Package 'RSurveillance'

October 1, 2014

Type Package

Title Design and analysis of disease surveillance activities

Version 0.0.1

Date 2014-09-29

Author Evan Sergeant

Maintainer Evan Sergeant <evan@ausvet.com.au>

Description This package provides a range of functions for the design and analysis of disease surveillance activities. These functions were originally developed for animal health surveillance activities but can be equally applied to aquatic animal, wildlife, plant and human health surveillance activities. Utilities are included for sample size calculation and analysis of representative surveys for disease freedom, risk-based studies for disease freedom and for prevalence estimation.

License GPL-2|GPL-3

LazyLoad yes

Imports epitools,epiR

R topics documented:

`adj.risk` *calculate adjusted risk*

Description

calculate adjusted risk

Usage

```
adj.risk(rr, ppr)
```

Arguments

rr	relative risk values (vector)
ppr	population proportions corresponding to rr values (vector of equal length)

Value

vector of adjusted risk values (in order corresponding to rr)

Examples

```
## Not run:
# examples for adj.risk
adj.risk(c(5, 1), c(0.1, 0.9))
adj.risk(c(5, 3, 1), c(0.1, 0.1, 0.8))

## End(Not run)
```

---

ap *estimate apparent prevalence*

Description

estimate apparent prevalence

Usage

```
ap(x, n, type = "wilson", conf = 0.95)
```

Arguments

x	number of positives in sample
n	sample size note: either x or n can be a vector, but at least one must be scalar
type	one of c("normal", "exact", "wilson", "jeffreys", "agresti-coull", "all"), default "wilson"
conf	level of confidence required, default = 0.95

Value

either 1) if type = "all", a list with 5 elements, each element a matrix with 6 columns, x, n, proportion, lower confidence limit, upper confidence limit, confidence level and CI method; or 2) a matrix of results for the chosen method

Examples

```
## Not run:
# examples for ap function
n<- 200
x<- 25
conf<- 0.95
ap(x, n)
ap(seq(10, 100, 10), 200, type = "agresti")
```

```
ap(seq(10, 100, 10), 200, type = "all")

## End(Not run)
```

`binom.agresti` *calculate agresti-coull confidence limits*

Description

calculate agresti-coull confidence limits

Usage

```
binom.agresti(x, n, conf = 0.95)
```

Arguments

`x`	number of positives in sample
`n`	sample size note: either x or n can be a vector, but at least one must be scalar
`conf`	level of confidence required, default 0.95

Value

a dataframe with 6 columns, x, n, proportion, lower confidence limit, upper confidence limit, confidence level and CI method

Examples

```
## Not run:
# test binom.agresti
binom.agresti(25, 200)
binom.agresti(seq(10, 100, 10), 200)
binom.agresti(50, seq(100, 1000, 100))

## End(Not run)
```

`binom.cp` *calculate clopper pearson exact confidence limits*

Description

calculate clopper pearson exact confidence limits

Usage

```
binom.cp(x, n, conf = 0.95)
```

Arguments

x	number of positives in sample
n	sample size note: either x or n can be a vector, but at least one must be scalar
conf	level of confidence required, default = 0.95

Value

a dataframe with 6 columns, x, n, proportion, lower confidence limit, upper confidence limit, confidence level and CI method

Examples

```
## Not run:
# test binom.cp
binom.cp(25, 200)
binom.cp(seq(10, 100, 10), 200)
binom.cp(50, seq(100, 1000, 100))

## End(Not run)
```

---

binom.jeffreys *calculate jeffreys confidence limits*

Description

calculate jeffreys confidence limits

Usage

```
binom.jeffreys(x, n, conf = 0.95)
```

Arguments

x	number of positives in sample
n	sample size note: either x or n can be a vector, but at least one must be scalar
conf	level of confidence required, default = 0.95

Value

a dataframe with 6 columns, x, n, proportion, lower confidence limit, upper confidence limit, confidence level and CI method

Examples

```
## Not run:
# test binom.jeffreys
binom.jeffreys(25, 200)
binom.jeffreys(seq(10, 100, 10), 200)
binom.jeffreys(50, seq(100, 1000, 100))

## End(Not run)
```

disc.prior *calculate discounted prior*

Description

calculate discounted prior

Usage

```
disc.prior(prior, p.intro)
```

Arguments

prior	prior probability of freedom before surveillance
p.intro	probability of introduction for time period (scalar or vactor equal length to sep)

Value

vector of discounted prior probability of freedom

Examples

```
## Not run:
# examples for disc.prior
disc.prior(0.5, 0.01)
disc.prior(0.95, c(0.001, 0.005, 0.01, 0.02, 0.05))
disc.prior(c(0.5, 0.6, 0.7, 0.8, 0.9, 0.95), 0.01)

## End(Not run)
```

epi.calc *calculate effective probability of infection*

Description

calculate effective probability of infection

Usage

```
epi.calc(pstar, rr, ppr)
```

Arguments

pstar	design prevalence
rr	relative risk values (vector)
ppr	population proportions corresponding to rr values (vector of equal length)

Value

list of 2 elements, a vector of EPI values and a vector of corresponding adjusted risks (in corresponding order to rr)

Examples

```
## Not run:
# examples for epi.calc
epi.calc(0.1, c(5, 1), c(0.1, 0.9))
epi.calc(0.02, c(5, 3, 1), c(0.1, 0.1, 0.8))

## End(Not run)
```

`n.2stage` *calculate 2-stage sample size*

Description

calculate 2-stage sample size

Usage

```
n.2stage(H = NA, N = NA, sep.sys = 0.95, sep.c, pstar.c, pstar.u,
  se = 1)
```

Arguments

`H`	population size = number of clusters or NA if not known
`N`	populaton size if known, scalar or vector
`sep.sys`	desired population sensitivity
`sep.c`	desired cluster-level sensitivity
`pstar.c`	specified cluster-level design prevalence (proportion or integer)
`pstar.u`	specified population-level design prevalence (proportion or integer)
`se`	unit sensitivity

Value

list of number of clusters to sample and sample size per cluster

Examples

```
## Not run:
# examples of n.2stage - checked
n.2stage(NA, NA, 0.95, 0.5, 0.01, 0.1, 0.95)
n.2stage(500, NA, 0.95, 0.5, 10, 0.1, 0.95)
n.2stage(1000, c(50, 100, 200, 500, 1000, 5000, NA), 0.95, 0.5, 0.01, 0.05, 0.8)
n.2stage(1000, c(50, 100, 200, 500, 1000, 5000, NA), 0.95, 0.5, 0.01, 1, 0.8)
n.2stage(1000, c(50, 100, 200, 500, 1000, 5000, NA), 0.9, 0.95, 1, 0.1, 0.8)

## End(Not run)
```

n.ap *sample size for apparent prevalence (simple proportion)*

Description

sample size for apparent prevalence (simple proportion)

Usage

```
n.ap(p, precision, conf = 0.95)
```

Arguments

p	expected proportion, scalar or vector of values
precision	absolute precision +/- proportion (half width of desired confidence interval), scalar or vector of values note: at least one of p and precision must be a scalar
conf	level of confidence required, default = 0.95

Value

a vector of sample sizes

Examples

```
## Not run:
# examples of n.ap
n.ap(0.5, 0.1)
n.ap(0.5, 0.1, conf=0.99)
n.ap(0.5, 0.1, N=1000)
n.ap(seq(0.1, 0.5, by = 0.1), 0.05)
n.ap(0.2, c(0.01, 0.02, 0.05, 0.1))

## End(Not run)
```

n.binom *calculate sample size assuming sampling with replacement (binomial)*

Description

calculate sample size assuming sampling with replacement (binomial)

Usage

```
n.binom(sep, pstar, se = 1)
```

Arguments

sep	desired population sensitivity
pstar	specified design prevalence
se	unit sensitivity

Value

vector of sample sizes

Examples

```
## Not run:
# examples for n.binom - checked
n.binom(sep=0.95, pstar=c(0.01, 0.02, 0.05, 0.1, 0.2))
n.binom(c(0.5, 0.8, 0.9, 0.95), 0.01)

## End(Not run)
```

`n.c.freecalc` *calculate optimum sample size and cut-point reactors for given population size and other parameters using freecalc algorithm all paramaters must be scalars*

Description

calculate optimum sample size and cut-point reactors for given population size and other parameters using freecalc algorithm all paramaters must be scalars

Usage

```
n.c.freecalc(N, sep = 0.95, c = 1, se, sp = 1, pstar, minSpH = 0.95)
```

Arguments

`N`	population size
`sep`	target population sensitivity
`c`	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
`se`	unit sensitivity
`sp`	unit specificity, default=1
`pstar`	design prevalence
`minSpH`	minimium desired population specificity

Value

a list of 3 elements, a dataframe with 1 row and six columns for the recommended sample size and corresponding values for population sensitivity (SeP), population specificity (SpP), N, c and pstar, a vector of SeP values and a vector of SpP values, for n = 1:N

Examples

```
## Not run:
# examples for n.c.hp
n.c.freecalc(120,0.95,c=5,se=0.9,sp=0.99,pstar=0.1, minSpH=0.9)[[1]]
n.c.freecalc(65,0.95,c=5,se=0.95,sp=0.99,pstar=0.05, minSpH=0.9)

## End(Not run)
```

`n.c.hp` *all paramaters must be scalars*

Description

all paramaters must be scalars

Usage

```
n.c.hp(N, sep = 0.95, c = 1, se, sp = 1, pstar, minSpH = 0.95)
```

Arguments

`N`	population size
`sep`	target population sensitivity
`c`	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
`se`	unit sensitivity
`sp`	unit specificity, default=1
`pstar`	design prevalence
`minSpH`	minimium desired population specificity

Value

a list of 3 elements, a dataframe with 1 row and six columns for the recommended sample size and corresponding values for population sensitivity (SeP), population specificity (SpP), N, c and pstar, a vector of SeP values and a vector of SpP values, for n = 1:N

Examples

```
## Not run:
# examples for n.c.hp
n.c.hp(65,0.95,c=5,se=0.95,sp=0.99,pstar=0.05, minSpH=0.9)[[1]]
tmp<- n.c.hp(120,0.95,c=5,se=0.9,sp=0.99,pstar=0.1, minSpH=0.9)

## End(Not run)
```

`n.freecalc` *calculate freecalc sample size for given population size, cut-point number of reactors and other parameters all paramaters must be scalars*

Description

calculate freecalc sample size for given population size, cut-point number of reactors and other parameters all paramaters must be scalars

Usage

```
n.freecalc(N, sep = 0.95, c = 1, se, sp = 1, pstar, minSpH = 0.95)
```

Arguments

N	population size
sep	target population sensitivity
c	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
se	unit sensitivity
sp	unit specificity, default=1
pstar	design prevalence
minSpH	minimium desired population specificity

Value

a list of 2 elements, a dataframe with 1 row and six columns for the recommended sample size and corresponding values for population sensitivity (SeP), population specificity (SpP), N, c and pstar and a dataframe of n rows with SeP and SpP values for each value of n up to the recommended value

Examples

```
## Not run:
# examples for n.freecalc
n.freecalc(65,0.95,c=1,se=0.95,sp=0.99,pstar=0.05, minSpH=0.9)[[1]]
n.freecalc(65,0.95,c=2,se=0.95,sp=0.99,pstar=0.05, minSpH=0.9)[[1]]
n.freecalc(65,0.95,c=3,se=0.95,sp=0.99,pstar=0.05, minSpH=0.9)

## End(Not run)
```

---

n.freedom *calculate sample size usng appropriate method for data provided*

Description

calculate sample size usng appropriate method for data provided

Usage

```
n.freedom(N = NA, sep = 0.95, pstar, se = 1)
```

Arguments

N	populaton size if known, scalar or vector
sep	desired population sensitivity
pstar	specified design prevalence (proportion or integer)
se	unit sensitivity

Value

vector of sample sizes, NA if N is specified and n>N

Examples

```
## Not run:
# examples for n.freedom - checked
n.freedom(NA, sep=0.95, pstar=0.01, se=1)
n.freedom(500, sep=0.95, pstar=0.01, se=1)
n.freedom(N=c(100, 500, 1000, 5000, 10000, 100000, NA), sep=0.95, pstar=0.01, se=1)
n.freedom(500, sep=0.95, pstar=0.01, se=c(0.5, 0.6, 0.7, 0.8, 0.9, 0.99, 1))

## End(Not run)
```

`n.hp` *calculate sample size for given population size, cut-point number of reactors and other parameters calculates sample size to achieve specified population sensitivity with population specificity >= specified minimum value if not possible to achieve target sep AND SpH>= minSpH, returns sample size for maximum achievable sep. all paramaters must be scalars*

Description

calculate sample size for given population size, cut-point number of reactors and other parameters calculates sample size to achieve specified population sensitivity with population specificity >= specified minimum value if not possible to achieve target sep AND SpH>= minSpH, returns sample size for maximum achievable sep. all paramaters must be scalars

Usage

```
n.hp(N, sep = 0.95, c = 1, se, sp = 1, pstar, minSpH = 0.95)
```

Arguments

`N`	population size
`sep`	target population sensitivity
`c`	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
`se`	unit sensitivity
`sp`	unit specificity, default=1
`pstar`	design prevalence
`minSpH`	minimium desired population specificity

Value

a list of 2 elements, a dataframe with 1 row and six columns for the recommended sample size and corresponding values for population sensitivity (SeP), population specificity (SpP), N, c and pstar and a dataframe of n rows with SeP and SpP values for each value of n up to the recommended value

Examples

```
## Not run:
# examples for n.hp
n.hp(65,0.95,c=1,se=0.95,sp=0.99,pstar=0.05, minSpH=0.9)[[1]]
n.hp(65,0.95,c=2,se=0.95,sp=0.99,pstar=0.05, minSpH=0.9)

## End(Not run)
```

`n.hypergeo` *calculate sample size assuming sampling without replacement (hypergeometric)*

Description

calculate sample size assuming sampling without replacement (hypergeometric)

Usage

```
n.hypergeo(sep, N, d, se = 1)
```

Arguments

`sep`	desired population sensitivity
`N`	population size
`d`	expected number of infected units in population (=pstar*N rounded to next integer)
`se`	unit sensitivity

Value

vector of sample sizes, NA if n>N

Examples

```
## Not run:
# examples for n.hypergeo - checked
n.hypergeo(0.95, N=100, d=1, se = 0.95)
n.hypergeo(sep=0.95, N=c(100, 200, 500, 1000, 10000), d=ceiling(0.01*c(100, 200, 500, 1000, 10000)))
n.hypergeo(c(0.5, 0.8, 0.9, 0.95), N=100, d=5)
n.hypergeo(0.95, N=80, d=c(1, 2, 5, 10))
n.hypergeo(0.95, N=80, d=c(1, 2, 5, 10), se = 0.8)

## End(Not run)
```

n.pfree *calculate sample size rqued to achieve target confidence of freedom*

Description

calculate sample size rqued to achieve target confidence of freedom

Usage

```
n.pfree(pfree, prior, p.intro, pstar, se, N = NA)
```

Arguments

pfree	desired probability of freedom (scalar or vector)
prior	prior probability of freedom before surveillance
p.intro	probability of introduction for time period (scalar or vactor equal length to sep)
pstar	design prevalence
se	unit sensitivity (scalar or vector)
N	population size

Value

vector of sample sizes

Examples

```
## Not run:
# examples for n.pfree
n.pfree(0.95, 0.5, 0.01, 0.05, 0.9)
n.pfree(0.95, 0.5, 0.01, 0.05, 0.9, N=300)
n.pfree(pfree = c(0.9, 0.95, 0.98, 0.99), prior = 0.7, 0.01, 0.01, 0.8, 1000)
n.pfree(0.95, 0.7, 0.01, 0.1, 0.96)

## End(Not run)
```

n.pooled *sample size for pooled testing for freedom*

Description

sample size for pooled testing for freedom

Usage

```
n.pooled(sep, k, pstar, pse, psp = 1)
```

Arguments

sep	desired population sensitivity (scalar or vector)
k	pool size (constant) (scalar or vector)
pstar	design prevalence
pse	pool-level sensitivity
psp	pool-level specificity

Value

vector of sample sizes

Examples

```
## Not run:
# examples for n.pooled
n.pooled(0.95, 5, 0.01, 1, 1)
n.pooled(0.95, 10, 0.1, 0.9, 1)
n.pooled(0.95, c(2, 5, 10, 20), 0.1, c(0.99, 0.98, 0.97, 0.95), 1)

## End(Not run)
```

---

n.rb *calculate sample size for risk-based sampling \- binomial*

Description

calculate sample size for risk-based sampling \- binomial

Usage

```
n.rb(pstar, rr, ppr, spr, se, sep)
```

Arguments

pstar	design prevalence (scalar)
rr	relative risk values (vector)
ppr	population proportions corresponding to rr values (vector of equal length)
spr	surveillance proportion for each risk group (vector equal length to rr, ppr)
se	unit sensitivity (fixed or vector same length as rr, ppr, n)
sep	required population sensitivity

Value

list of 2 elements, a vector of sample sizes for each risk group a scalar of total sample size, a vector of EPI values and a vector of adjusted risks

Examples

```
## Not run:
# examples for n.rb
n.rb(0.1, c(5, 3, 1), c(0.1, 0.10, 0.80), c(0.5, 0.3, 0.2), 0.9, 0.95)
n.rb(0.01, c(5, 1), c(0.1, 0.9), c(0.8, 0.2), c(0.9, 0.95), 0.95)

## End(Not run)
```

`n.rb.varse` *sample size for varying sensitivity*

Description

sample size for varying sensitivity

Usage

```
n.rb.varse(pstar, rr, ppr, spr, se, spr.rg, sep)
```

Arguments

`pstar`	design prevalence
`rr`	vector of relative risk values
`ppr`	vector of population proportions for each risk group - same length as rr
`spr`	vector of surveillance proportions for each risk group - same length as rr
`se`	vector of sensitivity values
`spr.rg`	matrix of proportions of samples for each sensitivity value in each risk group (rows - risk groups, columns = sensitivity values) row sums must equal 1
`sep`	required population sensitivity

Value

list of 3 elements, a matrix of sample sizes for each risk and sensitivity group, a vector of EPI values and a vector of mean sensitivity for each risk group

Examples

```
## Not run:
# examples for n.rb.varse
m<- rbind(c(0.8, 0.2), c(0.5, 0.5), c(0.7, 0.3))
n.rb.varse(0.01, c(5, 3, 1), c(0.1, 0.1, 0.8), c(0.4, 0.4, 0.2), c(0.92, 0.8), m, 0.95)

m<- rbind(c(0.8, 0.2), c(0.6, 0.4))
n.rb.varse(0.05, c(3, 1), c(0.2, 0.8), c(0.7, 0.3), c(0.95, 0.8), m, 0.95)

m<- rbind(c(1), c(1))
n.rb.varse(0.05, c(3, 1), c(0.2, 0.8), c(0.7, 0.3), c(0.95), m, 0.99)

## End(Not run)
```

n.tp *sample size for true prevalence*

Description

sample size for true prevalence

Usage

```
n.tp(p, se, sp, precision, conf = 0.95)
```

Arguments

p	estimated tp
se	test unit sensitivity
sp	test unit specificity
precision	absolute precision +/- proportion (half width of desired confidence interval)
conf	desired level of confidence for CI, default = 0.95

Value

a vector of sample sizes

Examples

```
## Not run:
# examples for n.tp
n.tp(0.1, 0.9, 0.99, 0.05)
n.tp(0.1, 0.9, 0.99, 0.05, conf = 0.99)
n.tp(c(0.05, 0.1, 0.2, 0.3, 0.4, 0.5), 0.9, 0.99, 0.05)
n.tp(0.5, 0.9, 0.99, c(0.01, 0.02, 0.05, 0.1, 0.2))

## End(Not run)
```

pfree.1 *calculate confidence of freedom for a single time period*

Description

calculate confidence of freedom for a single time period

Usage

```
pfree.1(sep, p.intro, prior = 0.5)
```

Arguments

sep	population sensitivity for time period (scalar or vector)
p.intro	probability of introduction for time period (scalar or vactor equal length to sep)
prior	prior probability of freedom before surveillance

Value

data.frame with columns for sep, p.intro, discounted prior, pfree, pfree.equ and prior.equ

Examples

```
## Not run:
# examples for pfree.1
pfree.1(0.8, 0.01, 0.5)
pfree.1(0.6, c(0.001, 0.005, 0.01, 0.02, 0.05), 0.5)
pfree.1(runif(10, 0.4, 0.6), 0.01, 0.5)
pfree.1(runif(10, 0.4, 0.6), runif(10, 0.005, 0.015), 0.5)

## End(Not run)
```

---

pfree.calc *calculate probability (confidence) of freedom for given prior, sep and p.intro over 1 or more time periods*

Description

calculate probability (confidence) of freedom for given prior, sep and p.intro over 1 or more time periods

Usage

```
pfree.calc(sep, p.intro, prior = 0.5)
```

Arguments

sep	population sensitivity for time period (scalar or vector)
p.intro	probability of introduction for time period (scalar or vactor equal length to sep)
prior	prior probability of freedom before surveillance

Value

data.frame with columns for sep, p.intro, discounted prior, pfree, pfree.equ and prior.equ

Examples

```
## Not run:
# examples for pfree.calc
pfree.calc(0.8, 0.01, 0.5)
pfree.calc(rep(0.6,24), 0.01, 0.5)
pfree.calc(runif(10, 0.4, 0.6), 0.01, 0.5)
pfree.calc(runif(10, 0.4, 0.6), runif(10, 0.005, 0.015), 0.5)

## End(Not run)
```

pfree.equ *calculate equilibrium pfree and equilibrium prior Pfree*

Description

calculate equilibrium pfree and equilibrium prior Pfree

Usage

```
pfree.equ(sep, p.intro)
```

Arguments

sep	population sensitivity for time period (scalar or vector)
p.intro	probability of introduction for time period (scalar or vactor equal length to sep)

Value

list of 2 vectors, equilibrium posterior probability of freedom and equilibrium prior (discounted) probability of freedom

Examples

```
## Not run:
# examples of pfree.equ
pfree.equ(runif(10, 0.4, 0.6), 0.01)
pfree.equ(0.8, 0.05)
pfree.equ(rep(0.9, 6), c(0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05))

## End(Not run)
```

pstar.calc *calculate design prevalence for given sample size anddesired population-level sensitivity*

Description

calculate design prevalence for given sample size anddesired population-level sensitivity

Usage

```
pstar.calc(N = NA, n, sep, se)
```

Arguments

N	populaton size if known, scalar or vector
n	sample size
sep	desired population sensitivity
se	unit sensitivity

Value

vector of design prevalence values

Examples

```
## Not run:
# examples of pstar.calc- checked
pstar.calc(NA, 280, 0.95, 0.98)
pstar.calc(500, 250, sep=0.95, se=1)
pstar.calc(N=c(100, 500, 1000, 5000, 10000, 100000, NA), n=30, sep=0.95, se=1)
pstar.calc(500, n=30, sep=0.95, se=c(0.5, 0.6, 0.7, 0.8, 0.9, 0.99, 1))

## End(Not run)
```

---

sd.tp *function to calculate variance/sd of tp assuming se and sp known exactly*

Description

function to calculate variance/sd of tp assuming se and sp known exactly

Usage

```
sd.tp(x, n, se, sp)
```

Arguments

x	number of positive results in sample
n	sample size
se	test unit sensitivity
sp	test unit specificity

Value

vector of standard deviation values for true prevalence estimates

Examples

```
## Not run:
# example of sd.tp
sd.tp(1:10, 20, 0.9, 0.99)

## End(Not run)
```

se.parallel — *calculate combined sensitivity for multiple tests in parallel (assuming independence)*

Description

calculate combined sensitivity for multiple tests in parallel (assuming independence)

Usage

```
se.parallel(se)
```

Arguments

se	vector of unit sensitivity values

Value

scalar of combined sensitivity, assuming independence

Examples

```
## Not run:
# examples for se.parallel
se.parallel(c(0.99, 0.95, 0.8))

## End(Not run)
```

se.series — *calculate combined sensitivity for multiple tests in series (assuming independence)*

Description

calculate combined sensitivity for multiple tests in series (assuming independence)

Usage

```
se.series(se)
```

Arguments

se	vector of unit sensitivity values

Value

scalar of combined sensitivity, assuming independence

Examples

```
## Not run:
# examples for se.series
se.series(c(0.99, 0.95, 0.8))

## End(Not run)
```

---

sep — *Population sensitivity function to calculate seh using most appropriate option depending on whether or not N provided*

Description

Population sensitivity function to calculate seh using most appropriate option depending on whether or not N provided

Usage

```
sep(N = NA, n, pstar, se = 1)
```

Arguments

N	vector of population sizes: NA or vector of same length as n
n	vector of sample sizes
pstar	design prevalence: single value as a proportion or integer
se	unit sensitivity: single value or vector same lengthh as n

Value

value vector of population-level sensitivities

Examples

```
## Not run:
# examples for sep - checked
sep(n=300, pstar=0.01, se=1)
sep(NA, 300, 0.01, 1)
sep(10000, 150, 0.02, 1)
sep(n=1:100, pstar = 0.05, se=0.95)
N<- seq(30, 100, by = 5)
se<- 0.95
pstar<- 0.1
n<- rep(30, length(N))
sep(N, n, pstar, se = se)
sep(rep(100, 10), seq(10, 100, by = 10), pstar = 1, se=0.99)
N<- c(55, 134, NA, 44, 256)
n<- c(15, 30, 28, 15, 33)
sep(N, n, 0.1, 0.95)

## End(Not run)
```

sep.binom — *Population sensitivity assuming sampling with replacement (binomial)*

Description

Population sensitivity assuming sampling with replacement (binomial)

Usage

```
sep.binom(n, pstar, se = 1, sp = 1)
```

Arguments

n	integer scalar or vector of number tested (sample size)
pstar	scalar or vector of design prevalence as proportion
se	unit sensitivity of test (proportion)
sp	unit specificity of test (proportion)

Value

vector of population-level sensitivities

Examples

```
## Not run:
# examples for sep.binom - checked
sep.binom(n=300, pstar = 0.02, se = 0.92)
tested<- seq(10,100, by=10)
prev<- 0.05
sens<- 0.9
sep.binom(tested, prev, sens)

## End(Not run)
```

sep.binom.imperfect — *any one input can be a vector, all others must be either scalars or vectors of the same length*

Description

any one input can be a vector, all others must be either scalars or vectors of the same length

Usage

```
sep.binom.imperfect(n, c = 1, se, sp = 1, pstar)
```

Arguments

n	sample size
c	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
se	unit sensitivity
sp	unit specificity, default=1
pstar	design prevalence

Value

a vector of population-level sensitivities

Examples

```
## Not run:
# examples for sep.imperfect.binom
sep.binom.imperfect(1:10*5, 2, 0.95, 0.98, 0.1)
sep.binom.imperfect(50, 1:5, 0.95, 0.98, 0.1)
sep.binom.imperfect(30, 2, 0.9, 0.98, 0.1)
sep.binom.imperfect(30, 1, 0.9, 0.98, 0.1)

## End(Not run)
```

---

sep.exact *Population sensitivity assuming census (all units tested)*

Description

Population sensitivity assuming census (all units tested)

Usage

```
sep.exact(d = 1, se = 1)
```

Arguments

d	expected number of infected units in population (=pstar*N rounded to next integer)
se	unit sensitivity of test (proportion)

Value

vector of population-level sensitivities

Examples

```
## Not run:
# examples for sep.exact - checked
sep.exact(d=1, se = 0.92)
inf<- 1:5
sens<- 0.8
sep.exact(d=inf, se=sens)
sep.exact(se=0.8, d = ceiling(0.01*c(10, 50, 100, 250, 500)))

## End(Not run)
```

`sep.freecalc` *all inputs are scalars*

Description

all inputs are scalars

Usage

```
sep.freecalc(N, n, c = 1, se, sp = 1, pstar)
```

Arguments

N	population size
n	sample size
c	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
se	unit sensitivity
sp	unit specificity, default=1
pstar	design prevalence

Value

a scalar of population-level sensitivity

Examples

```
## Not run:
# examples of sep.freecalc
sep.freecalc(150, 30, 2, 0.9, 0.98, 0.1)
sep.freecalc(150, 30, 1, 0.9, 0.98, 0.1)


## End(Not run)
```

`sep.hp` *any one input can be a vector, all others must be either scalars or vectors of the same length*

Description

any one input can be a vector, all others must be either scalars or vectors of the same length

Usage

```
sep.hp(N, n, c = 1, se, sp = 1, pstar)
```

Arguments

`N`	population size
`n`	sample size
`c`	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
`se`	unit sensitivity
`sp`	unit specificity, default=1
`pstar`	design prevalence

Value

a vector of population-level sensitivities

Examples

```
## Not run:
# examples of sep.hp
sep.hp(150, 1:5*10, 2, 0.9, 0.98, 0.1)
sep.hp(150, 30, 2, 0.9, 0.98, 15)
sep.hp(150, 30, 1, 0.9, 0.98, 15)
sep.hp(150, 30, 1, 0.9, 0.98, 0.1)

## End(Not run)
```

`sep.hypergeo` *Population sensitivity assuming sampling without replacement (hypergeometric approximation)*

Description

Population sensitivity assuming sampling without replacement (hypergeometric approximation)

Usage

```
sep.hypergeo(N, n, d, se = 1)
```

Arguments

N	population size
n	sample size (tested)
d	expected number of infected units in population (=pstar*N rounded to next integer)
se	unit sensitivity of test (proportion)

Value

value vector of population-level sensitivities

Examples

```
## Not run:
# examples for sep.hypergeo - checked
sep.hypergeo(N=100, n=50, d=1, se = 0.92)
inf<- 1:5
sens<- 0.8
sep.hypergeo(N=100, n=50, d=inf, se=sens)
N<- c(10, 50, 100, 250, 500)
sep.hypergeo(se=0.8, N=N, n=c(5, 25, 50, 125, 250), d = ceiling(0.01*N))

## End(Not run)
```

---

sep.pfree *calcuate population sensitivity for given PFree*

Description

calcuate population sensitivity for given PFree

Usage

```
sep.pfree(prior, pfree)
```

Arguments

prior	prior probability of freedom before surveillance (scalar or vector)
pfree	desired probability of freedom (scalar or vector)

Value

vector of population-level sensitivities

Examples

```
## Not run:
# examples of sep.pfree
sep.pfree(0.5, 0.95)
sep.pfree(c(0.5, 0.6, 0.7, 0.8, 0.9, 0.95), 0.99)
sep.pfree(0.5, c(0.8, 0.9, 0.95, 0.99))

## End(Not run)
```

sep.pooled *population sensitivity with pooled sampling*

Description

population sensitivity with pooled sampling

Usage

```
sep.pooled(r, k, pstar, pse, psp = 1)
```

Arguments

r	number of pools sampled (scalar or vector)
k	pool size (constant) (scalar or vector)
pstar	design prevalence
pse	pool-level sensitivity
psp	pool-level specificity

Value

list of 2 elements, vector of sep values and vector of spp values

Examples

```
## Not run:
# examples for sep.pooled
sep.pooled(60, 5, 0.01, 1, 1)
sep.pooled(4, 10, 0.1, 0.9, 1)
sep.pooled(1:10*5, 5, 0.02, 0.9, 0.99)
sep.pooled(10, 5, 0.05, c(0.8, 0.9, 0.95, 0.99), 1)

## End(Not run)
```

sep.prior *calcuate population sensitivity for given prior pfree*

Description

calcuate population sensitivity for given prior pfree

Usage

```
sep.prior(prior, p.intro)
```

Arguments

prior	prior probability of freedom before surveillance (scalar or vector)
p.intro	probability of introduction for time period (scalar or vactor equal length to sep)

Value

vector of population-level sensitivities

Examples

```
## Not run:
# examples of sep.prior
sep.prior(0.95, 0.01)
sep.prior(c(0.9, 0.95, 0.98, 0.99), 0.01)
sep.prior(0.95, c(0.001, 0.005, 0.01, 0.02, 0.05))

## End(Not run)
```

---

sep.rb.bin *calculate popuation sensitivity for single risk factor - binomial*

Description

calculate popuation sensitivity for single risk factor - binomial

Usage

```
sep.rb.bin(pstar, rr, ppr, n, se)
```

Arguments

pstar	design prevalence (scalar)
rr	relative risk values (vector)
ppr	population proportions corresponding to rr values (vector of equal length)
n	sample size per risk category (vector same length as rr and ppr)
se	unit sensitivity (fixed or vector same length as rr, ppr, n)

Value

list of 3 elements, a scalar of population-level sensitivity a vector of EPI values and a vector of corresponding Adjusted risks

Examples

```
## Not run:
# examples for sep.rb.bin
sep.rb.bin(0.1, c(5, 3, 1), c(0.1, 0.1, 0.8), c(5, 5, 5), 0.9)
sep.rb.bin(0.1, c(5, 1), c(0.1, 0.9), c(10, 5), c(0.95, 0.9))
sep.rb.bin(0.1, c(5, 1), c(0.1, 0.9), c(10, 5), c(0.9, 0.9))
sep.rb.bin(0.01, c(5, 1), c(0.1, 0.9), c(90, 50), c(0.9, 0.9))

## End(Not run)
```

`sep.rb.bin.varse` *calculate population sensitivity for varying unit sensitivity - binomial*

Description

calculate population sensitivity for varying unit sensitivity - binomial

Usage

```
sep.rb.bin.varse(pstar, rr, ppr, df)
```

Arguments

`pstar`	design prevalence (scalar)
`rr`	relative risk values (vector)
`ppr`	population proportions corresponding to rr values (vector of equal length)
`df`	dataframe of values for each sensitivity level col 1 = risk group index, col 2 = unit Se, col 3 = n

Value

list of 3 elements, a scalar of population-level sensitivity a vector of EPI values and a vector of corresponding Adjusted risks

Examples

```
## Not run:
# examples for sep.rb.bin.varse
rg<- c(1, 1, 2, 2)
se<- c(0.92, 0.85, 0.92, 0.85)
n<- c(80, 30, 20, 30)
df<- data.frame(rg, se, n)
sep.rb.bin.varse(0.01, c(5, 1), c(0.1, 0.9), df)

rg<- c(1, 1, 2, 2)
se<- c(0.95, 0.8, 0.95, 0.8)
n<- c(20, 10, 10, 5)
df<- data.frame(rg, se, n)
sep.rb.bin.varse(0.05, c(3, 1), c(0.2, 0.8), df)

rg<- c(rep(1, 30), rep(2, 15))
se<- c(rep(0.95, 20), rep(0.8, 10), rep(0.95, 10), rep(0.8, 5))
n<- rep(1, 45)
df<- data.frame(rg, se, n)
sep.rb.bin.varse(0.02, c(3, 1), c(0.2, 0.8), df)

rg<- c(1, 2, 3, 1, 2, 3)
se<- c(0.95, 0.95, 0.95, 0.8, 0.8, 0.8)
n<- c(20, 10, 10, 30, 5, 5)
df<- data.frame(rg, se, n)
sep.rb.bin.varse(0.01, c(5, 3, 1), c(0.1, 0.3, 0.6), df)

## End(Not run)
```

sep.rb.hypergeo *calculate population sensitivity for single risk factor - hypergeometric*

Description

calculate population sensitivity for single risk factor - hypergeometric

Usage

```
sep.rb.hypergeo(pstar, rr, N, n, se)
```

Arguments

pstar	design prevalence (scalar)
rr	relative risk values (vector)
N	Population size per risk category (vector same length as rr and ppr)
n	sample size per risk category (vector same length as rr and ppr)
se	unit sensitivity (fixed or vector same length as rr, ppr, n)

Value

list of 3 elements, a scalar of population-level sensitivity a vector of EPI values and a vector of corresponding Adjusted risks

Examples

```
## Not run:
# examples for sep.rb.bin
sep.rb.hypergeo(0.1, c(5, 3, 1), c(10, 10, 80), c(5, 5, 5), 0.9)
sep.rb.hypergeo(0.1, c(5, 1), c(15, 140), c(10, 5), c(0.95, 0.9))
sep.rb.hypergeo(0.1, c(5, 1), c(23, 180), c(10, 5), c(0.9, 0.9))
sep.rb.hypergeo(0.01, c(5, 1), c(100, 900), c(90, 50), c(0.9, 0.9))

## End(Not run)
```

sep.rb.hypergeo.varse *calculate population sensitivity for varying unit sensitivity - hypergeometric*

Description

calculate population sensitivity for varying unit sensitivity - hypergeometric

Usage

```
sep.rb.hypergeo.varse(pstar, rr, N, df)
```

Arguments

pstar	design prevalence (scalar)
rr	relative risk values (vector)
N	vector of population size corresponding to rr values (vector of equal length)
df	dataframe of values for each sensitivity level col 1 = risk group index, col 2 = unit Se, col 3 = n

Value

list of 5 elements, a scalar of population-level sensitivity a vector of EPI values, a vector of corresponding Adjusted risks a vector of sample sizes (n) per risk group and a vector of mean unit sensitivities per risk group

Examples

```
## Not run:
# examples for sep.rb.hypergeo.varse
rg<- c(1, 1, 2, 2)
se<- c(0.92, 0.85, 0.92, 0.85)
n<- c(80, 30, 20, 30)
df<- data.frame(rg, se, n)
sep.rb.hypergeo.varse(0.01, c(5, 1), c(200, 1800), df)

rg<- c(1, 1, 2, 2)
se<- c(0.95, 0.8, 0.95, 0.8)
n<- c(20, 10, 10, 5)
df<- data.frame(rg, se, n)
sep.rb.hypergeo.varse(0.05, c(3, 1), c(100, 400), df)

rg<- c(rep(1, 30), rep(2, 15))
se<- c(rep(0.95, 20), rep(0.8, 10), rep(0.95, 10), rep(0.8, 5))
n<- rep(1, 45)
df<- data.frame(rg, se, n)
sep.rb.hypergeo.varse(0.02, c(3, 1), c(100, 400), df)

rg<- c(1, 2, 3, 1, 2, 3)
se<- c(0.95, 0.95, 0.95, 0.8, 0.8, 0.8)
n<- c(20, 10, 10, 30, 5, 5)
df<- data.frame(rg, se, n)
sep.rb.hypergeo.varse(0.01, c(5, 3, 1), c(100, 300, 600), df)

## End(Not run)
```

---

sep.rb2.binom *1-stage risk-based sampling with 2 risk factors - binomial*

Description

1-stage risk-based sampling with 2 risk factors - binomial

Usage

```
sep.rb2.binom(pstar, rr1, ppr1, rr2, ppr2, n, se)
```

Arguments

pstar	design prevalence
rr1	relative risks for first level risk factor
ppr1	population proportions for first level risk factor
rr2	relative risks for second level risk factor, matrix, rows = levels of rr1, cols = levels of rr2
ppr2	population proportions for second level risk factor, matrix, rows = levels of rr1, cols = levels of rr2
n	matrix of number tested for each risk group (stratified
se	test unit sensitivity

Value

list of 4 elements, a scalar of population-level sensitivity a matrix of EPI values, a vector of corresponding Adjusted risks for the first risk factor and a matrix of adjusted risks for the second risk factor

Examples

```
## Not run:
# examples for sep.rb2.binom
pstar<- 0.01
rr1<- c(3, 1)
ppr1<- c(0.2, 0.8)
rr2<- rbind(c(4,1), c(4,1))
ppr2<- rbind(c(0.1, 0.9), c(0.3, 0.7))
se<- 0.8
n<- rbind(c(50, 20), c(20, 10))
sep.rb2.binom(pstar, rr1, ppr1, rr2, ppr2, n, se)

## End(Not run)
```

---

sep.rb2.hypergeo *1-stage risk-based sampling with 2 risk factors - hypergeometric*

Description

1-stage risk-based sampling with 2 risk factors - hypergeometric

Usage

```
sep.rb2.hypergeo(pstar, rr1, rr2, N, n, se)
```

Arguments

pstar	design prevalence
rr1	relative risks for first level risk factor
rr2	relative risks for second level risk factor
N	matrix of population size for each risk group (stratified
n	matrix of number tested (sample size) for each risk group (stratified
se	test unit sensitivity

Value

list of 6 elements, a scalar of population-level sensitivity a matrix of EPI values, a vector of corresponding Adjusted risks for the first risk factor and a matrix of adjusted risks for the second risk factor, a vector of population proportions for the first risk factor and a matrix of population proportions for the second risk factor

Examples

```
## Not run:
# examples for sep.rb2.hypergeo
pstar<- 0.01
rr1<- c(3, 1)
rr2<- rbind(c(4,1), c(4,1))
N<- rbind(c(100, 500), c(300, 1000))
n<- rbind(c(50, 20), c(20, 10))
se<- 0.8
sep.rb2.hypergeo(pstar, rr1, rr2, N, n, se)

## End(Not run)
```

---

sep.sys *population sensitivity from sampling of individual clusters*

Description

population sensitivity from sampling of individual clusters

Usage

```
sep.sys(H = NA, N = NA, n, pstar.c, pstar.u, se = 1)
```

Arguments

H	population size = number of clusters or NA if not known
N	population size within clusters NA if not provided, otherwise a vector of same length as n
n	sample size (tested)
pstar.c	cluster (herd) level design prevalence single value either proportion or integer
pstar.u	unit (animal) level design prevalence single value either proportion or integer
se	unit sensitivity of test (proportion)

Value

vector of population-level sensitivities

Note

if pstar.c is not a proportion N must be entered (and N>=n)

Examples

```
## Not run:
# examples for sep.sys - checked
H<- 500
N<- rep(1000, 150)
N[5]<- NA
n<- rep(30, 150)
pstar.u<- 0.1
pstar.c<- 0.01
se<- 0.98
sep.sys(H, N, n, pstar.c, pstar.u, se)
sep.sys(NA, N, n, 0.02, 0.05, 0.95)
N<- round(runif(105)*900+100)
n<- round(runif(105)*30+10)
sse<- sep.sys(1000, N, n, 0.02, 0.05, 0.9)
data.frame(N, n, sse[[2]])

## End(Not run)
```

`sep.var.se` *population sensitivity for varying unit sensitivity*

Description

population sensitivity for varying unit sensitivity

Usage

```
sep.var.se(N = NA, se, pstar)
```

Arguments

`N`	population size (number of units/clusters. N >= length(se)) or NA if unknown
`se`	vector of unit sensitivity values
`pstar`	specified design prevalence

Value

value vector of population-level sensitivities

Examples

```
## Not run:
# examples of sep.var.se - checked
sens<- c(rep(0.9, 50), rep(0.95, 100))
sep.var.se(NA, sens, 0.01)
sep.var.se(se=sens, pstar=0.01)
sep.var.se(N=500, sens, 0.01)
sep.var.se(NA, runif(150, 0.95, 0.99), 0.02)
sep.var.se(500, runif(150, 0.95, 0.99), 0.02)

## End(Not run)
```

sph *Population specificity*

Description

Population specificity

Usage

```
sph(n, sp)
```

Arguments

n	integer scalar or vector of number tested
sp	unit specificity of test (proportion)

Value

value vector of population-level specificities

Examples

```
## Not run:
# examples for sph - checked
sph(10, 0.9)
sph(c(10, 20, 50, 100), 0.99)
sph(100, c(0.999, 0.99, 0.98, 0.95, 0.9))

## End(Not run)
```

sph.binom *any one input can be a vector, all others must be either scalars or vectors of the same length*

Description

any one input can be a vector, all others must be either scalars or vectors of the same length

Usage

```
sph.binom(n, c = 1, sp)
```

Arguments

n	sample size
c	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
sp	unit specificity

Value

a vector of population-level specificities

Examples

```
## Not run:
# examples for sph.imperfect.sp
sph.binom(30, 2, 0.98)
sph.binom(30, 1, 0.98)
sph.binom(1:5*10, 2, 0.98)
sph.binom(100, 1:5, 0.98)
sph.binom(100, 3, 95:100/100)
sph.binom(c(5, 10, 15, 20, 30, 50, 100, 200), 2, 0.98)

## End(Not run)
```

---

sph.hp *HerdPlus herd specificity calculation any one input can be a vector, all others must be either scalars or vectors of the same length*

Description

HerdPlus herd specificity calculation any one input can be a vector, all others must be either scalars or vectors of the same length

Usage

```
sph.hp(N, n, c = 1, sp)
```

Arguments

N	population size
n	sample size
c	The cut-point number of reactors to classify a herd/flock as positive, default=1. if reactors < c result is negative, >= c is positive
sp	unit specificity

Value

a vector of population-level specificities

Examples

```
## Not run:
# examples of sph.hp
sph.hp(150, 30, 2, 0.98)
sph.hp(150, 30, 1, 0.98)
sph.hp(150, 1:5*10, 2, 0.98)
sph.hp(500, 30, 2, 95:100/100)

## End(Not run)
```

`sse.combined` *update between components to account for lack of independence*

Description

update between components to account for lack of independence

Usage

```
sse.combined(C = NA, pstar.c, rr, ppr, sep)
```

Arguments

`C`	NA or vector of population sizes (number of clusters) for each risk group
`pstar.c`	cluster level design prevalence
`rr`	cluster level relative risks
`ppr`	cluster level population proportions (not required if C is specified)
`sep`	list of sep values for clusters in each component and corresponding risk group. Each element is a dataframe, first column= clusterid, 2nd =rg.c, 3rd col = sep

Value

list of 2 elements, a matrix (or vector if C not specified) of population-level (surveillance system) sensitivities (binomial and hypergeometric and adjusted vs unadjusted) and a matrix of adjusted and unadjusted component sensitivities for each component

Examples

```
## Not run:
# example for sse.combined (checked in excel combined components.xlsx)
C<- c(300, 1200)
pstar<- 0.01
rr<- c(3,1)
ppr<- c(0.2, 0.8)
comp1<- data.frame(id=1:100, rg=c(rep(1,50), rep(2,50)), cse=rep(0.5,100))
comp2<- data.frame(id=seq(2, 120, by=2), rg=c(rep(1,25), rep(2,35)), cse=runif(60, 0.5, 0.8))
comp3<- data.frame(id=seq(5, 120, by=5), rg=c(rep(1,10), rep(2,14)), cse=runif(24, 0.7, 1))
sep<- list(comp1, comp2, comp3)
sse.combined(C, pstar, rr, sep = sep)
sse.combined(C=NA, pstar, rr, ppr, sep = sep)

## End(Not run)
```

`sse.rb.2stage` *calculate system sensitivity for 2 stage risk-based sampling*

Description

calculate system sensitivity for 2 stage risk-based sampling

Usage

```
sse.rb.2stage(C = NA, pstar.c, pstar.u, rr.c, ppr.c, rr.u, ppr.u, N = NA, n,
  rg, se)
```

Arguments

`C`	Population size (number of clusters), NA = unknown
`pstar.c`	cluster level design prevalence
`pstar.u`	unit level design prevalence
`rr.c`	cluster level relative risks (vector), NA if no cluster level risk factor if risk factor does not apply at either level use rr = c(1,1)
`ppr.c`	cluster level population proportions for risk categories (vector), NA if no cluster level risk factor
`rr.u`	unit level relative risks (vector), NA if no unit level risk factor if risk factor does not apply at either level use rr = c(1,1)
`ppr.u`	matrix, 1 row for each cluster, columns = unit level risk groups
`N`	cluster sizes NA or matrix of N for each risk group for each cluster, N=NA means cluster sizes not provided)
`n`	matrix, 1 row for each cluster, columns = unit level risk groups
`rg`	vector of cluster level risk group for each cluster
`se`	unit sensitivity for each cluster, scalar or vector of values for each cluster

Value

list of 2 elements, a scalar of population-level (surveillance system) sensitivity and a vector of cluster-level sensitivities

Examples

```
## Not run:
# examples for sse.rb.2stage
pstar.c<- 0.02
pstar.u<- 0.1
rr.c<- c(5, 1)
ppr.c<- c(0.1, 0.9)
rr.u<- c(3, 1)
se<- 0.9
n<- cbind(rep(10, 50), rep(5, 50))
rg<- c(rep(1, 30), rep(2, 20))
ppr.u<- cbind(rep(0.2, 50), rep(0.8, 50))
N<- cbind(rep(30, 50), rep(120, 50))
```

```
C<- 500
sse.rb.2stage(C=NA, pstar.c, pstar.u, rr.c, ppr.c, rr.u, ppr.u, N=NA, n, rg, se)
sse.rb.2stage(C, pstar.c, pstar.u, rr.c, ppr.c, rr.u, ppr.u, N=NA, n, rg, se)
sse.rb.2stage(C=NA, pstar.c, pstar.u, rr.c, ppr.c, rr.u, ppr.u, N, n, rg, se)
sse.rb.2stage(C, pstar.c, pstar.u, rr.c, ppr.c, rr.u, ppr.u, N, n, rg, se)

## End(Not run)
```

`tp` *estimate true prevalence*

Description

estimate true prevalence

Usage

```
tp(x, n, se, sp, type = "blaker", conf = 0.95)
```

Arguments

`x`	number of positive units (scalar)
`n`	sample size (no units sampled)
`se`	test unit sensitivity
`sp`	test unit specificity
`type`	one of c("normal", "c-p", "sterne", "blaker", "wilson", "all")
`conf`	desired level of confidence for CI, default = 0.95 note: all parameters must be scalars - won't work for vectors

Value

list with 2 elements, a matrix of apparent prevalence and lower and upper confidence limits and a matrix of true prevalence and lower and upper confidence limits using the chosen method(s)

Examples

```
## Not run:
# examples for tp
x<- 20
n<- 120
se<- 0.9
sp<- 0.99
conf<- 0.95
tp(x, n, se, sp, "all")
tp(x, n, se, sp, "c-p")
tp(x, n, 0.95, 0.9, "c-p")

## End(Not run)
```

`tp.normal` *normal approximation CI for tp*

Description

normal approximation CI for tp

Usage

```
tp.normal(x, n, se, sp, conf = 0.95)
```

Arguments

`x`	number of positive results in sample
`n`	sample size
`se`	test unit sensitivity
`sp`	test unit specificity
`conf`	desired level of confidence for CI, default = 0.95

Value

list with 2 elements, a matrix of apparent prevalence and wilson lower and upper confidence limits and a matrix of true prevalence and normal approximation lower and upper confidence limits

Examples

```
## Not run:
# examples for tp.normal
tp.normal(25, 120, 0.9, 0.99)
tp.normal(seq(5, 25, by=5), 120, 0.9, 0.99)

## End(Not run)
```

Index