Package ‘hint’

October 13, 2022

Type Package

Title Tools for Hypothesis Testing Based on Hypergeometric Intersection Distributions

Version 0.1-3

Date 2022-02-01

Author Alex T. Kalinka

Maintainer Alex T. Kalinka <alex.t.kalinka@gmail.com>

Description Hypergeometric Intersection distributions are a broad group of distributions that describe the probability of picking intersections when drawing independently from two (or more) urns containing variable numbers of balls belonging to the same n categories. <arXiv:1305.0717>.

License GPL (>= 2)

URL https://github.com/alextkalinka/hint

Imports graphics, grDevices

Encoding UTF-8

LazyLoad yes

NeedsCompilation yes

Repository CRAN

RoxygenNote 7.1.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Date/Publication 2022-02-02 14:40:02 UTC

R topics documented:

`add.distr` *add.distr*

Description

This function will add one or more distributions or hypothesis tests to an existing plot.

Usage

```
add.distr(..., cols = "blue", test.cols = "red")
```

Arguments

`...`	One or more distributions or objects of class hint.test.
`cols`	A character string vector naming the colours of the distributions. If length(cols) is less than the number of distributions, the colours will be recycled. Defaults to "blue".
`test.cols`	A character string vector naming the colours to use for the regions in which the cumulative probability of the hypothesis test was derived (if it exists). If length(test.cols) is less than the number of distributions, the colours will be recycled. Defaults to "red".

Value

Plots to the current device.

`Binomialintersection` *The Binomial Intersection Distribution*

Description

Density, distribution function, quantile function and random generation for the binomial intersection distribution.

Usage

```
dbint(n, A, range = NULL, log = FALSE)

pbint(n, A, vals, upper.tail = TRUE, log.p = FALSE)

qbint(p, n, A, upper.tail = TRUE, log.p = FALSE)

rbint(num = 5, n, A)
```

Arguments

`n`	An integer specifying the number of categories in the urns.
`A`	A vector of integers specifying the numbers of balls drawn from each urn. The length of the vector equals the number of urns.
`range`	A vector of integers specifying the intersection sizes for which probabilities (dhint) or cumulative probabilites (phint) should be computed (can be a single number). If range is NULL (default) then probabilities will be returned over the entire range of possible values.
`log`	Logical. If TRUE, probabilities p are given as log(p). Defaults to FALSE.
`vals`	A vector of integers specifying the intersection sizes for which probabilities (dhint) or cumulative probabilites (phint) should be computed (can be a single number). If range is NULL (default) then probabilities will be returned over the entire range of possible values.
`upper.tail`	Logical. If TRUE, probabilities are P(X >= v), else P(X <= v). Defaults to TRUE.
`log.p`	Logical. If TRUE, probabilities p are given as log(p). Defaults to FALSE.
`p`	A probability between 0 and 1.
`num`	An integer specifying the number of random numbers to generate. Defaults to 5.

Details

The binomial intersection distribution is given by

$$P(X = v|N) = \binom{b}{v} \left(\prod_{i=1}^{N-1} p_i \right)^v \left(1 - \prod_{i=1}^{N-1} p_i \right)^{b-v}$$

where b gives the sample size which is smallest. This is an approximation for the hypergeometric intersection distribution when n is large and b is small relative to the samples taken from the $N - 1$ other urns.

Examples

```
## Generate the distribution of intersections sizes:
dd <- dbint(20, c(10, 12, 11, 14))
## Restrict the range of intersections.
```

```
dd <- dbint(20, c(10, 12), range = 0:5)
## Generate cumulative probabilities.
pp <- pbint(29, c(15, 8), vals = 5)
pp <- pbint(29, c(15, 8), vals = 2, upper.tail = FALSE)
## Extract quantiles:
qq <- qbint(0.15, 23, c(12, 10))
## Generate random samples from Binomial intersection distributions.
rr <- rbint(num = 10, 18, c(9, 14))
```

`hint.dist.test` *hint.dist.test*

Description

Tests whether the absolute distance between two intersection sizes would be expected by chance, i.e. whether they fall into opposite tails of their respective Hypergeometric Intersection distributions.

Usage

```
hint.dist.test(d, n1, A1, n2, A2, q1 = 0, q2 = 0, alternative = "greater")
```

Arguments

`d` A positive integer specifying the observed distance to be tested.

`n1` An integer specifying the number of categories in the urns for the first distribution.

`A1` An integer vector specifying the number of balls drawn from urns for the first distribution.

`n2` An integer specifying the number of categories in the urns for the second distribution.

`A2` An integer vector specifying the number of balls drawn from the urns for the second distribution.

`q1` An integer specifying the number of categories with duplicates in the second urn of the first distribution. If 0 then the symmetric, singleton case is computed, otherwise the asymmetric, duplicates case is computed (see `Hyperintersection`).

`q2` An integer specifying the number of categories with duplicates in the second urn of the second distribution. If 0 then the symmetric, singleton case is computed, otherwise the asymmetric, duplicates case is computed (see `Hyperintersection`).

`alternative` A characer string specifying the hypothesis to be tested. Can be one of "greater", "less", or "two.sided".

Details

The distribution of absolute distances between two hypergeometric intersection sizes is given by

$$P(X = d) = \sum_{\{v_1, v_2\}_i \in D_d}^{|D_d|} P(v_{1_i} | n_1, a_1, b_1, ...) \cdot P(v_{2_i} | n_2, a_2, b_2, ...)$$

where D_d is the set of pairs of intersection sizes, $\{v_1, v_2\}$, with absolute differences of size d.

Value

An object of class `hint.dist.test`, which is a list containing the following components:

- `parameters` An integer vector giving the parameter values.
- `p.value` A numerical value giving the p-value associated with the test.
- `alternative` A character string naming the hypothesis that was tested.

`hint.test` *hint.test*

Description

Apply the hypergeometric intersection test to categorical data to test for enrichment or depletion of intersections between two samples.

Usage

```
hint.test(cats, draw1, draw2, alternative = "greater")
```

Arguments

`cats` A data frame or matrix with 3 columns; the first gives the category identifier, and the second and third give the number of balls belonging to this category in the first and second urns respectively.

`draw1` A vector of objects corresponding to the categories given in cats drawn from the first urn.

`draw2` A vector of objects corresponding to the categories given in cats drawn from the second urn.

`alternative` A characer string specifying the hypothesis to be tested. Can be one of "greater", "less", or "two.sided".

Details

The hypergeometric intersection distributions describe the distribution of intersection sizes when sampling without replacement from two separate urns in which reside balls belonging to the same n object categories (see `Hyperintersection`).

Value

An object of class `hint.test`, which is a list containing the following components:

- `parameters` An integer vector giving the parameter values.
- `p.value` A numerical value giving the p-value associated with the test.
- `alternative` A character string naming the hypothesis that was tested.

References

Kalinka, A. T. (2013). The probability of drawing intersections: extending the hypergeometric distribution. arXiv.1305.0717

`Hyperdistinct` *Drawing Distinct Categories from a Single Urn*

Description

Density, distribution function, quantile function and random generation for the distribution of distinct categories drawn from a single urn in which there are duplicates in q of the categories.

Usage

```
dhydist(n, a, q, range = NULL, log = FALSE)

phydist(n, a, q, vals, upper.tail = TRUE, log.p = FALSE)

qhydist(p, n, a, q, upper.tail = TRUE, log.p = FALSE)

rhydist(num = 5, n, a, q)
```

Arguments

`n`	An integer specifying the number of categories in the urn.
`a`	An integer specifying the number of balls drawn from the urn.
`q`	An integer specifying the number of categories in the urn which have duplicate members.
`range`	A vector of integers specifying the intersection sizes for which probabilities (dhydist) or cumulative probabilites (phydist) should be computed (can be a single number). If range is NULL (default) then probabilities will be returned over the entire range of possible values.
`log`	Logical. If TRUE, probabilities p are given as log(p). Defaults to FALSE.
`vals`	A vector of integers specifying the intersection sizes for which probabilities (dhydist) or cumulative probabilites (phydist) should be computed (can be a single number). If range is NULL (default) then probabilities will be returned over the entire range of possible values.
`upper.tail`	Logical. If TRUE, probabilities are P(X >= c), else P(X <= c). Defaults to TRUE.
`log.p`	Logical. If TRUE, probabilities p are given as log(p). Defaults to FALSE.
`p`	A probability between 0 and 1.
`num`	An integer specifying the number of random numbers to generate. Defaults to 5.

Examples

```
## Generate the distribution of distinct categories drawn from a single urn.
dd <- dhydist(20, 10, 12)
## Restrict the range of intersections.
dd <- dhydist(20, 10, 12, range = 5:10)
## Generate cumulative probabilities.
pp <- phydist(29, 15, 8, vals = 5)
pp <- phydist(29, 15, 8, vals = 2, upper.tail = FALSE)
## Extract quantiles:
qq <- qhydist(0.15, 23, 12, 10)
## Generate random samples based on this distribution.
rr <- rhydist(num = 10, 18, 9, 12)
```

`Hyperintersection` *The Hypergeometric Intersection Family of Distributions*

Description

The Hypergeometric Intersection Family of Distributions

Usage

```
dhint(n, A, q = 0, range = NULL, approx = FALSE, log = FALSE, verbose = TRUE)

phint(n, A, q = 0, vals, upper.tail = TRUE, log.p = FALSE)

qhint(p, n, A, q = 0, upper.tail = TRUE, log.p = FALSE)

rhint(num = 5, n, A, q = 0)
```

Arguments

`n` An integer specifying the number of categories in the urns.

`A` A vector of integers specifying the numbers of balls drawn from each urn. The length of the vector equals the number of urns.

`q` An integer specifying the number of categories in the second urn which have duplicate members. If q is 0 (default) then the symmetrical, singleton case is computed, otherwise the asymmetrical, duplicates case is computed (see Details).

`range` A vector of integers specifying the intersection sizes for which probabilities (dhint) or cumulative probabilites (phint) should be computed (can be a single number). If range is NULL (default) then probabilities will be returned over the entire range of possible values.

`approx` Logical. If TRUE, a binomial approximation will be used to generate the distribution.

`log` Logical. If TRUE, probabilities p are given as log(p). Defaults to FALSE.

`verbose` Logical. If TRUE, progress of calculation in the asymmetric, duplicates case is printed to the screen.

`vals` A vector of integers specifying the intersection sizes for which probabilities (dhint) or cumulative probabilites (phint) should be computed (can be a single number). If range is NULL (default) then probabilities will be returned over the entire range of possible values.

`upper.tail` Logical. If TRUE, probabilities are P(X >= c), else P(X <= c). Defaults to TRUE.

`log.p` Logical. If TRUE, probabilities p are given as log(p). Defaults to FALSE.

`p` A probability between 0 and 1.

`num` An integer specifying the number of random numbers to generate. Defaults to 5.

Details

The hypergeometric intersection distributions describe the distribution of intersection sizes when sampling without replacement from two separate urns in which reside balls belonging to the same n object categories. In the simplest case when there is exactly one ball in each category in each urn (symmetrical, singleton case), then the distribution is hypergeometric:

$$P(X = v) = \frac{\binom{a}{v}\binom{n-a}{b-v}}{\binom{n}{b}}$$

When there are three urns, the distribution is given by

$$P(X = v) = \frac{\binom{a}{v}\sum_i \binom{a-v}{i}\binom{n-a}{b-v-i}\binom{n-v-i}{c-v}}{\binom{n}{b}\binom{n}{c}}$$

If, however, we allow duplicates in $q \leq n$ of the categories in the second urn, then the distribution of intersection sizes is described by the following variant of the hypergeometric:

$$P(X = v) = \sum_{m=0}^{\alpha}\sum_{l=0}^{\beta}\sum_{j=0}^{l}\binom{n-q}{v-l}\binom{q}{l}\binom{q-l}{m}\binom{n-v-q+l}{a-v-m}\binom{l}{j}\binom{n+q-a-m-j}{b-v}/\binom{n}{a}\binom{n+q}{b}$$

Value

'dhint', 'phint', and 'qhint' return a data frame with two columns: v, the intersection size, and p, the associated p-values. 'rhint' returns an integer vector of random samples based on the hypergeometric intersection distribution.

References

Kalinka, A. T. (2013). The probability of drawing intersections: extending the hypergeometric distribution. arXiv.1305.0717

Examples

```
## Generate the distribution of intersections sizes without duplicates:
dd <- dhint(20, c(10, 12))
## Restrict the range of intersections.
dd <- dhint(20, c(10, 12), range = 0:5)
## Allow duplicates in q of the categories in the second urn:
dd <- dhint(35, c(15, 11), 22, verbose = FALSE)
## Generate cumulative probabilities.
pp <- phint(29, c(15, 8), vals = 5)
pp <- phint(29, c(15, 8), vals = 2, upper.tail = FALSE)
pp <- phint(29, c(15, 8), 23, vals = 2)
## Extract quantiles:
qq <- qhint(0.15, 23, c(12, 10))
qq <- qhint(0.15, 23, c(12, 10), 18)
## Generate random samples from Hypergeometric intersection distributions.
rr <- rhint(num = 10, 18, c(9, 14))
rr <- rhint(num = 10, 22, c(11, 17), 12)
```

`plot.hint.test` *plot.hint.test*

Description

This function visualises the results of a Hypergeometric Intersection test.

Usage

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

Arguments

`x`	An object of class ‘hint.test‘.
`...`	Additional arguments to be passed to ‘plot‘.

Details

Plots the relevant Hypergeometric Intersection distribution as a segment plot, and highlights the region where the observed statistic falls, i.e. the region from which the probability is computed (two.sided tests are visualised in one tail, the one with the smallest density). This can be especially useful for pedagogical purposes.

Value

Plots to the current device.

plotDistr *plotDistr*

Description

Plot a distribution or visualise the result of a hypothesis test.

Usage

```
plotDistr(
  distr,
  col = "black",
  test.col = "red",
  xlim = NULL,
  ylim = NULL,
  xlab = "Intersection size (v)",
  ylab = "Probability",
  add = FALSE,
  ...
)
```

Arguments

`distr` A data frame or matrix in which the first column gives random variable values, and the second gives probabilities. Can also be a vector (in which case random variables of 0:length(distr) will be automatically assigned, or an object of class hint.test.

`col` A character string naming the colour to use for the distribution. Defaults to "black".

`test.col` A character string naming the colour to use for the region in which the cumulative probability of the hypothesis test was derived (if it exists). Defaults to "red".

`xlim` A vector of two numbers giving the range for the x-axis. If NULL (default), then this is determined by the maximum and minimum values in distr.

`ylim` A vector of two numbers giving the range for the y-axis. If NULL (default), then this is determined by the maximum and minimum values in distr.

`xlab` A character string giving a label for the x-axis. Deafults to "Intersection size (v)".

`ylab` A character string giving a label for the y-axis. Deafults to "Probability".

`add` Logical. Whether the plot will be added to an existing plot or not. Defaults to FALSE.

`...` Additional arguments to be passed to plot.

Details

Visualising the results of a hypothesis test may often be of interest, but can be especially useful for pedagogical purposes.

Value

Plots to the current device.

`print.hint.test` *print.hint.test*

Description

Prints the resuls of ‘hint.test‘.

Usage

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

Arguments

`x`	An object of class ‘hint.test‘.
`...`	Additional arguments to be passed to ‘print‘.

Value

Prints output to the console.

Index