

# Package ‘binGroup’

February 14, 2012

**Title** Evaluation and experimental design for binomial group testing

**Version** 1.0-9

**Date** 2011-12-13

**Author** Boan Zhang, Christopher Bilder, Brad Biggerstaff, Frank Schaarschmidt

**Description** This package provides methods for estimation and hypothesis testing of proportions in group testing designs. It involves methods for estimating a proportion in a single population (assuming sensitivity and specificity 1 in designs with equal group sizes), as well as hypothesis tests and functions for experimental design for this situation. For estimating one proportion or the difference of proportions, a number of confidence interval methods are included, which can deal with various different pool sizes. Further, regression methods are implemented for simple pooling and matrix pooling designs.

**Maintainer** Frank Schaarschmidt <schaarschmidt@biostat.uni-hannover.de>

**License** GPL

**Repository** CRAN

**Date/Publication** 2011-12-21 17:46:11

## R topics documented:

binGroup-package . . . . .	2
bgtCI . . . . .	6
bgtPower . . . . .	8
bgtTest . . . . .	10
bgtvs . . . . .	12
bgtWidth . . . . .	14
binCI . . . . .	16
binDesign . . . . .	18
binPower . . . . .	20

binTest . . . . .	22
binWidth . . . . .	24
estDesign . . . . .	25
gt.control . . . . .	27
gtreg . . . . .	28
gtreg.mp . . . . .	31
hivsurv . . . . .	35
nDesign . . . . .	36
plot.bgtDesign . . . . .	38
plot.binDesign . . . . .	39
pooledBin . . . . .	40
pooledBinDiff . . . . .	43
predict.gt . . . . .	45
print.bgt . . . . .	46
print.bgtDesign . . . . .	47
print.binDesign . . . . .	48
print.gt.mp . . . . .	48
print.poolbindiff . . . . .	49
print.summary.gt . . . . .	49
residuals.gt . . . . .	50
sDesign . . . . .	51
sim.gt . . . . .	53
sim.mp . . . . .	55
summary.gt . . . . .	57
summary.gt.mp . . . . .	58
summary.poolbindiff . . . . .	60

## Index 61

---

binGroup-package	<i>Statistical Methods for Group Testing.</i>
------------------	---

---

## Description

This package provides methods for estimation and hypothesis testing of proportions in group testing designs. It involves methods for estimating a proportion in a single population (assuming sensitivity and specificity 1 in designs with equal group sizes), as well as hypothesis tests and functions for experimental design for this situation. For estimating one proportion or the difference of proportions, a number of confidence interval methods are included, which can deal with various different pool sizes. Further, regression methods are implemented for simple pooling and matrix pooling designs.

## Details

Package:	binGroup
Type:	Package
Version:	1.0-8
Date:	2011-01-04

License: GPL  
LazyLoad: no

### 1) One-sample case

Methods for calculating confidence intervals for a single population proportion from designs with equal group sizes (as described by Tebbs and Bilder, 2004 and Schaarschmidt, 2007) are implemented in the function `bgtCI`.

For the problem of choosing an adequate experimental design in the one-sample case with only one group size, the functions `estDesign`, `sDesign`, `nDesign` implement different iterative approaches, as exemplified by Swallow (1985) and Schaarschmidt (2007).

If a confidence interval for a single proportion shall be calculated based on a design involving groups of different group sizes, a number of methods described by Hepworth (1999) is available in the function `pooledBin`. The exact method described by Hepworth (1996) is implemented in the function `bgtvs`.

### 2) Two-sample case

The function `pooledBinDiff` provides a number of confidence interval methods for estimating the difference of proportions from two independent samples, allowing for groups of different group size (Biggerstaff, 2008).

### 3) Regression models

Two approaches (by Vansteelandt et al., 2000 and Xie, 2001) to estimate parameters of group testing regression models can be applied by calling `gtreg`. Once fitted, corresponding methods to extract residuals, calculate predictions and summarize the parameter estimates (including hypotheses tests) are available in the S3 methods `residuals.gt`, `predict.gt` and `summary.gt`.

Group testing regression models in settings with matrix pooling (Xie, 2001) can be fit using `gtreg.mp`.

## Author(s)

Boan Zhang, Christopher Bilder, Brad Biggerstaff, Frank Schaarschmidt

Maintainer: Frank Schaarschmidt <schaarschmidt@biostat.uni-hannover.de>

## References

Biggerstaff, B.J. (2008): Confidence interval for the difference of proportions estimated from pooled samples. *Journal of Agricultural Biological and Environmental Statistics*, 13(4), 478-496.

Hepworth, G. (1996) Exact confidence intervals for proportions estimated by group testing. *Biometrics* 52, 1134-1146.

Hepworth, G. (1999): *Estimation of proportions by group testing*. PhD Dissertation. Melbourne, Australia: The University of Melbourne.

Schaarschmidt, F. (2007) Experimental design for one-sided confidence intervals or hypothesis tests in binomial group testing. *Communications in Biometry and Crop Science* 2 (1), 32-40. <http://agrobiol.sggw.waw.pl/cbcs/>

Swallow, W.H. (1985) Group testing for estimating infection rates and probabilities of disease transmission. *Phytopathology* 75 (8), 882-889.

Tebbs, J.M. & Bilder, C.R. (2004) Confidence interval procedures for the probability of disease transmission in multiple-vector-transfer designs. *Journal of Agricultural, Biological and Environmental Statistics* 9 (1), 75-90.

Vansteelandt, S., Goetghebeur, E., and Verstraeten, T. (2000) Regression models for disease prevalence with diagnostic tests on pools of serum samples, *Biometrics*, 56, 1126-1133.

Xie, M. (2001) Regression analysis of group testing samples, *Statistics in Medicine*, 20, 1957-1969.

## Examples

```
# 1) One-sample problem

# 1.1) Confidence intervals for designs with equal group size (pool size),
# where
# n denotes the number of groups (pools),
# s denotes the common group size (number of individuals pooled per group),
# y denotes the number of groups tested positive.

# The following call reproduces the example given
# by Tebbs and Bilder (2004) for the two-sided 95-percent
# exact (Clopper-Pearson) interval:

bgtCI(n=24, y=3, s=7, conf.level=0.95,
      alternative="two.sided", method="CP")

# 1.2) Confidence intervals for designs with unequal group size (pool size):
# Keeping notation as above but allowing for (a limited number of) different
# group size s, the examples given in Hepworth (1996), Table 5 can be
# reproduced by calling:

bgtvs(n=c(2,3), s=c(5,2), y=c(0,0))
bgtvs(n=c(2,3), s=c(5,2), y=c(0,1))

# The function pooledBin provides different methods for the same problem,
# where x is the number of positive groups, m is the size of the groups and
# n is the number of groups with the corresponding sizes:

pooledBin(x=c(0,1), m=c(5,2), n=c(2,3), ci.method="score")
pooledBin(x=c(0,1), m=c(5,2), n=c(2,3), ci.method="lrt")
pooledBin(x=c(0,1), m=c(5,2), n=c(2,3), ci.method="bc-skew-score")

# 1.3) For experimental design based on the bias of the point estimate,
# as proposed by Swallow (1985): The values in Table 1 (Swallow, 1985),
# p.885 can be reproduced by calling:

estDesign(n=10, smax=100, p.tr=0.001)
estDesign(n=10, smax=100, p.tr=0.01)

# 2) Two-sample comparison

# Assume a design, where pools 5, 1, 1, 30, and 20 pools of size 10, 4, 1, 25, 50,
# respectively, are used to estimate the prevalence in two populations.
```

```

# In population 1, one out of 5 pools with 10 units is positive,
# while in population 2, two out of five pools with 10 units is positive as well as
# the one pool with only 1 unit.
# The difference of proportions is to be estimated.

x1 <- c(1,0,0,0,0)
m1 <- c(10,4,1,25,50)
n1 <- c(5,1,1,30,20)

x2 <- c(2,0,1,0,0)
m2 <- c(10,4,1,25,50)
n2 <- c(5,1,1,30,20)

pooledBinDiff(x1=x1, m1=m1,x2=x2, m2=m2, n1=n1, n2=n2, ci.method="lrt")

# 3) Regression models

# 3.1) Fitting a regression model
# A HIV surveillance data (used by Vansteelandt et al. 2000)
# can be analysed for the dependence of HIV prevalence
# on covariates AGE and EDUC., with sensitivity and specificity
# assumed to be 0.9 each.

data(hivsurv)
fit1 <- gtmreg(formula = groupres ~ AGE + EDUC., data = hivsurv,
  groupn = gnum, sens = 0.9, spec = 0.9, method = "Xie")
summary(fit1)

# 3.2) Fitting a regression model for matrix pooling data
# The function sim.mp is used to simulate a matrix pooling data set:

set.seed(9128)
sa1a<-sim.mp(par=c(-7,0.1), n.row=c(5,4), n.col=c(6,5),
  sens=0.95, spec=0.95)

str(sa1a)
sa1<-sa1a$dframe

## Not run:
fit2 <- gtmreg.mp(formula = cbind(col.resp, row.resp) ~ x, data = sa1,
  coln = coln, rown = rown, arrayn = arrayn,
  sens = 0.95, spec = 0.95, n.gibbs = 2000, trace = TRUE)

fit2
summary(fit2)

## End(Not run)

```

**Description**

Calculates the point estimate, the exact Clopper-Pearson and Blaker CI, the Score test derived Wilson and Agresti-Coull CI, the asymptotic second-order corrected interval fo Cai and the Wald CI for a single binomial proportion estimated from a binomial group testing trial. Assumes equal group sizes, an assay method classifying a group as positive if at least one unit in the group is positive, individuals units randomly assigned to the groups.

**Usage**

```
bgtCI(n, s, y, conf.level = 0.95,
      alternative = "two.sided", method = "CP")
```

**Arguments**

n	integer, specifying the number of groups (i.e. assays i.e. observations)
s	integer, specifying the common size of groups i.e. the number of individual units in each group
y	integer, specifying the number of positive groups
conf.level	nominal confidence level of the interval
alternative	character string defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' gives the only an upper bound with confidence level=conf.level 'greater' gives the only a lower bound with confidence level=conf.level and 'two.sided' gives a two-sided confidence interval with confidence level=conf.level
method	character string defining the method for CI calculation, where: "CP" is Clopper-Pearson, an exact tail interval showing symmetric coverage probability (inversion of two one-sided tests), "Blaker" is the Blaker interval, an exact interval, inversion of one two.sided test, therefore defined only two.sided, but shorter than the two-sided Clopper-Pearson CI. Both guarantee to contain the true parameter with at least conf.level*100 percent probability, "AC" is the Agresti-Coull (generalized Agresti-Coull) interval, asymptotic method, "Score" is Wilson Score, asymptotic method derived from inversion of the Score test, "SOC" is the second order corrected interval, asymptotic method for one-sided problems (for details see Cai, 2005), and "Wald" the Wald interval, which cannot be recommended.

**Details**

This function allows the computation of confidence intervals for binomial group testing as described in Tebbs & Bilder (2004) and Schaarschmidt (2007). If an actual confidence level greater or equal to that specified in the conf.level argument shall always be guaranteed, the exact method of Clopper-Pearson (method="CP") can be recommended for one-sided and the improved method of Blaker (2000) (method="Blaker") can be recommended for two-sided hypotheses. If a mean confidence level close to that specified in the argument conf.level is required, but moderate violation of this

level is acceptable, the Second-Order corrected (method="SOC"), Wilson Score (method="Score") or Agresti-Coull (method="AC") might be used (Brown, Cai, DasGupta, 2001; Cai 2005).

### Value

A list containing:

conf.int	a confidence interval for the proportion
estimate	the point estimator of the proportion

### Author(s)

Frank Schaarschmidt

### References

Blaker H (2000). Confidence curves and improved exact confidence intervals for discrete distributions. *The Canadian Journal of Statistics* 28 (4), 783-798.

Brown LD, Cai TT, DasGupta A (2001). Interval estimation for a binomial proportion. *Statistical Science* 16 (2), 101-133.

Cai TT (2005). One-sided confidence intervals in discrete distributions. *Journal of Statistical Planning and Inference* 131, 63-88.

Schaarschmidt F (2007). Experimental design for one-sided confidence intervals or hypothesis tests in binomial group testing. *Communications in Biometry and Crop Science* 2 (1), 32-40. <http://agrobiol.sggw.waw.pl/cbcs/>

Tebbs JM & Bilder CR (2004). Confidence interval procedures for the probability of disease transmission in multiple-vector-transfer designs. *Journal of Agricultural, Biological and Environmental Statistics*, 9 (1), 75-90.

### See Also

[pooledBin](#) for asymptotic confidence intervals and [bgtvs](#) for an exact confidence interval when designs with different group sizes are used

[bgtTest](#): for hypothesis tests in binomial group testing

### Examples

```
# See the example in Tebbs and Bilder (2004)
# the two.sided 95-percent
# Clopper-Pearson as default method:

bgtCI(n=24,y=3,s=7)
bgtCI(n=24,y=3,s=7,conf.level=0.95,
      alternative="two.sided", method="CP")

# other methods:
# Blaker CI is exact but shorter
# than Clopper-Pearson, only two.sided
```

```

bgtCI(n=24,y=3,s=7, alternative="two.sided",
      method="Blaker")

# the asymptotic Wilson CI might even
# be shorter:

bgtCI(n=24,y=3,s=7, alternative="two.sided",
      method="Score")

# one-sided confidence intervals:

bgtCI(n=24,y=3,s=7, alternative="less", method="CP")

# Wilson Score interval is less conservative
bgtCI(n=24,y=3,s=7, alternative="less", method="Score")

# the second-order corrected CI is even shorter
# in this situation:
bgtCI(n=24,y=3,s=7, alternative="less", method="SOC")

```

---

bgtPower	<i>Power to Reject a Hypothesis in Binomial Group Testing for One Proportion</i>
----------	--

---

### Description

Closed calculation of the Power to reject a hypothesis in a binomial group testing experiment using confidence intervals for decision. Closed calculation of bias of the point estimator for a given experimental design  $n$ ,  $s$  and the true, unknown proportion.

### Usage

```

bgtPower(n, s, delta, p.hyp, conf.level = 0.95,
         method = "CP", alternative = "two.sided")

```

### Arguments

$n$	integer, giving the number of groups i.e. assays i.e. observations, a vector of integers is also allowed
$s$	integer, giving the common size of groups i.e. the number of individual units in each group, a vector of integers is also allowed
delta	absolute difference between the threshold and the true proportion which shall be detectable with specified power, a vector is also allowed
p.hyp	number between 0 and 1, specifying the threshold proportion in the hypotheses
conf.level	confidence level required for the decision on the hypotheses

method	character string, specifying the Confidence interval method (see <a href="#">bgtCI</a> ) to be used
alternative	character string, defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' calculates the probability that p.hyp is excluded by an upper confidence limit for a true proportion p.hyp-delta, 'greater' calculates the probability that p.hyp is excluded by a lower confidence limit for a true proportion p.hyp+delta, 'two.sided' calculates $\min(\text{power}(p.hyp-delta, p.hyp+delta))$ for a two-sided CI, thus can result in much lower power.

### Details

The power of a confidence interval here is defined as the probability that a confidence interval or limit excludes the threshold parameter (p.hyp) of the null hypothesis, as described in Schaarschmidt(2007). I.e., the null hypothesis  $H_0: p \geq p.hyp$  might be rejected, if an upper confidence limit for p does not contain p.hyp. Due to discreteness, the power does not increase monotone for increasing number of groups n or group size s, but exhibits local maxima and minima, depending on n,s, conf.level, p.hyp. The power can be identical for different methods, depending on the particular combination of n, s, p.hyp, conf.level. Note that coverage probability and power are not necessarily symmetric for upper and lower bounds of binomial CI, especially for Wald, Wilson Score and Agresti-Coull CI.

Additional to the power, bias of the point estimator is calculated according to Swallow (1985).

If vectors are specified for n, s, and (or) delta, a matrix will be constructed and power and bias are calculated for each line in this matrix.

### Value

A matrix containing the columns

ns	a vector of total sample size n*s resulting from the latter
n	a vector of number of groups
s	a vector of group sizes
delta	a vector of delta
power	the power to reject the given null hypothesis, with the specified method and parameters of the first 4 columns
bias	the bias of the estimator for the specified n, s, and the true proportion

### References

*Schaarschmidt F (2007)*. Experimental design for one-sided confidence intervals or hypothesis tests in binomial group testing. *Communications in Biometry and Crop Science* 2 (1), 32-40. <http://agrobiol.sggw.waw.pl/cbcs/>

*Swallow WH (1985)*. Group testing for estimating infection rates and probabilities of disease transmission. *Phytopathology* Vol.75, N.8, 882-889.

**See Also**

**nDesign**: stepwise increasing  $n$  (for a fixed group size  $s$ ) until a certain power is reached within a restriction of bias of the estimator **sDesign**: stepwise increasing  $s$  (for a fixed number of groups) until a certain power is reached within a restriction of bias of the estimator **estDesign**: selection of an appropriate design to achieve minimal mean square error of the estimator

**Examples**

```
# Calculate the power for the design
# in the example given in Tebbs and Bilder(2004):
# n=24 groups each containing 7 insects
# if the true proportion of virus vectors
# in the population would be 0.04 (4 percent),
# the power to reject H0: p>=0.1 using an
# upper Clopper-Pearson ("CP") confidence interval
# can be calculated using the following call:

bgtPower(n=24, s=7, delta=0.06, p.hyp=0.1,
  conf.level=0.95, alternative="less", method="CP")

# c(), seq() or rep() might be used to explore development
# of power and bias for varying n, s, delta. How much can
# we decrease the number of groups (costly assays to be performed)
# by pooling the same number of 320 individuals to groups of
# increasing size without largely decreasing power?

bgtPower(n=c(320,160,80,64,40,32,20,10,5),
  s=c(1,2,4,5,8,10,16,32,64),
  delta=0.01, p.hyp=0.02)

# How does power develop for increasing differences
# delta between the true proportion and the threshold proportion?

bgtPower(n=50, s=10, delta=seq(from=0, to=0.01, by=0.001),
  p.hyp=0.01, method="CP")

# use a more liberal method:

bgtPower(n=50, s=10, delta=seq(from=0, to=0.01, by=0.001),
  p.hyp=0.01, method="SOC")
```

**Description**

Calculates p values for hypotheses tests of binomial proportions estimated from binomial group testing experiments against a threshold proportion in the hypotheses. Exact test, Score test and Wald test are available methods. Assumes equal group sizes, 100 percent sensitivity and specificity of the assays to test the groups, and individuals units randomly assigned to the groups with identical true probability of success.

**Usage**

```
bgtTest(n, y, s, p.hyp, alternative = "two.sided",
        method = "Exact")
```

**Arguments**

n	integer, number of groups (i.e. assays i.e. observations)
y	integer, number of positive groups
s	integer, common size of groups i.e. the number of individual units in each group
p.hyp	number between 0 and 1, specifying the hypothetical threshold proportion to test against
alternative	character string defining the alternative hypothesis, either 'two.sided', 'less' or 'greater'
method	character string defining the test method to be used: can be one of 'Exact' for an exact test corresponding to the Clopper-Pearson confidence interval 'Score' for a Score test, corresponding to the Wilson confidence interval 'Wald' for a Wald test corresponding to the Wald confidence interval, not recommended

**Value**

A list containing:

p.value	the p value of the test
estimate	the estimated proportion
p.hyp	as input
alternative	as input
method	as input

**References**

*Swallow WH (1985) Group testing for estimating infection rates and probabilities of disease transmission. Phytopathology 75 (8), 882-889.*

*Blyth, C and Still, H. (1983) Binomial confidence intervals. Journal of the American Statistical Association 78, 108-116.*

*Santner TJ and Duffy DE (1989) The statistical analysis of discrete data. Springer New York.*

*Remund KM, Dixon DA, Wright DL, Holden LR (2001) Statistical considerations on seed purity testing on transgenic traits. Seed Science Research (11), 101-119.*

**See Also**

[bgtCI](#) for confidence intervals in binomial group testing

**Examples**

```
# Assume the experiment: Assays are performed on
# n=10 groups, each group is a bulk sample
# of s=100 individuals, aim is to show that
# less than 0.5 percent ('p<0.005') of the units
# of the population show a detrimental trait (positive assay).
# The assay is sensitive to show a positive result if only 1
# unit in the bulk sample of 100 units is positive.
# y=1 positive assay and 9 negative assays are observed.

bgtTest(n=10,y=1,s=100,alternative="less",method="Exact",p.hyp=0.005)

# The exact test corresponds to the
# limits of the Clopper-Pearson confidence interval
# in the example of Tebbs and Bilder(2004):

bgtTest(n=24, y=3, s=7, alternative="two.sided",
method="Exact", p.hyp=0.0543)

bgtTest(n=24, y=3, s=7, alternative="two.sided",
method="Exact", p.hyp=0.0038)

# Further methods:

bgtTest(n=24, y=3, s=7, alternative="two.sided",
method="Score", p.hyp=0.0516)

bgtTest(n=24, y=3, s=7, alternative="two.sided",
method="Wald", p.hyp=0.0401)
```

---

bgtvs

*Confidence Interval for One Proportion in Group Testing with Variable Group Sizes*

---

**Description**

Calculates confidence intervals for a single proportion in binomial group testing if groups of different size are evaluated

**Usage**

```
bgtvs(n, s, y, conf.level = 0.95, alternative = "two.sided",
maxiter = 100)
```

**Arguments**

n	vector of integer values, specifying the number of groups of the different sizes
s	vector of integer values, specifying the group sizes, must be of same length as n
y	vector of integer values, specifying the number of positive groups among the n groups tested
conf.level	a single numeric value, the confidence level of the interval
alternative	a character string, with options "two.sided", "less", "greater"
maxiter	maximal number steps in iteration of confidence limits

**Details**

Hepworth (1996) describes methods for constructing confidence intervals in binomial group testing, if groups of different size are used. Currently, only the exact method (Hepworth, 1996, equation 5, Table.5) is implemented. Note, that the algorithm becomes computationally very expensive if the number of different groups becomes larger than 3.

**Value**

A list containing

conf.int	a numeric vector, the lower and upper limits of the confidence interval
estimate	the point estimate
conf.level	as input
alternative	as input
input	a matrix containing the input values of n (number of groups), s (group size), and y (number of positive pools)

moreover, some of the input arguments.

**Author(s)**

Frank Schaarschmidt

**References**

*Hepworth, G (1996): Exact confidence intervals for proportions estimated by group testing. Biometrics 52, 1134-1146.*

**See Also**

[pooledBin](#) for asymptotic methods to calculate confidence intervals for one proportion in designs with a number of different pool sizes. Note that pooledBin can handle larger number of different pool sizes than bgtvs

**Examples**

```

# Consider a very simple example,
# given in Hepworth (1996), table 5:
# 2 groups each containing 5 units,
# and 3 groups, each containing 2 units

# In the first setting (n=2, s=5) y=1 positive group
# has been observed, in the second setting (n=3, s=2),
# y=2 positive have been observed.

bgtvs(n=c(2,3), s=c(5,2), y=c(1,2))

#####

# Recalculate the example given in
# Hepworth (1996), table 5:

bgtvs(n=c(2,3), s=c(5,2), y=c(0,0))
bgtvs(n=c(2,3), s=c(5,2), y=c(0,1))
bgtvs(n=c(2,3), s=c(5,2), y=c(0,2))
bgtvs(n=c(2,3), s=c(5,2), y=c(0,3))
bgtvs(n=c(2,3), s=c(5,2), y=c(1,0))
bgtvs(n=c(2,3), s=c(5,2), y=c(1,1))
bgtvs(n=c(2,3), s=c(5,2), y=c(1,2))
bgtvs(n=c(2,3), s=c(5,2), y=c(1,3))
bgtvs(n=c(2,3), s=c(5,2), y=c(2,0))
bgtvs(n=c(2,3), s=c(5,2), y=c(2,1))
bgtvs(n=c(2,3), s=c(5,2), y=c(2,2))
bgtvs(n=c(2,3), s=c(5,2), y=c(2,3))

```

---

bgtWidth

*Expected Width of Confidence Intervals in Binomial Group Testing*


---

**Description**

Calculation of expected value of the width of confidence intervals for one proportion in binomial group testing, in dependence of the number of groups, group size, confidence level and an assumed true proportion. Available for the confidence interval methods in `bgtCI(binGroup)`.

**Usage**

```

bgtWidth(n, s, p, conf.level = 0.95, alternative = "two.sided",
method = "CP")

```

**Arguments**

`n` integer, giving the number of groups i.e. assays i.e. observations, vector of integers is also allowed

s	integer, giving the common size of groups i.e. the number of individual units in each group, vector of integers is also allowed
p	assumed true proportion of individuals showing the trait to be estimated, vector is also allowed
conf.level	required confidence level of the interval
alternative	character string, defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' calculates the expected width between the assumed true proportion p and the upper conf.level*100 percent-bound of a one-sided CI, 'greater' calculates the expected width between the true assumed proportion p and the lower conf.level*100 percent-bound of a one-sided CI, 'two.sided' calculates the expected width between the lower and the upper bound of a two-sided conf.level*100 percent-CI.
method	character string as in the method argument in <a href="#">bgtCI</a>

### Details

For calculation of expected interval width in the standard binomial estimation see, e.g., Brown et al. (2001). The calculation in case of binomial group testing is simply done by replacing the binomial probabilities by the probabilities  $P(Y=y)$  for group testing (see Tebbs and Bilder, 2004)

### Value

A matrix containing the columns

ns	the resulting total number of units n*s
n	number of groups
s	group size
p	the assumed true proportion
	and the calculated
expCIwidth	expected value of CI width as defined under argument alternative

### Author(s)

Frank Schaarschmidt

### References

Brown LD, Cai TT, DasGupta A (2001) Interval estimation for a binomial proportion. *Statistical Science* 16 (2), 101-133.

Schaarschmidt F (2007) Experimental design for one-sided confidence intervals or hypothesis tests in binomial group testing. *Communications in Biometry and Crop Science* 2 (1), 32-40. <http://agrobiol.sggw.waw.pl/cbcs/>

Tebbs JM & Bilder CR (2004) Confidence interval procedures for the probability of disease transmission in multiple-vector-transfer designs. *Journal of Agricultural, Biological and Environmental Statistics* 9 (1), 75-90.

**Examples**

```
# There is a minimal expected CI length, if
# group size s is increased (fixed other parameters)
# the corresponding group size might be chosen:

bgtWidth(n=20, s=seq(from=1, to=200, by=10),
  p=0.01, alternative="less", method="CP" )

# and this depends largely on the assumed proportion p:

bgtWidth(n=20, s=seq(from=1, to=200, by=10),
  p=0.05, alternative="less", method="CP" )

bgtWidth(n=20, s=seq(from=1, to=200, by=10),
  p=0.005, alternative="less", method="CP" )
```

binCI

*Confidence Intervals for One Binomial Proportion***Description**

Calculates the exact Clopper-Pearson and Blaker, the asymptotic second-order corrected, Wilson, Agresti-Coull and Wald confidence interval for a single binomial proportion

**Usage**

```
binCI(n, y, conf.level = 0.95, alternative = "two.sided",
  method = "CP")

binCP(n, y, conf.level=0.95, alternative="two.sided")
binBlaker(n,y,conf.level=0.95, tolerance=1e-04, alternative="two.sided")
binAC(n, y, conf.level=0.95, alternative="two.sided")
binSOC(n, y,conf.level=0.95,alternative="two.sided")
binWald(n, y, conf.level=0.95, alternative="two.sided")
binWilson(n, y,conf.level=0.95,alternative="two.sided")
```

**Arguments**

n	number of trials (number of individuals under observation)
y	number of successes (number of individuals showing the trait of interest)
conf.level	nominal confidence level
alternative	character string defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' gives the only an upper bound with confidence level=conf.level 'greater' gives the only a lower bound with confidence level=conf.level and 'two.sided' gives a two-sided confidence interval with confidence level=conf.level

method	character string defining the method for CI calculation: where "CP" is Clopper-Pearson, an exact tail interval showing symmetric coverage probability (inversion of two one-sided tests), "Blaker" is the Blaker interval, an exact interval, inversion of one two-sided test, therefore defined only two.sided, but shorter than the two-sided Clopper-Pearson CI. Both guarantee to contain the true parameter with at least $\text{conf.level} \times 100$ percent probability, "AC" is Agresti-Coull, generalized Agresti-Coull interval, asymptotic method, "Score" is Wilson Score, asymptotic method derived from inversion of the Score test, "SOC" is the second order corrected interval, asymptotic method for one-sided problems (for details see Cai, 2005), and "Wald" the Wald interval, which cannot be recommended.
tolerance	precision of computation for the bounds of the Blaker interval

### Details

This function allows computation of confidence intervals for a binomial proportion from a standard binomial experiment. If an actual confidence level greater or equal to that specified in the `conf.level` argument shall always be guaranteed, the exact method of Clopper-Pearson (`method="CP"`) can be recommended for one-sided and the improved method of Blaker (`method="Blaker"`) can be recommended for two-sided hypotheses. If a mean confidence level close to that specified in the argument `conf.level` is required, but moderate violation of this level is acceptable, the Second-Order corrected (`method="SOC"`), Wilson Score (`method="Wilson"`) or Agresti-Coull (`method="AC"`) might be used, where SOC has the most symmetric coverage and Wilson and Agresti-Coull are in tendency conservative for the upper bound and proportions close to 0 and for the lower bound and proportions close to 1. The Wald CI might be used for large number of observations  $n > 10000$  or intermediate proportions.

For discussion of CI for a single binomial proportion see Brown et al. (2001) for two-sided and Cai (2005) for one-sided intervals.

### Value

A list containing:

<code>conf.int</code>	the estimated confidence interval
<code>estimate</code>	the point estimator

And the `method`, `conf.level` and `alternative` specified in the function call.

### Author(s)

Frank Schaarschmidt

### References

- Blaker H (2000) Confidence curves and improved exact confidence intervals for discrete distributions. The Canadian Journal of Statistics 28 (4), 783-798.*
- Brown LD, Cai TT, DasGupta A (2001) Interval estimation for a binomial proportion. Statistical Science 16 (2), 101-133.*
- Cai TT(2005) One-sided confidence intervals in discrete distributions. Journal of Statistical Planning and Inference 131, 63-88.*

**See Also**

[binom.test](#) for the exact confidence interval and test, [binTest](#) to calculate p.values of the exact, Score and Wald test.

**Examples**

```
# Default method is the two-sided 95% Clopper-Pearson CI:
```

```
binCI(n=200, y=10)
```

```
# other methods might result in
# shorter intervals (but asymmetric coverage):
```

```
binCI(n=200,y=10, method="Blaker")
```

```
binCI(n=200,y=10, method="Score")
```

---

binDesign

*Sample Size Iteration for One Parameter Binomial Problem*


---

**Description**

This function increases the sample size until a maximal sample size or a prespecified power is achieved.

**Usage**

```
binDesign(nmax, delta, p.hyp, conf.level = 0.95,
power = 0.8, method = "CP", alternative = "two.sided")
```

**Arguments**

nmax	integer, maximal number of trials (individuals under observation) allowed in the iteration
delta	absolute difference between the threshold and the true proportion which shall be detectable with the specified power
p.hyp	threshold proportion to test against in the hypothesis, specify as a value between 0 and 1
conf.level	Confidence level of the decision, default is 0.95
power	Level of power to be achieved to be specified as a probability between 0 and 1
method	character string specifying the CI method to be used for evaluation, see argument method in <code>bgtCI</code>

alternative character string, defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' calculates the probability that p.hyp is excluded by an upper confidence limit given that the true proportion is p.hyp-delta, 'greater' calculates the probability that p.hyp is excluded by a lower confidence limit given that the true proportion is p.hyp+delta. 'two.sided' calculates  $\min(\text{power}(p.\text{hyp}-\text{delta}, p.\text{hyp}+\text{delta}))$  for a two-sided CI, thus can result in much lower power.

### Details

The power of a confidence interval here is defined as the probability that a confidence interval or limit excludes the threshold parameter ( $p.\text{hyp}$ ) of the hypothesis.

This function increases the number of trials (number of individuals under observation) until a pre-specified power is reached. Since the power does not increase monotone with increasing  $n$  for binomial proportions but oscillates between local maxima and minima, the simple iteration given here will generally result in selecting those  $n$ , for which the given CI method shows a local minimum of coverage if the null hypothesis is true. The power can be identical for different methods, depending on the particular combination of  $n$ ,  $p.\text{hyp}$ ,  $\text{conf.level}$ .

Especially for large  $n$ , the calculation time may become large (particularly for Blaker). Then only the range of sample size which is of interest can be specified in  $n\text{max}$ , f.e. as:  $n\text{max}=c(150,300)$ . Alternatively, the function `binPower` might be used instead to calculate power and bias only for some particular combinations of  $n$ ,  $\text{delta}$ ,  $p.\text{hyp}, \dots$ .

Note that coverage probability and power are not necessarily symmetric for upper and lower bound of binomial CI.

The results can be visualized by application of the function `plot()` to the returned object of this function.

### Value

a list

powerout power value, either the power of the first  $n$  exceeding the pre-specified power, or the maximal power achieved in the specified range of  $n$  if the specified power has not been reached

nout corresponding sample size  $n$  (the number of trials) for which the prespecified power is reached, or the sample size  $n$  for which the maximal power has been reached within the specified range, if pre-specified power has not been reached.

and additional input and iteration values needed only for the function `plot.binDesign`.

### Author(s)

Frank Schaarschmidt

### References

Schaarschmidt, F. (2007). Experimental design for one-sided confidence intervals or hypothesis tests in binomial group testing. *Communications in Biometry and Crop Science* 2 (1), 32-40. <http://agrobiol.sggw.waw.pl/cbcs/>

**See Also**

[binPower](#) for calculation of power, [plot.binDesign](#) for plot of the results

**Examples**

```
# Find a sample size for which the power to
# reject H0: p >= 0.1 in favor of HA: p < 0.1 is
# at least 0.9 (90 percent) in case that the
# true proportion is 0.04 (i.e. an absolute delta
# of 0.06 to the threshold proportion p.hyp=0.1)
# The exact one sided Clopper-Pearson CI shall be
# used with default confidence level = 0.95.

sasi<-binDesign( nmax=200, delta=0.06,
  p.hyp=0.1, alternative="less", method="CP", power=0.9)
sasi

#### One can also plot the result:

plot(sasi)

# For larger sample sizes iteration can be very time consuming.
# Better to use only a smaller range of n then:

sasi<-binDesign( nmax=c(200,300), delta=0.03, p.hyp=0.1,
  alternative="less", method="CP", power=0.9)
sasi
```

---

binPower

*Power Calculation for One Parameter Binomial Problem*

---

**Description**

Closed calculation of the power to reject a hypothesis using confidence intervals for a single binomial proportion, for specified sample size  $n$ , `conf.level` and an assumed absolute difference to the threshold parameter under the null hypothesis.

**Usage**

```
binPower(n, delta, p.hyp, conf.level = 0.95,
  alternative = "two.sided", method = "CP")
```

**Arguments**

n	number of trials n in the binomial experiment, specify as a single integer
delta	assumed absolute difference of the true proportion to the threshold proportion under the null hypothesis
p.hyp	threshold proportion under the null hypothesis
conf.level	nominal confidence level of the interval
alternative	character string, defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' calculates the 'power of the upper confidence limit' for a true proportion p.hyp - delta, 'greater' calculates the 'power of the lower confidece limit' for a true proportion of p.hyp + delta. 'two.sided' calculates $\min(\text{power}(p.hyp - delta, p.hyp + delta))$ for a two-sided CI, thus can result in much lower power.
method	Character string, specifying the confidence interval method (see <a href="#">binCI</a> ) to be used

**Details**

The power of a confidence interval here is defined as the probability that a confidence interval or limit excludes the threshold parameter (p.hyp) of the null hypothesis. E.g., the null hypothesis  $H_0: p \geq 0.005$  can be rejected, if an upper confidence limit for p does not contain p.hyp=0.005. In case that a delta of 0.002 shall be detectable, this function calculates the probability, that an interval of a given method will exclude p.hyp=0.005 if the true proportion = 0.003. Due to discreteness, the power does not increase monotone for increasing sample size (number of trials or individuals under observation) n, but exhibits local maxima and minima, depending on n, conf.level and p.hyp. The power can be identical for different methods, depending on the particular combination of n, p.hyp, conf.level. Note, that coverage probability and power are not necessarily symmetric for upper and lower bound of binomial CI, especially for Wald, Wilson Score and Agresti-Coull CI.

**Value**

A list containing

power                    the power which is achieved for the specified parameters and method

**Author(s)**

Frank Schaarschmidt

**See Also**

[binDesign](#) for iteration of a sample size n for which a specified power is reached

**Examples**

```
# What is the probability to reject the null hypothesis
# H0: p >= 0.02 in order to show that the alternative
# hypothesis HA: p < 0.02 is very likely in the first
```

```

# example of if 200 seeds are taken from a seed lot and
# are checked for the proportion of defectives.
# Assume a true proportion under the alternative:
# p = 0.01, i.e. a absolute difference delta = 0.01
# to the threshold proportion p.hyp=0.02.
# The null hypothesis can be rejected if the threshold
# p.hyp=0.02 is excluded by an 95 percent upper bound of the
# Clopper-Pearson CI. What is the probability of this event?

binPower(n=200, delta=0.01, p.hyp=0.02,
  alternative="less", method="CP")

# Assuming a lower true proportion (if one is satisfied
# also with the situation that we can only reject H0
# in case that the seed lot has a very high purity, e.g.
# only a proportion of 0.001 defectives )

binPower(n=200, delta=0.019, p.hyp=0.02,
  alternative="less", method="CP")

# Or use a higher sample size:

binPower(n=600, delta=0.01, p.hyp=0.02,
  alternative="less", method="CP")

```

---

binTest

*Hypothesis tests for One Binomial Proportion*


---

### Description

Calculates p-values for hypothesis tests of a single binomial proportion.

### Usage

```
binTest(n, y, p.hyp, alternative = "two.sided",
  method = "Exact")
```

### Arguments

n	single integer value, number of trials (number of individuals under observation)
y	single integer value, number of successes (number of individuals showing the trait of interest)
p.hyp	single numeric value between 0 and 1, specifying the hypothetical threshold proportion to test against
alternative	character string defining the alternative hypothesis, either 'two.sided', 'less' or 'greater'

method character string defining the test method to be used: can be one of "Exact" for an exact test corresponding to the Clopper-Pearson confidence interval, uses `binom.test(stats)` "Score" for a Score test, corresponding to the Wilson confidence interval "Wald" for a Wald test corresponding to the Wald confidence interval

### Value

A list containing:

p.value	the p value of the test
estimate	the estimated proportion
p.hyp	as input
alternative	as input
method	as input

### Author(s)

Frank Schaarschmidt

### References

*Santner, T.J. and Duffy, D.E. (1989)* The statistical analysis of discrete data. Springer Verlag New York Berlin Heidelberg. Chapter 2.1.

### See Also

`binom.test(stats)` for the exact test and corresponding confidence interval

### Examples

```
# 200 seeds are taken from a seed lot.  
# 2 are found to be defective.  
# H0: p >= 0.02 shall be rejected in favor of HA: p < 0.02.  
# The exact test shall be used for decision:  
  
binTest(n=200, y=2, p.hyp=0.02, alternative="less", method="Exact" )
```

binWidth

*Expected Confidence Interval Width for One Binomial Proportion***Description**

Calculation of expected value of the width of confidence intervals in a binomial experiment, in dependence of the number of trials (number of individuals under observation), confidence level and an assumed true proportion. Available for the confidence interval methods in binCI(binGroup).

**Usage**

```
binWidth(n, p, conf.level = 0.95,
         alternative = "two.sided", method = "CP")
```

**Arguments**

n	integer, giving the number of trials (i.e. number of individuals under observation)
p	assumed true proportion of individuals showing the trait to be estimated
conf.level	required confidence level of the interval
alternative	character string, defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' calculates the expected width between the assumed true proportion p and the upper conf.level*100 percent-bound of a one-sided CI, 'greater' calculates the expected width between the assumed true proportion p and the lower conf.level*100 percent-bound of a one-sided CI, 'two.sided' calculates the expected width between the lower and the upper bound of a two-sided conf.level*100 percent-CI.
method	character string defining the method for CI calculation: where "CP" is Clopper-Pearson, an exact tail interval showing symmetric coverage probability (inversion of two one-sided tests), "Blaker" is the Blaker interval, an exact interval, inversion of one two.sided test, therefore defined only two.sided, but shorter than the two-sided Clopper-Pearson CI. Both guarantee to contain the true parameter with at least conf.level*100 percent probability, "AC" is Agresti-Coull, generalized Agresti-Coull interval, asymptotic method, "Score" is Wilson Score, asymptotic method derived from inversion of the Score test, "SOC" is the second order corrected interval, asymptotic method for one-sided problems (for details see Cai, 2005), and "Wald" the simple Wald-type interval.

**Details**

For calculation of expected interval width in the standard binomial estimation see Brown et al. (2001).

**Value**

A list containing:

expCIWidth      the expected value of the width of the confidence interval for the specified arguments

and the alternative, p and n which are specified in the function call.

**Author(s)**

Frank Schaarschmidt

**See Also**

[binDesign](#) for experimental design for hypothesis testing

**Examples**

```
# methods differ slightly in length when sample sizes are large:
```

```
binWidth(n=200,p=0.02,alternative="two.sided",  
method="CP")$expCIWidth
```

```
binWidth(n=200,p=0.02,alternative="two.sided",  
method="Blaker")$expCIWidth
```

```
binWidth(n=200,p=0.02,alternative="two.sided",  
method="Score")$expCIWidth
```

```
# but do more for small sample sizes and intermediate p:
```

```
binWidth(n=20,p=0.2,alternative="two.sided",  
method="CP")$expCIWidth
```

```
binWidth(n=20,p=0.2,alternative="two.sided",  
method="Blaker")$expCIWidth
```

```
binWidth(n=20,p=0.2,alternative="two.sided",  
method="Score")$expCIWidth
```

**Description**

Find the group size  $s$  for a fixed number of assays  $n$  and an assumed true proportion  $p.tr$  for which the mean squared error (mse) of the point estimator is minimal and bias is within a restriction. For experimental design in binomial group testing as recommended by Swallow (1985), if main objective is estimation.

**Usage**

```
estDesign(n, smax, p.tr, biasrest = 0.05)
```

**Arguments**

<code>n</code>	integer, fixed sample size (number of assays)
<code>smax</code>	integer, maximal group size allowed in planning of the design
<code>p.tr</code>	assumed true proportion of the 'positive' trait in the population to be tested, specify as a value between 0 and 1
<code>biasrest</code>	value between 0 and 1 specifying the absolute bias maximally allowed

**Details**

Swallow (1985) recommends to use the upper bound of the expected range of true proportion  $p.tr$  for optimization of the design. For further details see the reference. Up to now, specify  $n < 1020$ .

**Value**

the group size  $s$ , for which the mse of the estimator is minimal for the given  $n$ ,  $p.tr$  or the group size  $s$  for which bias restriction  $biasrest$  is just not violated, and for this particular group size  $s$ : a list containing:

<code>varp</code>	the variance of the estimator
<code>mse</code>	the mean square error of the estimator
<code>bias</code>	the bias of the estimator
<code>exp</code>	the expected value of the estimator

**Author(s)**

Frank Schaarschmidt

**References**

Swallow WH (1985) Group testing for estimating infection rates and probabilities of disease transmission. *Phytopathology* Vol.75, N.8, 882-889.

**See Also**

[nDesign](#), [sDesign](#) for choice of the binomial group testing design according to the power in a hypothesis test

## Examples

```
### Compare table 1 in Swallow(1985),885:  
estDesign(n=10, smax=100, p.tr=0.001)  
estDesign(n=10, smax=100, p.tr=0.01)  
estDesign(n=25, smax=100, p.tr=0.05)  
estDesign(n=40, smax=100, p.tr=0.25)  
estDesign(n=200, smax=100, p.tr=0.3)
```

---

gt.control

*Auxiliary for Controlling Group Testing Regression*

---

## Description

Auxiliary function to control fitting parameters of EM algorithm used internally in [gtreg.mp](#) and [EM.mp](#) or [gtreg](#) and [gtreg.fit](#) with method = "Xie".

## Usage

```
gt.control(tol = 0.005, n.gibbs = 1000, n.burnin = 20,  
maxit = 500, trace = FALSE, time = TRUE)
```

## Arguments

tol	convergence criterion
n.gibbs	the Gibbs sample size to be used in each E step for the EM algorithm (default is 1000), for matrix pooling or simple pooling with retests (Dorfman's procedure)
n.burnin	the number of samples in the burn-in period (default is 20), for matrix pooling or simple pooling with retests (Dorfman's procedure)
maxit	maximal number of iterations in the EM algorithm
trace	logical indicating if output should be printed for each iteration, defaults to FALSE
time	logical indicating if the length of time for the model fitting should be printed, defaults to TRUE

## Value

A list with components named as the arguments

**Examples**

```
# The default settings:
gt.control()
```

---

 gtreg
 

---

*Fitting Group Testing Models*


---

**Description**

gtreg is a function to fit the group testing regression model specified through a symbolic description of the linear predictor and descriptions of the group testing setting.

**Usage**

```
gtreg(formula, data, groupn, retest = NULL, sens = 1,
      spec = 1, linkf = c("logit", "probit", "cloglog"),
      method = c("Vansteelandt", "Xie"), sens.ind = NULL,
      spec.ind = NULL, start = NULL, control = gt.control(...), ...)
```

```
gtreg.fit(Y, X, groupn, sens, spec, linkf, start=NULL)
```

```
EM(Y, X, groupn, sens, spec, linkf, start = NULL,
   control = gt.control())
```

```
EM.ret(Y, X, groupn, ret, sens, spec, linkf, sens.ind,
       spec.ind, start = NULL, control = gt.control())
```

**Arguments**

formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	an optional data frame, list or environment (or object coercible by <code>as.data.frame</code> ) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>gtreg</code> is called.
groupn	a vector, list or data frame of the group numbers that designates individuals to groups.
retest	a vector, list or data frame of individual retest results for Dorfman's retesting procedure. Default value is NULL for no retests. See 'Details' for how to code it.
sens	sensitivity of the test, set to be 1 by default.
spec	specificity of the test, set to be 1 by default.

<code>sens.ind</code>	sensitivity of the individual retests, set to be equal to <code>sens</code> if not specified otherwise.
<code>spec.ind</code>	specificity of the individual retests, set to be equal to <code>spec</code> if not specified otherwise.
<code>linkf</code>	a character string specifying one of the three link functions for a binomial model: "logit" (default) or "probit" or "cloglog".
<code>method</code>	The method to fit the model, must be one of "Vansteelandt" (default) or "Xie". The option "Vansteelandt" finds estimates by directly maximizing the likelihood function based on the group responses while the option "Xie" uses the EM algorithm to maximize the likelihood function in terms of the unobserved individual responses.
<code>start</code>	starting values for the parameters in the linear predictor.
<code>control</code>	a list of parameters for controlling the fitting process in method "Xie". See the documentation for <a href="#">gt.control</a> for details.
<code>Y</code>	For <code>gtreg.fit</code> , <code>EM</code> and <code>EM.ret</code> : the vector of the group responses.
<code>X</code>	For <code>gtreg.fit</code> , <code>EM</code> and <code>EM.ret</code> : the design matrix of the covariates.
<code>ret</code>	For <code>EM.ret</code> : a vector containing individual retest results.
<code>...</code>	arguments to be passed by default to <a href="#">gt.control</a> : see argument <code>control</code>

## Details

A typical predictor has the form `groupresp ~ covariates` where `response` is the (numeric) group response vector and `covariates` is a series of terms which specifies a linear predictor for individual responses. Note that it is actually the unobserved individual responses, not the observed group responses, which are modeled by the covariates here. In `groupresp`, a 0 denotes a negative response and a 1 denotes a positive response, where the probability of an individual positive response is being modeled directly. A terms specification of the form `first + second` indicates all the terms in `first` together with all the terms in `second` with duplicates removed. The terms in the formula will be re-ordered so that main effects come first, followed by the interactions, all second-order, all third-order and so on; to avoid this pass a terms object as the formula.

A specification of the form `first:second` indicates the set of terms obtained by taking the interactions of all terms in `first` with all terms in `second`. The specification `first*second` indicates the cross of `first` and `second`. This is the same as `first + second + first:second`.

Three workhorse functions `gtreg.fit`, `EM` and `EM.ret`, where the first corresponds to Vansteelandt's method and the last two corresponds to Xie's method, are called by `gtreg` to carry out the model fitting. The `gtreg.fit` function uses the `optim` function with default method "Nelder-Mead" to maximize the likelihood function of the observed group responses. If this optimization method produces a Hessian matrix of all zero elements, the "SANN" method in `optim` is employed to find the coefficients and Hessian matrix. For "SANN" method, the number of iterations in `optim` is set to be 10000.

The `EM` and `EM.ret` function apply Xie's EM algorithm to the likelihood function written in terms of the unobserved individual responses; the functions use `glm.fit` to update the parameter estimates within each M step. The `EM` function is used when there are no retests and `EM.ret` is used when individual retests are available. Thus, within `retest`, individual observations in observed positive groups are 0 (negative) or 1 (positive); the remaining individual observations are NAs meaning that

no retest is performed for them. The EM.retest function uses Gibbs sampling in each E-step to estimate the conditional probabilities. Retests cannot be used with Vansteelandt's method; a warning message will be given in this case, and the individual retests will be ignored in the model fitting. There could be slight differences in the estimates between the Vansteelandt's and Xie's methods (when retests are not available) due to different convergence criteria.

The data used here should be in the form of simple pooling - meaning that each individual appears in exactly one pool. When only the group responses are observed, the null degrees of freedom are the number of groups minus 1 and the residual degrees of freedom are the number of groups minus the number of parameters. When individual retests are observed too, it is an open research question for what the degrees of freedom should be; the degrees of freedom are given currently as if only the group responses are observed.

For the background on the use of optim, see help(optim).

## Value

gtreg returns an object of class "gt". See later in this section. The function summary (i.e., [summary.gt](#)) can be used to obtain or print a summary of the results. The group testing functions predict (i.e., [predict.gt](#)) and residuals (i.e., [residuals.gt](#)) can be used to extract various useful features of the value returned by gtreg. An object of class "gt" is a list containing at least the following components:

coefficients	a named vector of coefficients
hessian	estimated Hessian matrix of the negative log likelihood function, serves as an estimate of the information matrix
residuals	the response residuals, difference of the observed group responses and the fitted group responses.
fitted.values	the fitted mean values of group responses.
deviance	the deviance between the fitted model and the saturated model.
aic	Akaike's An Information Criterion, minus twice the maximized log-likelihood plus twice the number of coefficients
Gibbs.sample.size	If retest is used, this is the number of Gibbs samples generated in each E-step.
null.deviance	The deviance for the null model, comparable with deviance. The null model will include only the intercept if there is one in the model.
counts	For Vansteelandt's method: the number of iterations in optim; For Xie's method: the number of iterations in the EM algorithm.
df.residual	the residual degrees of freedom.
df.null	the residual degrees of freedom for the null model.
z	the vector of group responses.
call	the matched call.
formula	the formula supplied.
terms	the terms object used.
method	the method ("Vansteelandt" or "Xie") used to fit the model.
link	the link function used in the model.

**Author(s)**

Boan Zhang

**References**

Xie, M. (2001), Regression analysis of group testing samples, *Statistics in Medicine*, 20, 1957-1969.

Vansteelandt, S., Goetghebeur, E., and Verstraeten, T. (2000), Regression models for disease prevalence with diagnostic tests on pools of serum samples, *Biometrics*, 56, 1126-1133.

**See Also**

[summary.gt](#), [predict.gt](#) and [residuals.gt](#) for gt methods. [gtreg.mp](#) for the group testing regression model in the matrix pooling setting.

**Examples**

```
data(hivsurv)

fit1 <- gtreg(formula = groupres ~ AGE + EDUC., data = hivsurv,
             groupn = gnum, sens = 0.9, spec = 0.9, method = "Xie")
fit1

## --- Continuing the Example from '?sim.gt':

set.seed(46)
gt.data <- sim.gt(par = c(-12, 0.2), sample.size = 700, group.size = 5)
fit2 <- gtreg(formula = gres ~ x, data = gt.data, groupn = groupn)
fit2

set.seed(21)
gt.data <- sim.gt(par = c(-12, 0.2), sample.size = 700, group.size = 6,
                sens = 0.95, spec = 0.95, sens.ind = 0.98, spec.ind = 0.98)
fit1 <- gtreg(formula = gres ~ x, data = gt.data, groupn = groupn,
             retest = retest, method = "X", sens = 0.95, spec = 0.95, sens.ind = 0.98,
             spec.ind = 0.98, trace = TRUE)
summary(fit1)
```

**Description**

gtreg.mp is a function to fit the group testing regression model in the matrix pooling setting specified through a symbolic description of the linear predictor and descriptions of the group testing setting.

**Usage**

```
gtreg.mp(formula, data, coln, rown, arrayn, retest = NULL,
  sens = 1, spec = 1, linkf = c("logit", "probit", "cloglog"),
  sens.ind = NULL, spec.ind = NULL, start = NULL,
  control = gt.control(...), ...)
```

```
EM.mp(col.resp, row.resp, X, coln, rown, sqn, ret, sens, spec,
  linkf, sens.ind, spec.ind, start = NULL, control = gt.control())
```

**Arguments**

formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	an optional data frame, list or environment (or object coercible by <code>as.data.frame</code> to a data frame) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>gtreg.mp</code> is called.
coln	a vector, list or data frame that specifies column group number for each sample
rown	a vector, list or data frame that specifies row group number for each sample
arrayn	a vector, list or data frame that specifies array number for each sample
retest	a vector, list or data frame of individual retest results. A 0 denotes negative and 1 denotes positive. A NA denotes that no retest is performed for that individual. Default value is NULL for no retests.
sens	sensitivity of the group tests, set to be 1 by default.
spec	specificity of the group tests, set to be 1 by default.
sens.ind	sensitivity of the individual retests, set to be equal to <code>sens</code> if not specified otherwise.
spec.ind	specificity of the individual retests, set to be equal to <code>spec</code> if not specified otherwise.
linkf	a character string specifying one of the three link functions for a binomial model: "logit" (default) or "probit" or "cloglog".
start	starting values for the parameters in the linear predictor.
control	a list of parameters for controlling the fitting process. See the documentation for <a href="#">gt.control</a> for details.
col.resp	For <code>EM.mp</code> : vector of group responses of column pools for all samples. 0 denotes negative and 1 denotes positive.
row.resp	For <code>EM.mp</code> : vector of group responses of row pools for all samples. 0 denotes negative and 1 denotes positive.
X	For <code>EM.mp</code> : the design matrix of the covariates.
sqn	For <code>EM.mp</code> : a vector that specifies array number
ret	For <code>EM.mp</code> : a vector containing individual retest results
...	arguments to be passed to <a href="#">gt.control</a> : see argument <code>control</code>

## Details

With matrix pooling, individual samples are placed in a matrix-like grid where samples are pooled within each row and within each column. This leads to two kinds of group responses: row and column group responses. Thus, a typical predictor has the form `cbind(col.resp, row.resp) ~ covariates` where `col.resp` is the (numeric) column group response vector and `row.resp` is the (numeric) row group response vector. The `covariates` term is a series of terms which specifies a linear predictor for individual responses. Note that it is actually the unobserved individual responses, not the observed group responses, which are modeled by the `covariates`. In `col.resp` and `row.resp`, a 0 denotes a negative response and 1 denotes a positive response, where the probability of an individual positive response is being modeled directly. A terms specification of the form `first + second` indicates all the terms in `first` together with all the terms in `second` with duplicates removed. The terms in the formula will be re-ordered so that main effects come first, followed by the interactions, all second-order, all third-order and so on; to avoid this pass a terms object as the formula.

A specification of the form `first:second` indicates the set of terms obtained by taking the interactions of all terms in `first` with all terms in `second`. The specification `first*second` indicates the cross of `first` and `second`. This is the same as `first + second + first:second`.

`EM.mp` is the workhorse function. It applies Xie's EM algorithm to the likelihood function written in terms of the unobserved individual responses. In each E step, the Gibbs sampling technique is used to estimate the conditional probabilities. Because of the large number of Gibbs samples needed to achieve convergence, the model fitting process could be quite slow, especially when multiple positive rows and columns are observed. In this case, we can either increase the Gibbs sample size to help achieve convergence or loosen the convergence criteria by increasing `tol` at the expense of perhaps poorer estimates. If follow-up retests are performed, the retest results going into the model will help achieve convergence faster with the same Gibbs sample size and convergence criteria. In each M step, we use `glm.fit` to update the parameter estimates

## Value

`gtreg.mp` returns an object of class `"gt.mp"` which inherits from the class `"gt"`. See later in this section. The function `summary` (i.e., `summary.gt.mp`) can be used to obtain or print a summary of the results. The group testing function `predict` (i.e., `predict.gt`) can be used to make predictions on `"gt.mp"` objects. An object of class `"gt.mp"` is a list containing at least the following components:

<code>coefficients</code>	a named vector of coefficients.
<code>hessian</code>	estimated Hessian matrix of the negative log likelihood function, serves as an estimate of the information matrix
<code>counts</code>	the number of iterations performed in the EM algorithm.
<code>Gibbs.sample.size</code>	the number of Gibbs samples generated in each E step.
<code>call</code>	the matched call.
<code>formula</code>	the formula supplied.
<code>terms</code>	the terms object used.
<code>link</code>	the link function used in the model.

**Author(s)**

Boan Zhang

**References**

Xie, M. (2001), Regression analysis of group testing samples, *Statistics in Medicine*, 20, 1957-1969.

**See Also**

[summary.gt.mp](#) and [predict.gt](#) for `gt.mp` methods. [gtreg](#) for the group testing regression model in the simple pooling setting.

**Examples**

```
## --- Continuing the Example from '?sim.mp':
# 5*6 and 4*5 matrix
set.seed(9128)
sa1a<-sim.mp(par=c(-7,0.1), n.row=c(5,4), n.col=c(6,5),
  sens=0.95, spec=0.95)
sa1<-sa1a$dframe

## Not run:
fit1 <- gtreg.mp(formula = cbind(col.resp, row.resp) ~ x, data = sa1,
  coln = coln, rown = rown, arrayn = arrayn,
  sens = 0.95, spec = 0.95, n.gibbs = 2000, trace = TRUE)

fit1
summary(fit1)

## End(Not run)

## Here is an example of how long this fitting process may take. For the
## following simulated data, it takes a computer with 2.2GHZ processor and
## 3GB RAM about 6 minutes to achieve convergence.
set.seed(9012)
sa2a<-sim.mp(par=c(-7,0.1), n.row=c(10,10,10,10), n.col=c(10,10,10,10),
  sens=0.95, spec=0.95)
sa2<-sa2a$dframe

## Not run:
fit2 <- gtreg.mp(formula = cbind(col.resp, row.resp) ~ x, data = sa2,
  coln = coln, rown = rown, arrayn = arrayn, retest = retest,
  sens = 0.95, spec = 0.95, start = c(-7, 0.1))

fit2
summary(fit2)

## End(Not run)
```

---

`hivsurv`*Data from an HIV surveillance project*

---

### Description

The `hivsurv` data set comes from an HIV surveillance project discussed in Verstraeten et al. (1998) and Vansteelandt et al. (2000). The purpose of the study was to estimate the HIV prevalence among pregnant Kenyan women in four rural locations of the country, using both individual and group testing responses. Blood tests were administered to each participating woman, and 4 covariates were obtained on each woman. Because the original group responses are unavailable, individuals are artificially put into groups of 5 here to form group responses. Only the 428 complete observations are given here.

### Usage

```
data(hivsurv)
```

### Format

A data frame with 428 observations on the following 8 variables.

DATE the date when each sample was collected

PAR. parity (number of children)

AGE age (in years)

MA.ST. marital status (1: single; 2: married (polygamous); 3: married (monogamous); 4: divorced; 5: widow)

EDUC. highest attained education level (1: no schooling; 2: primary school; 3: secondary school; 4: higher)

HIV individual response of HIV diagnosis (0: negative; 1: positive)

gnum the group number that designates individuals into groups

groupres the group response calculated from artificially formed groups

### Source

Verstraeten, T., Farah, B., Duchateau, L., Matu, R. (1998), Pooling sera to reduce the cost of HIV surveillance: a feasibility study in a rural Kenyan district, *Tropical Medicine & International Health*, 3, 747-750.

Vansteelandt, S., Goetghebeur, E., and Verstraeten, T. (2000), Regression models for disease prevalence with diagnostic tests on pools of serum samples, *Biometrics*, 56, 1126-1133.

### Examples

```
data(hivsurv)
```

```
str(hivsurv)
```

---

nDesign

---

*Iterate Sample Size in One Parameter Group Testing*


---

### Description

Increasing number of groups (assays, bulk samples) for a fixed group size in a binomial group testing design, until a pre-specified power is achieved. At the same time, bias of the estimator is controlled. A hypothetical threshold proportion  $p_{\text{hyp}}$  and the absolute difference  $\delta$  to be detected have to be specified.

### Usage

```
nDesign(nmax, s, delta, p.hyp, conf.level = 0.95,
power = 0.8, alternative = "two.sided", method = "CP", biasrest = 0.05)
```

### Arguments

nmax	either a single integer giving the maximal number of individuals $n$ allowed in the iteration, or a vector of two integers giving the range of $n$ in which power shall be iterated
s	integer, fixed group size (number of units per group)
delta	absolute difference between the threshold and the true proportion which shall be detectable with specified power
p.hyp	threshold proportion to test against in the hypothesis, specify as a value between 0 and 1
conf.level	confidence level of the decision, default is 0.95
power	level of power to be achieved to be specified as a probability between 0 and 1
alternative	character string, defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' calculates the probability that $p_{\text{hyp}}$ is excluded by an upper confidence limit for a true proportion $p_{\text{hyp}} - \delta$ , 'greater' calculates the probability that $p_{\text{hyp}}$ is excluded by a lower confidence limit for a true proportion $p_{\text{hyp}} + \delta$ . 'two.sided' calculates $\min(\text{power}(p_{\text{hyp}} - \delta, p_{\text{hyp}} + \delta))$ for a two.sided CI, thus can result in much lower power.
method	character string specifying the CI method to be used for evaluation, see argument method in bgtCI
biasrest	value between 0 and 1 specifying the absolute bias maximally allowed

### Details

The power of a confidence interval here is defined as the probability that a confidence interval or limit excludes the threshold parameter ( $p_{\text{hyp}}$ ) of the hypothesis.

This function increases the number of groups (i.e. number of observations or assays in binomial group testing) until a pre-specified power is reached or the maximal number of groups  $n_{\text{max}}$  (specified in the function call) is reached. Since the power does not increase monotone with increasing

n for binomial proportions but oscillates between local maxima and minima, the simple iteration given here will generally result in selecting those n, for which the given CI method shows a local minimum of coverage if the null hypothesis is true. Bias decreases monotone with increasing the number of groups (if other parameters are fixed) The resulting Problems of choosing a number of groups which results in satisfactory power, are solved in the following manner:

In case that the pre-specified power can be reached within the given range of n, the smallest n will be returned for which at least this power is reached, as well as the actual power for this n.

In case that the pre-specified power is not reached within the given value, that n is returned for which maximal power is achieved, and the corresponding value of power.

In case that biasrestriction is violated even for the largest n within the given range of n, simply that n will be returned for which power was largest in the given range. Due to discreteness of binomial distribution, power can be zero for one-sided hypothesis over a range of n.

The power can be identical for different methods, depending on the particular combination of n, s, p.hyp, conf.level.

Especially for large n, the calculation time may become large (particularly for Blaker). Then only the range of sample size which is of interest can be specified in nmax, f.e. as: nmax=c(150,300). Alternatively, the function bgtPower might be used instead to calculate power and bias only for some particular combinations of n, s, delta, p.hyp,... .

## Value

A list containing

nout	the number of groups (assays or bulk samples) necessary reach the power with the specified parameters
powerout	the power for the specified parameters and selected number of groups n
biasout	the bias for the specified parameters and the selected number of groups n

and a number of values specified in the function call or produced in the iteration, which are only necessary to apply the function plot() on objects of class 'nDesign'

## Author(s)

Frank Schaarschmidt

## References

*Schaarschmidt F (2007)*. Experimental design for one-sided confidence intervals or hypothesis tests in binomial group testing. *Communications in Biometry and Crop Science* 2 (1), 32-40. <http://agrobiol.sggw.waw.pl/cbcs/>

*Swallow WH (1985)*. Group testing for estimating infection rates and probabilities of disease transmission. *Phytopathology* Vol.75, N.8, 882-889.

**See Also**

`plot.nDesign` to plot the iteration of this function

`bgtPower`: calculation of power and bias depending on  $n$ ,  $s$ ,  $\delta$ ,  $p_{\text{hyp}}$ ,  $\text{conf.level}$ , `method`

`sDesign`: function for stepwise increasing group size  $s$  for a given  $n$  in order to achieve sufficient power within a biasrestriction `estDesign`: function to choose group size  $s$  according to the minimal mse of the estimator, as given in Swallow (1985)

**Examples**

```
## Assume one aims to show that a proportion is smaller
## 0.005 (i.e. 0.5 per cent) with a power
## of 0.80 (i.e. 80 per cent) if the unknown proportion
## in the population is 0.003 (i.e. 0.3 per cent),
## thus, to detect a delta of 0.002.
## The Clopper Pearson CI shall be used.
## The maximal group size because of limited
## sensitivity of assay might be s=20 and we
## can only afford to perform maximally 100 assays:
```

```
nDesign(nmax=100, s=20, delta=0.002, p.hyp=0.005,
        alternative="less", method="CP", power=0.8)
```

```
## A power of 80 per cent can not be reached but
## only 30 percent with n=100
## One might accept to show significance only for a
## lower true proportion = 0.001 i.e accepting to be
## able to show significance only if true proportion
## is delta=0.004 smaller than the threshold
```

```
nDesign(nmax=100, s=20, delta=0.004, p.hyp=0.005,
        alternative="less", method="CP", power=0.8)
```

```
test<-nDesign(nmax=100, s=30, delta=0.004, p.hyp=0.005,
             alternative="less", method="CP", power=0.8)
```

```
plot(test)
```

---

plot.bgtDesign

*Plot Results of nDesign or sDesign*

---

**Description**

Plots the results of the iteration performed by `nDesign` or `sDesign` in order to find an experimental design with sufficient power.

**Usage**

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

**Arguments**

x                    An object of class "nDesign" or "sDesign" as can be created by [nDesign](#) and [sDesign](#), respectively.

...                   further arguments to be passed to [plot](#)

**Examples**

```
plot(nDesign(nmax=100, s=30, delta=0.004, p.hyp=0.005,  
          alternative="less", method="Score", power=0.8))
```

---

plot.binDesign                    *Plot Results of binDesign*

---

**Description**

Plot function to visualize the power curve.

**Usage**

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

**Arguments**

x                    an object of class "binDesign, as can be created by `link{binDesign}`"

...                   plot parameters as described in [par](#)

**Value**

A plot.

**See Also**

[binPower](#) for calculation of power

**Examples**

```

# Find a sample size for which the power to reject
# H0: p >= 0.1 in favor of HA: p < 0.1
# is at least 0.9 (90 percent) in case that
# the true proportion is 0.04 (i.e. an absolute delta
# of 0.06 to the threshold proportion p.hyp=0.1)
# The exact one sided Clopper-Pearson CI shall be used
# with default confidence level = 0.95.

sasi<-binDesign( nmax=200, delta=0.06, p.hyp=0.1,
  alternative="less", method="CP", power=0.9)

sasi

# Plot the result

# plot(sasi)

# for larger sample sizes this can be very time consuming.
# Better to use only a smaller range of n then:

sasi<-binDesign( nmax=c(200,300), delta=0.03, p.hyp=0.1,
  alternative="less", method="CP", power=0.9)

plot(sasi)

```

---

pooledBin

*Confidence intervals for a single proportion*


---

**Description**

Calculates confidence intervals for a single proportion based on pooled testing experiments containing various different pool sizes.

**Usage**

```

pooledBin(x, m, n = rep(1, length(x)),
  pt.method = c("bc-mle", "mle", "mir"),
  ci.method = c("skew-score", "bc-skew-score", "score",
    "lrt", "wald", "mir"),
  scale = 1, alpha = 0.05, tol = .Machine$double.eps^0.5)

```

**Arguments**

x	a vector, specifying the observed number of positive pools, among the number of pools tested (n)
m	a vector of pool sizes, must have the same length as, x
n	a vector of the corresponding number of pools of sizes m
pt.method	a character string, specifying the point estimate to compute, with the following options: "bc-mle" bias-corrected MLE, the default; "mle" MLE, and "mir" MIR.
ci.method	a character string, specifying the confidence interval to compute, with options: "skew-score" skewness-corrected, the default, "score" the score, "bc-skew-score" bias- and skewness-corrected "lrt" likelihood ratio test, "wald" wald, and "mir" MIR.
scale	a single numeric, coefficient to scale the point estimates and intervals bounds in the print and summary method ( <a href="#">print.poolbin</a> , <a href="#">summary.poolbin</a> )
alpha	a single numeric, specifying the type-I-error level
tol	accuracy required for iterations in internal functions

**Details**

Note, that the methods "mir" and "wald" can not be recommended, because they return too narrow intervals in relevant situations, "mir" because it ignores the pooling, and "wald" because it relies on simple large sample methods. For computational details and simulation results of the remaining methods, see Biggerstaff (2008).

**Value**

A list with elements

p	the estimated proportion
lcl	the lower confidence limit
ucl	the upper confidence limit
pt.method	the method used for point estimation
ci.method	the method used for interval estimation
alpha	the type-I-error level
x	the numbers of postive pools
m	the size of the pools
n	the numbers of pools with corresponding pool sizes m
scale	Scaling coefficient for the output

**Author(s)**

Brad Biggerstaff

## References

*Biggerstaff BJ (2008): Confidence interval for the difference of proportions estimated from pooled samples. Journal of Agricultural Biological and Environmental Statistics 2008, 13(4):478-496.*

## See Also

[bgtvs](#) to compute exact confidence intervals for one proportion in designs with different pool sizes. Note that [bgtvs](#) can only deal with a limited number of different pool sizes.

[bgtCI](#) to compute exact or asymptotic confidence intervals for one proportion in designs with a common pool size in all pools.

## Examples

```
# Consider an imaginary example, where pools of size
# 1, 5, 10 and 50 are tested, 5 pools of each size
# among the 5 pools with size 1 and 5, no pool is positive,
# while among the 5 pools of size 10 and 50, 1 and 2 positive
# pools are identified, respectively.

x1 <- c(0,0,1,2)
m1 <- c(1,5,10,50)
n1 <- c(5,5,5,5)

pooledBin(x=x1, m=m1, n=n1)
pooledBin(x=x1, m=m1, n=n1, scale=1000)

pooledBin(x=x1, m=m1, n=n1)

summary(pooledBin(x=x1, m=m1, n=n1), scale=1000)

# For another population, tested with the same design, one might find:
# 1 positive result among the pools pooling 5 elements,
# no positive result among the pools pooling 10 elements,
# 4 positive results among the pools pooling 50 elements,

x2<-c(0,1,0,4)
m2 <- c(1,5,10,50)
n2 <- c(5,5,5,5)

pooledBin(x=x2, m=m2, n=n2)

# Some other methods for the confidence bounds:

pooledBin(x=x2, m=m2, n=n2, ci.method="lrt")
pooledBin(x=x2, m=m2, n=n2, ci.method="score")
```

---

pooledBinDiff                      *Confidence intervals for the difference of proportions*

---

### Description

The function calculates confidence intervals for the difference of two proportions based on pooled testing experiments containing various different pool sizes.

### Usage

```
pooledBinDiff(x1, m1, x2, m2, n1 = rep(1, length(x1)),
              n2 = rep(1, length(x2)), pt.method = c("bc-mle", "mle", "mir"),
              ci.method = c("skew-score", "bc-skew-score", "score", "lrt", "wald", "mir"),
              scale = 1, alpha = 0.05, tol = .Machine$double.eps^0.5)
```

### Arguments

x1	a vector, specifying the observed number of positive pools, among the number of pools tested (n1) in population 1
m1	a vector of corresponding pool sizes in population 1, must have the same length as x1
x2	a vector, specifying the observed number of positive pools, among the number of pools tested (n2) in population 2
m2	a vector of corresponding pool sizes in population 2, must have the same length as x2
n1	a vector of the corresponding number of pools with sizes m1
n2	a vector of the corresponding number of pools with sizes m2
pt.method	a character string, specifying the point estimate to compute, with the following options: "bc-mle" bias-corrected MLE, the default; "mle" MLE, and "mir" MIR.
ci.method	a character string, specifying the confidence interval to compute, with options: "skew-score" skewness-corrected, the default, "score" the score, "bc-skew-score" bias- and skewness-corrected "lrt" likelihood ratio test. Further, the options "wald" wald and "mir" MIR are available but are recommended.
scale	a single numeric, coefficient to scale the point estimates and intervals bounds in the print and summary method ( <a href="#">print.poolbindiff</a> , <a href="#">summary.poolbindiff</a> )
alpha	a single numeric, specifying the type-I-error level
tol	accuracy required for iterations in internal functions

### Details

Note, that the methods "mir" and "wald" can not be recommended, because they return too narrow intervals in relevant situations, "mir" because it ignores the pooling, and "wald" because it relies on simple large sample methods. For computational details and simulation results of the remaining methods, see Biggerstaff (2008).

**Value**

A list with elements

d	the estimated difference of proportions
lc1	the lower confidence limit
uc1	the upper confidence limit
pt.method	the method used for point estimation
ci.method	the method used for interval estimation
alpha	the type-I-error level
scale	Scaling coefficient for the output
x1	the numbers of positive pools in population 1
m1	the size of the pools in population 1
n1	the numbers of pools with corresponding pool sizes m1 in population 1
x2	the numbers of positive pools in population 2
m2	the size of the pools in population 2
n2	the numbers of pools with corresponding pool sizes m1 in population 2

**Author(s)**

Brad Biggerstaff

**References**

*Biggerstaff BJ (2008): Confidence interval for the difference of proportions estimated from pooled samples. Journal of Agricultural Biological and Environmental Statistics 2008, 13(4):478-496.*

**Examples**

```
# Consider an imaginary example, where pools of size
# 1, 5, 10 and 50 are tested, 5 pools of each size. The same
# design is applied to estimate the prevalence in
# two populations:
# In population 1, among the 5 pools with size 1 and 5,
# no positive pool is observed,
# while among the 5 pools of size 10 and 50, 1 and 2 positive
# pools are identified, respectively.

# In population 2, 1 positive result is observed
# among the 5 pools each pooling 5 elements,
# no positive result among the pools pooling 10 elements,
# 4 positive results among the pools pooling 50 elements,

x1 <- c(0,0,1,2)
m <- c(1,5,10,50)
n <- c(5,5,5,5)
```

```

x2<-c(0,1,0,4)

pooledBinDiff(x1=x1, m1=m, x2=x2, m2=m, n1=n, n2=n)

summary(pooledBinDiff(x1=x1, m1=m, x2=x2, m2= m, n1=n, n2=n), scale=1000)

# Compare recommended methods:

pooledBinDiff(x1=x1, m1=m, x2=x2, m2= m, n1=n, n2=n, pt.method="mle",
  ci.method="lrt")

pooledBinDiff(x1=x1, m1=m, x2=x2, m2= m, n1=n, n2=n, pt.method="mle",
  ci.method="score")

pooledBinDiff(x1=x1, m1=m, x2=x2, m2= m, n1=n, n2=n, pt.method="mle",
  ci.method="skew-score")

```

---

predict.gt

---

*Predict Method for Group Testing Model Fits*


---

## Description

Obtains predictions for individual observations and optionally estimates standard errors of those predictions from objects of class "gt" or "gt.mp" returned by gtmreg and gtmreg.mp, respectively.

## Usage

```

## S3 method for class 'gt'
predict(object, newdata, type = c("link", "response"),
  se.fit = FALSE, conf.level = NULL, na.action = na.pass, ...)

```

## Arguments

object	a fitted object of class "gt" or "gt.mp".
newdata	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.
type	the type of prediction required. The option "link" is on the scale of the linear predictors; the alternative "response" is on the scale of the response variable. Thus for the binomial model the "link" predictions are of log-odds (probabilities on logit scale) and type = "response" gives the predicted probabilities.
se.fit	logical switch indicating if standard errors are required.
conf.level	confidence level of the interval of the predicted values.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
...	currently not used

**Details**

If newdata is omitted the predictions are based on the data used for the fit. When newdata is present and contains missing values, how the missing values will be dealt with is determined by the na.action argument. In this case, if na.action = na.omit omitted cases will not appear, whereas if na.action = na.exclude they will appear (in predictions and standard errors), with value NA. See also napredict.

**Value**

If se = FALSE, a vector or matrix of predictions. If se = TRUE, a list with components

fit	Predictions
se.fit	Estimated standard errors
lower	Lower bound of the confidence interval if calculated
upper	Upper bound of the confidence interval if calculated

**Author(s)**

Boan Zhang

**Examples**

```
data(hivsurv)

fit1 <- glogit(formula = groupres ~ AGE + EDUC., data = hivsurv,
  groupn = gnum, sens = 0.9, spec = 0.9, linkf = "logit", method = "V")
pred.data <- data.frame(AGE = c(15, 25, 30), EDUC. = c(1, 3, 2))
predict(object = fit1, newdata = pred.data, type = "link", se.fit = TRUE)
predict(object = fit1, newdata = pred.data, type = "response",
  se.fit = TRUE, conf.level = 0.9)
predict(object = fit1, type = "response", se.fit = TRUE, conf.level = 0.9)
```

---

print.bgt

*Print Functions for Group Testing CIs and Tests for One Proportion*

---

**Description**

Print objects of class "bgtCI", "bgtTest", "bgtvs", "binCI", and "binTest"

**Usage**

```
## S3 method for class 'bgtCI'  
print(x, ...)  
## S3 method for class 'binCI'  
print(x, ...)  
## S3 method for class 'bgtTest'  
print(x, ...)  
## S3 method for class 'binTest'  
print(x, ...)  
## S3 method for class 'bgtvs'  
print(x, ...)
```

**Arguments**

x                    an object of the corresponding class  
...                   currently only digits is passed to signif for appropriate rounding

**Value**

A print out.

---

print.bgtDesign            *Print Functions for nDesign and sDesign*

---

**Description**

Print function for results of functions [nDesign](#), [nDesign](#).

**Usage**

```
## S3 method for class 'nDesign'  
print(x, ...)  
## S3 method for class 'sDesign'  
print(x, ...)
```

**Arguments**

x                    An object of class "nDesign" or "sDesign"  
...                   currently only digits is passed to signif for appropriate rounding

---

print.binDesign      *Print Function for binDesign*

---

**Description**

Print objects of class "binDesign"

**Usage**

```
## S3 method for class 'binDesign'
print(x, ...)
```

**Arguments**

x                    an object of class "binDesign"  
 ...                  currently only digits is passed to signif for appropriate rounding

---

print.gt.mp            *Print methods for objects of classes "gt" and "gt.mp"*

---

**Description**

Print methods for objects of classes "gt" and "gt.mp"

**Usage**

```
## S3 method for class 'gt.mp'
print(x, digits = max(3, getOption("digits") - 3), ...)
## S3 method for class 'gt'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

**Arguments**

x                    objects of class "gt" and "gt.mp", respectively  
 digits              the digits for rounding  
 ...                  currently not used

**Value**

A print out

---

```
print.poolbindiff      Print methods for classes "poolbin" and "poolbindiff"
```

---

**Description**

Print methods for objects of classes "poolbin" and "poolbindiff"

**Usage**

```
## S3 method for class 'poolbin'
print(x, scale = x$scale, ...)
## S3 method for class 'poolbindiff'
print(x, scale = x$scale, ...)
```

**Arguments**

x	An object of class "poolbin" or "poolbindiff" ( <a href="#">pooledBin</a> , <a href="#">pooledBinDiff</a> )
scale	A coefficient to scale the point estimate and interval bounds
...	further arguments to be passed to print

**Value**

A print out.

---

```
print.summary.gt      Print Functions for summary.gt.mp and summary.gt
```

---

**Description**

Print function for objects obtained by calling [summary.gt.mp](#) and [summary.gt](#)

**Usage**

```
## S3 method for class 'summary.gt.mp'
print(x, digits = max(3, getOption("digits") - 3),
      signif.stars = getOption("show.signif.stars"), ...)
## S3 method for class 'summary.gt'
print(x, digits = max(3, getOption("digits") - 3),
      signif.stars = getOption("show.signif.stars"), ...)
```

**Arguments**

x	an object of class "summary.gt.mp" or "summary.gt"
digits	digits for rounding
signif.stars	logical, indicating whether significance stars should be shown
...	further arguments to be passed to printCoefmat

**Value**

A print out.

---

residuals.gt

*Extract Model Residuals From a Fitted Group Testing Model*

---

**Description**

`residuals.gt` is a function which extracts model residuals from objects of class "gt" returned by [gtreg](#).

**Usage**

```
## S3 method for class 'gt'  
residuals(object, type = c("deviance", "pearson", "response"),...)
```

**Arguments**

<code>object</code>	an object of class "gt" from which the model residuals are to be extracted.
<code>type</code>	the type of residuals which should be returned. The alternatives are: "deviance" (default), "pearson", and "response".
<code>...</code>	currently not used

**Value**

Residuals of group responses extracted from the object `object`.

**Author(s)**

Boan Zhang

**Examples**

```
data(hivsurv)  
  
fit1 <- gtreg(formula = groupres ~ AGE * EDUC., data = hivsurv, groupn = gnum,  
             linkf = "probit")  
residuals.gt(object = fit1, type = "pearson")  
residuals.gt(object = fit1, type = "deviance")
```

sDesign

*Iterate Group Size for a One-Parameter Group Testing Problem***Description**

Increasing of group size  $s$  for a fixed number of groups  $n$  in a binomial group testing design, until a pre-specified power is achieved. Control of bias during iteration. A hypothetical threshold proportion  $p.hyp$  and the absolute difference  $\delta$  to be detected have to be specified.

**Usage**

```
sDesign(n, smax, delta, p.hyp, conf.level = 0.95,
        power = 0.8, alternative = "two.sided", method = "CP", biasrest = 0.05)
```

**Arguments**

<code>n</code>	integer, fixed sample size (number of assays)
<code>smax</code>	integer, maximal group size allowed in planning of the design
<code>delta</code>	absolute difference between the threshold and the true proportion which shall be detectable with specified power
<code>p.hyp</code>	threshold proportion to test against in the hypothesis, specify as a value between 0 and 1
<code>conf.level</code>	confidence level of the decision, default is 0.95
<code>power</code>	level of power to be achieved to be specified as a probability between 0 and 1
<code>alternative</code>	character string, defining the alternative hypothesis, either 'two.sided', 'less' or 'greater' where 'less' calculates the probability that $p.hyp$ is excluded by an upper confidence limit for a true proportion $p.hyp - \delta$ , 'greater' calculates the probability that $p.hyp$ is excluded by a lower confidence limit for a true proportion of $p.hyp + \delta$ . 'two.sided' calculates $\min(\text{power}(p.hyp - \delta, p.hyp + \delta))$ for a two.sided CI, thus can result in much lower power. Note that coverage probability and power are not necessarily symmetric for upper and lower bound of binomial CI.
<code>method</code>	character string specifying the CI method to be used for evaluation, see argument method in <a href="#">bgtCI</a>
<code>biasrest</code>	value between 0 and 1 specifying the absolute bias maximally allowed

**Details**

The power of a confidence interval here is defined as the probability that a confidence interval or limit excludes the threshold parameter ( $p.hyp$ ) of the hypothesis. This function increases size of groups (number of units in a bulk sample) until a pre-specified power is reached. Since the power does not increase monotone with increasing  $s$  for binomial proportions but oscillates between local maxima and minima, the simple iteration given here will generally result in selecting those  $s$ , for which the given CI method shows a local minimum of coverage if the null hypothesis is true. Since the positive bias of the estimator in group testing increases with increasing group size, it is checked

whether bias is smaller than a pre-specified value (bias.rest). If bias violates this restriction for a given combination n, s, delta, s will not be further increased the actual power of the last acceptable group size s is returned.

### Value

A list containing:

sout	the group size necessary to met the conditions
powerout	the exact power for the specified parameters and the group size
biasout	the bias for the specified parameters and the iterated group size

and a number of values specified in the function call or produced in the iteration.

### References

*Schaarschmidt F (2007) Experimental design for one-sided confidence intervals or hypothesis tests in binomial group testing. Communications in Biometry and Crop Science 2 (1), 32-40. <http://agrobiol.sggw.waw.pl/cbcs/>*

*Swallow WH (1985) Group testing for estimating infection rates and probabilities of disease transmission. Phytopathology 75 (8), 882-889.*

### See Also

[plot.sDesign](#) can be used to plot th iteration of this function

[bgtPower](#): calculation of power and bias depending on n, s, delta, p.hyp, conf.level, method  
[nDesign](#): function to iteratively optimize sample size(number of groups)n for a given group size s  
[estDesign](#): function to choose group size s according to the minimal mse of the estimator, as given in Swallow (1985)

### Examples

```
## Assume that objective is to show that a proportion
## is smaller than 0.005 (i.e. 0.5%) with a
## power of 0.80 (i.e. 80%) if the unknown proportion
## in the population is 0.003 (i.e. 0.3%), thus a
## delta = 0.002 shall be detected. A 95-per-cent
## Clopper-Pearson CI (corresponding to an exact test)
## shall be used. The maximal number of groups might
## be 30 where the assay sensitivity is not limited
## until groupsize s = 100.
```

```
sDesign(smax=100, n=30, delta=0.002, p.hyp=0.005,
  alternative="less", method="CP", power=0.8)
```

```
## One might accept to detect delta=0.004,
## i.e. accepting to reject H0: p>=0.005 with
## power 80 per cent when the true proportion is 0.001:
```

```
sDesign(smax=100, n=30, delta=0.004, p.hyp=0.005,
```

```

alternative="less", method="CP", power=0.8)

sDesign(smax=100, n=30, delta=0.004, p.hyp=0.005,
alternative="less", method="Score", power=0.8)

```

---

sim.gt

*Simulation Function for Group Testing Data*


---

### Description

Simulates data in group testing form ready to be fit by `gtreg`.

### Usage

```

sim.gt(x = NULL, gshape = 20, gscale = 2, par,
linkf = c("logit", "probit", "cloglog"),
sample.size, group.size, sens = 1, spec = 1,
sens.ind = NULL, spec.ind = NULL)

```

### Arguments

<code>x</code>	a matrix of user-submitted covariates to simulate the data with, defaults to NULL in which case a gamma distribution is used to generate the covariates automatically
<code>gshape</code>	shape parameter of gamma distribution, must be non-negative, set to be 20 by default
<code>gscale</code>	scale parameter of gamma distribution, must be strictly positive, set to be 2 by default
<code>par</code>	the true coefficients in the linear predictor
<code>sample.size</code>	sample size of simulated data
<code>linkf</code>	a character string specifying one of the three link functions to be used: "logit" (default) or "probit" or "cloglog"
<code>group.size</code>	group size in pooling individual samples
<code>sens</code>	sensitivity of the group tests, set to be 1 by default.
<code>spec</code>	specificity of the group tests, set to be 1 by default.
<code>sens.ind</code>	sensitivity of the individual retests, set to be equal to <code>sens</code> if not specified otherwise.
<code>spec.ind</code>	specificity of the individual retests, set to be equal to <code>spec</code> if not specified otherwise.

## Details

sim.gt generates group testing data in simple pooling form. The covariates are either specified by the `x` argument or they are generated from a gamma distribution with a given `gshape` and `gscale`. The individual probabilities are calculated from the covariates, the coefficients given in `par`, and the link function specified through `linkf`. The true binary individual responses are then simulated from the individual probabilities. The true group responses are found from the individual responses within the groups (i.e., if at least one response is positive, the group is positive; otherwise, the group response is negative). Finally, the observed group responses are simulated using the given `sens` and `spec`. Individual retests are simulated from `sens.ind` and `spec.ind` for samples in observed positive groups. Note that with a given `group.size`, the last group may have less individuals.

## Value

sim.gt returns a data frame with the following columns:

gres	the group response
x	the covariate
groupn	the group number
ind	the actual individual response
retest	the results of individual retests

## Author(s)

Boan Zhang

## See Also

[gtreg](#), [gtreg.mp](#)

## Examples

```
set.seed(46)
gt.data <- sim.gt(par = c(-12, 0.2), sample.size = 700, group.size = 5)

x1 <- sort(runif(100, 0, 30))
x2 <- rgamma(100, shape=17, scale=1.5)
gt.data <- sim.gt(x = cbind(x1,x2), par=c(-14, 0.2, 0.3),
  group.size = 4, sens = 0.98, spec = 0.98)
```

---

sim.mp	<i>Simulation Function for Group Testing Data with Matrix Pooling Design</i>
--------	--

---

### Description

Simulates data in group testing form ready to be fit by `gtreg.mp`.

### Usage

```
sim.mp(x = NULL, gshape = 20, gscale = 2, par,
       linkf = c("logit", "probit", "cloglog"),
       n.row, n.col, sens = 1, spec = 1,
       sens.ind = NULL, spec.ind = NULL)
```

### Arguments

<code>x</code>	a matrix of user-submitted covariates to simulate the data with, defaults to NULL in which case a gamma distribution is used to generate the covariates automatically.
<code>gshape</code>	shape parameter of gamma distribution, must be non-negative, set to be 20 by default
<code>gscale</code>	scale parameter of gamma distribution, must be strictly positive, set to be 2 by default
<code>par</code>	the true coefficients in the linear predictor
<code>linkf</code>	a character string specifying one of the three link functions to be used: "logit" (default) or "probit" or "cloglog"
<code>n.row</code>	a vector that specifies the number of rows in each matrix, a scalar if only one matrix is simulated
<code>n.col</code>	a vector that specifies the number of columns in each matrix, a scalar if only one matrix is simulated
<code>sens</code>	sensitivity of the group tests, set to be 1 by default.
<code>spec</code>	specificity of the group tests, set to be 1 by default.
<code>sens.ind</code>	sensitivity of the individual retests, set to be equal to <code>sens</code> if not specified otherwise.
<code>spec.ind</code>	specificity of the individual retests, set to be equal to <code>spec</code> if not specified otherwise.

## Details

sim.mp generates group testing data in matrix pooling form. The covariates are either specified by the `x` argument or they are generated from a gamma distribution with a given `gshape` and `gscale`. The individual probabilities are calculated from the covariates, the coefficients given in `par` and the link function specified through `linkf`. The true binary individual responses are then simulated from the individual probabilities. The individuals are organized into (by column) one or more matrices specified by `n.row` and `n.col`, and the true group responses are found (i.e., if at least one response is positive, the group is positive; otherwise, the group response is negative). The observed row and column group responses are then simulated using the given `sens` and `spec` values. Individual retests are simulated from `sens.ind` and `spec.ind` for individuals that lie on the intersection of an observed positive row and an observed positive column. In the case where no column (row) tests positive in a matrix, all the individuals in any observed positive rows (columns) will be assigned a simulated retest result. If no column or row is observed positive, NULL is returned.

## Value

sim.mp returns a list with the components `dframe`: the data frame that is actually to be fit, `ind`: the true individual responses presented in matrices and `prob`: the individual probabilities.

`dframe` is a data frame with columns

<code>col.resp</code>	the column group response
<code>row.resp</code>	the row group response
<code>x</code>	the covariate
<code>arrayn</code>	the array number
<code>coln</code>	the column group number
<code>rown</code>	the row group number
<code>retest</code>	the results of individual retests

## Author(s)

Boan Zhang

## See Also

[gtreg.mp](#) for the corresponding function to fit the model.

## Examples

```
# 5*6 and 4*5 matrix
set.seed(9128)
sa1a<-sim.mp(par=c(-7,0.1), n.row=c(5,4), n.col=c(6,5),
  sens=0.95, spec=0.95)
sa1<-sa1a$dframe
```

summary.gt

*Summary Method for Group Testing Model (Simple Pooling) Fits***Description**

Produce a summary list for objects of class "gt" returned by gtregr.

**Usage**

```
## S3 method for class 'gt'
summary(object, ...)
```

**Arguments**

object	a fitted object of class "gt".
...	currently not used.

**Details**

print.summary.gt is the print function that formats the coefficients, standard errors, etc. and additionally gives 'significance stars' if signif.stars is TRUE. The coefficients component of the result gives the estimated coefficients and their estimated standard errors, together with their ratio. This third column is labeled z ratio using Wald tests. A fourth column gives the two-tailed p-value corresponding to the z ratio based on a Wald test. (It is possible that there are no residual degrees of freedom from which to estimate it. In that case the estimate is NaN.)

**Value**

summary.gt returns an object of class "summary.gt", a list with components

call	the component from object.
link	the component from object.
deviance	the component from object.
aic	the component from object.
df.residual	the component from object.
null.deviance	the component from object.
df.null	the component from object.
Gibbs.sample.size	the component from object if simple pooling with retests (Dorfman's procedure) is used.
deviance.resid	the deviance residuals: see <a href="#">residuals.gt</a>
coefficients	the matrix of coefficients, standard errors, z-values and p-values. Aliased coefficients are omitted.
counts	the component from object.
method	the component from object.
cov.mat	the estimated covariance matrix of the estimated coefficients.

**Author(s)**

Boan Zhang

**See Also**

[gtreg](#) for creating an object of class "gt", and [print.summary.gt](#) for some options of changing the print out following `summary.gt`.

**Examples**

```
## --- Continuing the Example from '?gtreg':  
  
data(hivsurv)  
  
fit1 <- gtreg(formula = groupres ~ AGE + EDUC., data = hivsurv,  
             groupn = gnum, sens = 0.9, spec = 0.9, method = "Xie")  
  
summary(fit1)
```

---

`summary.gt.mp`*Summary Method for Group Testing Model (Matrix Pooling) Fits*

---

**Description**

Produce a summary list for objects of class "gt.mp" returned by [gtreg.mp](#).

**Usage**

```
## S3 method for class 'gt.mp'  
summary(object, ...)
```

**Arguments**

<code>object</code>	a fitted object of class "gt.mp".
<code>...</code>	currently not used.

**Details**

`print.summary.gt.mp` is the print function that formats the coefficients, standard errors, etc. and additionally gives 'significance stars' if `signif.stars` is TRUE. The coefficients component of the result gives the estimated coefficients and their estimated standard errors, together with their ratio. This third column is labeled z ratio using Wald tests. A fourth column gives the two-tailed p-value corresponding to the z ratio based on a Wald test.

**Value**

summary.gt.mp returns an object of class "summary.gt.mp", a list with components

call	the component from object.
link	the component from object.
coefficients	the matrix of coefficients, standard errors, z-values and p-values.
counts	the component from object.
Gibbs.sample.size	the component from object.
cov.mat	the estimated covariance matrix of the estimated coefficients.

**Author(s)**

Boan Zhang

**See Also**

[gtreg.mp](#) for creating an object of class "gtreg.mp", [print.summary.gt.mp](#) gives some hints how to change the print-out of summary.gt.mp

**Examples**

```
## --- Continuing the Example from '?sim.mp' and '?gtreg.mp':
# 5*6 and 4*5 matrix

set.seed(9128)
sa1a<-sim.mp(par=c(-7,0.1), n.row=c(5,4), n.col=c(6,5),
  sens=0.95, spec=0.95)
sa1<-sa1a$dframe

## Not run:
fit1mp <- gtreg.mp(formula = cbind(col.resp, row.resp) ~ x, data = sa1,
  coln = coln, rown = rown, arrayn = arrayn,
  sens = 0.95, spec = 0.95, linkf = "logit", n.gibbs = 1000, tol = 0.005)

summary(fit1mp)

## End(Not run)
```

---

summary.poolbindiff    *Summary methods for "poolbin" and "poolbindiff"*

---

**Description**

Summary Method for One-Sample and Two-sample confidence intervals (various pool sizes)

**Usage**

```
## S3 method for class 'poolbin'  
summary(object, scale = object$scale, ...)  
## S3 method for class 'poolbindiff'  
summary(object, scale = object$scale, ...)
```

**Arguments**

object	An object of class "poolbin" or "poolbindiff" ( <a href="#">pooledBin</a> , <a href="#">pooledBinDiff</a> )
scale	A coefficient to scale the point estimate and interval bounds
...	further arguments to be passed to print

**Value**

A print out

# Index

- \*Topic **datagen**
  - sim.gt, 53
  - sim.mp, 55
- \*Topic **datasets**
  - hivsurv, 35
- \*Topic **hplot**
  - plot.bgtDesign, 38
  - plot.binDesign, 39
- \*Topic **htest**
  - bgtCI, 6
  - bgtPower, 8
  - bgtTest, 10
  - bgtvs, 12
  - bgtWidth, 14
  - binCI, 16
  - binDesign, 18
  - binGroup-package, 2
  - binPower, 20
  - binTest, 22
  - binWidth, 24
  - estDesign, 25
  - nDesign, 36
  - pooledBin, 40
  - pooledBinDiff, 43
  - sDesign, 51
- \*Topic **misc**
  - gt.control, 27
- \*Topic **models**
  - binGroup-package, 2
  - gtreg, 28
  - predict.gt, 45
  - residuals.gt, 50
- \*Topic **package**
  - binGroup-package, 2
- \*Topic **pooled testing**
  - pooledBinDiff, 43
- \*Topic **print**
  - print.bgt, 46
  - print.bgtDesign, 47
  - print.binDesign, 48
  - print.gt.mp, 48
  - print.poolbindiff, 49
  - print.summary.gt, 49
  - summary.gt, 57
  - summary.gt.mp, 58
  - summary.poolbindiff, 60
- \*Topic **regression**
  - binGroup-package, 2
  - gtreg, 28
  - gtreg.mp, 31
  - predict.gt, 45
  - residuals.gt, 50
- bgtAC (bgtCI), 6
- bgtBlaker (bgtCI), 6
- bgtCI, 3, 6, 9, 12, 15, 42, 51
- bgtCP (bgtCI), 6
- bgtPower, 8, 38, 52
- bgtPowerI (bgtPower), 8
- bgtSOC (bgtCI), 6
- bgtTest, 7, 10
- bgtvs, 3, 7, 12, 42
- bgtWald (bgtCI), 6
- bgtWidth, 14
- bgtWidthI (bgtWidth), 14
- bgtWilson (bgtCI), 6
- binAC (binCI), 16
- binBlaker (binCI), 16
- binCI, 16, 21
- binCP (binCI), 16
- binDesign, 18, 21, 25
- binGroup (binGroup-package), 2
- binGroup-package, 2
- binom.test, 18
- binPower, 20, 20, 39
- binPowerI (binPower), 20
- binSOC (binCI), 16
- binTest, 18, 22
- binWald (binCI), 16

binWidth, 24  
binWilson (binCI), 16

EM (gtreg), 28  
EM.mp, 27  
EM.mp (gtreg.mp), 31  
estDesign, 3, 10, 25, 38, 52

gt.control, 27, 29, 32  
gtreg, 3, 27, 28, 34, 50, 54, 58  
gtreg.fit, 27  
gtreg.mp, 3, 27, 31, 31, 54, 56, 58, 59

hivsurv, 35

msep (estDesign), 25

nDesign, 3, 10, 26, 36, 39, 47, 52

par, 39  
plot, 39  
plot.bgtDesign, 38  
plot.binDesign, 20, 39  
plot.nDesign, 38  
plot.nDesign (plot.bgtDesign), 38  
plot.sDesign, 52  
plot.sDesign (plot.bgtDesign), 38  
pooledBin, 3, 7, 13, 40, 49, 60  
pooledBinDiff, 3, 43, 49, 60  
predict.gt, 3, 30, 31, 34, 45  
print.bgt, 46  
print.bgtCI (print.bgt), 46  
print.bgtDesign, 47  
print.bgtTest (print.bgt), 46  
print.bgtvs (print.bgt), 46  
print.binCI (print.bgt), 46  
print.binDesign, 48  
print.binTest (print.bgt), 46  
print.gt (print.gt.mp), 48  
print.gt.mp, 48  
print.nDesign (print.bgtDesign), 47  
print.poolbin, 41  
print.poolbin (print.poolbindiff), 49  
print.poolbindiff, 43, 49  
print.sDesign (print.bgtDesign), 47  
print.summary.gt, 49, 58  
print.summary.gt.mp, 59

residuals.gt, 3, 30, 31, 50, 57

sDesign, 3, 10, 26, 38, 39, 51  
sim.gt, 53  
sim.mp, 55  
summary.gt, 3, 30, 31, 49, 57  
summary.gt.mp, 34, 49, 58  
summary.poolbin, 41  
summary.poolbin (summary.poolbindiff),  
60  
summary.poolbindiff, 43, 60