

Package ‘RSurveillance’

October 9, 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	2
ap	3
binom.agresti	4
binom.cp	4
binom.jeffreys	5
disc.prior	6
epi.calc	6
n.2stage	7
n.ap	8
n.binom	8
n.c.freecalc	9
n.c.hp	10
n.freecalc	10
n.freedom	11

n.hp	12
n.hypergeo	13
n.pfree	13
n.pooled	14
n.rb	15
n.rb.varse	15
n.tp	16
pfree.1	17
pfree.calc	18
pfree.equ	18
pstar.calc	19
sd.tp	20
se.parallel	20
se.series	21
sep	21
sep.binom	22
sep.binom.imperfect	23
sep.exact	23
sep.freecalc	24
sep.hp	25
sep.hypergeo	25
sep.pfree	26
sep.pooled	27
sep.prior	27
sep.rb.bin	28
sep.rb.bin.varse	29
sep.rb.hypergeo	30
sep.rb.hypergeo.varse	30
sep.rb2.binom	31
sep.rb2.hypergeo	32
sep.sys	33
sep.var.se	34
sph	35
sph.binom	35
sph.hp	36
sse.combined	37
sse.rb.2stage	38
tp	39
tp.normal	40

Index **41**

adj.risk

Adjusted risk

Description

Calculates adjusted risk for given relative risk and population proportions

Usage

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

Arguments

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

Value

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

Examples

```
# 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))
```

ap	<i>Apparent prevalence</i>
----	----------------------------

Description

Estimates apparent prevalence and confidence limits for given sample size and result

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	method for estimating CI, 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

```
# 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")
```

binom.agresti	<i>Agresti-Coull confidence limits</i>
---------------	--

Description

Calculates Agresti-Coull confidence limits for a simple proportion (apparent prevalence)

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

```
# test binom.agresti
binom.agresti(25, 200)
binom.agresti(seq(10, 100, 10), 200)
binom.agresti(50, seq(100, 1000, 100))
```

binom.cp	<i>Clopper-Pearson exact confidence limits</i>
----------	--

Description

Calculates Clopper-Pearson exact binomial confidence limits for a simple proportion (apparent prevalence)

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

```
# test binom.cp
binom.cp(25, 200)
binom.cp(seq(10, 100, 10), 200)
binom.cp(50, seq(100, 1000, 100))
```

binom.jeffreys	<i>Jeffreys confidence limits</i>
----------------	-----------------------------------

Description

Calculates Jeffreys confidence limits for a simple proportion (apparent prevalence)

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

```
# test binom.jeffreys
binom.jeffreys(25, 200)
binom.jeffreys(seq(10, 100, 10), 200)
binom.jeffreys(50, seq(100, 1000, 100))
```

disc.prior	<i>Discounted prior probability of freedom</i>
------------	--

Description

Calculates the discounted prior probability of disease freedom, after adjusting for the probability of disease exceeding the design prevalence during the time period

Usage

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

Arguments

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

Value

vector of discounted prior probability of freedom

Examples

```
# 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)
```

epi.calc	<i>Effective probability of infection (EPI)</i>
----------	---

Description

Calculates effective probability of infection (adjusted design prevalence) for each risk group

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 to rr)

Value

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

Examples

```
# 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))
```

n.2stage	<i>2-stage freedom sample size</i>
----------	------------------------------------

Description

Calculates sample sizes for a 2-stage representative survey (sampling of clusters and units within clusters) for disease freedom or detection, assuming imperfect test sensitivity, perfect (100

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

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

Examples

```
# 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)
```

n.ap	<i>Sample size for apparent prevalence</i>
------	--

Description

Calculates sample size for estimating 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 equivalent to half the width of the 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

```
# examples of n.ap
n.ap(0.5, 0.1)
n.ap(0.5, 0.1, conf=0.99)
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))
```

n.binom	<i>Binomial sample size</i>
---------	-----------------------------

Description

Calculates sample size for demonstrating freedom or detecting disease using binomial approach and assuming imperfect test sensitivity, perfect (100) representative sampling

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

```
# 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)
```

n.c.freecalc	<i>Freecalc optimum sample size and specified cut-point number of reactors</i>
--------------	--

Description

Calculates optimum sample size and cut-point reactors to achieve specified population sensitivity, 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	test unit sensitivity
sp	test unit specificity, default=1
pstar	design prevalence
minSpH	minimum 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

```
# 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)
```

n.c.hp	<i>Hypergeometric (HerdPlus) optimum sample size and specified cut-point number of reactors</i>
--------	---

Description

Calculates optimum sample size and cut-point reactors to achieve specified population sensitivity, for given population size and other parameters, 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	test unit sensitivity
sp	test unit specificity, default=1
pstar	design prevalence
minSpH	minimum 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

```
# 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)
```

n.freecalc	<i>Freecalc sample size for a finite population and specified cut-point number of reactors</i>
------------	--

Description

Calculates sample size required for a specified population sensitivity, for a given population size, cut-point number of reactors and other parameters, using Freecalc algorithm. 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	test unit sensitivity
sp	test unit specificity, default=1
pstar	design prevalence
minSpH	minimum 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

```
# 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)
```

n.freedom	<i>Freedom sample size</i>
-----------	----------------------------

Description

Calculates sample size for demonstrating freedom or detecting disease using the appropriate method, depending on whether or not N provided (hypergeometric if N provided, binomial otherwise), assuming imperfect test sensitivity, perfect (100 and representative sampling

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

```
# 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))
```

n.hp

Hypergeometric (HerdPlus) sample size for finite population and specified cut-point number of reactors

Description

Calculates sample size to achieve specified population sensitivity with population specificity \geq specified minimum value, for given population size, cut-point number of reactors and other parameters, all parameters 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, \geq c is positive
se	test unit sensitivity
sp	test unit specificity, default=1
pstar	design prevalence
minSpH	minimum 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. Returns sample size for maximum achievable sep if it is not possible to achieve target sep AND SpP \geq minSpH.

Examples

```
# 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)
```

n.hypergeo	<i>Hypergeometric sample size</i>
------------	-----------------------------------

Description

Calculates sample size for demonstrating freedom or detecting disease using hypergeometric approximation and assuming imperfect test sensitivity, perfect (100 representative sampling

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

```
# 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)
```

n.pfree	<i>Sample size to achieve desired (posterior) probability of freedom</i>
---------	--

Description

Calculates the sample size required to achieve a given value for probability of disease 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 vector equal length to sep)
pstar	design prevalence
se	unit sensitivity (scalar or vector)
N	population size

Value

vector of sample sizes

Examples

```
# 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)
```

n.pooled

Sample size for pooled testing for freedom

Description

Calculates sample size to achieve desired population-level sensitivity, assuming pooled sampling and allowing for imperfect sensitivity and specificity of the pooled test

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

```
# 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)
```

n.rb	<i>Risk-based sample size</i>
------	-------------------------------

Description

Calculates sample size for risk-based sampling for a single risk factor and using binomial method

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

```
# 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)
```

n.rb.varse	<i>Risk-based sample size for varying unit sensitivity</i>
------------	--

Description

Calculates sample size for risk-based sampling for a single risk factor and varying unit sensitivity, using binomial method

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 smatrix 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

```
# 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)
```

n.tp

Sample size for true prevalence

Description

Calculates sample size for estimating true prevalence using normal approximation

Usage

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

Arguments

p	estimated true prevalence
se	test sensitivity
sp	test specificity
precision	absolute precision, +/- proportion equal to half the width of the desired confidence interval
conf	desired level of confidence for CI, default = 0.95

Value

a vector of sample sizes

Examples

```
# 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))
```

pfree.1	<i>Probability of freedom for single time period</i>
---------	--

Description

Calculates the posterior probability (confidence) of disease freedom (negative predictive value) 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 vector 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

```
# 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)
```

pfree.calc	<i>Probability of freedom over time</i>
------------	---

Description

Calculates the probability (confidence) of disease 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 vector 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

```
# 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)
```

pfree.equ	<i>Equilibrium probability of freedom</i>
-----------	---

Description

Calculates equilibrium probability of disease freedom and equilibrium prior probability of freedom, before discounting for, probability of introduction

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 vector equal length to sep)

Value

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

Examples

```
# 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))
```

pstar.calc

Design prevalence back-calculation

Description

Calculates design prevalence required for given sample size and desired population-level sensitivity, assuming imperfect test sensitivity, perfect (100 representative sampling

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

```
# 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))
```

sd.tp	<i>Standard deviation of true prevalence estimate</i>
-------	---

Description

Calculates the standard deviation of true prevalence estimate assuming se and sp known exactly, used to calculate normal approximation CI for estimate

Usage

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

Arguments

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

Value

vector of standard deviation values for true prevalence estimates

Examples

```
# example of sd.tp
sd.tp(1:10, 20, 0.9, 0.99)
```

se.parallel	<i>Sensitivity of tests in parallel</i>
-------------	---

Description

Calculates the combined sensitivity for multiple tests interpreted in parallel (assuming independence)

Usage

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

Arguments

se	vector of unit sensitivity values
----	-----------------------------------

Value

scalar of combined sensitivity, assuming independence

Examples

```
# examples for se.parallel
se.parallel(c(0.99, 0.95, 0.8))
```

se.series	<i>Sensitivity of tests in series</i>
-----------	---------------------------------------

Description

Calculates the combined sensitivity for multiple tests interpreted in series (assuming independence)

Usage

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

Arguments

se vector of unit sensitivity values

Value

scalar of combined sensitivity, assuming independence

Examples

```
# examples for se.series
se.series(c(0.99, 0.95, 0.8))
```

sep	<i>Population sensitivity</i>
-----	-------------------------------

Description

Calculates population sensitivity using appropriate method, depending on whether or not N provided (hypergeometric if N provided, binomial otherwise), assuming perfect (100 test specificity and representative sampling

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 length as n

Value

a vector of population-level sensitivities

Examples

```
# 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)
```

sep.binom

*Binomial Population sensitivity***Description**

Calculates population sensitivity for detecting disease, assuming imperfect test sensitivity and specificity and representative sampling, using binomial distribution (assumes large or unknown population size)

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), default = 1
sp	unit specificity of test (proportion), default = 1

Value

vector of population-level sensitivities

Examples

```
# 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)
```

sep.binom.imperfect *Binomial population sensitivity for imperfect test*

Description

Calculates population sensitivity for a large or unknown population and allowing for imperfect test sensitivity and specificity, using Binomial distribution

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	test unit sensitivity
sp	test unit specificity, default=1
pstar	design prevalence

Value

a vector of population-level sensitivities

Examples

```
# 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)
```

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

Description

Calculates population sensitivity for detecting disease assuming imperfect test sensitivity and perfect (100) and a census of all units in the population

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

```
# 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)))
```

sep.freecalc

FreeCalc population sensitivity for imperfect test

Description

Calculates population sensitivity for a finite population and allowing for imperfect test sensitivity and specificity, using Freecalc method

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	test unit sensitivity
sp	test unit specificity, default=1
pstar	design prevalence - assumed or target prevalence for detection of disease in the population

Value

population-level sensitivity

Examples

```
# 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)
```

sep.hp	<i>Hypergeometric (HerdPlus) population sensitivity for imperfect test</i>
--------	--

Description

Calculates population sensitivity for a finite population and allowing for imperfect test sensitivity and specificity, using Hypergeometric distribution

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	test unit sensitivity
sp	test unit specificity, default=1
pstar	design prevalence

Value

a vector of population-level sensitivities

Examples

```
# 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)
```

sep.hypergeo	<i>Hypergeometric Population sensitivity</i>
--------------	--

Description

Calculates population sensitivity for detecting disease, assuming imperfect test sensitivity and perfect (100 and representative sampling, using hypergeometric approximation (assumes known population size),

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

a vector of population-level sensitivities

Examples

```
# 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))
```

sep.pfree	<i>Population sensitivity to achieve desired (posterior) probability of freedom</i>
-----------	---

Description

Calculates the population sensitivity required to achieve a given value for probability of disease freedom

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

a vector of population-level sensitivities

Examples

```
# 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))
```

sep.pooled	<i>Pooled population sensitivity</i>
------------	--------------------------------------

Description

Calculates population sensitivity assuming pooled sampling and allowing for imperfect sensitivity and specificity of the pooled test

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

```
# 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)
```

sep.prior	<i>Population sensitivity to achieve desired prior probability of freedom</i>
-----------	---

Description

Calculates the population sensitivity required to achieve a given value for the prior (discounted) probability of disease freedom

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 vector equal length to sep)

Value

a vector of population-level sensitivities

Examples

```
# 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))
```

sep.rb.bin

Binomial risk-based population sensitivity

Description

Calculates risk-based population sensitivity with a single risk factor, using binomial method (assumes a large population)

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 to rr)
n	sample size per risk category (vector same length as rr and ppr)
se	unit sensitivity (fixed value 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

```
# 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))
```

sep.rb.bin.varse	<i>Binomial risk-based population sensitivity for varying unit sensitivity</i>
------------------	--

Description

Calculates population sensitivity for a single risk factor and varying unit sensitivity using binomial method (assumes large population)

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 to rr)
df	dataframe of values for each sensitivity level, col 1 = risk group index, col 2 = unit Se, col 3 = n (sample size for that risk group and unit sensitivity)

Value

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

Examples

```
# 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)
```

sep.rb.hypergeo	<i>Hypergeometric risk-based population sensitivity</i>
-----------------	---

Description

Calculates risk-based population sensitivity with a single risk factor, using the hypergeometric method (assuming a finite and known population size)

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 value or a vector the 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

```
# 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))
```

sep.rb.hypergeo.varse	<i>Hypergeometric risk-based population sensitivity for varying unit sensitivity</i>
-----------------------	--

Description

Calculates population sensitivity for a single risk factor and varying unit sensitivity using hypergeometric approximation method (assumes known population size)

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 to rr)
df	dataframe of values for each sensitivity level col 1 = risk group index, col 2 = unit Se, col 3 = n (sample size for risk group and unit sensitivity)

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

```
# 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)
```

sep.rb2.binom

*Binomial risk-based population sensitivity for 2 risk factors***Description**

Calculates risk-based population sensitivity for two risk factors, using binomial method (assumes a large population)

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 (rows = levels of rr1, cols = levels of rr2)
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

```
# 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)
```

sep.rb2.hypergeo

Hypergeometric risk-based population sensitivity for 2 risk factors

Description

Calculates risk-based population sensitivity for two risk factors, using hypergeometric approximation method (assumes a known population size)

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 (rows = levels of rr1, cols = levels of rr2)

n	matrix of number tested (sample size) for each risk group (rows = levels of rr1, cols = levels of rr2)
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

```
# 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)
```

sep.sys	<i>2-stage population sensitivity</i>
---------	---------------------------------------

Description

Calculates population-level (system) sensitivity for representative 2-stage sampling (sampling of clusters and units within clusters), assuming imperfect test sensitivity, perfect (100)

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, default = NA
N	population size within clusters NA if not provided, otherwise a vector of same length as n, default = NA
n	sample size (vector of number tested per cluster)
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), default = 1

Value

vector of population-level sensitivities

Note

if pstar.c is not a proportion N must be entered (and $N \geq n$)

Examples

```
# 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]])
```

sep.var.se

*Population sensitivity for varying unit sensitivity***Description**

calculates population-level sensitivity where unit sensitivity varies and using the appropriate method, depending on whether or not N provided (hypergeometric if N provided, binomial otherwise), assuming perfect (100 test specificity and representative sampling

Usage

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

Arguments

N	population size (number of units or clusters, $N \geq \text{length}(se)$) or NA if unknown
se	vector of unit sensitivity values
pstar	specified design prevalence

Value

a vector of population-level sensitivities

Examples

```
# 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)
```

sph	<i>Population specificity</i>
-----	-------------------------------

Description

Calculates population specificity assuming representative sampling

Usage

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

Arguments

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

Value

a vector of population-level specificities

Examples

```
# 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))
```

sph.binom	<i>Binomial population specificity for imperfect test</i>
-----------	---

Description

Calculates population specificity for a large or unknown population, using the Binomial distribution and adjusting for cut-point number of reactors

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	test unit specificity

Value

a vector of population-level specificities

Examples

```
# 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)
```

sph.hp

Hypergeometric population specificity calculation

Description

Calculates population specificity for a finite population and imperfect test, using Hypergeometric distribution

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	test unit specificity

Value

a vector of population-level specificities

Examples

```
# 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)
```

sse.combined

*System sensitivity by combining multiple components***Description**

Calculates overall system sensitivity for multiple components, accounting for lack of independence (overlap) between components

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

```
# 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)
```

sse.rb.2stage

*Two-stage risk-based system sensitivity***Description**

Calculates system sensitivity for 2 stage risk-based sampling, using either binomial or hypergeometric approximation

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 (default)
pstar.c	cluster level design prevalence
pstar.u	unit level design prevalence
rr.c	cluster level relative risks (vector), use rr.c = c(1,1) if risk factor does not apply
ppr.c	cluster level population proportions for risk categories (vector), NA if no cluster level risk factor
rr.u	unit level relative risks (vector), use rr.u = c(1,1) if risk factor does not apply
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

```
# 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)
```

tp	<i>True prevalence</i>
----	------------------------

Description

Estimates true prevalence and confidence limits for given sample size and result, according to specified method

Usage

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

Arguments

x	number of positive units (scalar)
n	sample size (no. units sampled) (scalar)
se	test sensitivity (scalar)
sp	test specificity (scalar)
type	method for estimating CI, one of c("normal", "c-p", "sterne", "blaker", "wilson", "all")
conf	desired level of confidence for CI, default = 0.95 (scalar)

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

```
# 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")
```

tp.normal

Normal approximation confidence limits for true prevalence

Description

Estimates true prevalence and confidence limits for estimates based on normal approximation

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

```
# examples for tp.normal
tp.normal(25, 120, 0.9, 0.99)
tp.normal(seq(5, 25, by=5), 120, 0.9, 0.99)
```


Index

*Topic **methods**

adj.risk, 2
ap, 3
binom.agresti, 4
binom.cp, 4
binom.jeffreys, 5
disc.prior, 6
epi.calc, 6
n.2stage, 7
n.ap, 8
n.binom, 8
n.c.freecalc, 9
n.c.hp, 10
n.freecalc, 10
n.freedom, 11
n.hp, 12
n.hypergeo, 13
n.pfree, 13
n.pooled, 14
n.rb, 15
n.rb.varse, 15
n.tp, 16
pfree.1, 17
pfree.calc, 18
pfree.equ, 18
pstar.calc, 19
sd.tp, 20
se.parallel, 20
se.series, 21
sep, 21
sep.binom, 22
sep.binom.imperfect, 23
sep.exact, 23
sep.freecalc, 24
sep.hp, 25
sep.hypergeo, 25
sep.pfree, 26
sep.pooled, 27
sep.prior, 27
sep.rb.bin, 28
sep.rb.bin.varse, 29
sep.rb.hypergeo, 30
sep.rb.hypergeo.varse, 30

sep.rb2.binom, 31
sep.rb2.hypergeo, 32
sep.sys, 33
sep.var.se, 34
sph, 35
sph.binom, 35
sph.hp, 36
sse.combined, 37
sse.rb.2stage, 38
tp, 39
tp.normal, 40

adj.risk, 2
ap, 3

binom.agresti, 4
binom.cp, 4
binom.jeffreys, 5

disc.prior, 6

epi.calc, 6

n.2stage, 7
n.ap, 8
n.binom, 8
n.c.freecalc, 9
n.c.hp, 10
n.freecalc, 10
n.freedom, 11
n.hp, 12
n.hypergeo, 13
n.pfree, 13
n.pooled, 14
n.rb, 15
n.rb.varse, 15
n.tp, 16

pfree.1, 17
pfree.calc, 18
pfree.equ, 18
pstar.calc, 19

sd.tp, 20
se.parallel, 20

se.series, [21](#)
sep, [21](#)
sep.binom, [22](#)
sep.binom.imperfect, [23](#)
sep.exact, [23](#)
sep.freecalc, [24](#)
sep.hp, [25](#)
sep.hypergeo, [25](#)
sep.pfree, [26](#)
sep.pooled, [27](#)
sep.prior, [27](#)
sep.rb.bin, [28](#)
sep.rb.bin.varse, [29](#)
sep.rb.hypergeo, [30](#)
sep.rb.hypergeo.varse, [30](#)
sep.rb2.binom, [31](#)
sep.rb2.hypergeo, [32](#)
sep.sys, [33](#)
sep.var.se, [34](#)
sph, [35](#)
sph.binom, [35](#)
sph.hp, [36](#)
sse.combined, [37](#)
sse.rb.2stage, [38](#)

tp, [39](#)
tp.normal, [40](#)