#D91E49".
col_gray	A character string specifying the colour for the Gray curve. Default is "#939597".
base_size	A positive numeric scalar specifying the base font size (in points) passed to theme_bw(). Default is 28.
...	Further arguments passed to or from other methods (ignored).

Details

For design = 'controlled' or design = 'external', the x-axis represents the treatment-minus-control difference $\theta = \pi_t - \bar{\pi}_c$, where $\bar{\pi}_c$ is the mean of the supplied pi_c values. For design = 'uncontrolled', the x-axis represents π_t directly.

Vertical reference lines are drawn at the decision thresholds:

- When prob = 'posterior': lines at θ_{TV} and θ_{MAV} (converted to the x-axis scale).
- When prob = 'predictive': a single line at θ_{NULL}.

Value

Invisibly returns a ggplot object.

`plot.pbayesdecisionprob1cont` *Plot Method for pbayesdecisionprob1cont Objects*

Description

Displays an operating characteristics curve of Go/NoGo/Gray decision probabilities against the true treatment effect for continuous endpoint results returned by `pbayesdecisionprob1cont`.

Usage

```
## S3 method for class 'pbayesdecisionprob1cont'
plot(
  x,
  title = NULL,
  xlab = NULL,
  col_go = "#658D1B",
  col_nogo = "#D91E49",
  col_gray = "#939597",
  base_size = 28,
  ...
)
```

Arguments

`x`	An object of class `pbayesdecisionprob1cont`.
`title`	A character string for the plot title. Defaults to `NULL` (no title displayed).
`xlab`	A character string or expression for the x-axis label. Defaults to `NULL`, which auto-generates a label based on `design`.
`col_go`	A character string specifying the colour for the Go curve. Default is `"#658D1B"`.
`col_nogo`	A character string specifying the colour for the NoGo curve. Default is `"#D91E49"`.
`col_gray`	A character string specifying the colour for the Gray curve. Default is `"#939597"`.
`base_size`	A positive numeric scalar specifying the base font size (in points) passed to `theme_bw()`. Default is `28`.
`...`	Further arguments passed to or from other methods (ignored).

Details

For `design = 'controlled'` or `design = 'external'`, the x-axis represents the treatment-minus-control difference $\theta = \mu_t - \bar{\mu}_c$, where $\bar{\mu}_c$ is the mean of the supplied `mu_c` values. For `design = 'uncontrolled'`, the x-axis represents μ_t directly.

Vertical reference lines are drawn at the decision thresholds:

- When `prob = 'posterior'`: lines at θ_{TV} and θ_{MAV} (converted to the x-axis scale).
- When `prob = 'predictive'`: a single line at θ_{NULL}.

Value

Invisibly returns a `ggplot` object.

`plot.pbayesdecisionprob2bin` *Plot Method for pbayesdecisionprob2bin Objects*

Description

Displays operating characteristics for two-binary-endpoint results returned by `pbayesdecisionprob2bin`.

Usage

```
## S3 method for class 'pbayesdecisionprob2bin'
plot(
  x,
  which = "Go",
  title = NULL,
  xlab = NULL,
  ylab = NULL,
  col_go = "#658D1B",
  col_nogo = "#D91E49",
  col_gray = "#939597",
  base_size = 28,
  ...
)
```

Arguments

`x`	An object of class `pbayesdecisionprob2bin`.
`which`	A character string specifying which decision probability to plot. Must be one of `"Go"`, `"Gray"`, `"NoGo"`, `"all"`, or `"overlay"`. Default is `"Go"`.
`title`	A character string for the plot title. Defaults to `NULL` (no title displayed).
`xlab`	A character string or expression for the x-axis label. Defaults to `NULL`, which auto-generates a label based on `design`.

ylab	A character string or expression for the y-axis label. Defaults to NULL, which auto-generates a label based on design.
col_go	A character string specifying the high-end fill colour for the Go probability gradient. Default is "#658D1B".
col_nogo	A character string specifying the high-end fill colour for the NoGo probability gradient. Default is "#D91E49".
col_gray	A character string specifying the high-end fill colour for the Gray probability gradient. Default is "#939597".
base_size	A positive numeric scalar specifying the base font size (in points) passed to theme_bw(). Default is 28.
...	Further arguments passed to or from other methods (ignored).

Details

When the input scenarios form a regular grid over (pi_t1, pi_t2) (i.e., every combination of the unique values of pi_t1 and pi_t2 is present) and rho_t is constant, the function produces a **filled tile plot**: each panel (Go, Gray, NoGo) is coloured by its own probability on a continuous gradient (white to the panel colour), so intensity directly reflects the probability magnitude. A solid white contour line is overlaid at the corresponding decision threshold (gamma_go for the Go panel, gamma_nogo for the NoGo panel, and their mean for the Gray panel) to mark the boundary where the probability equals the threshold. Otherwise the function falls back to a **scatter plot** in which point colour encodes the decision probability on a continuous scale.

When which = "all", the three panels are arranged side-by-side using gridExtra::grid.arrange, so each panel retains its own independent colour scale. This requires the **gridExtra** package.

For design = 'controlled' or design = 'external', both axes are expressed as treatment-minus-control differences: $\theta_1 = \pi_{t1} - \bar{\pi}_{c1}$ and $\theta_2 = \pi_{t2} - \bar{\pi}_{c2}$, where $\bar{\pi}_{c1}$ and $\bar{\pi}_{c2}$ are the means of the supplied pi_c1 and pi_c2 vectors. For design = 'uncontrolled', the axes represent π_{t1} and π_{t2} directly.

Vertical and horizontal reference lines are drawn at the decision thresholds:

- When prob = 'posterior': vertical lines at θ_{TV1} and θ_{MAV1} (x-axis) and horizontal lines at θ_{TV2} and θ_{MAV2} (y-axis).
- When prob = 'predictive': a single vertical line at θ_{NULL1} and a single horizontal line at θ_{NULL2}.

Value

Invisibly returns a ggplot object (single panel) or a gtable object (which = "all").

plot.pbayesdecisionprob2cont *Plot Method for pbayesdecisionprob2cont Objects*

Description

Displays operating characteristics for two-continuous-endpoint results returned by `pbayesdecisionprob2cont`.

Usage

```
## S3 method for class 'pbayesdecisionprob2cont'
plot(
  x,
  which = "Go",
  title = NULL,
  xlab = NULL,
  ylab = NULL,
  col_go = "#658D1B",
  col_nogo = "#D91E49",
  col_gray = "#939597",
  base_size = 28,
  ...
)
```

Arguments

`x`	An object of class `pbayesdecisionprob2cont`.
`which`	A character string specifying which decision probability to plot. Must be one of `"Go"`, `"Gray"`, `"NoGo"`, `"all"`, or `"overlay"`. Default is `"Go"`.
`title`	A character string for the plot title. Defaults to `NULL` (no title displayed).
`xlab`	A character string or expression for the x-axis label. Defaults to `NULL`, which auto-generates a label based on `design`.
`ylab`	A character string or expression for the y-axis label. Defaults to `NULL`, which auto-generates a label based on `design`.
`col_go`	A character string specifying the high-end fill colour for the Go probability gradient. Default is `"#658D1B"`.
`col_nogo`	A character string specifying the high-end fill colour for the NoGo probability gradient. Default is `"#D91E49"`.
`col_gray`	A character string specifying the high-end fill colour for the Gray probability gradient. Default is `"#939597"`.
`base_size`	A positive numeric scalar specifying the base font size (in points) passed to `theme_bw()`. Default is `28`.
`...`	Further arguments passed to or from other methods (ignored).

Details

When the input scenarios form a regular grid over (mu_t1, mu_t2) (i.e., every combination of the unique values of mu_t1 and mu_t2 is present) and rho_t is constant, the function produces a **filled tile plot**: each panel (Go, Gray, NoGo) is coloured by its own probability on a continuous gradient (white to the panel colour), so intensity directly reflects the probability magnitude. Otherwise the function falls back to a **scatter plot** in which point colour encodes the decision probability on a continuous scale.

When which = "all", the three panels are arranged side-by-side using gridExtra::grid.arrange, so each panel retains its own independent colour scale. This requires the **gridExtra** package.

For design = 'controlled' or design = 'external', both axes are expressed as treatment-minus-control differences: $\theta_1 = \mu_{t1} - \bar{\mu}_{c1}$ and $\theta_2 = \mu_{t2} - \bar{\mu}_{c2}$, where $\bar{\mu}_{c1}$ and $\bar{\mu}_{c2}$ are the means of the supplied mu_c1 and mu_c2 vectors. For design = 'uncontrolled', the axes represent μ_{t1} and μ_{t2} directly.

Vertical and horizontal reference lines are drawn at the decision thresholds:

- When prob = 'posterior': vertical lines at θ_{TV1} and θ_{MAV1} (x-axis) and horizontal lines at θ_{TV2} and θ_{MAV2} (y-axis).
- When prob = 'predictive': a single vertical line at θ_{NULL1} and a single horizontal line at θ_{NULL2}.

Value

Invisibly returns a ggplot object (single panel) or a gtable object (which = "all").

---

print.pbayesdecisionprob1bin

Print Method for pbayesdecisionprob1bin Objects

Description

Displays a formatted summary of Go/NoGo/Gray decision probabilities for binary endpoint results returned by pbayesdecisionprob1bin.

Usage

```
## S3 method for class 'pbayesdecisionprob1bin'
print(x, digits = 4, ...)
```

Arguments

x	An object of class pbayesdecisionprob1bin.
digits	A positive integer specifying the number of decimal places for probability values. Default is 4.
...	Further arguments passed to or from other methods (ignored).

Value

Invisibly returns `x`.

`print.pbayesdecisionprob1cont`
Print Method for pbayesdecisionprob1cont Objects

Description

Displays a formatted summary of Go/NoGo/Gray decision probabilities for continuous endpoint results returned by `pbayesdecisionprob1cont`.

Usage

```
## S3 method for class 'pbayesdecisionprob1cont'
print(x, digits = 4, ...)
```

Arguments

`x`	An object of class `pbayesdecisionprob1cont`.
`digits`	A positive integer specifying the number of decimal places for probability values. Default is 4.
`...`	Further arguments passed to or from other methods (ignored).

Value

Invisibly returns `x`.

`print.pbayesdecisionprob2bin`
Print Method for pbayesdecisionprob2bin Objects

Description

Displays a formatted summary of Go/NoGo/Gray decision probabilities for two-binary-endpoint results returned by `pbayesdecisionprob2bin`.

Usage

```
## S3 method for class 'pbayesdecisionprob2bin'
print(x, digits = 4, ...)
```

Arguments

x	An object of class pbayesdecisionprob2bin.
digits	A positive integer specifying the number of decimal places for probability values. Default is 4.
...	Further arguments passed to or from other methods (ignored).

Value

Invisibly returns x.

---

print.pbayesdecisionprob2cont

Print Method for pbayesdecisionprob2cont Objects

Description

Displays a formatted summary of Go/NoGo/Gray decision probabilities for two-continuous-endpoint results returned by pbayesdecisionprob2cont.

Displays a formatted summary of Go/NoGo/Gray decision probabilities for two-continuous-endpoint results returned by pbayesdecisionprob2cont.

Usage

```
## S3 method for class 'pbayesdecisionprob2cont'
print(x, digits = 4, ...)

## S3 method for class 'pbayesdecisionprob2cont'
print(x, digits = 4, ...)
```

Arguments

x	An object of class pbayesdecisionprob2cont.
digits	A positive integer specifying the number of decimal places for probability values. Default is 4.
...	Further arguments passed to or from other methods (ignored).

Value

Invisibly returns x.

Invisibly returns x.

ptdiff_MC *Cumulative Distribution Function of the Difference of Two Independent t-Distributed Variables via Monte Carlo Simulation*

Description

Calculates the cumulative distribution function (CDF) of the difference between two independent non-standardised t-distributed random variables using Monte Carlo simulation. Specifically, computes $P(T_t - T_c \leq q)$ or $P(T_t - T_c > q)$, where $T_k \sim t(\mu_k, \sigma_k^2, \nu_k)$ for $k \in \{t, c\}$.

Usage

```
ptdiff_MC(nMC, q, mu_t, mu_c, sd_t, sd_c, nu_t, nu_c, lower.tail = TRUE)
```

Arguments

nMC	A positive integer giving the number of Monte Carlo draws. Typical values range from 10,000 (quick estimates) to 100,000 or more (high precision). Larger values reduce Monte Carlo error at the cost of computation time.
q	A numeric scalar representing the quantile threshold.
mu_t	A numeric scalar or vector giving the location parameter of the t-distribution for the treatment group.
mu_c	A numeric scalar or vector giving the location parameter of the t-distribution for the control group.
sd_t	A positive numeric scalar or vector giving the scale parameter of the t-distribution for the treatment group.
sd_c	A positive numeric scalar or vector giving the scale parameter of the t-distribution for the control group.
nu_t	A numeric scalar giving the degrees of freedom of the t-distribution for the treatment group. Must be greater than 2 for finite variance.
nu_c	A numeric scalar giving the degrees of freedom of the t-distribution for the control group. Must be greater than 2 for finite variance.
lower.tail	A logical scalar; if TRUE (default), the function returns $P(T_t - T_c \leq q)$, otherwise $P(T_t - T_c > q)$.

Details

The algorithm proceeds as follows:

1. Generate an nMC x n matrix of standard t draws for T_t (ν_t degrees of freedom), then scale and shift each column by sd_t[i] and mu_t[i].
2. Repeat for T_c with ν_c degrees of freedom.
3. Compute the nMC x n difference matrix $D = T_t - T_c$.
4. Return colMeans(D > q) as the estimated $P(T_t - T_c > q)$ for each parameter set.

All operations are matrix-based (no R-level loop over parameter sets), so performance scales well with n. However, note that the matrix size is `nMC` x `n`, so memory usage grows linearly with both `nMC` and n. When n is large (e.g., `nsim` x `n_scenarios` in `pbayesdecisionprob1cont`), memory requirements can become prohibitive; in such cases prefer `CalcMethod = 'MM'`.

Monte Carlo error is approximately $\sqrt{p(1-p)/\mathrm{nMC}}$; near $p = 0.5$ this is roughly $0.5/\sqrt{\mathrm{nMC}}$.

Value

A numeric scalar or vector in `[0, 1]`. When `mu_t`, `mu_c`, `sd_t`, or `sd_c` are vectors of length n, a vector of length n is returned.

Examples

```
# P(T_t - T_c > 3) with equal parameters
ptdiff_MC(nMC = 1e5, q = 3, mu_t = 2, mu_c = 0, sd_t = 1, sd_c = 1,
          nu_t = 17, nu_c = 17, lower.tail = FALSE)

# P(T_t - T_c > 1) with unequal scales
ptdiff_MC(nMC = 1e5, q = 1, mu_t = 5, mu_c = 3, sd_t = 2, sd_c = 1.5,
          nu_t = 10, nu_c = 15, lower.tail = FALSE)

# P(T_t - T_c > 0) with different degrees of freedom
ptdiff_MC(nMC = 1e5, q = 0, mu_t = 1, mu_c = 1, sd_t = 1, sd_c = 1,
          nu_t = 5, nu_c = 20, lower.tail = FALSE)

# Lower tail: P(T_t - T_c <= 2)
ptdiff_MC(nMC = 1e5, q = 2, mu_t = 3, mu_c = 0, sd_t = 1.5, sd_c = 1.2,
          nu_t = 12, nu_c = 15, lower.tail = TRUE)

# Vectorised usage
ptdiff_MC(nMC = 1e5, q = 1, mu_t = c(2, 3, 4), mu_c = c(0, 1, 2),
          sd_t = c(1, 1.2, 1.5), sd_c = c(1, 1.1, 1.3),
          nu_t = 10, nu_c = 10, lower.tail = FALSE)
```

`ptdiff_MM` *Cumulative Distribution Function of the Difference of Two Independent t-Distributed Variables via Moment-Matching Approximation*

Description

Calculates the cumulative distribution function (CDF) of the difference between two independent non-standardised t-distributed random variables using a Moment-Matching approximation. Specifically, computes $P(T_t - T_c \leq q)$ or $P(T_t - T_c > q)$, where $T_k \sim t(\mu_k, \sigma_k^2, \nu_k)$ for $k \in \{t, c\}$.

Usage

```
ptdiff_MM(q, mu_t, mu_c, sd_t, sd_c, nu_t, nu_c, lower.tail = TRUE)
```

Arguments

q	A numeric scalar representing the quantile threshold.
mu_t	A numeric scalar or vector giving the location parameter of the t-distribution for the treatment group.
mu_c	A numeric scalar or vector giving the location parameter of the t-distribution for the control group.
sd_t	A positive numeric scalar or vector giving the scale parameter of the t-distribution for the treatment group.
sd_c	A positive numeric scalar or vector giving the scale parameter of the t-distribution for the control group.
nu_t	A numeric scalar giving the degrees of freedom of the t-distribution for the treatment group. Must be greater than 4 for finite fourth moment.
nu_c	A numeric scalar giving the degrees of freedom of the t-distribution for the control group. Must be greater than 4 for finite fourth moment.
lower.tail	A logical scalar; if TRUE (default), the function returns $P(T_t - T_c \leq q)$, otherwise $P(T_t - T_c > q)$.

Details

The difference $D = T_t - T_c$ is approximated by a single non-standardised t-distribution $t(\mu^*, {\sigma^*}^2, \nu^*)$ whose parameters are determined by matching the first two even moments of D:

- $\mu^* = \mu_t - \mu_c$.
- ${\sigma^*}^2$ is obtained from the second-moment equation.
- ν^* is obtained from the fourth-moment equation.

The approximation requires $\nu_t > 4$ and $\nu_c > 4$ for finite fourth moments. It is exact in the normal limit ($\nu \to \infty$) and works well in practice when $\nu > 10$. Because it reduces to a single call to pt(), it is orders of magnitude faster than the numerical integration method (ptdiff_NI) and is fully vectorised.

Value

A numeric scalar or vector in [0, 1]. When mu_t, mu_c, sd_t, or sd_c are vectors of length n, a vector of length n is returned.

Examples

```
# P(T_t - T_c > 3) with equal parameters
ptdiff_MM(q = 3, mu_t = 2, mu_c = 0, sd_t = 1, sd_c = 1,
          nu_t = 17, nu_c = 17, lower.tail = FALSE)

# P(T_t - T_c > 1) with unequal scales
ptdiff_MM(q = 1, mu_t = 5, mu_c = 3, sd_t = 2, sd_c = 1.5,
          nu_t = 10, nu_c = 15, lower.tail = FALSE)

# P(T_t - T_c > 0) with different degrees of freedom
```

```
ptdiff_MM(q = 0, mu_t = 1, mu_c = 1, sd_t = 1, sd_c = 1,
          nu_t = 5, nu_c = 20, lower.tail = FALSE)

# Lower tail: P(T_t - T_c <= 2)
ptdiff_MM(q = 2, mu_t = 3, mu_c = 0, sd_t = 1.5, sd_c = 1.2,
          nu_t = 12, nu_c = 15, lower.tail = TRUE)

# Vectorised usage
ptdiff_MM(q = 1, mu_t = c(2, 3, 4), mu_c = c(0, 1, 2),
          sd_t = c(1, 1.2, 1.5), sd_c = c(1, 1.1, 1.3),
          nu_t = 10, nu_c = 10, lower.tail = FALSE)
```

`ptdiff_NI` *Cumulative Distribution Function of the Difference of Two Independent t-Distributed Variables via Numerical Integration*

Description

Calculates the cumulative distribution function (CDF) of the difference between two independent non-standardised t-distributed random variables using numerical integration. Specifically, computes $P(T_t - T_c \leq q)$ or $P(T_t - T_c > q)$, where $T_k \sim t(\mu_k, \sigma_k^2, \nu_k)$ for $k \in \{t, c\}$.

Usage

```
ptdiff_NI(q, mu_t, mu_c, sd_t, sd_c, nu_t, nu_c, lower.tail = TRUE)
```

Arguments

`q`	A numeric scalar representing the quantile threshold.
`mu_t`	A numeric scalar or vector giving the location parameter of the t-distribution for the treatment group.
`mu_c`	A numeric scalar or vector giving the location parameter of the t-distribution for the control group.
`sd_t`	A positive numeric scalar or vector giving the scale parameter of the t-distribution for the treatment group.
`sd_c`	A positive numeric scalar or vector giving the scale parameter of the t-distribution for the control group.
`nu_t`	A numeric scalar giving the degrees of freedom of the t-distribution for the treatment group. Must be greater than 2 for finite variance.
`nu_c`	A numeric scalar giving the degrees of freedom of the t-distribution for the control group. Must be greater than 2 for finite variance.
`lower.tail`	A logical scalar; if `TRUE` (default), the function returns $P(T_t - T_c \leq q)$, otherwise $P(T_t - T_c > q)$.

Details

The upper-tail probability is obtained via the convolution formula:

$$P(T_t - T_c > q) = \int_{-\infty}^{\infty} F_{t(\mu_c,\sigma_c^2,\nu_c)}(x-q)\, f_{t(\mu_t,\sigma_t^2,\nu_t)}(x)\, dx$$

where $f_{t(\mu_t,\sigma_t^2,\nu_t)}$ is the density of T_t and $F_{t(\mu_c,\sigma_c^2,\nu_c)}$ is the CDF of T_c. The integral is evaluated by adaptive Gauss-Kronrod quadrature via `stats::integrate`.

When the input parameters are vectors, `mapply` applies the scalar integration function across all parameter sets.

Advantages:

- Exact within numerical precision.
- Handles arbitrary parameter combinations.

Computational note: this method is substantially slower than the Moment-Matching approximation (`ptdiff_MM`) because it calls `integrate()` once per parameter set. For large-scale simulation (many parameter sets), prefer `CalcMethod = 'MM'` in `pbayesdecisionprob1cont`.

Value

A numeric scalar or vector in `[0, 1]`. When `mu_t`, `mu_c`, `sd_t`, or `sd_c` are vectors of length n, a vector of length n is returned.

Examples

```
# P(T_t - T_c > 3) with equal parameters
ptdiff_NI(q = 3, mu_t = 2, mu_c = 0, sd_t = 1, sd_c = 1,
          nu_t = 17, nu_c = 17, lower.tail = FALSE)

# P(T_t - T_c > 1) with unequal scales
ptdiff_NI(q = 1, mu_t = 5, mu_c = 3, sd_t = 2, sd_c = 1.5,
          nu_t = 10, nu_c = 15, lower.tail = FALSE)

# P(T_t - T_c > 0) with different degrees of freedom
ptdiff_NI(q = 0, mu_t = 1, mu_c = 1, sd_t = 1, sd_c = 1,
          nu_t = 5, nu_c = 20, lower.tail = FALSE)

# Lower tail: P(T_t - T_c <= 2)
ptdiff_NI(q = 2, mu_t = 3, mu_c = 0, sd_t = 1.5, sd_c = 1.2,
          nu_t = 12, nu_c = 15, lower.tail = TRUE)

# Vectorised usage
ptdiff_NI(q = 1, mu_t = c(2, 3, 4), mu_c = c(0, 1, 2),
          sd_t = c(1, 1.2, 1.5), sd_c = c(1, 1.1, 1.3),
          nu_t = 10, nu_c = 10, lower.tail = FALSE)
```

`rdirichlet` *Generate Random Samples from a Dirichlet Distribution*

Description

Generates random samples from a Dirichlet distribution using the Gamma representation: if $Y_i \sim \text{Gamma}(\alpha_i, 1)$ independently for $i = 1, \ldots, K$, then $(Y_1/S, \ldots, Y_K/S) \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_K)$, where $S = \sum_{i=1}^{K} Y_i$.

Usage

```
rdirichlet(n, alpha)
```

Arguments

`n`	A positive integer specifying the number of random vectors to generate.
`alpha`	A numeric vector of length $K \geq 2$ containing positive concentration parameters of the Dirichlet distribution. All elements must be strictly positive.

Details

The Dirichlet distribution is a multivariate generalisation of the Beta distribution and is commonly used as a conjugate prior for multinomial proportions in Bayesian statistics.

The probability density function is:

$$f(x_1, \ldots, x_K) = \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$

where $x_i > 0$ and $\sum_{i=1}^{K} x_i = 1$.

Key properties:

- Each marginal follows a Beta distribution: $X_i \sim \text{Beta}\left(\alpha_i,\, \sum_{l \neq i} \alpha_l\right)$.
- $E[X_i] = \alpha_i / \sum_{l=1}^{K} \alpha_l$.
- Components are negatively correlated unless one component dominates.

Implementation steps:

1. Generate independent $Y_i \sim \text{Gamma}(\alpha_i, 1)$ for each $i = 1, \ldots, K$.
2. Normalise: $X_i = Y_i / \sum_{l=1}^{K} Y_l$.

Value

A numeric matrix of dimensions `n` x `K` where each row is one random draw from the Dirichlet distribution, with all elements in `[0, 1]` and each row summing to 1. When `n = 1`, a numeric vector of length `K` is returned.

Examples

```
# Example 1: Generate 5 samples from Dirichlet(1, 1, 1) - uniform on simplex
samples <- rdirichlet(5, c(1, 1, 1))
print(samples)
rowSums(samples)  # Each row should sum to 1

# Example 2: Generate samples with unequal concentrations
samples <- rdirichlet(1000, c(2, 5, 3))
colMeans(samples)  # Expected values: approximately c(0.2, 0.5, 0.3)

# Example 3: Sparse Dirichlet (small alpha values)
samples <- rdirichlet(100, c(0.1, 0.1, 0.1, 0.1))
head(samples)  # Most weight concentrated on one component

# Example 4: Concentrated Dirichlet (large alpha values)
samples <- rdirichlet(100, c(100, 100, 100))
colMeans(samples)  # Concentrated around c(1/3, 1/3, 1/3)

# Example 5: Bayesian update with Jeffreys prior for 4 categories
prior_alpha     <- c(0.5, 0.5, 0.5, 0.5)
observed_counts <- c(10, 5, 8, 7)
posterior_samples <- rdirichlet(1000, prior_alpha + observed_counts)
colMeans(posterior_samples)  # Posterior mean
```

Index