Package ‘DiagTestKit’

August 5, 2019

Type Package

Title Functions used in evaluating sensitivity and specificity at CVB Statistics

Description A package written by CVB Statistics to estimate the sensitivity and specificity of an experimental diagnostic test kit in accordance with CVB STATWI0002 supporting the 2018 revision to VSM 800.73.

Version 0.5.4

Date 2019-08-15

Depends R (>= 3.5)

Imports data.table, plyr, methods

VignetteBuilder knitr

Suggests knitr, testthat, ggplot2

License CC0

URL https://www.aphis.usda.gov/aphis/ourfocus/animalhealth/veterinary-biologics/biologics-regulations-and-guidance/ct_vb_statwi,
https://github.com/ABS-dev/DiagTestKit/blob/master/README.md

BugReports https://github.com/ABS-dev/DiagTestKit/issues

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Author CVB Statistics [cre],
CVB Statistics [aut] (starting 2017),
David Siev [aut] (until 2019)

Maintainer CVB Statistics <CVB.Data.Help@usda.gov>

R topics documented:

`betaParm` *Convert Beta Parameterizations*

Description

Convert between the paramaterizations of a beta distribution.

Usage

```
betaParm(B, to = "alpha.beta")
```

Arguments

`B`	`vector` A named vector specifying non-NULL values for 2 parameters. e.g. c(alpha=NA, beta=NA, mu=.6, theta=NA, phi=1.6, sigma2=NA) or just c(mu=.6, phi=1.6)
`to`	Specification of desired parameters, options are one of 'alpha.beta', 'mu.phi', 'mu.theta' or 'mu.sigma2'.

Value

`vector` A named vector with values for the parameters specified in the 'to' argument of the input.

Author(s)

DiagTestKit-package

cellS *Obtain cell counts*

Description

This function creates expected cell counts (and probabilities) for a specific test pattern based on the diagnostic characteristics of the reference test(s) and experimental test.

Usage

```
cellS(SnR, SpR, Prev, SnE, SpE, sus.perc, N, nstates)
```

Arguments

`SnR` `data.frame` Each column corresponds to one reference test. Row 1 contains the sensitivity for the reference test(s). Row 2 contains the probability of a suspect result as a fraction of the non-correct test result. This is a value between 0 and 1 (inclusive). Namely, P(T? | D+) = ψ = δ * (1 - π) where δ is the second row for a given column (reference test). $\delta = \frac{\psi}{(1-\pi)}$. Use a zero for a 2-state test (i.e. no suspect region).

`SpR` `data.frame` Each column corresponds to one reference test. Row 1 contains the specificity for each reference test. Row 2 contains the probability of a suspect result as a fraction of the non-correct test result. This is a value between 0 and 1 (inclusive). Namely, P(T? | D-) = ϕ = γ * (1 - θ) where γ is the second row for a given column (reference test). $\gamma = \frac{\phi}{(1-\theta)}$. Use a zero for a 2-state test (i.e. no suspect region).

`Prev` `vector` A named vector containing the prevalence for each population sampled.

`SnE` Sensitivity of the experimental test kit.

`SpE` Specificity of the experimental test kit.

`sus.perc` `vector` A vector containing 2 elements, c(δ, γ) for the experimental test kit. A vector of zeros for a 2-state experimental kit. δ and γ are values between 0 and 1 (inclusive) corresponding to the proportion of the remaining probability (i.e. 1 - π or 1 - θ) that is suspect (ψ or ϕ). $\delta = \frac{\psi}{(1-\pi)}$ and $\gamma = \frac{\phi}{(1-\theta)}$.

`N` `vector` A named vector containing the sample size for each population sampled.

`nstates` `vector` A vector with length one greater than the number of reference tests. The first element is the number of states of the experimental test and the remaining entries are the number of states of each reference test (using the same ordering as SnR and SpR).

Value

`vector` A vector of expected counts corresponding to the properties of the reference and experimental tests. The expected counts are obtained based on a conditional independence assumption of all test methods.

Author(s)

DiagTestKit-package

`cloppearSnSp` *Binomial confidence interval, Clopper-Pearson method.*

Description

Evaluate binomial confidence interval using Clopper-Pearson method. A function written by CVB Statistics to estimate the sensitivity and specificity of an experimental diagnostic test kit in accordance with CVB STATWI0002.

Usage

```
cloppearSnSp(dat, alpha = 0.05, est.Sn = TRUE)
```

Arguments

`dat`	`data.frame` A data frame with a column for the experimental test results, a column for the infallible reference test results, and a column for the corresponding count. The column name for the experimental test results must contatin 'exp' and the column name for the infallible reference test results must include 'ref'. The counts should be the last column.
`alpha`	Complement of confidence level.
`est.Sn`	logical (TRUE/FALSE) Indicating if the sensitivity and its confidence interval should be supplied (TRUE) or if the specificity and its confidence interval should be supplied (FALSE).

Value

An object of type cp that extends `list`.

calcVal Named vector of point estimates and estimated simulated intervals. See below.

data Test and Total values of the data. See below.

alpha Complement of the confidence interval as provided above.

If `est.Sn == TRUE`

`calcVal` is a list with the following elements

- **Sn** Sensitivity estimate.
- **Sn.LL** Lower confidence limit for sensitivity.
- **Sn.UL** Upper confidence limit for sensitivity.

`data` is a list with the following elements

- **Test.Positive** Number of experimental test positives.
- **Total.Positive** Total number of positive samples.

If `est.Sn == FALSE`

`calcVal` is a list with the following elements

- **Sp** Specificity estimate.
- **Sp.LL** Lower confidence limit for specificity.
- **Sp.UL** Upper confidence limit for specificity.

`data` is a list with the following elements

- **Test.Negative** Number of experimental test negatives.
- **Total.Negative** Total number of negative samples.

Author(s)

DiagTestKit-package

References

Clopper CJ, Pearson ES, 1934. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika* 26:404-413.

Examples

```
CP.Sn <- cloppearSnSp(dat = dat_infal, est.Sn = TRUE)
CP.Sn
# Sn = P(T+|D+): 0.987013 (95% CI: 0.953876, 0.998423)
CP.Sp <- cloppearSnSp(dat = dat_infal, est.Sn = FALSE)
CP.Sp
# Sp = P(T-|D-): 0.970297 (95% CI: 0.915643, 0.915643)
```

`cp-class` *cp-class*

Description

cp-class

`data1` *Data set for Example 1 in vignette "DiagTestKit Examples"*

Description

Samples randomly selected from a single population are tested by one 2–state fallible reference test and a 2–state experimental test.

Usage

```
data1
```

Format

A data frame with four rows and three variables:

exp_result result of experimental test (positive, negative).

ref1_result result of reference test (positive, negative).

count number of samples with the unique testing combination.

`data2` *Data set for Example 2 in vignette "DiagTestKit Examples"*

Description

Samples randomly selected from 3 populations are tested by one 2–state fallible reference test and a 2–state experimental test.

Usage

```
data2
```

Format

A data frame with 12 rows and four variables:

population population identifier (A, B, C.

exp_result result of experimental test (positive, negative).

ref_result result of reference test (positive, negative).

count number of samples with the unique testing combination for the specific population.

`data3` *Data set for Example 3 in vignette "DiagTestKit Examples"*

Description

Samples randomly selected from single population tested by one 3–state fallible reference test and a 2–state experimental test.

Usage

```
data3
```

Format

A data frame with six rows and three variables:

exp_result result of experimental test (positive, negative).

ref_result result of reference test (positive, negative, suspect).

count number of samples with the unique testing combination.

`data4` *Data set for Example 4 in vignette "DiagTestKit Examples"*

Description

Samples randomly selected from 3 populations tested by one fallible 3–state reference test and a 2–state experimental test.

Usage

```
data4
```

Format

A data frame with 18 rows and four variables:

population population identifier (A, B, C).

exp_result result of experimental test (positive, negative).

ref_result result of reference test (positive, negative, suspect).

count number of samples with the unique testing combination for the specific population.

`data5` *Data set for Example 5 in vignette "DiagTestKit Examples"*

Description

Samples randomly selected from 2 populations tested by one 2–state reference test and a 3–state experimental test.

Usage

```
data5
```

Format

A data frame with 11 rows and four variables:

Population population identifier (A, B).

exp_result result of experimental test (positive, negative, suspect).

ref_result result of reference test (positive, negative).

count number of samples with the unique testing combination for the specific population.

`data6` *Data set for Example 6 in vignette "DiagTestKit Examples"*

Description

Samples randomly selected from a single population tested with two 2–state reference test and a 2–state experimental test.

Usage

```
data6
```

Format

A data frame with 8 rows and four variables:

exp_result result of experimental test (positive, negative).

ref1_result result of first reference test (positive, negative).

ref2_result result of second reference test (positive, negative).

count number of samples with the unique testing combination.

data7 *Data set for Example 7 in vignette "DiagTestKit Examples"*

Description

Samples randomly selected from 2 populations tested with two 3–state reference test and a 2–state experimental test.

Usage

```
data7
```

Format

A data frame with 39 rows and five variables:

Population population identifier (A, B).

exp_result result of experimental test (positive, negative).

ref1_result result of first reference test (positive, negative, suspect).

ref2_result result of second reference test (positive, negative, suspect).

count number of samples with the unique testing combination for the specific population.

data8 *Data set for Example 8 in vignette "DiagTestKit Examples"*

Description

Samples randomly selected from 3 populations tested by three 2–state reference tests and a 3–state experimental test.

Usage

```
data8
```

Format

A data frame with 72 rows and six variables:

Population population identifier (A, B, C).

exp_result result of experimental test (positive, negative, suspect).

ref1_result result of first reference test (positive, negative).

ref2_result result of second reference test (positive, negative).

ref3_result result of third reference test (positive, negative).

count number of samples with the unique testing combination for the specific population.

dat_dichot *Assay Validation Sensitivity and Specificity (Diagnostic Kit Format) example from CVB Data Guide.*

Description

Results from testing by diagnostic test kits with a dichotomous response for assay validation. See CVB Data Guide Appendix 1.8. DiagnosticKitDichotomous.zip/Diagnostic Kit Dichotomous SenSpec Example/dichotomoussenspec_deviceinfo.csv

Usage

```
dat_dichot
```

Format

A data frame with 202 rows and 14 variables:

deviceID Device identifier; unique for each row.

serialID Identifier of the preparation used.

tech Identifier of the technician performing testing.

sampleID Unique identifier for the sample being tested.

animalID Unique identifier for an animal.

specimen Type of specimen (wholeblood, serum, plasma).

species Animal species.

mfg_date Date of preparation manufacturer.

date Test date.

visual_read Test interpretation by visual reading.

instr_read Test interpretation by instrument reading.

control_read Test interpretation of the control.

ref_result Test interpretation of the reference.

prod_code Veterinary Services Product Code

Source

https://www.aphis.usda.gov/animal_health/vet_biologics/publications/DiagnosticKitDichotomous.zip

`dat_infal` *Counts data used in vignette "DiagTestKit GettingStarted" section 4*

Description

Counts of the positive and negative results for experimental and reference tests. A 2-state experimental test when an infallible reference test has been used to determine the true disease status of each sample.

Usage

```
dat_infal
```

Format

A data frame with 4 rows and 3 variables:

Experimental Result of the experimental test (positive or negative).

Reference Result of the reference test (positive or negative).

Count Number of samples observed with the unique testing combination.

`emp.hpd` *Calculate the empirical hpd.*

Description

Empirical highest posterior density by shortest length interval.

Usage

```
emp.hpd(X, alpha)
```

Arguments

`X`	vector of values
`alpha`	1 - confidence

Value

highest posterior density (1-alpha) interval

Note

Uses type 7 `quantile`. Also used in package `MF`

Author(s)

DiagTestKit-package

`estimateSnSp` *Estimate Sensitivity and Specificity*

Description

A function written by CVB Statistics to estimate the sensitivity and specificity of an experimental diagnostic test kit in accordance with CVB STATWI0002.

Usage

```
estimateSnSp(dat, Sn.ref, Sp.ref, prev.pop, nsim = 1000,
  control = NULL)
```

Arguments

`dat` `data.frame` This is a data frame where the first column includes information for the population sampled (if more than one population is sampled). The next column is the possible outcomes of the experimental test followed by one column for the possible outcomes for each reference test (one column per test). The last column of the data frame provides the number of samples with each pattern of test outcomes. The columns must be included in the order described. If more than one population is sampled, the column name for the column containing the population information must be 'population'. The column containing the test results for the experimental test must have 'exp' in the name, such as experimental, experiment, exp, Exp, etc. The column names containing the reference test results much contain 'ref' in the name, such as Ref1, Ref2, ref1_results, Reference2, etc.

`Sn.ref` `data.frame` Each column corresponds to one reference test. Row 1 contains the sensitivity for the reference test(s). Row 2 contains the probability of a suspect result as a fraction of the non-correct test result. This is a value between 0 and 1 (inclusive). Namely, P(T? | D+) = ψ = δ * (1 - π) where δ is the second row for a given column (reference test). $\delta = \frac{\psi}{(1-\pi)}$. Use a zero for a 2-state test (i.e. no suspect region). Alternatively, if all reference tests are 2-state tests, the sensitivities can be input as a named vector. Specifically, each element in the vector must be given a name which includes 'ref' (see above) and the column names (or names of the elements within the vector) must match those for Sp.ref.

`Sp.ref` `data.frame` Each column corresponds to one reference test. Row 1 contains the specificity for each reference test. Row 2 contains the probability of a suspect result as a fraction of the non-correct test result. This is a value between 0 and 1 (inclusive). Namely, P(T? | D-) = ϕ = γ * (1 - θ) where γ is the second row for a given column (reference test). $\gamma = \frac{\phi}{(1-\theta)}$. Use a zero for a 2-state test (i.e. no suspect region). Alternatively, if all reference tests are 2-state tests, the specificities can can be input as a named vector. Specifically, each element in the vector must be given a name which includes 'ref' (see above) and the column names (or names of the elements within the vector) must match those for Sn.ref.

`prev.pop` `vector` A named vector containing the prevalence for each population sampled. The names in the vector must match the population labels used in 'dat'.

`nsim` The number of simulations to draw from the sensitivity and specificity distribution(s) for each reference test and the prevalence distribution from each population.

`control` list of control values to replace defaults. See `estimateSnSpControl` for details.

Value

An object of type `snsp` that extends `list`.

calcVal Point estimates and estimated simulated intervals for properties of the experimental kit. See below.

detailOut Detailed output values. See below.

input Simulated values. See below.

`calcVal`

A list with the following values which will include the following for both 2- and 3-state experimental tests –

- **Nsim** Number of simulations performed.
- **Confidence** 1 - α.
- **SnPE** Sensitivity point estimate obtained as the median of the estimated values.
- **SnInterval** Estimated simulated interval for sensitivity.
- **SpPE** Specificity point estimate obtained as the median of the estimated values.
- **SpInterval** Estimated simulated interval for specificity.

If three states, the list will also include –

- **SusDisPosPE** Point estimate for the probability of test suspect given disease positive (ψ) which is the median of the calculated values (ψ = δ*(1-π)).
- **SusDisPosInterval** Estimated simulated interval for the probability of test suspect given disease positive (ψ).
- **SusDisNegPE** Point estimate for the probability of test suspect given disease negative (ϕ) which is the median of the calculated values (ϕ = γ*(1-θ)).
- **SusDisNegInterval** Estimated simulated interval for the probability of test suspect given disease negative (ϕ).

`detailOut`

A list with the following detailed output values which will include the following for both 2- and 3-state experimental tests –

- **Exp.Sn** `vector` The optimized values for the sensitivity of the experimental test kit.
- **Exp.Sp** `vector` The optimized values for the specificity of the experimental test kit.
- **Converge** `vector` Each entry is an integer code detailing the convergence of the optimization for each iteration. 0 indicates successful completion. See also `optim`.
- **Message** `vector` Each entry includes a character string providing any additional information returned by the optimizer or NULL. See also `optim`.

If three states, the list will also inlcude –

- **Exp.pos.p** `vector` The optimized values for the proportion of the remaining probability (1-Sn) that corresponds to a suspect region for diseased samples, namely δ.
- **Exp.sus.pos** `vector` The values for P(T? | D+) (ψ) calculated from Exp.sn and Exp.pos.p. P(T?|D+) = δ * (1 - π).
- **Exp.neg.p** `vector` The optimized value for the proportion of the remaining probability (1-Sp) that corresponds to a suspect region for non-diseased samples, namely γ.
- **Exp.sus.neg** `vector` The values for P(T? | D-) (ϕ) calculated from Exp.sp and Exp.neg.p. P(T?|D-) = γ * (1 - θ).

`input`

A list containing the seed used and the simulated values.

- **seed** The seed used in the random generation of the distributions of sensitivity and specificity for all reference tests and prevalence of each population. See also `set.seed`
- **Sn.sims** `matrix` The simulated values for the sensitivity of each reference test and ψ where ψ was specified in the second row of Sn.ref (or zero if Sn.ref was a vector). The first two columns correspond to the first reference test, columns 3 and 4 to the second reference test if it exists, etc.
- **Sp.sims** `matrix` The simulated values for the specificity of each reference test and ϕ where ϕ was specified in the second row of Sp.ref (or zero is Sp.ref was a vector). The first two columns correspond to the first reference test, columns 3 and 4 to the second reference test if it exists, etc.
- **prev.sims** `matrix` The simulated values of prevalence for each population. Each column correspond to one population.

Author(s)

DiagTestKit-package

See Also

`estimateSnSpControl`

Examples

```
data.1 <- data.frame(exp_result = rep(c('positive', 'negative'), each = 2),
                     ref1_result = rep(c('positive', 'negative'), 2),
                     count = c(82, 11, 5, 22))
example.1 <- estimateSnSp(dat = data.1, Sn.ref = data.frame(ref = c(0.90, 0)),
                     Sp.ref = data.frame(ref=c(0.99, 0)), prev.pop=c(A=0.80),
                     control = estimateSnSpControl(seed = 64725))
example.1

# 1000  simulations
# 95 % Interval Estimates
#
#               Point.Estimate     Lower  Upper
# Sn = P(T+|D+)      0.9449821 0.9019639      1
# Sp = P(T-|D-)      0.9062769 0.7523346      1


## Not run:
```

```
data.2 <- data.frame(Population = rep(LETTERS[1:3], each = 24),
                exp_result = rep(rep(c('negative', 'positive', 'suspect'), each = 8), 3),
                     ref1_result = rep(rep(c('negative', 'positive'), each = 4), 9),
                     ref2_result = rep(rep(c('negative', 'positive'), each = 2), 18),
                     ref3_result = rep(c('negative', 'positive'), 36),
                     count = c(3, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 5, 1, 8, 11, 62,
                               0, 0, 0, 0, 0, 0, 0, 2, 27, 2, 3, 0, 4, 0, 1, 1, 0,
                               0, 1, 4, 1, 6, 7, 41, 0, 0, 0, 0, 0, 0, 0, 2, 57, 5,
                               6, 1, 9, 1, 1, 0, 0, 0, 0, 1, 0, 2, 2, 12, 1, 0, 0,
                               0, 0, 0, 0, 0))
example.2 <- estimateSnSp(dat = data.2,
  Sn.ref = data.frame(ref1 = c(0.92, 0),
                      ref2 = c(0.88, 0),
                      ref3 = c(0.85, 0)),
  Sp.ref = data.frame(ref1 = c(0.86, 0),
                      ref2 = c(0.90, 0),
                      ref3 = c(0.92, 0)),
  prev.pop = c(A = 0.95, B = 0.62, C = 0.18),
  control = estimateSnSpControl(seed = 865213))
# 1000  simulations
# 95 % Interval Estimates

#                Point.Estimate     Lower      Upper
# Sn = P(T+|D+)      0.96541704 0.8879949 1.00000000
# Sp = P(T-|D-)      0.98351924 0.9016964 1.00000000
# SsP = P(T?|D+)     0.02568427 0.0000000 0.06339616
# SsN = P(T?|D-)     0.01534125 0.0000000 0.05604950

## End(Not run)
```

`estimateSnSpControl` *control values for estimateSnSp*

Description

The values supplied in the function–call replace the defaults and a list with all possible arguments is returned. The returned list is used as the `control` argument to the function `estimateSnSp`.

Usage

```
estimateSnSpControl(seed = NULL, Sn.distn = NULL, Sn.spread = NULL,
  Sp.distn = NULL, Sp.spread = NULL, prev.distn = NULL,
  prev.spread = NULL, tolerance = 0.001, alpha = 0.05,
  step.size = 1e-06, parm = NULL, rep.iter = TRUE, iter.n = 50)
```

Arguments

`seed` The seed used in the random generation of the distributions of sensitivity and specificity for all reference tests and prevalence of each population. See also `set.seed`.

`Sn.distn` `vector` A named vector with length equal to the number of reference tests. Determines which disibution should be used for sampling sensitivity of each reference test. Inputs are ’beta’ or ’triangular’. Defaults to ’beta’ for each reference test.

`Sn.spread` `vector` A named vector with length equal to the number of reference tests. Describes the width of the distribution for the sensitivity of each reference test. Inputs are 'wide', 'medium', or 'narrow'. Defaults to 'wide' for each reference test.

`Sp.distn` `vector` A named vector with length equal to the number of reference tests. Determines which disibution should be used for sampling specificity of each reference test. Inputs are 'beta' or 'triangular'. Defaults to 'beta' for each reference test.

`Sp.spread` `vector` A named vector with length equal to the number of reference tests. Describes the width of the distribution for the specificity of each reference test. Inputs are 'wide', 'medium', or 'narrow'. Defaults to 'wide' for each reference test.

`prev.distn` `vector` A named vector with length equal to the number of populations. Determines which disibution should be used for sampling the prevalence of each population. Inputs are 'beta' or 'triangular'. Defaults to 'beta'.

`prev.spread` `vector` A named vector with length equal to the number of populations. Describes the width of the distribution for the prevalence of each population. Inputs are 'wide', 'medium', or 'narrow'. Defaults to 'wide' for each population.

`tolerance` Setting a limit on the pgtol used in the `optim` function with the 'L-BFGS-B' method. See also `optim`. Defaults to 1E-03.

`alpha` Significance levels. Defaults to 0.05.

`step.size` Provides the level of resolution in values simulated from a triangular distribution. Defaults to 1E-06.

`parm` `vector` Starting values for the optimization of the parameters of the experimental test. If the experimental test has 2 states, this vector is of length two with elements corresponding to sensitivity and specificity, respectively. If the experimental test has 3 states, this vector is of length 4 with elements corresponding to sensitivity (π), the proportion of 1-Sn corresponding to the suspect region for disease positive samples (δ), specificity (θ), and the proportion of 1-Sp corresponding to the suspect region for disease negative samples (γ). All values are between 0 and 1, inclusive.

`rep.iter` logical (TRUE/FALSE) Indicates if updates should be printed regarding the number of iterations completed. Defaluts to TRUE.

`iter.n` integer indicating the frequency of updates for the number of iterations completed. Defaluts to 50.

Value

A list with the following elements (as defined above): `seed, Sn.disn, Sn.spread,Sp.distn, Sp.spread, prev.distn, prev.spread, tolerance, step.size, parm`

Author(s)

DiagTestKit-package

Examples

```
estimateSnSpControl()
estimateSnSpControl(seed = 64725)
```

`get.simulated.values` *Get Simulated Values*

Description

Simulate values for use in optimization. This function is used to obtain draws from a distribution for the sensitivity and specificity for each reference test and for the prevalence of each population tested.

Usage

```
get.simulated.values(means, distn, spread, nsim, step.size, prevalence)
```

Arguments

`means`	`vector` A named vector containing point estimates for the prevalence of each population (when prevalence = TRUE, see below) or a data frame where each column corresponds to a reference test and the rows are sensitivity (π) and ψ (or specificity (θ) and ϕ).
`distn`	`vector` A vector of same length as `means`. Values may be one of `NULL`, 'beta', or 'triangular'. `NULL` will be treated as 'beta'
`spread`	`vector` A vector of same length as `means`. Values may be one of `NULL`, 'wide', 'medium', or 'narrow'. `NULL` will be treated as 'wide'.
`nsim`	The number of simulations to draw from the sensitivity and specificity distribution(s) for each reference test or the prevalence distribution from each population.
`step.size`	Provides the level of resolution in values simulated from a triangular distribution.
`prevalence`	logical (TRUE/FALSE) TRUE indicates that the function is simulating values of prevalence. This will determine the structure of the output.

Value

final.mat A matrix of simulated values. If prevalence is TRUE, final.mat will have the number of columns corresponding to the number of populations sampled else if prevalence is FALSE, final.mat will have number of columns twice the number of reference tests. The columns are sensitivity (or specificity) of the first reference test, the probability of a suspect result as a fraction of the non-correct test result (i.e. either δ or γ) for the first reference and continues in the same pattern for all reference tests.

Author(s)

DiagTestKit-package

get.values *Optimization of Sensitivity and Specificity*

Description

Determine final optimized values for the sensitivity and specificity of an experimental test kit (and probability of suspect given disease positive and given disease negative for a 3-state kit).

Usage

```
get.values(dat, SnR.vec, SpR.vec, prev.vec, N.vec, nstates, tolerance,
  rep.iter, iter.n, parm = NULL)
```

Arguments

`dat` `vector` A vector of counts ordered in a manner consistent with output from the cellS function.

`SnR.vec` `data.frame` Each column corresponds to one reference test. Row 1 contains the sensitivity for the reference test(s). Row 2 contains the probability of a suspect result as a fraction of the non-correct test result. This is a value between 0 and 1 (inclusive). Namely, P(T? | D+) = ψ = δ * (1 - π) where δ is the second row for a given column (reference test). $\delta = \frac{\psi}{(1-\pi)}$. Use a zero for a 2-state test (i.e. no suspect region).

`SpR.vec` `data.frame` Each column corresponds to one reference test. Row 1 contains the specificity for each reference test. Row 2 contains the probability of a suspect result as a fraction of the non-correct test result. This is a value between 0 and 1 (inclusive). Namely, P(T? | D-) = ϕ = γ * (1 - θ) where γ is the second row for a given column (reference test). $\gamma = \frac{\phi}{(1-\theta)}$. Use a zero for a 2-state test (i.e. no suspect region).

`prev.vec` `vector` A named vector containing the prevalence for each population sampled.

`N.vec` `vector` A named vector containing the sample size for each population sampled.

`nstates` `vector` A vector with length one more than the number of reference tests. The first element is the number of states of the experimental test and the remaining entries are the number of states of each reference test (using the same ordering as SnR.vec and SpR.vec).

`tolerance` Setting a limit on the pgtol used in the optim function with the 'L-BFGS-B' method. See also `optim`.

`rep.iter` logical (TRUE/FALSE) Indicates if updates should be printed regarding the number of iterations completed.

`iter.n` integer indicating the frequency of updates for the number of iterations completed.

`parm` `vector` A vector of starting values to be used for the optimization that is passed to `minCell`. For a 2-state experimental test, this is a vector of length 2 with entries (π, θ). For a 3-state experimental test, this is a vector of length 4 with entries (π, δ, θ, γ). See also `estimateSnSp`.

Value

A list:

The following will be returned for both 2 and 3-state experimental tests –

- sens.final `vector` The optimized values for the sensitivity of the experimental test kit.
- spec.final `vector` The optimized values for the specificity of the experimental test kit.
- converge `vector` Each entry is an integer code detailing the convergence of the optimization for each iteration. 0 indicates successful completion. See also `optim`.
- message `vector` Each entry includes a character string giving any additional information returned by the optimizer or NULL. See also `optim`.

If three states –

- δ `vector` The optimized values for the probability of a suspect result as a fraction of the non-correct test result for diseased samples.
- γ `vector` The optimized value for the probability of a suspect result as a fraction of the non-correct test result for non-diseased samples.

Author(s)

DiagTestKit-package

`minCell` *minimize cell*

Description

A function used for optimizing the values of sensitivity and specificity (and δ and γ for a 3-state kit). The objective function minimizes the sum of the squared deviations (expected - observed cell counts).

Usage

```
minCell(parm, SnR, SpR, Prev, xdat, N, nstates)
```

Arguments

`parm` `vector` A vector of starting values to be used for the optimization that is passed to `minCell`. For a 2-state experimental test, this is a vector of length 2 with entries (π, θ) For a 3-state experimental test, this is a vector of length 4 with entries (π, δ, θ, γ). See also `estimateSnSp`.

`SnR` `data.frame` Each column corresponds to one reference test. Row 1 contains the sensitivity for the reference test(s). Row 2 contains the probability of a suspect result as a fraction of the non-correct test result. This is a value between 0 and 1 (inclusive). Namely, P(T? | D+) = ψ = δ * (1 - π) where δ is the second row for a given column (reference test). $\delta = \frac{\psi}{(1-\pi)}$. Use a zero for a 2-state test (i.e. no suspect region).

`SpR` `data.frame` Each column corresponds to one reference test. Row 1 contains the specificity for each reference test. Row 2 contains the probability of a suspect result as a fraction of the non-correct test result. This is a value between 0 and 1 (inclusive). Namely, P(T? | D-) = ϕ = γ * (1 - θ) where γ is the second row for a given column (reference test). $\gamma = \frac{\phi}{(1-\theta)}$. Use a zero for a 2-state test (i.e. no suspect region).

`Prev` `vector` A named vector containing the prevalence for each population sampled.

`xdat` `vector` A vector of the observed cell counts.

`N` `vector` A named vector containing the sample size for each population sampled passed to `cellS`.

`nstates` `vector` A vector with length one more than the number of reference tests. The first element is the number of states of the experimental test and the remaining entries are the number of states of each reference test (using the same ordering as SnR and SpR).

Value

The sum of the squared deviations between the expected and observed cell counts.

Author(s)

DiagTestKit-package

`SampDist` *Create Triangular Distribution*

Description

Creates a discrete step/triangular distribution that can be used to sample values for sensitivity or specificity of a reference test or prevalence of a population.

Usage

```
SampDist(m, w, h, threestate = FALSE, suspect = 2/3,
  stepwidth = 0.005, sumOne = TRUE)
```

Arguments

`m` This is a point estimate for the parameter in which you are obtaining the distribution, e.g. sensitivity, specificity, or prevalence.

`w` `vector` A vector that provides the half widths of the 3 regions, (w1 closest, w3 farthest).

`h` `vector` A vector of "y" (pseduo-value until scaled to be a probability) corresponding to the height of the shoulder and the height of the plateau.

`threestate` logical (TRUE/FALSE) Indicates whether or not there is a "suspect" region (i.e. positive/suspect/negative).

`suspect` A fraction that indicates what percentage of the remaining probability would be assigned to the suspect region. For instance, if the function gives sensitivity and then the probability of "suspect" is (1 - sensitivity)*suspect.

`stepwidth`	distance between the 'x' in the discrete distribution, resolution of possible observations of the created distribution.
`sumOne`	whether to expresss 'p' as a proportion of its sum.

Value

`data.frame` of 'x', 'y', and 'p'.

Author(s)

DiagTestKit-package

`snsp-class` *snsp-class*

Description

snsp-class

`updateAlpha` *Update alpha values for existing simulation*

Description

Report interval estimates with updated alpha values, using a previously evaluated simulation.

Usage

```
updateAlpha(x, newAlpha)
```

Arguments

`x`	output from `estimateSnSp`
`newAlpha`	updated alpha value. Must be within [0, 1]

Value

an object of type `snsp`. See output for `estimateSnSp`

Author(s)

DiagTestKit-package

See Also

`estimateSnSpControl`

Examples

```
data.1 <- data.frame(exp_result = rep(c('positive', 'negative'), each = 2),
                     ref1_result = rep(c('positive', 'negative'), 2),
                     count = c(82, 11, 5, 22))
example.1 <- estimateSnSp(dat = data.1, Sn.ref = data.frame(ref = c(0.90, 0)),
                     Sp.ref = data.frame(ref=c(0.99, 0)), prev.pop=c(A=0.80),
                     control = estimateSnSpControl(seed = 64725))
example.1a <- updateAlpha(example.1, newAlpha = 0.25)
example.1a

# 1000  simulations
# 75 % Interval Estimates
#
#                    Point.Estimate     Lower     Upper
# Sn = P(T+|D+)      0.9449821          0.9053901 0.9791017
# Sp = P(T-|D-)      0.9062769          0.8336064 1.0000000
```

Index