Package ‘MethodDev’

July 24, 2019

Type Package

Title Dosen't have a title

Version 0.1.3

Author Bin Zhuo

Maintainer Bin Zhuo <bzhuo@amgen.com>

Description This package is developed to maintain commonly used functions that
I write for the work at simulation team in Design and Innovation team within Amgen.

License What license is it under?

Encoding UTF-8

RdMacros Rdpack

Imports Rdpack,
dplyr,
foreach,
gsDesign,
baseUtility

LazyData true

RoxygenNote 6.1.1

Suggests testthat,
knitr,
rmarkdown

VignetteBuilder knitr

R topics documented:

`bvn_prob` *upper tail probability of a standard binormal distribution*

Description

calculate the bivariate normal probabilities for a given h,k and rho

Usage

```
bvn_prob(h, k, rho)
```

Arguments

`h`	the cutoff value for x; will integrate from 'h' to 'Inf'
`k`	the cutoff value for y; will integrate from 'k' to 'Inf'
`rho`	the correlation of the bivariate normal distribution.

Details

This function calculates the following quantity,

$$L(h, k, \rho) = \frac{1}{\sqrt{1-\rho^2}} \int_h^\infty \int_k^\infty e^{-\frac{x^2-2\rho xy+y^2}{2(1-\rho^2)}} dxdy$$

using formula Eq.(3) in the reference.

Value

a p value

References

Genz A (2004). "Numerical computation of rectangular bivariate and trivariate normal and t probabilities." *Statistics and Computing*, **14**(3), 251–260.

https://link.springer.com/content/pdf/10.1023/B:STCO.0000035304.20635.31.pdf

calc_p1_or *odds ratio and probabilities*

Description

odds ratio and probabilities

Usage

```
calc_p1_or(p0, p1 = NULL, or = NULL)
```

Arguments

`p0, p1`	probability in the control/treatment arm
`or`	odds ratio which is expressed as `odds_treatment/odds_control`

Details

this two functions calculates odds ratio based on two probabilities, or probability of treatment given odds ratio and probability in the control arm

Value

probability in the treatment arm, or odds ratio

Examples

```
p0 <- 0.59; p1 <- 0.812; or <- 3;
calc_p1_or(p0 = p0, or = or);
calc_p1_or(p0 = p0, p1 = p1)
```

compute_lambda *Calculate λ's based on the formula*

Description

Calculate λ's based on the formula

Usage

```
compute_lambda(os, osp, pfs)
```

Arguments

`os`	median os
`osp`	median OS after progression
`pfs`	median PFS

Value

a vector of lambda's and the theoretical correlation between OS and PFS

cong_dat_gen *Simulate data for all comers/enrichment population*

Description

Simulate data for all comers/enrichment population

Usage

```
cong_dat_gen(nsbj, alloc = c(1, 1), b_size = 2, rate,
  enrichment = FALSE, marker_prob = c(0.7, 0.3),
  marker_name = c("DLL3+", "DLL3-"), par_ctrl_pos, par_ctrl_neg,
  par_trt_pos, par_trt_neg, ...)
```

Arguments

nsbj	number of subjects to be simulated
alloc	allocation vector, length corresponds to number of arms; as to be integer; enter 1 if single arm
b_size	block size, has to be multiple of sum(alloc), enter 1 if single arm
rate	enrollment rate per unit time
enrichment	Is this data generated for enrichment population?
marker_prob	vector of prevalence probability of different category, if doesn't add up to 1, will automatically standardize and generates warning. For enrichment population, just set `marker_prob = 1`
marker_name	vector of names of different subgroup
par_trt_pos, par_trt_neg, par_ctrl_pos, par_ctrl_neg	parameter specification for treatment/control and biomarker positive/negative population. For enrichment data generation, `par_trt_neg` and `par_ctrl_neg` are set to be `NULL`
...	other parameters from function `enrl_dat_gen`

Value

a tibble of simulated survival data

cong_final_analysis *final analysis for Cong's method*

Description

final analysis for Cong's method

Usage

```
cong_final_analysis(snapshot, marker_positive = "DLL3+",
  alpha1 = 0.0125, alpha2 = 0.0125)
```

Arguments

snapshot	the data snapshot
marker_positive	the label for biomarker positive population
alpha1	significance level for testing all-comers population
alpha2	significance level for testing biomarker positive population

Value

a tibble summarizing the analysis results

---

cong_simu_trial *Trial process simulation*

Description

This function simulates survival data, performs enrichment analysis, based on which results, it enrolls biomarker positive population when enrichment is needed, or continues as the original trial when enrichment is not needed, and lastly, performs final analysis. Note that the final analysis is done on two analysis sets: all-comers and biomarker positive population; a win on either population will result in a positive outcome.

Usage

```
cong_simu_trial(n_allcomer, n_enrichment, alloc = c(1, 1), b_size = 2,
  rate, marker_positive = "DLL3+", marker_negative = "DLL3-",
  marker_prob = c(0.7, 0.3), sbj = 100, fu_time_ia = 2,
  n_event = 162, cutoff = 0, alpha1 = 0.0125, alpha2 = 0.0125,
  par_ctrl_pos = list(orr = 0.2, pfs_shape = 1, pfs_median = 7, corr =
  0), par_ctrl_neg = list(orr = 0.1, pfs_shape = 1, pfs_median = 6, corr
  = 0), par_trt_pos = list(orr = 0.1, pfs_shape = 1, pfs_median = 6, corr
  = 0), par_trt_neg = list(orr = 0.3, pfs_shape = 1, pfs_median = 10,
  corr = 0))

cong_simu_trial_parallel(nsim = 10, ncores = 4, ...)
```

Arguments

n_allcomer	number of subjects for all comers
n_enrichment	increased sample size for enrichment group
alloc	allocation vector, length corresponds to number of arms; as to be integer; enter 1 if single arm
b_size	block size, has to be multiple of sum(alloc), enter 1 if single arm
rate	enrollment rate per unit time
marker_positive, marker_negative	a string specifying which marker is negative/positive
marker_prob	vector of prevalence probability of different category, if doesn’t add up to 1, will automatically standardize and generates warning

sbj	number of subjects for analysis of enrichment decision
n_event	the desired number of events for final analysis. Note that this parameter is used to decide time cutoff for final analysis; therefore n_event should be only counted among all-comer populations to proect the integraty of the trial.
cutoff	the cutoff value to determine if enrichment is needed or not
alpha1, alpha2	significance level for testing all-comers or biomarker positive population, respectively
par_trt_pos, par_trt_neg, par_ctrl_pos, par_ctrl_neg	parameter specification for treatment/control and biomarker positive/negative population
marker_name	vector of names of different subgroup
ia_time_fu	the follow-up time for decision of enrichment analysis

---

gen_ospfs *generate correlated endpoints of PFS and OS*

Description

generate correlated endpoints of PFS and OS

Usage

```
gen_ospfs(nsbj, mos, mosp = NA, mpfs)
```

Arguments

nsbj	number of subjects to be simulated
mos	median overall survival time
mosp	median overall survival since progression. If mosp = NA, then OS and PFS will be generated using Theorem 1. If a numerical value is assigned to mosp, then OS and PFS will be generated using Theorem 2.
mpfs	median PFS

Details

This function generates correlated PFS and OS based on a paper published in 2009. Specifically, Let $X_1 \sim \exp(\lambda_1), X_2 \sim \exp(\lambda_2), X_3 \sim \exp(\lambda_3)$, where X_i has pdf $f_X(x_i) = \lambda_i \exp(-\lambda_i x_i)$.

- Theorem 1. If $PFS = \min(X_1, X_2), OS = X_2$ then

$$Corr(PFS, OS) = \frac{\lambda_2}{\lambda_1 + \lambda_2}$$

- Theorem 2. If we define $PFS = \min(X_1, X_2)$, $OS = PFS$ if $PFS = X_2$ and $OS = X_1 + X_3$ otherwise, then

$$Corr(PFS, OS) = \frac{\lambda_3}{\sqrt{\lambda_1^2 + 2\lambda_1\lambda_2 + \lambda_3^2}}$$

Value

a tibble with correlated PFS and OS

References

Fleischer F, Gaschler-Markefski B, Bluhmki E (2009). "A statistical model for the dependence between progression-free survival and overall survival." *Statistics in Medicine*, **28**(21), 2669–2686.

Examples

```
mos <- 6.7; mosp  <- 5.4; mpfs <- 1.93
y1 <- gen_ospfs(nsbj = 1e5, mos = mos, mosp = mosp, mpfs = mpfs)
cor(y1)
y2 <- gen_ospfs(nsbj = 1e5, mos, mosp = NA, mpfs)
all.equal(cor(y2)[1,2],mpfs/mos, tolerance = 1e-3)
```

---

ia_pval   *Decide interim analysis cutoff value for group sequential design*

Description

Decide interim analysis cutoff value for group sequential design

Usage

```
ia_pval(info_fraction, alpha = 0.025, beta = 0.1, ...)
```

Arguments

`info_fraction`	the information fraction for each look, default set to be 1, meaning no interim analysis. Can take vector
`alpha`	the desired type 1 error
`beta`	the type 2 error, equal to `1 - power`
`...`	other parameters inherited from `[gsDesign]gsDesign`

Value

a tibble containing number of looks and efficacy/futility p values

Examples

```
ia_pval(alpha = 0.025, beta = 0.1, info_fraction = c(0.5, 0.7))
```

`pos_two_grid` *Posterior probabilities for given sample size of n0 and n1*

Description

for a given pair of `n0` and `n1`, search all the possible sample space, calculates posterior probability, the probailities under the null and the alternative. This function is used to evaluate the power and type 1 error for a given pair of `n0` and `n1`

Usage

```
pos_two_grid(n0, n1, p0 = 0.25, p1 = 0.538, delta = (p1 - p0)/2,
  ab0 = NULL, ab1 = NULL)
```

Arguments

`n0, n1`	the sample size for control/treatment group
`p0`	the underlying probability of response rate for the control arm
`p1`	the hypothesized ORR for treatment
`delta`	the difference of the two proportions to be detected

Details

For a given pair of `n0` and `n1`, after specifying appropriate prior parameters, it calculates the posterior probability $P(p_1 - p_0 > \delta)$, the probability $P(X_0 = x_0|n_0, p_0) \cdot P(X_1 = x_1|n_1, p_0)$ under the null, and $P(x = x_0|n_0, p_0) \cdot P(X = x_1|n_1, p_1)$ under the alternative for each pair of of observed x_0 and x_1. Note that X_0, X_1 are assumed to be independent and follow binomial distribution.

The prior for control group, i.e. $p_0 \sim Beta(a_0, b_0)$, are derived based on $a/(a + b) = p_0$ and $a + b = n_0/2$ where n_0 is the sample size for control arm. The prior for treatment group is obtained such that $a_1 + b_1 = 2$ and $a_1 = 2p_0$.

Value

a tibble which contains the calculated probabilities

See Also

`pos_two`

Examples

```
library(dplyr)
r1 <- pos_two_grid(n0 = 75, n1 = 75, p0 = 0.59, p1= 0.812)
# power
r1 %>% filter(prob_post > 0.68) %>% select(prob_alt) %>% sum
# type 1 error
r1 %>% filter(prob_post > 0.68) %>% select(prob_null) %>% sum
```

pos_two_grid_search *Power and type 1 error calculation by grid search*

Description

for a given pair of n0 and n1, search all the possible sample space, calculates posterior probability, the probailities under the null and the alternative. This function is used to evaluate the power and type 1 error for a given pair of n0 and n1

Usage

```
pos_two_grid_search(p0, p1, ..., n0 = seq(50, 70, by = 5), n1 = n0,
  cutoff = seq(0.05, 0.2, by = 0.05), eval_success = TRUE,
  ncores = NA, ab0 = NULL)
```

Arguments

p0	response rate in the control arm
p1	the hypothesized ORR for treatment
...	other parameters inherited from pos_two_grid
n0	sample size for control, can be a vector
n1	sample size for treatment, must be of the same length as n0
cutoff	the cutoff value (can be a vector) to claim a decision (either success or failure)
eval_success	Is this for evaluating probability of success? If TRUE, then it evaluates $$P(p_1 - p_0 > \delta) > U;$$ otherwise it evaluates $$P(p_1 - p_0 > \delta) < L,$$ where L or U correspond to cutoff.
ncores	number of cores to be used for fast parallel computing. If not specified, it will use number of cores available - 1
ab0	a data frame or NULL. If a data frame, it should contain, in each row, the prior for corresponding sample size n0. If NULL, then prior_ab will be called internally to calculate the prior.

Details

For a given pair of n0 and n1, after specifying appropriate prior parameters, it calculates the posterior probability $P(p_1 - p_0 > \delta)$, the probability $P(X_0 = x_0|n_0, p_0) \cdot P(X_1 = x_1|n_1, p_0)$ under the null, and $P(x = x_0|n_0, p_0) \cdot P(X = x_1|n_1, p_1)$ under the alternative for each pair of of observed x_0 and x_1. Note that X_0, X_1 are assumed to be independent and follow binomial distribution.

The prior for control group, i.e. $p_0 \sim Beta(a_0, b_0)$, are derived based on $a/(a + b) = p_0$ and $a + b = n_0/2$ where n_0 is the sample size for control arm. The prior for treatment group is obtained such that $a_1 + b_1 = 2$ and $a_1 = 2p_0$.

Value

a tibble containing each scenario associated with its power and type 1 error

See Also

pos_two_grid

Examples

```
temp1 <- pos_two_grid_search(p0 = 0.59, p1 = 0.812,
                n0 = seq(50, 80, by =5),
                cutoff = seq(0.5, 0.8, 0.02),
                ncores = NA,
                eval_success = TRUE)
```

---

prior_ab — *calculate prior parameters for a given beta distribution*

Description

calculate prior parameters for a given beta distribution

Usage

```
prior_ab(n, p)
```

Arguments

n	the size of the prior beta distribution $a + b = n/2$
p	the prior mean $\frac{a}{a+b} = p$

Value

the parameters a and b for $Beta(a, b)$

---

rand_arm — *generate block randomized arms*

Description

generate block randomized arms

Usage

```
rand_arm(nsbj, ratio, arm_name = paste("arm", 1:length(ratio), sep =
  "_"))
```

Arguments

nsbj	an integer for total number of subjects to be randominzed
ratio	the allocation ratio
arm_name	a vector of characters for arms

Value

a vector of length ‘nsbj‘ with randomized treatment arms

Examples

```
rand_arm(nsbj = 1, ratio = c(1, 1))
rand_arm(nsbj = 12, ratio = c(2, 2, 1))
rand_arm(nsbj = 4, ratio = c(1, 2, 0, 1))
```

`rand_timein` *Generate enrollment time by piecewise enrollment rate*

Description

Generate enrollment time by piecewise enrollment rate

Usage

```
rand_timein(nsbj, rate, starttime)
```

Arguments

`nsbj`	number of subject enrolled
`rate`	a vector (or a single value) specifying the enrollment rate at each piece
`starttime`	a vector (or a single value) specifying starting time for corresponding enrollment rate. `starttime` always starts with 0, whether it's a vector or a single value.

Value

a tibble where the first column is enrollment time, and the second column indicates the piece sequence

Examples

```
rate <- c(7, 14, 30)
starttime <- c(0, 1, 3)
timein1 <- rand_timein(nsbj = 300, rate = rate, starttime = starttime)
```

snapshot_by_event *data snapshot by desired event size*

Description

this function calculates the time cut for desired event size and then the censor indicator. It has been verified against EAST software.

Usage

```
snapshot_by_event(dat, n_event)
```

Arguments

dat	the data frame containing, at least, the following variables • `timein` patient arrival time • `pfs` progression or surivival time • `lfu` lost to follow up time or dropout time
n_event	desired number of events for analysis

Value

the same data with extra columns `timecut` (the calander time cut), `pfs_censor` (the censoring indicator, with 1 = event and 0 = censor), `ongoing` (whether the status is still ongoing by timecut).

survival_test *Run survival analysis*

Description

Run survival analysis

Usage

```
survival_test(snapshot, pval_eff = 0.025, pval_fu = NA, is_trt = NA)
```

Arguments

snapshot	the data set obtained from `take_snapshot`
pval_eff, pval_fu	the significance level to claim a success/futility for interim analysis or success/failure for final: set `pval_eff = NA` and assign `pval_fu` a positive value between 0 and 1 if it's just for interim futility; set `pval_fu = NA` and assign `pval_eff` a positive value between 0 and 1 if it's just for interim efficacy; if it's interim analysis for both efficacy and futility, then must have `pval_eff` < `pval_fu`; if it's for final analysis, then `pval_eff` and `pval_fu` must both be specified and set to be equal;
is_trt	user-defined treatment group. If `is_trt = NA` then the second arm number shown in data will be the treatment arm.

Details

this function takes the snapshot and runs survival analysis to get the log rank test p-value, the 95 survival time.

Value

a data frame containing the results of the test

`tail_prob` *Calculate probability of success or failure*

Description

Calculate probability of success or failure

Usage

```
tail_prob(dat, cutoff, prob1, prob2, eval_success = TRUE)
```

Arguments

`dat`	the object returned by `pos_two_grid`
`cutoff`	the cutoff value to claim a success/failure
`prob1`	the posterior probability
`prob2`	the probability under the null or the alternative
`eval_success`	Is this for evaluating probability of success?

Value

a p value

`test_bm_neg` *Enrichment decision making*

Description

Enrichment decision making

Usage

```
test_bm_neg(cong_dat, marker_negative = "DLL3-", endpoint = "resp",
  sbj = 100, fu_time_ia = 2, cutoff = 0)
```

Arguments

`cong_dat`	a data set generated by `cong_dat_gen`
`marker_negative`	a string specifying which marker is negative
`endpoint`	the endpoint used to calculate the decision rule
`sbj`	number of subjects to be included in analysis of biomarker negative
`fu_time_ia`	minimum follow-up time for eligible evaluation
`cutoff`	the cutoff chosen to make the decision

Details

This function performs analysis for the biomarker negative population, then decides if enrichment is needed

Value

a tibble with decision included (see column `need_enrichment`)

Index