Package 'gsbDesign'

January 28, 2024

Type Package

Version 1.0-3

Date 2024-01-23

Title Group Sequential Bayes Design

Depends gsDesign, lattice, grid,

Imports stats, graphics, grDevices, utils, ggplot2

Description Group Sequential Operating Characteristics for Clinical,
Bayesian two-arm Trials with known Sigma and Normal Endpoints,
as described in Gerber and Gsponer (2016) <doi:10.18637/jss.v069.i11>.

License GPL-3

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2024-01-28 11:20:11 UTC

Author Florian Gerber [aut],
Thomas Gsponer [aut],
Bjoern Bornkamp [cre] (maintainer)

Maintainer Bjoern Bornkamp <bbnkmp@mail.de>

R topics documented:

gsbDesign-package *Group Sequential Bayesian Design*

Description

The gsbDesign package allows to evaluate the operating characteristics for a group sequential design with Bayesian success/futility criteria and prior information. A clinical trial with two arms, a normal endpoint and an arbitrary number of interim analyses can be evaluated. The success and futility criteria at each interim analysis are based on the posterior distribution for the true treatment difference (delta). An arbitrary number of success and futility criteria can be specified at each interim analysis. The success criteria are of the form:

$$P(\delta > ds|data) > ps$$

And the futility criteria are of the form:

$$P(\delta < df|data) > pf$$

Here ds and df are user-specified effect thresholds, and ps and pf are user-defined probability thresholds.
Prior information can either be specified for the true treatment difference (delta), or for the true effects in the two treatment arms. Only normal prior distributions can be used. In all cases informative priors are specified in terms of a mean and an effective sample size defined relative to sigma.
The user also has to specify the number of patients for each stage of the group sequential design, and the standard deviation (sigma) of the endpoint (assumed to be known).
The operating characteristics are either evaluated for a user-specified grid of true treatment differences, or for a grid or set of true treatment means for the two arms. The operating characteristics of main interest are the probabilities of success and futility at each interim analysis, and the expected sample size.
The main function of the package is gsb(). More detailed information can be found in the help of function gsb().

Details

Package:	gsbDesign
Type:	Package
Version:	1.0-3
Date:	2024-01-23
License:	GNU General Public License >=3
LazyLoad:	yes

Author(s)

Florian Gerber <florian.gerber@math.uzh.ch>, Thomas Gsponer

References

- Gerber F, Gsponer T (2016). gsbDesign: An R Package for Evaluating the Operating Characteristics of a Group Sequential Bayesian Design. Journal of Statistical Software, 69(11), 1-23, DOI: 10.18637/jss.v069.i11
- Gsponer T, Gerber F, Bornkamp B, Ohlssen D, Vandemeulebroecke M, Schmidli H (2014). A Practical Guide to Bayesian Group Sequential Designs. Pharmaceutical Statistics, 13(1) 71-80, DOI: 10.1002/pst.1593
- Berry SM, Carlin BP, Lee JJ, and Mueller P (2010). Bayesian Adaptive Methods for Clinical Trials. Chapman&Hall/CRC, London.
- Jennison C, and Turnbull BW (2000). Group Sequential Methods with Applications to Clinical Trials. Chapman&Hall/CRC, London.
- Spiegelhalter DJ, Abrams KR, and Myles, J. P. (2004). Bayesian Approaches to Clinical Trials and Health Care Evaluation. Wiley, New York.

See Also

gsb

---

gsb *Group Sequential Bayesian Design*

Description

The gsbDesign package allows to evaluate the operating characteristics for a group sequential design with Bayesian success/futility criteria and prior information. A clinical trial with two arms, a normal endpoint and an arbitrary number of interim analyses can be evaluated. The success and futility criteria at each interim analysis are based on the posterior distribution for the true treatment difference (delta). An arbitrary number of success and futility criteria can be specified at each interim analysis. The success criteria are of the form:

$$P(\delta > ds|data) > ps$$

And the futility criteria are of the form:

$$P(\delta < df|data) > pf$$

Here ds and df are user-specified effect thresholds, and ps and pf are user-defined probability thresholds.
Prior information can either be specified for the true treatment difference (delta), or for the true effects in the two treatment arms. Only normal prior distributions can be used. In all cases informative priors are specified in terms of a mean and an effective sample size defined relative to sigma.
The user also has to specify the number of patients for each stage of the group sequential design, and the standard deviation (sigma) of the endpoint (assumed to be known).
The operating characteristics are either evaluated for a user-specified grid of true treatment differences, or for a grid or set of true treatment means for the two arms. The operating characteristics of main interest are the probabilities of success and futility at each interim analysis, and the expected sample size.
The main function of the package is gsb().

Usage

```
gsb(design = NULL, simulation = NULL)

gsbDesign(nr.stages = NULL,
          patients = NULL,
          sigma = 1,
          criteria.success = NULL,
          criteria.futility = NULL,
          prior.difference = "non-informative",
          prior.control = "non-informative",
          prior.treatment = "non-informative")

gsbSimulation(truth = NULL,
              type.update = c("treatment effect","per arm"),
              method = c("numerical integration", "simulation", "both"),
              grid.type = c("table","plot","sliced","manually"),
              nr.sim = 50000,
              warnings.sensitivity = 100,
              seed = NULL)
```

Arguments

design	an object of class gsbDesign containing information on the trial design. I.e. information on the number of stages (nr.stages), the number of patients (patients), and the standard deviation (sigma), the decision criteria (criteria.success, criteria.futility) and prior information (prior.difference, prior.control, prior.treatment). This object can be created with function gsbDesign() as shown in the examples below.
simulation	an object of class gsbSimulation containing information on how to evaluate the operating characteristics. I.e. on the true effects (truth, grid.type), the Bayesian update (type.update), whether simulation or numerical integration is used (method) and simulation parameters (nr.sim, warnings.sensitivity, seed). This object can be created with function gsbSimulation() as shown in the examples below.
nr.stages	a numeric of length 1 specifying the number of stages (interim and final analyses).
patients	a numeric of length 1 in order to specify an equal number of patients in the control and treatment arm in every stage. Enter a numeric of length 2 in order to specify the number of patients in the control and treatment arm separately. If the number of patients is not equal for all stages enter a n x 2 matrix. n denotes the number of stages. More details are shown in the examples below.
sigma	a numeric of length 1 in order to specify an equal standard deviation sigma for both arms. If the sigma is not equal in both arms enter a numeric of length 2 specifying c(sigma.control, sigma.treatment). More details are shown in the examples below.

`criteria.success`
: a `numeric` or `matrix` containing the success criteria ds and ps. More details are shown in the examples below.

`criteria.futility`
: a `numeric` or `matrix` containing the futility criteria df and pf. More details are shown in the examples below.

`prior.difference`
: if `prior.difference = "non-informative"` a non-informative (flat) prior on difference is used. Enter an informative prior as `numeric` as shown in the examples below. If this prior is specified the argument `type.update` as to be on "treatment effect".

`prior.control`
: if `prior.control = "non-informative"` a non-informative (flat) prior is used in the control arm. Enter an informative prior as `numeric` as shown in the examples below. If this prior is specified the argument `type.update` has to be "per arm".

`prior.treatment`
: if `prior.treatment = "non-informative"` a non-informative (flat) prior is used in the treatment arm. Enter an informative prior as `numeric` as shown in the examples below. If this prior is specified the argument `type.update` has to be "per arm".

`truth`
: specifies the truth to evaluate.

  If `type.update = "treatment effect"` this argument specifies the true treatment effect delta (= treatment - placebo). It has the form `truth = c(min, max, n)` where `min` and `max` indicates the range of the true treatment effect and `n` specifies the number of true values. (See example 1 below.)
  Alternatively a `numeric` containing true delta values can be entered. To use this set `grid.type = "manually"`. (example 2).
  If `type.update = "per arm"` and `grid.type = "table"` the argument has to be specified as `list` containing a vector of true control values and true treatment values.

  If `type.update = "per arm"` and `grid.type = "sliced"` the argument has to be specified as `list` containing a vector of true control values and a vector of true deltas (= treatment - control).

  If `type.update = "per arm"` and `grid.type = "plot"` the argument has to be specified as follows:
  `truth = c(min.placebo, max.placebo,min.treatment, max.treatment, n)`, here `n` is the number of grid points.

  If `type.update = "per arm"` and `grid.type = "manually"` the argument has to be a n x 2 - matrix containing the true placebo values in the first column and the corresponding treatment values in the second column.

`type.update`
: If `type.update = "treatment effect"`, the Bayesian update from prior to posterior is calculated on treatment effect delta. If `type.update = "per arm"`, the update is calculated separately in the placebo and the treatment arm. In this case

it is possible to enter prior information in only one arm. For type.update = "per arm" only a simulation method is implemented.

method — If the type.update = "treatment effect", the operating characteristics can be obtained by simulation or by numerical integration, which is faster. So method can be set to "simulation", "numerical integration" or "both". We used method = "both" to check, whether the different methods yield the same results. If type.update = "per arm" only a simulation method is implemented.

grid.type — If type.update = "per arm", there are 4 possibilities to specify the truth:
1. grid.type = "table" for presenting the results in a table.
2. grid.type = "sliced" for presenting the results in a table.
3. grid.type = "plot" optimized grid for plotting the results.
4. grid.type = "manually" for specifying a grid in a matrix.
If type.update = "treatment effect" only grid.type = "table" and grid.type = "manually" are valid inputs. The argument truth should be specified according to the argument grid.type.

nr.sim — a numeric of length 1 specifying the number of simulations.

warnings.sensitivity — a numeric of length 1. If the number of (nr.sim) is smaller than this integer value during the simulation, a warning message is printed. The number of simulations is decreasing in each stage because only the simulated trials with no decision in the precedent stages are remaining for simulation. Thus the accuracy decreases in each stage. warnings.sensitivity has to be specified if method = "simulation" or method = "both".

seed — a numeric of length 1 to set a seed value for the random number generator. If seed = "generate" a new seed value is generated.

Details

See Gerber and Gsponer (2016) for more details.

Value

OC — a data.frame containing the resulting operating characteristics per stage (i.e. the probabilities of success, of futility and of success or futility), the cumulative operating characteristics per stage and the expected sample size per stage.

boundary — a data.frame containing the boundaries, i.e. the bounds on posterior scale and the standardized bounds. This is available if type.update = "treatment effect" and method = "numerical integration" or method = "both".

design — same as input.

simulation — same as input.

delta.grid — an object of class gsbDelta.grid and matrix containing the grid of true control and true treatment values. Available if type.update = "per arm".

warnings — a matrix containing warnings if any. If a simulation was done (i.e. method = "simulation" or method = "both") the number of simulations is decreasing in each stage because only the simulated trials with no decision in the precedent stages are remaining for simulation. Thus the accuracy decreases in each stage.

warnings indicates the stages (and the deltas) for which the number of remaining simulations is below the threshold specified in warnings.sensitivity.

system.time contains information on the used time for the simulation / numerical integration.

Author(s)

Florian Gerber <florian.gerber@math.uzh.ch>, Thomas Gsponer

References

- Gerber F, Gsponer T (2016). gsbDesign: An R Package for Evaluating the Operating Characteristics of a Group Sequential Bayesian Design. Journal of Statistical Software, 69(11), 1-23, DOI: 10.18637/jss.v069.i11
- Gsponer T, Gerber F, Bornkamp B, Ohlssen D, Vandemeulebroecke M, Schmidli H (2014). A Practical Guide to Bayesian Group Sequential Designs. Pharmaceutical Statistics, 13(1) 71-80, DOI: 10.1002/pst.1593
- Berry SM, Carlin BP, Lee JJ, and Mueller P (2010). Bayesian Adaptive Methods for Clinical Trials. Chapman&Hall/CRC, London.
- Jennison C, and Turnbull BW (2000). Group Sequential Methods with Applications to Clinical Trials. Chapman&Hall/CRC, London.
- Spiegelhalter DJ, Abrams KR, and Myles, JP (2004). Bayesian Approaches to Clinical Trials and Health Care Evaluation. Wiley, New York.

See Also

plot.gsbMainOut, tab, gsbBayesUpdate, gsbCriteria, gsbDesign-package

Examples

```
## E X A M P L E 1: Update on treatment effect, flat prior
##
## A. Trial Design:
## ----------------
## A.1 2 stages (interim + final):
##     --> nr.stages = 2
## A.2 10 patients per arms and stages. (total 2*2*10 = 40 patients)
##     --> patients = 10
## A.3 Sigma in both arms = 10
##     --> sigma = 10
## A.3 Criteria:
##     stop for success, if P( delta > 0 | data ) >= 0.8
##                       AND P( delta > 7 | data ) >= 0.5
##     --> criteria.success = c(0,0.8,7,0.5)
##     stop for futility, if P( delta < 2 | data ) >= 0.8
##     --> criteria.futility = c(2,0.8)
## A.4 Prior:
##     --> prior = "non-informative"

design1 <- gsbDesign(nr.stages = 2,
                     patients = 10,
```

```
                    sigma = 10,
                    criteria.success = c(0,0.8, 7, 0.5),
                    criteria.futility = c(2,0.8),
                    prior.difference = "non-informative")
design1

## B. Simulation Settings
## ----------------------
## B.1 True treatment effects to be evaluated = seq(-10,20,60)
##     --> truth = c(-10,20,60)
## B.2 Bayesian update on treatment effect delta (= treatment - control)
##     --> type.update = "treatment effect"

simulation1 <- gsbSimulation(truth=c(-10,20,60),
                             type.update="treatment effect")
simulation1

## C.1 Calculate the operating characteristics
x1 <- gsb(design=design1, simulation=simulation1)
x1

## D.1 Table the probabilities of success
t1.1 <- tab(x1, "success", digits=2)
t1.1

## D.2 Table the cumulative probabilities of futility at delta = c(-5,0,5.57)
##     (for 5.57 a linear interpolation is used.)
t1.2 <- tab(x1, "cumulative futility", atDelta = c(-5,0,5.57), digits=5)
t1.2

## D.3 Table the expected sample size (digits == 0 --> ceiling)
t1.3 <- tab(x1, "sample size", atDelta= c(-5,0,5,16), digits=0)
t1.3

## E.1 Plot the operating characteristics
plot(x1)

## E.2 Plot the operating characteristics
plot(x1,"cumulative all")

## E.3 Plot the expected sample size
plot(x1, what="sample size")

## F.1 Boundaries / criteria
x1$boundary
plot(x1, what="boundary")
plot(x1, what="std.boundary")


## E X A M P L E 2: Update on treatment effect, informative prior
##
## A. Trial design:
## ----------------
```

```
## A.1 3 stages (interims + final):
##     --> nr.stages = 3
## A.2 10 patients per stage in control arm
##     15 patients per stage in treatment arm
##     (i.e. total 3 * ( 10 + 15 ) = 75 patients)
##     --> patients = c(10,15)
## A.3 Sigma in control arm = 9, sigma in treatment arm = 12
##     --> sigma = c(9,12)
## A.3 Criteria:
##     stop for success, if P( delta > 0 | data ) >= 0.8
##                       AND P( delta > 7 | data ) >= 0.5
##     --> criteria.success = c(0,0.8,7,0.5)
##     not stop for futility, i.e. no futility criteria
##     --> criteria.futility = NA
## A.4 Prior on difference:
##     prior difference = 3
##     informative prior equivalent to:
##     5 patients in control arm; 2 patients in treatment arm
##     --> prior = c(3,5,2)

design2a <- gsbDesign(nr.stages = 3,
                      patients = c(10,15),
                      sigma=c(9,12),
                      criteria.success = c(0,0.8,7,0.5),
                      criteria.futility = NA,
                      prior.diff = c(3,5,2))
design2a

## A similar design with 3 success criteria can be specified as follows
## A.3 criteria:
##     Stage 1: stop for success, if P( delta > 0  | data ) >= 0.8
##                               AND if P( delta > 10 | data ) >= 0.5
##                               AND if P( delta > 14 | data ) >= 0.4
##     Stage 2: stop for success, if P( delta > 0  | data ) >= 0.8
##                               AND if P( delta > 9  | data ) >= 0.5
##                               AND if P( delta > 13 | data ) >= 0.4
##     Stage 3: stop for success, if P( delta > 0  | data ) >= 0.8
##                               AND if P( delta > 7  | data ) >= 0.5
##                               AND if P( delta > 12 | data ) >= 0.4
##     --> criteria.success = rbind(c(0,0.8, 10,0.5, 14,0.4),
##                                  c(0,0.8,  9,0.5, 13,0.4),
##                                  c(0,0.8,  7,0.5, 12,0.4))

design2b <- gsbDesign(nr.stages = 3,
                      patients = c(10,15),
                      sigma = c(9,12),
                      criteria.success = rbind(c(0,0.8, 10,0.5, 14,0.4),
                                               c(0,0.8, 9,0.5,  13,0.4),
                                               c(0,0.8, 7,0.5,  12,0.4)),
                      criteria.futility = NA,
                      prior.diff = c(3,5,2))
design2b
```

```
## B. Simulation Settings
## ----------------------
## B.1 True treatment effects to be evaluated from -5 to 30
##     --> truth = -5:30
## B.2 To enter the values in this format set grid.type = "manually"
##     --> grid.type = "manually"

## B.2 Bayesian update on treatment effect delta (treatment - control)
##     --> type.update = "treatment effect"

simulation2 <- gsbSimulation(truth = -5:30,
                             grid.type ="manually",
                             type.update = "treatment effect")
simulation2

## C. Calculate the operating characteristics
x2a <- gsb(design = design2a, simulation = simulation2)
x2b <- gsb(design = design2b, simulation = simulation2)
x2a
x2b

## D. Table the cumulative probabilities of success of 'design2b'
##    at delta = c(-5,0,5.57). For 5.57 a linear interpolation is used.
t2b <- tab(x2b, "cumulative success", atDelta = c(-5,0,5.57), digits=5)
t2b

## E. Plot the operating characteristics of 'design2a' and 'design2b'
plot(x2a)
plot(x2b)
plot(x2a,"cumulative all")

## F.1 Boundaries / criteria of 'design2b'
x2b$boundary
plot(x2b, what="boundary")
plot(x2b, what="std.boundary")




## E X A M P L E 3: Update on treatment effect, informative prior
##
## A. Trial Design
## ---------------
## A.1 3 stages (interims + final):
##     --> nr.stages = 3
## A.2 Patients:
##         Stage 1: 10 patients in control arm; 15 patients in treatment arm
##         Stage 2: 20 patients in control arm; 30 patients in treatment arm
##         Stage 3: 30 patients in control arm; 45 patients in treatment arm
##     --> patients = rbind(c(10,15),c(20,30),c(30,45))
## A.3 Sigma in control arm = 9 ; in treatment arm = 12
##     --> sigma = c(9,12)
```

```
## A.4 Success criteria for all stages:
##     stop for success, if P( delta > 0 | data ) >= 0.8
##                       AND P( delta > 7 | data ) >= 0.5
##      --> criteria.success = c(0,0.8,7,0.5)
## A.5 Futility criteria:
##         Stage 1: no futility criteria
##         Stage 2: stop for futility, if P( delta < 2 | data ) >= 0.8
##         Stage 3: stop for futility, if P( delta < 2 | data ) >= 0.8
##     --> criteria.futility = rbind(c(NA,NA),c(2,0.8),c(2,0.8))
## A.6 Prior on treatment effect:
##         difference  = 3;
##         informative prior equivalent to:
##         2 placebo patient; 1 treatment patient
##     --> prior.difference = c(3,2,1)

design3 <- gsbDesign(nr.stages = 3,
                     patients = rbind(c(10,15),c(20,30),c(30,45)),
                     sigma=c(9,12),
                     criteria.success = c(0,0.8,7,0.5),
                     criteria.futility = rbind(c(NA,NA),c(2,0.8),c(2,0.8)),
                     prior.difference = c(3,2,1))
design3

## B. Simulation Settings
## ----------------------
## B.1 True treatment effects to be evaluated at seq(-5,20,15)
##     --> truth = c(-5,20,15)
## B.2 Bayesian update on treatment effect delta (= treatment - control)
##     --> type.update = "treatment effect"
## B.3 Operating characteristics are evaluated by simulation and
##     numerical integration to double check the results
##     --> method = "both"
## B.4 Number of simulations = 5000
##     --> nr.sim = 5000
## B.5 If the number of simulated trials is smaller than 300
##     during the simulation print a warning.
##     --> warnings.sensitivity = 300
## B.6 A seed value is set to 13
##     --> seed = 13

simulation3 <- gsbSimulation(truth = c(-5,20,15),
                             type.update = "treatment effect",
                             method = "both",
                             nr.sim = 5000,
                             warnings.sensitivity = 300,
                             seed = 13)
simulation3

## C. Calculate the operating characteristics
x3 <- gsb(design = design3, simulation = simulation3)
x3

## D. The summary(x3) is almost the same as print(x3) but its entries
```

```
##    can be saved as list.
s3 <- summary(x3)
names(s3)

## E.1 Plot the operating characteristics
plot(x3)

## E.2 Plot the operating characteristics obtained by simulation and
##     numerical integration in one plot. The lines should be identical (then
##     only one line is visible)
plot(x3, "both")
plot(x3, "cumulative both")



## E X A M P L E 4 - Boundaries / Criteria
## See how the Bayesian boundaries change within 10 stages.
##
## A. Trial Design:
## ----------------
## A.1 10 stages (interims + final):
##     --> nr.stages = 10
## A.2 10 patients per arm and stage. (total 2*10*10 = 200 patients)
##     --> patients = 10
## A.3 sigma in both arms = 10
##     --> sigma = 10
## A.3 Criteria:
##     stop for success, if P( delta > 0 | data ) >= 0.8
##                       AND P( delta > 7 | data ) >= 0.5
##     --> criteria.success = c(0,0.8,7,0.5)
##     stop for futility, if P( delta < 2 | data ) >= 0.8
##     --> criteria.futility = c(2,0.8)
## A.4 Prior:
##     --> prior = "non-informative"

design4 <- gsbDesign(nr.stages=10,
                     patients=10,
                     sigma=10,
                     criteria.success=c(0,0.8, 7, 0.5),
                     criteria.futility=c(2,0.8),
                     prior.difference="non-informative")
design4


## B. Simulation Settings
## ------------------------------------------
## B.1 True treatment effects to be evaluated = seq(-10,20,60)
##     --> truth = c(-10,20,60)
## B.2 Bayesian update on treatment effect delta (= treatment - control)
##     --> type.update = "treatment effect"

simulation4 <- gsbSimulation(truth=c(-10,20,60),
                             type.update="treatment effect")
```

```
simulation4

## C. Calculate the operating characteristics
x4 <- gsb(design = design4, simulation = simulation4)
x4

## D. Boundaries / criteria
x4$boundary
plot(x4, what="boundary")
plot(x4, what="std.boundary")


## E X A M P L E 5 - Bayesian update "per arm",
##
## A. Trial Design:
## ----------------
## A.1 3 stages (interims + final):
##     --> nr.stages = 3
## A.2 12 patients per stage in control arm
##     20 patients per stage in treatment arm
##     (i.e. total 3 * ( 12 + 20 ) = 96 patients)
##     --> patients = c(12,20)
## A.3 sigma in both arms = 10
##     --> sigma = 10
## A.3 Criteria:
##     stop for success, if P( delta > 0 | data ) >= 0.8
##                       AND P( delta > 7 | data ) >= 0.5
##     --> criteria.success = c(0,0.8,7,0.5)
##     stop for futility, if P( delta < 2 | data ) >= 0.8
##     --> criteria.futility = c(2,0.8)
## A.4 Prior:
##     informative prior equivalent to:
##     2 patients in control arm with mean = 0
##     --> prior.control = c(0,2)
##     1 patient in treatment arm with mean = 7
##     --> prior.treatment = c(7,1)

design5 <- gsbDesign(nr.stages=3,
                     patients=c(12,20),
                     sigma=10,
                     criteria.success=c(0,0.8,7,0.5),
                     criteria.futility=c(2,0.8),
                     prior.control=c(0,2),
                     prior.treatment=c(7,1))
design5

## B.  Simulation Settings: - with table grid
## ------------------------------------------
## B.1 True control/treatment values:
##       control = seq(1,5,0.5)
##       treatment = seq(1,7,1)
##       --> truth = list(seq(1,5,0.5),seq(1,7,1))
## B.2 Output optimized to create table
```

```
##       --> grid.type = "table"
## B.3 Bayesian update per arm
##       --> type.update = "per arm"
## B.4 Number of simulations = 5000 (which is low)
##       --> nr.sim = 5000
## B.5 If the number of simulations is smaller than 2000
##       print a warning.
##       --> warnings.sensitivity = 2000
## B.6 A seed value is set to 13
##       --> seed = 13

simulation5.table <- gsbSimulation(truth = list(seq(1,5,0.5), seq(1,7,1)),
                                   grid.type = "table",
                                   type.update = "per arm",
                                   nr.sim = 5000,
                                   warnings.sensitivity = 2000,
                                   seed = 13)
simulation5.table

## The same grid can be specified manually by
simulation5.manually <- gsbSimulation(truth = as.matrix(expand.grid(seq(1,5,0.5),seq(1,7,1))),
                                      grid.type = "manually",
                                      type.update = "per arm",
                                      nr.sim = 5000,
                                      warnings.sensitivity = 2000,
                                      seed = 13)
simulation5.manually

## To specify a grid optimized for sliced plotting with
## control values from -10 to 0 and treatment values from -10 to 25

simulation5.sliced <- gsbSimulation(truth = list(control=seq(-10,0,2), delta=seq(-10,25,4)),
                                    grid.type = "sliced",
                                    type.update = "per arm",
                                    nr.sim = 5000,
                                    warnings.sensitivity = 2000,
                                    seed = 13)
simulation5.sliced

## To specify a grid optimized for plotting with
## control values from 1 to 5 and treatment values from 1 to 7
## with approximately 20 values enter:
simulation5.plot <- gsbSimulation(truth = c(1,5,1,7,20),
                                  grid.type = "plot",
                                  type.update = "per arm",
                                  nr.sim = 5000,
                                  warnings.sensitivity = 2000,
                                  seed = 13)
simulation5.plot

## C. Use function gsb
x5.table <- gsb(design5,simulation5.table)
x5.sliced <- gsb(design5,simulation5.sliced)
```

```
x5.plot <- gsb(design5,simulation5.plot)
x5.table

## D. Tables
## D.1 For any grid a table in long format can be obtained
t5.1 <- tab(x5.table,"cumulative futility")
head(t5.1)

t5.2 <- tab(x5.sliced,"all")
head(t5.2)

## D.2 For the "table" grid there are additionally tables in wide format available.
t5.3 <- tab(x5.table,"success", wide=TRUE)
t5.3

## Fix a stage, e.g. stage 2, to get a matrix
t5.3[,,2]

## D.2 Set delta.control to '3' to get a matrix
t5.3["contr 3",,]

# D.3 Plot results
plot(x5.table)
plot(x5.plot)
plot(x5.sliced)
plot(x5.sliced, sliced=TRUE)
plot(x5.sliced, sliced=TRUE, range.control=c(-4,0))
plot(x5.sliced, what="success", sliced=TRUE, range.control=c(-4,0))

## the plot can differ because the number of simulations "nr.sim"
## is low and because the grids are different
plot(x5.plot,"sample size", color=FALSE)
head(tab(x5.table,"sample size"))
```

---

gsbBayesUpdate *Bayesian Update*

Description

Bayesian update from prior and data to posterior for normally distributed data with known sigma.

Usage

```
gsbBayesUpdate(alpha, beta, meanData, precisionData, with.alpha = TRUE)
```

Arguments

alpha	vector of prior means.
beta	vector of prior precisions.

meanData	vector of means from data.
precisionData	vector of precisions from data.
with.alpha	logical. If with.alpha = TRUE, alpha, beta, meanData and precisionData has to be specified and the posterior means, posterior precisions and weights are returned. Else only beta and precisionData has to be specified and the posterior precisions and weights are returned.

Value

alpha	posterior means. Only if with.alpha = TRUE.
beta	posterior precisions.
weight	weights of the priors relative to the whole information after updating.

Note

This function is used in the function gsb().

Author(s)

Florian Gerber <florian.gerber@math.uzh.ch>, Thomas Gsponer

See Also

gsb

Examples

```
## One dimensional case, with.alpha = FALSE
gsbBayesUpdate(beta=10,precisionData=20, with.alpha=FALSE)

## Two dimensional case, with.alpha = TRUE
gsbBayesUpdate(alpha=c(5,6),beta=c(10,11),meanData=c(10,11),
               precisionData=c(20,21),with.alpha=TRUE)
```

---

gsbCriteria *Criteria on posterior scale*

Description

Transforms the criteria on posterior-scale.

Usage

```
gsbCriteria(criteria, priorMean, postPrecision, weight)
```

Arguments

criteria	an array with criteria. Can be created with the function gsbDesign().
priorMean	a vector of prior means.
postPrecision	a vector of posterior precisions. Can be created with function gsbBayesUpdate().
weight	a vector of weights from the bayesian update. Can be created with function gsbBayesUpdate().

Value

CS	vector of success criteria on posterior scale
CF	vector of futility criteria on posterior scale

Note

This function is used in function gsb().

Author(s)

Florian Gerber <florian.gerber@math.uzh.ch>, Thomas Gsponer

See Also

gsb

---

plot.gsbMainOut *Plot methods*

Description

Methods for plotting the results of gsb().

Usage

```
## S3 method for class 'gsbMainOut'
plot(x,
     what=c("all", "cumulative all",
            "both", "cumulative both",
            "sample size", "success", "futility",
            "success or futility", "indeterminate", "cumulative success",
            "cumulative futility", "cumulative success or futility",
            "cumulative indeterminate", "boundary",
            "std.boundary","delta.grid","patients"),
     range.delta = "default",
     stages = "default",
     delta.grid = TRUE,
     color = TRUE,
```

```
     smooth = 100,
     contour = TRUE,
     export = FALSE,
     path = tempdir(),
     sliced = FALSE,
     range.control="default", ...)

## S3 method for class 'gsbSimulation'
plot(x,...)

## S3 method for class 'gsbDesign'
plot(x,...)
```

Arguments

x	object of appropriate class.
what	a character string to choose a plot. It should correspond to one level of OC$type where OC is an object of the output of gsb(). Additional possibilities are what = "all" to plot the success-, futility- and success or futility-probabilities, what = "cumulative all" to plot the cumulative success-, cumulative futility- and cumulative success or futility-probabilities, what = "boundary" or what = "std.boundary" to plot the bounds, what = "patients" for a histogram of the patients per stage and what = "delta.grid" to plot the grid of delta's.
range.delta	a vector of length 2. For choosing the plot limits manually set range.delta = c(min, max) for a Bayesian update on "treatment effect" or range.delta = c(control.min, control.max, treatment.min, treatment.max) for a Bayesian update "per arm". If range.delta = "default" the range of the plot is chosen so that all delta's are covered.
stages	a vector of length 2 containing the number of the lowest and highest stage, which should be plotted. If stages = "default" all stages are plotted (expect for what = "sample size" where only the last stage is plotted.)
delta.grid	logical. If TRUE the delta grid is plotted too.
color	logical. If TRUE the plot is colored.
smooth	a vector of length 1. A higher number makes the plot 'smoother' if type.update = "per arm".
contour	logical. If TRUE contour lines are added to the plot.
export	logical. If TRUE the plot is save as .png-file.
path	character. to specify the location to which to table should be exported. the default 'tempdir()' exports the table to a temporary directory.
sliced	logical. If TRUE the contour plot for type.update = "per arm" is shown in several 2D plots. In order to use this option the argument grid.type of gsbSimulation() has to be sliced.
range.control	a vector of length 2 or \"default\". If special = TRUE the range of the control values can be set manually as vector c(min, max).
...	further arguments passed to or from other methods.

Value

Returns an object of class "trellis"

Author(s)

Florian Gerber <florian.gerber@math.uzh.ch>, Thomas Gsponer

References

uses the R-package 'lattice'.

See Also

gsb, xyplot

Examples

```
## please see examples of function 'gsb'.
## ---------------------------------------


## ---------------------------------------
## alternative plots can be created for example
## with package 'ggplot2'.

des <- gsbDesign(nr.stages=2,
                     patients=10,
                     sigma=10,
                     criteria.success=c(0,0.8, 7, 0.5),
                     criteria.futility=c(2,0.8),
                     prior.difference="non-informative")

sim <- gsbSimulation(truth=c(-10,20,60),
                             type.update="treatment effect")


x <- gsb(des,sim)

## get data.frame with operating characteristics
datgraph <- x$OC


## prepare for plot
sub <- c("success", "futility", "success or futility")
datgraph2 <- subset(datgraph,datgraph$type %in% sub)
datgraph2$type <- as.factor(paste(datgraph2$type))
datgraph2$value[datgraph2$type=="cumulative success or futility"] <-
1-datgraph2$value[datgraph2$type=="cumulative success or futility"]
levels(datgraph2$type) <- c("1)cumulative futility" ,"3)cumulative success","2)indeterminate")
datgraph2$type=as.factor(paste(datgraph2$type))
levels(datgraph2$type) <- c("cumulative futility" ,"indeterminate","cumulative success")
```

```
datgraph2 <- datgraph2[order(datgraph2$delta),]


## plots
library(ggplot2)
p1 <- qplot(delta,value,geom="blank",color=type,facets=.~stage,data=datgraph2,
xlab=expression(delta))

p1+geom_line(size=1.5)+scale_color_manual(values = c("cumulative futility" = "dark red",
"indeterminate" = "orange", "cumulative success" = "dark green"))

p2=p1+geom_area(aes(x = delta,y=value,fill=type))

p2+scale_fill_manual(values = c("cumulative futility" = "dark red",
"indeterminate" = "orange", "cumulative success" = "dark green"))
```

---

tab *get tables.*

Description

This function creates tables from the output of function `gsb()`.

Usage

```
tab(x,
    what=c("all", "cumulative all", "success", "futility",
           "indeterminate", "success or futility",
           "cumulative success", "cumulative futility",
           "cumulative indeterminate", "cumulative success or futility",
           "sample size"),
    atDelta = "default",
    wide=FALSE,
    digits = 3,
    export = FALSE,
    sep = ",",
    path = tempdir())
```

Arguments

`x`	object of class `gsbMainOut` which is returned by the function `gsb()`.
`what`	`string` to specify the content of the table. It should correspond one level of `OC$type` in the output of `gsb()`.
`atDelta`	if `"default"` the table shows the operating characteristics evaluated at the actually calculated true values entered in argument `truth` of function `gsbSimulation`. Alternatively a `numeric` of arbitrary length can be entered. Then the operating

	characteristics are displayed for the true value delta entered in this `numeric` using a linear interpolation. This only works if `type.update = "treatment effect"`.
`wide`	`logical`. If TRUE a zable in wide format is produced. Only possible if `grid.type = "table"`.
`digits`	`numeric` of length 1 specifying the number of digits which should be displayed in the table. The function `round()` is used. If `what = "sample size"` and `digits = 0`, the function uses `ceiling()` instead.
`export`	`logical`. if `TRUE` the table is exported and saved as .cvs-file.
`sep`	`character`. the field separator string. if the table is exported the values are separated by this string.
`path`	`character`. to specify the location to which to table should be exported. the default 'tempdir()' exports the table to a temporary directory.

Value

Returns a matrix with the results from the output of function `gsb()`.

Author(s)

Florian Gerber <florian.gerber@math.uzh.ch>

Examples

```
## please see examples of function 'gsb'.
```

Index