Package: clinfun (via r-universe)

October 28, 2026

Title Clinical Trial Design and Data Analysis Functions

Version 1.1.6

Depends R (>= 3.0.0), graphics, stats

Imports mvtnorm

Suggests knitr, rmarkdown, survival

Author Venkatraman E. Seshan [aut, cre], Karissa Whiting [aut]

Description Utilities to make your clinical collaborations easier if not fun. It contains functions for designing studies such as Simon 2-stage and group sequential designs and for data analysis such as Jonckheere-Terpstra test and estimating survival quantiles.

Maintainer Venkatraman E. Seshan <seshanv@mskcc.org>

License GPL (>= 2)

NeedsCompilation yes

VignetteBuilder knitr

Repository https://cran.r-universe.dev

Date/Publication 2026-08-29 12:30:12 UTC

RemoteUrl https://github.com/cran/clinfun

RemoteRef HEAD

RemoteSha b6d6b5b8ac0d42b5e6541b9576b2750b1ea7b659

Contents

aucVardiTest 2
calogrank 3
coxphCPE 4
coxphERR 5
coxphQuantile 6
deltaAUC 7
fedesign 8
gsdesign 10

jonckheere.test 11
ktau 12
oc.twostage.bdry 13
permlogrank 14
ph2simon 14
ph2single 16
power.ladesign 16
pselect 17
roc.area.test 19
roc.curve 20
roc.perm.test 21
ROCanalysis 23
toxbdry 23
twostage.admissible 25
twostage.inference 26

Index **27**

`aucVardiTest` *Two-Sample Tests for Growth Curves under Dependent Right Censoring*

Description

Permutation test for comparing growth curves across tow groups under dependent right censoring.

Usage

```
aucVardiTest(meas, grp, tim=NULL, cgrps=NULL, nperm=5000)
```

Arguments

`meas`	Matrix of measurements where the rows are the subjects and columns the time-points. At least one value should not be missing in each row. For example they can be tumor sizes measured over time.
`grp`	Group indicator for each subject. There must be at least two different groups. This can represent each subject's treatment.
`tim`	Times at which the measurements in `meas` are taken. If missing, the times are set to 1 through `ncol(meas)`.
`cgrps`	The two groups that are being compared. If missing the first two groups will be compared.
`nperm`	Number of permutations for the reference distribution.

Details

The test statistic is defined as the sum of pairwise differences in the partial areas under the growth curve. For each pair of subjects the partial area is computed until the smaller of the maximum followup times. For each subject, linear interpolation is is used to fill-in missing values prior to the maximum followup time. The reference distribution of obtained by permuting the group labels.

Value

returns a list with objects ostat, pstat and p.value which are the observed test statistic for the two groups being compared, values of the statistics when the group labels are permuted

References

Vardi Y., Ying Z. and Zhang C.H. (2001). Two-Sample Tests for Growth Curves under Dependent Right Censoring. *Biometrika* 88, 949-960.

Examples

```
grp <- sample(1:3, 100, replace=TRUE)
grp0 <- LETTERS[grp]
maxfup <- sample(5:20, 100, replace=TRUE)
meas <- matrix(NA, 100, 20)
for(i in 1:100) {
  meas[i, 1:maxfup[i]] <- cumsum((3+0.04*grp[i]) + rnorm(maxfup[i]))
}
aucVardiTest(meas, grp)
aucVardiTest(meas, grp0, cgrps=c("C","B"))
```

---

calogrank *Survival curves analysis of covariance*

Description

Logrank test to compare survival curves adjusting for covariates

Usage

```
calogrank(ftime, fstatus, grp, cvt, strat=NULL)
```

Arguments

ftime	failure times
fstatus	status indicator
grp	group indicator
cvt	continuous covariates used for adjusted analysis
strat	stratification variable

Details

calograrank is the covariate adjusted version of k-sample survdiff. The function in its current form only does basic error checking.

References

Heller G. and Venkatraman E.S. (2004) A nonparametric test to compare survival distributions with covariate adjustment. *JRSS-B* 66, 719-733.

Examples

```
## Not run:   library(survival)
  data(pbc)
  pbc1 <- pbc
  pbc1$trt[pbc1$trt == -9] <- NA
  pbc1$copper[pbc1$copper == -9] <- NA
  # only death (2) is considered; transplant(1) is censored
  calograrank(pbc1$time, pbc1$status==2, pbc1$trt, pbc1[,c("copper")])
  calograrank(pbc1$time, pbc1$status==2, pbc1$trt,
                                  pbc1[,c("protime", "copper")])
## End(Not run)
```

---

coxphCPE *Concordance Probability Estimate for Cox model*

Description

Calculates the Concordance Probability Estimate for a Cox proportional hazards model. Both the Gonen and Heller (2005) version for continuous risk score and Heller and Mo (2016) for discrete risk score can be calculated.

Usage

```
coxphCPE(phfit, out.ties=FALSE)
```

Arguments

phfit	output from a proportional hazards fit.
out.ties	binary flag to decide if pairs with tied risk scores should be used.

Value

coxphCPE returns a vector with CPE, smooth.CPE and se.CPE which are the estimate, the smoothed estimate and its standard error respectively.

References

Gonen M and Heller G. (2005) Concordance probability and discriminatory power in proportional hazards regression. *Biometrika* 92, 965-970.

Heller G and Mo Q. (2016). Estimating the concordance probability in a survival analysis with a discrete number of risk groups. *Lifetime Data Analysis*, 22,263-79.

Examples

```
## Not run:  library(survival)
  data(pbc)
  pbcfit <- coxph(Surv(time, status==2) ~ trt + log(copper), pbc,
    subset=(trt>0 & copper>0))
  coxphCPE(pbcfit)

## End(Not run)
```

---

coxphERR *Heller Explained Relative Risk*

Description

Calculates the contribution of a subset of covariates to the explained relative risk derived from the full Cox proportional hazards model.

Usage

```
coxphERR(phfit, ngamma=NULL)
```

Arguments

phfit	The output from a proportional hazards fit.
ngamma	A vector of indices corresponding to covariates of interest. If missing (default), the explained relative risk is computed for the full model.

Details

The object phfit should be the result of a call to coxph with the option x=TRUE.

Value

The function coxphERR returns the vector (ERR, se.ERR). The first component ERR represents the contribution of a subset of covariates to the explained relative risk estimate of the full model. If a set of covariates is not provided, then it computes the estimate of the full model. The second component se.ERR is the standard error of the estimate.

References

Heller G. (2012) A measure of explained risk in the proportional hazards model. *Biostatistics*

Examples

```
## Not run:
 library(survival)
 ovarianfit <- coxph(Surv(futime, fustat) ~ age + resid.ds + rx +
                                          ecog.ps, data=ovarian,x=T)
 # Compute the explained relative risk (ERR) and
 # its standard error (se.ERR) for the full model.
 coxphERR(ovarianfit)
 # Compute the contribution of age and ECOG performance status to
 # the explained relative risk. Age and ECOG performance status are
 # the first and fourth covariates in the model.
 coxphERR(ovarianfit, c(1,4))

## End(Not run)
```

---

coxphQuantile *Survival time quantile as a function of covariate*

Description

Given a specified quantile level p, draws a curve between survival time and specified covariate. The curve represents the p-th quantile of survival as a function of covariate.

Usage

```
coxphQuantile(phfit, xrange, p=0.5, whichx=1, otherx=NULL, ...)
```

Arguments

phfit	a coxph object.
xrange	the range of covariate values for which the quantiles of survival times are computed.
p	the probability level for the quantile (default is median).
whichx	if there are more than one covariates in the Cox model, the one chosen for the quantile plot.
otherx	the values for other covariates in the Cox model. If missing uses their average values.
...	additional parameters to be passed on to the lines command.

Details

This function is used to draw quantile curves. It requires a plot of the data (time & covariate of interest) to be present. See example.

It invisibly returns the observed failure times and the covariate values at which the estimated survival probability is (exactly) p.

References

Heller G. and Simonoff J.S. (1992) Prediction in censored survival data: A comparison of the proportional hazards and linear regression models. *Biometrics* 48, 101-115.

Examples

```
## Not run:   library(survival)
data(pbc)
pbcfit <- coxph(Surv(time, status==2) ~ trt + log(copper), pbc,
                      subset=(trt>0 & copper>0))
plot(log(pbc$copper[pbc$trt>0 & pbc$copper>0]), pbc$time[pbc$trt>0 &
  pbc$copper>0], pch=c("o","x")[1+pbc$status[pbc$trt>0 & pbc$copper>0]],
  xlab="log Copper", ylab="Survival time")
coxphQuantile(pbcfit, c(2.5,6), whichx=2, otherx=1)
coxphQuantile(pbcfit, c(2.5,6), p=0.75, whichx=2, otherx=2, col=2)
## End(Not run)
```

---

deltaAUC *Comparing the AUC from ROC curves from nested binary regression*

Description

Testing the differences in the area under the ROC curves (AUC) from nested maximum rank correlation models.

Usage

```
deltaAUC(y, x, z)
```

Arguments

y	binary response variable
x	matrix of set of covariates that is the basis of the existing (reduced) model
z	matrix of set of covariates that are added to to get the new (full) model

Details

The models are fit using maximum rank correlation (MRC) method which is an alternate approach to logistic regression. In MRC the area under the ROC curve (AUC) is maximized as opposed to the likelihood in logistic regression. Due to invariance of AUC to location and scale shifts one of the parameters (anchor variable) is set to 1.

The first variable (column) in x is used as the anchor variable.

The IPMN data set used as an example in the paper below is included. The columns are high risk lesion (V1), recent weight loss (V2), main duct involvement (V4), presence of a solid component in imaging (V3), and lesion size (V5).

Value

It returns a list with the following elements

par.full	the MRC estimate of parameters for the full model
par.red	the MRC estimate of parameters for the reduced model
results	matrix od results which gives the full reduced model AUCs along with the test statistic and p-value

References

Heller G., Seshan V.E., Moskowitz C.S. and Gonen M. (2016) Inference for the difference in the area under the ROC curve derived from nested binary regression models. *Biostatistics* 18, 260-274.

Examples

```
data(ipmn)
deltaAUC(ipmn$V1, cbind(ipmn$V4, ipmn$V3, ipmn$V5), ipmn$V2)
```

---

fedesign *Trial Designs Based On Fisher's Exact Test*

Description

Calculates sample size, effect size and power based on Fisher's exact test

Usage

```
fe.ssize(p1, p2, alpha=0.05, power=0.8, r=1, npm=5, mmax=1000)
CPS.ssize(p1, p2, alpha=0.05, power=0.8, r=1)
fe.mdor(ncase, ncontrol, pcontrol, alpha=0.05, power=0.8)
mdrr(n, cprob, presp, alpha=0.05, power=0.8, niter=15)
fe.power(d, n1, n2, p1, alpha = 0.05)
or2pcase(pcontrol, OR)
```

Arguments

p1	response rate of standard treatment
p2	response rate of experimental treatment
d	difference = p2-p1
pcontrol	control group probability
n1	sample size for the standard treatment group
n2	sample size for the standard treatment group
ncontrol	control group sample size
ncase	case group sample size

alpha	size of the test (default 5%)
power	power of the test (default 80%)
r	treatments are randomized in 1:r ratio (default r=1)
npm	the sample size program searches for sample sizes in a range (+/- npm) to get the exact power
mmax	the maximum group size for which exact power is calculated
n	total number of subjects
cprob	proportion of patients who are marger positive
presp	probability of response in all subjects
niter	number of iterations in binary search
OR	odds-ratio

Details

CPS.ssize returns Casagrande, Pike, Smith sample size which is a very close to the exact. Use this for small differences p2-p1 (hence large sample sizes) to get the result instantaneously.

Since Fisher's exact test orders the tables by their probability the test is naturally two-sided.

fe.ssize return a 2x3 matrix with CPS and Fisher's exact sample sizes with power.

fe.mdor return a 3x2 matrix with Schlesselman, CPS and Fisher's exact minimum detectable odds ratios and the corresponding power.

fe.power returns a Kx2 matrix with probabilities (p2) and exact power.

mdrr computes the minimum detectable P(resp|marker+) and P(resp|marker-) configurations when total sample size (n), P(response) (presp) and proportion of subjects who are marker positive (cprob) are specified.

or2pcase give the probability of disease among the cases for a given probability of disease in controls (pcontrol) and odds-ratio (OR).

References

Casagrande, JT., Pike, MC. and Smith PG. (1978). An improved approximate formula for calculating sample sizes for comparing two binomial distributions. *Biometrics* 34, 483-486.

Fleiss, J. (1981) Statistical Methods for Rates and Proportions.

Schlesselman, J. (1987) Re: Smallest Detectable Relative Risk With Multiple Controls Per Case. *Am. J. Epi.*

gsdesign *Group Sequential Designs*

Description

Functions to calculate sample size for group sequential designs

Usage

```
gsdesign.binomial(ifrac, pC, pE, r = 1, sig.level = 0.05, power = 0.8,
  delta.eb=0.5, delta.fb = NULL, alternative = c("two.sided",
  "one.sided"), pooled.variance = FALSE, CPS = TRUE, tol=0.00001, ...)
gsdesign.normal(ifrac, delta, sd = 1, r = 1, sig.level = 0.05,
  power = 0.8, delta.eb = 0.5, delta.fb = NULL, alternative =
  c("two.sided", "one.sided"), tol=0.00001, ...)
gsdesign.survival(ifrac, haz.ratio, r = 1, sig.level = 0.05,
  power = 0.8, delta.eb = 0.5, delta.fb = NULL, alternative =
  c("two.sided", "one.sided"), tol=0.00001, ...)
```

Arguments

ifrac	information fraction or the ratio of current sample size or number of events to the total sample size or number of events. This should be an increasing vector of numbers from 0 to 1 with the last one being 1. If just 1 is given a fixed sample design is derived.
pC	prob of success of the standard therapy (for binomial data)
pE	prob of success of the experimental therapy (for binomial data)
delta	true difference in means (for normal data)
sd	standard deviation (for normal data)
haz.ratio	hazard ratio (for survival comparison)
r	treatment allocation of r (default=1) experimental per 1 control.
sig.level	significance level (type I error probability)
power	power of test (1 minus type II error probability)
delta.eb	power for efficacy boundary in the Pocock (=0) to O'Brien-Fleming (=0.5) family (default is 0.5)
delta.fb	power for futility boundary in the Pocock (=0) to O'Brien-Fleming (=0.5) family (default is NULL i.e. no futility boundary is requested.)
alternative	one- or two-sided test.
pooled.variance	whether the test statistic is standardized by pooled (2*pbar*(1-pbar)) or unpooled variance (pC*(1-pC) + pE*(1-pE)). Default is unpooled variance.
CPS	whether continuity correction is used for sample size calculation as in Casagrande, Pike & Smith. Default is to use it.
tol	tolerance level for multivariate normal probability computation.
...	additional options passed on the pmvnorm function.

Details

The futility boundary is not returned when delta.fb is not specified i.e. stopping for futility is not requested. The futility boundary is non-binding. That is the significance level is not adjusted to account for early stopping for futility. This makes the test a bit conservative in that the true size is less than the nominal level.

If the alternative is two-sided by default the futility boundary will also be two-sided i.e. continuation region is wedge shaped. However, if the goal is to show the superiority of the experimental treatment then futility boundary should be one sided. This can be achieved by deriving the boundaries for one-sided alternative and significance level set at half of the value used for two sided alternative. See the examples section for a representative design for which the trial cannot be stopped at the first look for futility.

The Casagrande-Pike-Smith type continuity correction is obtained using the formula

$$n * [1 + \sqrt{1 + 4/(|pC - pE| * n)}]^2$$

where n is the uncorrected sample size.

Value

a list with ifrac, sig.level, power, alternative, delta.eb, delta.fb and:

`efbdry`	the critical value to use at the different looks. For two-sided alternative the absolute test statistic should exceed this.
`futbdry`	the critical value to use at the different looks. For two-sided alternative the absolute test statistic should be below this.
`sample.size`	the sample size per arm for binomial/normal data.
`num.events`	the total number of failures which should be converted to number of subjects using censoring proportion.

Examples

```
gsdesign.normal(1:4/4, 0.25, sig.level=0.05, alt="t", delta.fb=0.5)
gsdesign.normal(1:4/4, 0.25, sig.level=0.025, alt="o", delta.fb=0.5)
```

---

jonckheere.test *Exact/permutation version of Jonckheere-Terpstra test*

Description

Jonckheere-Terpstra test to test for ordered differences among classes

Usage

```
jonckheere.test(x, g, alternative = c("two.sided", "increasing",
                "decreasing"), nperm=NULL)
```

Arguments

x, g	data and group vector
alternative	means are monotonic (two.sided), increasing, or decreasing
nperm	number of permutations for the reference distribution. The default is null in which case the permutation p-value is not computed. Recommend that the user set nperm to be 1000 or higher if permutation p-value is desired.

Details

jonckheere.test is the exact (permutation) version of the Jonckheere-Terpstra test. It uses the statistic

$$\sum_{k<l}\sum_{ij} I(X_{ik} < X_{jl}) + 0.5I(X_{ik} = X_{jl}),$$

where i, j are observations in groups k and l respectively. The asymptotic version is equivalent to cor.test(x, g, method="k"). The exact calculation requires that there be no ties and that the sample size is less than 100. When data are tied and sample size is at most 100 permutation p-value is returned.

References

Jonckheere, A. R. (1954). A distribution-free k-sample test again ordered alternatives. *Biometrika* 41:133-145.

Terpstra, T. J. (1952). The asymptotic normality and consistency of Kendall's test against trend, when ties are present in one ranking. *Indagationes Mathematicae* 14:327-333.

Examples

```
set.seed(1234)
g <- rep(1:5, rep(10,5))
x <- rnorm(50)
jonckheere.test(x+0.3*g, g)
x[1:2] <- mean(x[1:2]) # tied data
jonckheere.test(x+0.3*g, g)
jonckheere.test(x+0.3*g, g, nperm=5000)
```

---

ktau *Kendall's tau-b estimate*

Description

Calculates the Kendall's tau-b.

Usage

```
ktau(x, y)
```

Arguments

x	first variable
y	second variable

Details

ktau computes the same quantity as cor(x, y, method="kendall"). It uses a faster algorithm than pairwise comparisons used by cor.

Value

ktau returns Kendall's tau-b.

Examples

```
set.seed(1234)
x <- rnorm(10000); y <- x+rnorm(10000)
cor(x, y, method="k")
clinfun:::ktau(x,y)
```

---

oc.twostage.bdry *Two-stage boundary operating characteristics*

Description

Calculates the operating characteristics of a two-stage boundary.

Usage

```
oc.twostage.bdry(pu, pa, r1, n1, r, n)
```

Arguments

pu	unacceptable response rate
pa	response rate that is desirable
r1	first stage threshold to declare treatment undesirable
n1	first stage sample size
r	overall threshold to declare treatment undesirable
n	total sample size

Value

oc.twostage.bdry returns the type I and II error rates as well as the probability of early temination and expected sample size under pu for a specific boundary.

permlogrank *Permutation version of survdiff*

Description

Small sample survdiff using permutation reference distributions.

Usage

```
permlogrank(formula, data, subset, na.action, rho=0, nperm=5000)
```

Arguments

nperm	number of permutations for the reference distribution
formula, data, subset, na.action, rho	see survdiff for details

Details

permlogrank is the permutation version of k-sample survdiff. see survdiff in survival package for details.

References

Heller G, Venkatraman ES. (1996). Resampling procedures to compare two survival distributions in the presence of right censored data. *Biometrics* 52:1204-1213.

ph2simon *Simon's 2-stage Phase II design*

Description

Calculates Optimal and Minimax 2-stage Phase II designs given by Richard Simon

Usage

```
ph2simon(pu, pa, ep1, ep2, nmax=100)
## S3 method for class 'ph2simon'
print(x, ...)
## S3 method for class 'ph2simon'
plot(x, ...)
```

Arguments

pu	unacceptable response rate; baseline response rate that needs to be exceeded for treatment to be deemed promising
pa	response rate that is desirable; should be larger than pu
ep1	threshold for the probability of declaring drug desirable under pu (target type 1 error rate); between 0 and 1
ep2	threshold for the probability of rejecting the drug under pa (target type 2 error rate); between 0 and 1
nmax	maximum total sample size (default 100; can be at most 1000)
x	object returned by ph2simon
...	arguments to be passed onto plot and print commands called within

Value

ph2simon returns a list with pu, pa, alpha, beta and nmax as above and:

out — matrix of best 2 stage designs for each value of total sample size n. The 6 columns in the matrix are:

r1	number of responses needed to exceeded in first stage
n1	number of subjects treated in first stage
r	number of responses needed to exceeded at the end of trial
n	total number of subjects to be treated in the trial
EN(pu)	expected number pf patients in the trial under pu
PET(pu)	probability of stopping after the first stage under pu

Trial is stopped early if <= r1 responses are seen in the first stage and treatment is considered desirable only when >r responses seen.

Methods (by generic)

- `print(ph2simon)`: formats and returns the minimax, optimal and any admissible designs.
- `plot(ph2simon)`: plots the expected sample size against the maximum sample size as in Jung et al., 2001

References

Simon R. (1989). Optimal Two-Stage Designs for Phase II Clinical Trials. *Controlled Clinical Trials* 10, 1-10.

Jung SH, Carey M and Kim KM. (2001). Graphical Search for Two-Stage Designs for Phase II Clinical Trials. *Controlled Clinical Trials* 22, 367-372.

Jung SH, Lee T, Kim K, and George, SL. (2004). Admissible two-stage designs for phase II cancer clinical trials. *Statistics in medicine* 23(4), 561-569.

See Also

`twostage.inference`, `oc.twostage.bdry`

Examples

```
ph2simon(0.2, 0.4, 0.1, 0.1)
ph2simon(0.2, 0.35, 0.05, 0.05)
ph2simon(0.2, 0.35, 0.05, 0.05, nmax=150)
```

`ph2single` *Exact single stage Phase II design*

Description

Calculates the exact one stage Phase II design

Usage

```
ph2single(pu,pa,ep1,ep2,nsoln=5)
```

Arguments

`pu`	unacceptable response rate
`pa`	response rate that is desirable
`ep1`	threshold for the probability of declaring drug desirable under p0
`ep2`	threshold for the probability of rejecting the drug under p1
`nsoln`	number of designs with given alpha and beta

Value

ph2single returns a data frame with variables: n, r, and the Type I and Type II errors. Treatment desirable if >r responses seen.

`power.ladesign` *Power of k-sample rank test under Lehmann alternative*

Description

Functions to calculate the power of rank tests for animal studies.

Usage

```
  power.ladesign(gsize, odds.ratio, sig.level = 0.05, statistic =
     c("Kruskal-Wallis", "Jonckheere-Terpstra"), alternative =
     c("two.sided", "one.sided"), nrep=1e+6)
  ## S3 method for class 'ladesign'
print(x,...)
```

Arguments

gsize	sample size of the k (= length of vector) groups.
odds.ratio	odds ratio parameters for the k-1 groups. The first group is considered the control.
sig.level	the significance level of the test (default = 0.05)
statistic	the test statistic for the k-group comparison. Is one of Kruskal-Wallis (default) or Jonckeere-Terpstra.
alternative	one- or two-sided test. Valid only for the Jonckheere-Terpstra test.
nrep	number of reps (default 1 million) for Monte Carlo.
x	object of class ladesign returned by power.ladesign
...	arguments to be passed on left for S3 method consistency.

Details

Although the power for Jonckheere-Terpstra test is calculated for any set of odds ratio, the test is meant for monotone alternative. Thus it is preferable to specify odds ratios that are monotonically increasing with all values larger than 1 or decreasing with all values smaller than 1.

Value

returns a list with objects group.size, odds.ratio, statistic, sig.level and power. The "print" method formats the output.

References

Heller G. (2006). Power calculations for preclinical studies using a K-sample rank test and the Lehmann alternative hypothesis. *Statistics in Medicine* 25, 2543-2553.

Examples

```
power.ladesign(c(9,7), 4, statistic="K")
power.ladesign(c(9,7,9), c(2,4), statistic="J")
power.ladesign(c(9,7,9), c(2,4), statistic="J", alt="o")
```

---

pselect *Probability of selection under pick the winner rule*

Description

Calculates the probability of selecting the treatment with the higher response rate under the pick the winner rule.

Usage

```
pselect(n, p, min.diff=NULL, min.resp=NULL)
```

Arguments

n	sample size for each treatment arm. This is either a single integer or a vector of two integers for the special case of comparing two treatments with unequal sample sizes
p	vector of response rates for the treatments.
min.diff	this is the number of responses or the rate by which the best treatment should be superior to the others to be chosen. This must be a positive integer or a rate between 0 and 1. If missing it defaults to 1 for the equal sample size case but quits with a warning for the unequal sample size case.
min.resp	the minimum number of responses in each treatment arm for it to be considered further. If missing defaults to 0.

Value

the function returns a list with:

prob.none.worthy	is the probability that no treatment has the minimum number of responses specified in min.resp. this element is present only if min.resp is greater than 0 for at least one arm.
prob.inconclusive	this is the probability that the best treatment has the requisite min.resp responses but exceeds the second best by less than min.diff responses (rate) provided the second best also has at least min.resp responses.
prob.selection	this is a matrix which for each treatment gives the response probability and the probability of selecting it i.e. the number of responses in the chosen arm is at least min.resp and either none of the remaining arms exceed the min.resp threshold or the chosen (best) arm is better than the second best by at least min.diff responses (rate).

References

Simon R, Wittes RE, Ellenberg SS. (1985). Randomized phase II clinical trials. *Cancer Treat Rep* 69, 1375-1381.

Examples

```
# selection when no diffrence i.e. type I error
pselect(18, c(0.2, 0.2, 0.2))
# selection probability
pselect(18, c(0.2, 0.2, 0.4))
pselect(26, c(0.2, 0.2, 0.4), min.diff=2, min.resp=3)
# unequal sample size case
pselect(c(27,54), c(0.5, 0.65), min.diff=0.05)
# unequal sample size case
pselect(c(27,54), c(0.5, 0.65), min.diff=0.05, min.resp=c(14,27))
```

`roc.area.test` *Nonparametric area under the ROC curve*

Description

Computes the nonparametric area under the ROC curve and its variance based on U-statistic theory (DeLong et al, 1988).

Usage

```
  roc.area.test(markers, status)
  ## S3 method for class 'roc.area.test'
print(x, ...)
```

Arguments

`markers`	The marker values for each subject. If there are more than one markers then this should be a matrix.
`status`	binary disease status indicator
`x`	object of class roc.area.test output from this function.
`...`	optional arguments to the print function.

Details

It calculates the area and its variance. For more than one marker it calculates the statistic to test for the equality of all AUCs. This statistic has a standard normal reference distribution for two variables and chi-square with number of variables minus 1.

Value

a list with the following elements

`area`	estimated area.
`var`	estimated variance (matrix).
`stat`	test statistic for equality of AUCs. Is not returned when only one diagnostic marker is present.
`p.value`	the p-value for the test of equality (2-sided).
`df`	the degrees of freedom of the chi-square.

The "print" method formats and returns the output.

References

DeLong, E. R., D. M. DeLong, and D. L. Clarke-Pearson. 1988. Comparing the areas under two or more correlated receiver operating characteristic curves: A nonparametric approach. *Biometrics* 44:837-845.

Examples

```
g <- rep(0:1, 50)
x <- rnorm(100) + g
y <- rnorm(100) + g
z <- rnorm(100) + g
roc.area.test(cbind(x,y), g)
roc.area.test(cbind(x,y,z), g)
y1 <- y + 0.75*g
roc.area.test(cbind(x,y1), g)
```

---

roc.curve *Empirical ROC curve*

Description

Computes the empirical ROC curve for a diagnostic tool.

Usage

```
  roc.curve(marker, status, method=c("empirical"))
  ## S3 method for class 'roc.curve'
print(x, ...)
  ## S3 method for class 'roc.curve'
plot(x, PRC=FALSE, ...)
  ## S3 method for class 'roc.curve'
lines(x, PRC=FALSE, ...)
```

Arguments

marker	the marker values for each subject.
status	binary disease status indicator
method	the method for estimating the ROC curve. Currently only the empirical curve is implemented.
x	object of class roc.curve output from this function.
PRC	flag to tell whether ROC or Precision-Recall curve plotted.
...	optional arguments to the print, plot and lines functions.

Details

The computation is based on assuming that larger values of the marker is indicative of the disease. So for a given threshold x0, TPR is P(marker >= x0|status =1) and FPR is P(marker >= x0|status =0). This function computes the empirical estimates of TPR and FPR.

Value

a list with the following elements

marker	the diagnostic marker being studied.
status	binary disease
tpr	true positive rates for all thresholds.
fpr	true positive rates for all thresholds.
ppv	positive predictive values for all thresholds.
npv	negative predictive values for all thresholds.

The "print" method returns the nonparametric AUC and its s.e.

The "plot" and "lines" methods can be used to draw a new plot and add to an existing plot of ROC curve.

Examples

```
g <- rep(0:1, 50)
x <- rnorm(100) + g
y <- rnorm(100) + 1.5*g
o <- roc.curve(x, g)
plot(o)
lines(roc.curve(y, g), col=2)
```

---

roc.perm.test *Permutation test to compare ROC curve*

Description

Computes the test statistic and permutation reference distribution for comparing paired or unpaired ROC curves.

Usage

```
roc.perm.test(marker, status, marker2=NULL, group=NULL,
              nperm=2500, mp=NULL)
## S3 method for class 'roc.perm.test'
print(x, ...)
## S3 method for class 'roc.perm.test'
plot(x, ...)
```

Arguments

marker	marker values for each subject.
status	binary disease status indicator.
marker2	second diagnostic marker for the same subjects (paired).
group	indicator of which diagnostic test was used (unpaired).
nperm	number of permutations for the reference distribution.
mp	mixing proportion for the unpaired case when proportion of diseased subjects can differ.
x	object of class roc.perm.test output from this function.
...	optional arguments to print and plot functions.

Details

This function implements the permutation method described in the Venkatraman and Begg (1996) paper for the paired case and the Venkatraman (2000) paper for the unpaired case.

The function detects whether the data are paired or unpaired by testing which of the options marker2 and group is specified. If both are missing it will stop with an error message. At present exactly one should be missing.

Value

an object of class roc.perm.test with the following elements

ostat	test statistic from the observed data.
pstat	test statistic from permuted data.
p.value	the p-value for the test of equality (2-sided).

The "print" method formats and returns the statistic and p-value. The "plot" method plots the density from the permutation reference distribution and marks the location of the observed statistic.

References

Venkatraman, E.S. and Begg, C.B. (1996). A distribution-free procedure for comparing receiver operating characteristic curves from a paired experiment. *Biometrika* 83, 835-848.

Venkatraman, E.S. (2000) A permutation test to compare receiver operating characteristic curves. *Biometrics* 56(4):1134-8.

Examples

```
d <- rep(0:1, 50)
x <- rnorm(100) + 1.2*d
y <- rnorm(100) + 1.2*d
oo <- roc.perm.test(x, d, marker2=y)
plot(oo)
oo <- roc.perm.test(c(x,y), c(d,d), group=rep(1:2,each=100))
plot(oo)
```

ROCanalysis *Functions to plot and compare ROC curves.*

Description

These functions can be used for nonparametric analysis of ROC curves.

Details

The relevant functions are `roc.curve`, `roc.area.test` and `roc.perm.test`. See the individual functions for usage details.

toxbdry *Stopping rule for toxicity/futility monitoring*

Description

Computes a stopping rule and its operating characteristics for toxicity monitoring based repeated significance testing.

Usage

```
toxbdry(pLo, pHi, n, cP0=0.1, cP1=0.9, ngrid=6, niter=10, delta=0,
          priority=c("null","alt"))
futilbdry(rLo, rHi, n, size=0.1, power=0.9, ngrid=3, niter=10, delta=0.5)
bdrycross.prob(n, r, ptox)
## S3 method for class 'toxbdry'
print(x, ...)
## S3 method for class 'futilbdry'
print(x, ...)
```

Arguments

`pLo`	the toxicity rate that is acceptable.
`rLo`	baseline (null) response rate.
`pHi`	the toxicity rate that is too high and hence unacceptable.
`rHi`	desirable response rate. Stop when it is too unlikely.
`n`	vector of times (sample size) when toxicty/response is monitored.
`r`	vector of maximum acceptable toxicities (non-responders for futility) corresponding to n.
`ptox`	the toxicity rates for which the operating characteristics are calculated. For futility this is the non-response rate.

cP0	boundary crossing probability under pLo i.e. type I error or the probability of declaring a treatment with toxicity rate pLo unacceptable.
cP1	boundary crossing probability under pHi i.e. power or the probability of declaring a treatment with toxicity rate pHi unacceptable.
size	probability of calling drug effective if response rate is rLo.
power	probability of calling drug effective if response rate is rHi.
ngrid	the number of toxicity rates from pLo to pHi for which the operating characteristics are computed.
niter	the number of iterations run to obtain the boundary.
delta	power determining the shape of the boundary. Should be between 0 (default) and 0.5.
priority	the error threshold to prioritize when the max sample size is too small to have both error thresholds satisfied. Default is the null i.e. error under pLo.
x	object returned by the function toxbdry.
...	additional arguments to print.

Details

The shape parameter delta is used to determine the stopping boundary with 0 corresponding to the Pocock boundary where the same significance level is used for all looks and 0.5 corresponding to the O'Brien-Fleming boundary which has smaller probability of stopping at early looks.

Default value of delta for toxicity monitoring is 0; value between 0.1 and 0.2 is a reasonable choice to make it less likely to stop early. Default values of delta for futility stopping is 0.5.

For toxicity monitoring two sets of probabilities - pstop and pcross - are given which correspond to probability of stopping early and probability of declaring the treatment too toxic with the full complement of study subjects accrued and treated.

For futility monitoring instead two sets of probabilities - pstop and peffective - are given corresponding to the probability of stopping early for futility and probability of finishing the trial and declaring it a success. Note that peffective is the complement of pcross in toxicity monitoring.

The futility boundary can have a -1 in earlier looks which means that even zero responses is not sufficient for stopping at that look.

The exact calculations in this function are done along the lines of the method in Chapter 12 of Jennison and Turnbull (2000). Ivanova, Qaqish and Schell (2005) have an illustrative paper.

Value

the function returns a list with:

looks	when toxicty is monitored - same as input n.
lo.bdry	lower boundary is a vector of maximum acceptable number of toxicities corresponding the number of subjects in n. The boundary crossing probability for this is slightly above cP0.
hi.bdry	upper boundary is a vector of maximum acceptable number of toxicities corresponding the number of subjects in n. The boundary crossing probability for this is slightly below cP0.

bdry.oc	the operating characteristics i.e the toxicity rate, the probability of crossing, stopping (i.e. cross before the last observation) and the expected sample size for both the low (lo) and high (hi) boundaries.
bdry.alpha	the alpha levels for testing at each look for the two boundaries.

stopping for toxicity is done when the number of toxicities exceeds the boundary i.e. the boundary gives the maximum acceptable number.

References

Jennison C and Turnbull BW. (2000). Group Sequential Methods with Applications to Clinical Trials. *Chapman and Hall/CRC*

Ivanova A, Qaqish BF and Schell MJ. (2005). Continuous Toxicity Monitoring in Phase II Trials in Oncology. *Biometrics* 61, 540-545.

Examples

```
toxbdry(0.2, 0.35, c(20,40,60,75))
toxbdry(0.2, 0.3, c(20,40,60,75), cP0=0.15, cP1=0.8)
# continuous monitoring
toxbdry(0.1, 0.3, 2:30)
# prioritize cP1 error threshold
toxbdry(0.1, 0.3, 2:25, priority="alt")
```

---

twostage.admissible *Admissible design options between Minimax and Optimal*

Description

Lists the admissible design options between

Usage

```
twostage.admissible(x)
```

Arguments

x	output from ph2simon call

Value

twostage.admissible returns design options that are admissible (Jung et al, 2004). The output is a matrix with 8 columns: r1, n1, r, n, EN(p0), PET(p0), qLo, qHi. The columns qLo and qHi give the range of probability values for which the particular design is admissible.

References

Jung SH, Lee T, Kim K, and George, SL. (2004). Admissible two-stage designs for phase II cancer clinical trials. *Statistics in medicine* 23(4), 561-569.

Examples

```
oo = ph2simon(0.5, 0.7, 0.05, 0.1)
twostage.admissible(oo)
```

`twostage.inference` *Inference following a two-stage design for binary response*

Description

Calculates the p-value, UMVUE and CI for the data from a study using a two stage design for response.

Usage

```
twostage.inference(x, r1, n1, n, pu, alpha=0.05)
```

Arguments

`x`	number of responses observed at the end of the study
`r1`	first stage threshold to declare treatment undesirable
`n1`	first stage sample size
`n`	total sample size
`pu`	unacceptable response rate (null hypothesis)
`alpha`	the confidence level. For consistency with the design use the same value from the design. (default is 0.05)

Value

twostage.inference returns the UMVUE (Jung & Kim, 2004), p-value and CI (Koyama & Chen, 2008). The CI has confidence level 1-2*alpha and the one-sided (1-alpha) interval consistent with the design is obtained by changing the upper confidence limit (UCL) to 1.

References

Jung SH and Kim KM. (2004). On the estimation of the binomial probability in multistage clinical trials. *Statistics in Medicine* 23, 881-896.

Koyama T and Chen H. (2008). Proper inference from Simon's two-stage designs. *Statistics in Medicine* 27, 3145-3154.

Index

∗ **design**
  fedesign, 8
  gsdesign, 10
  oc.twostage.bdry, 13
  ph2simon, 14
  ph2single, 16
  power.ladesign, 16
  pselect, 17
  toxbdry, 23
  twostage.admissible, 25
  twostage.inference, 26
∗ **htest**
  aucVardiTest, 2
  calogrank, 3
  deltaAUC, 7
  jonckheere.test, 11
  permlogrank, 14
  roc.area.test, 19
  roc.perm.test, 21
∗ **multivariate**
  ktau, 12
∗ **survival**
  coxphCPE, 4
  coxphERR, 5
  coxphQuantile, 6

aucVardiTest, 2

bdrycross.prob *(*toxbdry*)*, 23

calogrank, 3
coxphCPE, 4
coxphERR, 5
coxphQuantile, 6
CPS.ssize *(*fedesign*)*, 8

deltaAUC, 7

fe.mdor *(*fedesign*)*, 8
fe.power *(*fedesign*)*, 8
fe.ssize *(*fedesign*)*, 8
fedesign, 8
futilbdry *(*toxbdry*)*, 23

gsdesign, 10

ipmn *(*deltaAUC*)*, 7

jonckheere.test, 11

ktau, 12

lines.roc.curve *(*roc.curve*)*, 20

mdrr *(*fedesign*)*, 8

oc.twostage.bdry, 13, *15*
or2pcase *(*fedesign*)*, 8

permlogrank, 14
ph2simon, 14
ph2single, 16
plot.ph2simon *(*ph2simon*)*, 14
plot.roc.curve *(*roc.curve*)*, 20
plot.roc.perm.test *(*roc.perm.test*)*, 21
power.ladesign, 16
print.futilbdry *(*toxbdry*)*, 23
print.ladesign *(*power.ladesign*)*, 16
print.ph2simon *(*ph2simon*)*, 14
print.roc.area.test *(*roc.area.test*)*, 19
print.roc.curve *(*roc.curve*)*, 20
print.roc.perm.test *(*roc.perm.test*)*, 21
print.toxbdry *(*toxbdry*)*, 23
pselect, 17

roc.area.test, 19
roc.curve, 20
roc.perm.test, 21
ROCanalysis, 23
rocanalysis *(*ROCanalysis*)*, 23

toxbdry, 23
twostage.admissible, 25
twostage.inference, *15*, 26