Package ‘convdistr’

April 20, 2021

Type Package
Title Convolute Probabilistic Distributions
Version 1.5.3
Date 2021-04-19
URL https://github.com/johnaponte/convdistr
Description Convolute probabilistic distributions using the random generator function of each distribution. A new random number generator function is created that perform the mathematical operation on the individual random samples from the random generator function of each distribution. See the documentation for examples.
License GPL (>= 3)
Encoding UTF-8
Suggests spelling, testthat, knitr, rmarkdown
Imports stats, graphics, pryr, extraDistr, dplyr, tidyr, ggplot2, RColorBrewer, SHELF, MASS, shiny
RoxygenNote 7.1.1
VignetteBuilder knitr
Language en-US
NeedsCompilation no
Author Aponte John [aut, cre] (<https://orcid.org/0000-0002-3014-3673>)
Maintainer Aponte John <john.j.aponte@gmail.com>
Repository CRAN
Date/Publication 2021-04-20 07:30:02 UTC

R topics documented:

  add_total ................................................................. 2
  BETA ................................................................. 3
  BINOMIAL ............................................................ 4
  cinqnum ............................................................... 5
  cinqnum.DIRAC ....................................................... 6
add_total

Description

This function returns a DISTRIBUTION with a new dimension created by row sum of the dimensions of the distribution.

Usage

```
add_total(p_distribution, p_totalname = "TOTAL")
```
Arguments

- `p_distribution` an object of class `DISTRIBUTION`
- `p_totalname` the name of the new dimension

Details

Only works with multidimensional distributions.

Value

- a `DISTRIBUTION`

Author(s)

John J. Aponte

Examples

```r
  d1 <- new_DIRICHLET(c(0.2,0.5,0.3))
  d2 <- add_total(d1)
```

---

**BETA**

Factory for a BETA distribution object

Description

Returns an BETA distribution object that produce random numbers from a beta distribution using the `rbeta` function

Usage

```r
  new_BETA(p_shape1, p_shape2, p_dimnames = "rvar")
  new_BETA_lci(p_mean, p_lci, p_uci, p_dimnames = "rvar")
  new_BETA_lci2(p_mean, p_lci, p_uci, p_dimnames = "rvar")
```

Arguments

- `p_shape1` non-negative parameters of the Beta distribution
- `p_shape2` non-negative parameters of the Beta distribution
- `p_dimnames` A character that represents the name of the dimension
- `p_mean` A numeric that represents the expected value of the proportion
- `p_lci` A numeric for the lower 95% confidence interval
- `p_uci` A numeric for the upper 95% confidence interval
Value

An object of class DISTRIBUTION, BETA

Functions

- new_BETA_lci: Constructor based on confidence intervals. Preserve expected value.
- new_BETA_lci2: Constructor based on ML confidence intervals

Note

When using confidence intervals, the shape parameters are obtained using the following formula:
\[ \text{varp} = \frac{(p_{uci} - p_{lci})}{4^2} \]
\[ \text{shape1} = p_{mean} \times \frac{(p_{mean} \times (1 - p_{mean})/\text{varp} - 1)}{\text{varp}} \]
\[ \text{shape2} = (1 - p_{mean}) \times \frac{(1 - p_{mean}) \times (1 - p_{mean})/\text{varp} - 1)}{\text{varp}} \]

new_BETA_lci2 estimate parameters using maximum likelihood
myDistr <- new_BETA_lci2(0.30,0.25,0.35)
myDistr$rfunc(10)

Author(s)

John J. Aponte

Examples

myDistr <- new_BETA(1,1)
myDistr$rfunc(10)
myDistr <- new_BETA_lci(0.30,0.25,0.35)
myDistr$rfunc(10)

BINOMIAL

Factory for a BINOMIAL distribution object

Description

Returns a BINOMIAL distribution object that produce random numbers from a binomial distribution using the rbinom function

Usage

new_BINOMIAL(p_size, p_prob, p_dimnames = "rvar")

Arguments

- p_size: integer that represent the number of trials
- p_prob: probability of success
- p_dimnames: A character that represents the name of the dimension
Value

An object of class `DISTRIBUTION`, `BINOMIAL`

Author(s)

John J. Aponte

Examples

```r
myDistr <- new_BINOMIAL(1000, 0.3)
myDistr$rfunc(10)
```

Description

Make a list with 5 numbers of the distribution (mean, sd, lci, uci, median).

Usage

```
cinqnum(x, ...)
```

Arguments

- `x` an object of class `DISTRIBUTION`
- `...` further parameters

Details

Uses the stored seed to have the same sequence always and produce the same numbers. This is an internal function for the summary function.

Value

A vector with the mean, sd, lci, uci and median values.

Author(s)

John J. Aponte
cinqnum.DISTRIBUTION

Description
And optimized version for DIRAC distributions

Usage
## S3 method for class 'DISTRIBUTION'
cinqnum(x, n)

Arguments
x an object of class DISTRIBUTION
n number of drawns

Value
a list of NA

Author(s)
John J. Aponte

Generic function for a distribution

Description
Generate n random numbers from the distribution, using the seed of the object, so always return the same value. Internal function to be used in the summary

Usage
## S3 method for class 'DISTRIBUTION'
cinqnum(x, n)

Arguments
x an object of class DISTRIBUTION
n number of drawns

Value
a list with the mean, sd, 95
Author(s)

John J. Aponte

---

Description

And optimized version for NA distribution

Usage

```r
## S3 method for class 'NA'
cinqnum(x, n)
```

Arguments

- `x`: an object of class `DISTRIBUTION`
- `n`: number of drawns

Value

a list of NA

Author(s)

John J. Aponte

---

convdistr

convdistr: A package useful for convolution of distributions.

Description

The convdistr package provides tools to define `DISTRIBUTION` objects and make mathematical operations with them. It keeps track of the results as if they were scalar numbers but maintaining the ability to obtain random samples of the convoluted distributions.
Make the convolution of two or more DISTRIBUTION objects

Description

The convolution of the simple algebraic operations is made by the operation of individual draws of the distributions. The DISTRIBUTION objects must have the same dimensions.

Usage

new_CONVOLUTION(listdistr, op, omit_NA = FALSE)

new_SUM(..., omit_NA = FALSE)

## S3 method for class 'DISTRIBUTION'
e1 + e2

new_SUBTRACTION(..., omit_NA = FALSE)

## S3 method for class 'DISTRIBUTION'
e1 - e2

new_MULTIPLICATION(..., omit_NA = FALSE)

## S3 method for class 'DISTRIBUTION'
e1 * e2

new_DIVISION(..., omit_NA = FALSE)

## S3 method for class 'DISTRIBUTION'
e1 / e2

Arguments

listdistr a list of DISTRIBUTION objects

op a function to convolute `+`, `*`, `/`

omit_NA if TRUE, NA distributions will be omitted

... DISTRIBUTION objects or a list of distribution objects

e1 object of class DISTRIBUTION

e2 object of class DISTRIBUTION

Details

If any of the distributions is of class NA (NA_DISTRIBUTION) the result will be a new distribution of class NA unless the omit_NA option is set to TRUE
CONVOLUTION_assoc

Value

and object of class CONVOLUTION, DISTRIBUTION

Functions

• new_SUM: Sum of distributions
• new_SUBTRACTION: Subtraction for distributions
• new_MULTIPLICATION: Multiplication for distributions
• new_DIVISION: DIVISION for distributions

Author(s)

John J. Aponte

Examples

x1 <- new_NORMAL(0,1)
x2 <- new_UNIFORM(1,2)
new_CONVOLUTION(list(x1,x2), `+`)
new_SUM(x1,x2)
x1 + x2
new_SUBTRACTION(x1,x2)
x1 - x2
new_MULTIPLICATION(list(x1,x2))
x1 * x2
new_DIVISION(list(x1,x2))
x1 / x2

CONVOLUTION_assoc  Convolution with association of dimensions

Description

In case of different dimensions of the distribution this function perform the operation on the com-
mon distributions and add without modifications the other dimensions of the distribution.

Usage

new_CONVOLUTION_assoc(dist1, dist2, op)
new_SUM_assoc(dist1, dist2)
new_SUBTRACTION_assoc(dist1, dist2)
new_MULTIPLICATION_assoc(dist1, dist2)
new_DIVISION_assoc(dist1, dist2)
**CONVOLUTION_comb**

**Arguments**

- **dist1**: an object of class `DISTRIBUTION`
- **dist2**: and object of class `DISTRIBUTION`
- **op**: one of `'+', '-', '*','/'`

**Details**

If distribution A have dimensions a and b and distribution B have dimensions b and c, the A + B would produce a distribution with dimensions a, c, b+b,

**Value**

an object of class `DISTRIBUTION`

**Functions**

- **new_SUM_assoc**: Sum of distributions
- **new_SUBTRACTION_assoc**: Subtraction of distributions
- **new_MULTIPLICATION_assoc**: Multiplication of distributions
- **new_DIVISION_assoc**: Division of distributions

**Author(s)**

John J. Aponte

**Examples**

```r
x1 <- new_MULTINORMAL(c(0,1), matrix(c(1,0.5,0.5,1),ncol=2), p_dimnames = c("A","B"))
x2 <- new_MULTINORMAL(c(10,1), matrix(c(1,0.4,0.4,1),ncol=2), p_dimnames = c("B","C"))
new_CONVOLUTION_assoc(x1,x2, `+`)
new_SUM_assoc(x1,x2)
new_SUBTRACTION_assoc(x1,x2)
new_MULTIPLICATION_assoc(x1,x2)
new_DIVISION_assoc(x1,x2)
```

**Description**

In case of different dimensions of the distribution this function perform the operation on the combination of the distributions of both distribution.
Usage

new_CONVOLUTION_comb(dist1, dist2, op, p_dimnames)
new_SUM_comb(dist1, dist2)
new_SUBTRACTION_comb(dist1, dist2)
new_MULTIPLICATION_comb(dist1, dist2)
new_DIVISION_comb(dist1, dist2)

Arguments

dist1: an object of class DISTRIBUTION

dist2: and object of class DISTRIBUTION

op: one of '+','-','*','/'

p_dimnames: a character vector with the name of the dimensions. If missing the combination of the individual dimensions will be used

Details

If distribution A have dimensions a and b and distribution B have dimensions b and c, the A + B would produce a distribution with dimensions a_b,a_c,b_b, b_c

Value

an object of class DISTRIBUTION

Functions

- new_SUM_comb: Sum of distributions
- new_SUBTRACTION_comb: Subtraction of distributions
- new_MULTIPLICATION_comb: Multiplication of distributions
- new_DIVISION_comb: Division of distributions

Note

In case of the same dimensions, only the first combination is taken

Author(s)

John J. Aponte
Examples

```r
x1 <- new_MULTINORMAL(c(0,1), matrix(c(1,0.5,0.5,1),ncol=2), p_dimnames = c("A","B"))
x2 <- new_MULTINORMAL(c(10,1), matrix(c(1,0.4,0.4,1),ncol=2), p_dimnames = c("B","C"))
new_CONVOLUTION_comb(x1,x2, `+`)  
new_SUM_comb(x1,x2)
new_SUBTRACTION_comb(x1,x2)
new_MULTIPLICATION_comb(x1,x2)
new_DIVISION_comb(x1,x2)
```

---

**DIRAC**

Factory for a DIRAC distribution object

---

**Description**

Returns an DIRAC distribution object that always return the same number, or the same matrix of numbers in case multiple dimensions are setup

**Usage**

```r
new_DIRAC(p_scalar, p_dimnames = "rvar")
```

**Arguments**

- `p_scalar` A numeric that set the value for the distribution
- `p_dimnames` A character that represents the name of the dimension

**Value**

An object of class DISTRIBUTION, DIRAC

**Author(s)**

John J. Aponte

**Examples**

```r
myDistr <- new_DIRAC(1)
myDistr$rfunc(10)
```
**DIRICHLET**

*Factory for a DIRICHLET distribution object*

**Description**

Returns an DIRICHLET distribution object that draw random numbers generated by the function `rdirichlet`

**Usage**

`new_DIRICHLET(p_alpha, p_dimnames)`

**Arguments**

- `p_alpha`: k-value vector for concentration parameter. Must be positive
- `p_dimnames`: A vector of characters for the names of the k-dimensions

**Details**

A name can be provided for the dimensions. Otherwise `rvar1, rvar2, ..., rvark` will be assigned

**Value**

An object of class `DISTRIBUTION, p_distribution$distribution, TRUNCATED`

**Author(s)**

John J. Aponte

**Examples**

```r
myDistr <- new_DIRICHLET(c(0.3,0.2,0.5), c("a","b","c"))
myDistr$rfunc(10)
```

---

**DISCRETE**

*Factory for a DISCRETE distribution object*

**Description**

Returns an DISCRETE distribution object that sample from the vector `p_supp` of options with probability the vector of probabilities `p_prob`.

**Usage**

`new_DISCRETE(p_supp, p_prob, p_dimnames = "rvar")`
Arguments

- **p_supp**: A numeric vector of options
- **p_prob**: A numeric vector of probabilities.
- **p_dimnames**: A character that represents the name of the dimension

Value

An object of class DISTRIBUTION, DISCRETE

Note

If the second argument is missing, all options will be sampled with equal probability. If provided, the second argument would add to 1 and must be the same length that the first argument.

Author(s)

John J. Aponte

Examples

```r
myDistr <- new_DISCRETE(p_supp=c(1,2,3,4), p_prob=c(0.40,0.30,0.20,0.10))
myDistr$rfunc(10)
```

---

**DISTRIBUTION**

**DISTRIBUTION class**

Description

DISTRIBUTION is a kind of abstract class (or interface) that the specific constructors should implement.

Details

It contains 4 fields

- **distribution**: A character with the name of the distribution implemented
- **seed**: A numerical that is used for details to produce reproducible details of the distribution
- **oval**: Observed value. Is the value expected. It is used as a number for the mathematical operations of the distributions as if they were a simple scalar
- **rfunc**: A function that generates random numbers from the distribution. Its only parameter `n` is the number of draws of the distribution. It returns a matrix with as many rows as `n`, and as many columns as the dimensions of the distributions
The DISTRIBUTION objects could support multidimensional distributions for example **DIRICHLET**. The names of the dimensions should coincide with the names of the oval vector. If only one dimension, the default name is *rvar*.

It is expected that the *rfunc* is included in the creation of new distributions by convolution so the environment should be carefully controlled to avoid reference leaking that is possible within the R language. For that reason, *rfunc* should be created within a `restrict_environment` function.

Once the object is instanced, the fields are immutable and should not be changed. If the seed needs to be modified, a new object can be created using the `set_seed` function.

Objects are defined for the following distributions:

- **UNIFORM**
- **NORMAL**
- **BETA**
- **TRIANGULAR**
- **POISSON**
- **EXPONENTIAL**
- **DISCRETE**
- **DIRAC**
- **DIRICHLET**
- **TRUNCATED**
- **NA_DISTRIBUTION**

**Value**

A DISTRIBUTION object

**Author(s)**

John J. Aponte

---

**DISTRIBUTION\_factory**  
* A factory of DISTRIBUTION classes

**Description**

Generate a function that creates DISTRIBUTION objects

**Usage**

`DISTRIBUTION\_factory(distname, rfunction, ovalfunc)`
**EXPONENTIAL**

**Arguments**

- **distname**: name of the distribution. By convention they are upper case
- **rfunction**: a function to generate random numbers from the distribution
- **ovalfunc**: a function that calculate the oval value, should used only the same arguments that the rfunction

**Value**

A function that is able to create **DISTRIBUTION** objects.

**Note**

The function return a new function, that have as arguments the formals of the rfunction plus a new argument dimnames for the dimension names. If The distribution is unidimensional, the default value dimnames = "rvar" will works well, but if not, the dimnames argument should be specified when the generated function is used as in the example for the new_MyDIRICHLET

**Author(s)**

John J. Aponte

**Examples**

```r
code here
```

**Description**

Returns an EXPONENTIAL distribution object that produce random numbers from an exponential distribution using the rexp function

**Usage**

```r
ew_EXPONENTIAL(p_rate, p_dimnames = "rvar")
```
**Arguments**

- `p_rate` A numeric that represents the rate of events
- `p_dimnames` A character that represents the name of the dimension

**Value**

An object of class DISTRIBUTION, EXPONENTIAL

**Author(s)**

John J. Aponte

**Examples**

```r
myDistr <- new_EXPONENTIAL(5)
myDistr$rfunc(10)
```

---

**Description**

Fits a beta distribution based on quantiles

**Usage**

```r
fitbeta_ml(point, lci, uci)
fitbeta(point, lci, uci)
```

**Arguments**

- `point` Point estimates corresponding to the median
- `lci` Lower limit (quantile 0.025)
- `uci` Upper limit (quantile 0.975)

**Value**

parameters shape1 and shape2 of a beta distribution

**Functions**

- `fitbeta_ml`: using ML to estimate parameters
- `fitbeta`: preserve the expected value
Note
This is a wrap of the `fitdist` to obtain the best parameters for a beta distribution based on quantiles. When using confidence intervals (not ML), the shape parameters are obtained using the following formula:

\[
varp = (p_{uci} - p_{ci})/4^2 \\
shape1 = p_{mean} \times (p_{mean} \times (1 - p_{mean})/varp - 1) \\
shape2 = (1 - p_{mean}) \times (p_{mean} \times (1 - p_{mean})/varp - 1)
\]

Author(s)
John J. Aponte

See Also
`fitdist`

Examples
```r
fitbeta_ml(0.45, 0.40, 0.50)
fitbeta(0.45, 0.40, 0.50)
```

Description
Fits a Dirichlet distribution based on the parameters of Beta distributions

Usage
```r
fitdirichlet(..., plotBeta = FALSE, n.fitted = "opt")
```

Arguments
- `...`: named vectors with the distribution parameters `shape1`, `shape2`
- `plotBeta`: if TRUE a ggplot of the densities are plotted
- `n.fitted`: Method to fit the values

Details
Each one of the arguments is a named vector with values for `shape1`, `shape2`. Values from `fitbeta` are suitable for this. This is a wrap of `fitDirichlet`

Value
a vector with the parameters for a Dirichlet distribution
ggDISTRIBUTION

Author(s)
John J. Aponte

See Also
fitDirichlet

Examples

```r
a <- fitbeta(0.3, 0.2, 0.4)
c <- fitbeta(0.2, 0.1, 0.3)
b <- fitbeta(0.5, 0.4, 0.6)
fitdirichlet(cat1=a, cat2=b, cat3=c)
```

Description

Plot of DISTRIBUTION objects using ggplot2

Usage

```r
ggDISTRIBUTION(x, n = 10000)
```

Arguments

- `x` an object of class DISTRIBUTION
- `n` number of observation

Value

a ggplot object with the density of the distribution

Examples

```r
x <- new_NORMAL(0,1)
foreach(x, plot(x, plot=FALSE))
ggDISTRIBUTION(x)
y <- new_DIRICHLET(c(10,20,70))
ggDISTRIBUTION(x)
```
LOGNORMAL  Factory for a LOGNORMAL distribution object

Description
Returns a LOGNORMAL distribution object that produce random numbers from a log normal distribution using the \texttt{rlnorm} function

Usage
\begin{verbatim}
new_LOGNORMAL(p_meanlog, p_sdlog, p_dimnames = "rvar")
\end{verbatim}

Arguments
- \texttt{p_meanlog} mean of the distribution on the log scale
- \texttt{p_sdlog} A numeric that represents the standard deviation on the log scale
- \texttt{p_dimnames} A character that represents the name of the dimension

Value
An object of class \texttt{DISTRIBUTION}, \texttt{LOGNORMAL}

Author(s)
John J. Aponte

Examples
\begin{verbatim}
myDistr <- new_LOGNORMAL(0,1)
myDistr$rfunc(10)
\end{verbatim}

metadata  Metadata for a DISTRIBUTION

Description
Shows the distribution and the oval values of a \texttt{DISTRIBUTION} object

Usage
\begin{verbatim}
metadata(x)
\end{verbatim}

## S3 method for class 'DISTRIBUTION'
metadata(x)

## Default S3 method:
metadata(x)
NA_DISTRIBUTION

Arguments

x a DISTRIBUTION object

Value

A data.frame with the metadata of the distributions

Methods (by class)

• DISTRIBUTION: Metadata for DISTRIBUTION objects
• default: Metadata for other objects

Note

The number of columns depends on the dimensions of the distribution. There will be one column distribution with the name of the distribution and one column for each dimension with the names from the oval field.

Author(s)

John J. Aponte

Description

Returns an NA distribution object that always return NA_real. This is useful to handle NA. By default only one dimension rvar is produced, but if several names are provided more columns will be added to the return matrix.

Usage

new NA(p_dimnames = "rvar")

Arguments

p_dimnames A character that represents the the names of the dimensions. By default only one dimension with name rvar

Value

An object of class DISTRIBUTION, NA

Author(s)

John J. Aponte
Examples

```r
myDistr <- new NAFTA(p_dimnames = "rvar")
myDistr$rfunc(10)
```

---

**new_MIXTURE**

*Mixture of DISTRIBUTION objects*

Description

Produce a new distribution that obtain random draws of the mixture of the DISTRIBUTION objects

Usage

```r
new_MIXTURE(listdistr, mixture)
```

Arguments

- `listdistr`: a list of DISTRIBUTION objects
- `mixture`: a vector of probabilities to mixture the distributions. Must add 1 If missing the draws are obtained from the distributions with the same probability

Value

an object of class MIXTURE, DISTRIBUTION

Author(s)

John J. Aponte

Examples

```r
x1 <- new NORMAL(0,1)
x2 <- new NORMAL(4,1)
x3 <- new NORMAL(6,1)
new_MIXTURE(list(x1,x2,x3))
```
**new_MULTINORMAL**

**Multivariate Normal Distribution**

**Description**

Return a DISTRIBUTION object that draw random numbers from a multivariate normal distribution using the mvrnorm function.

**Usage**

```
new_MULTINORMAL(p_mu, p_sigma, p_dimnames, tol = 1e-06, empirical = FALSE)
```

**Arguments**

- `p_mu`: a vector of means
- `p_sigma`: a positive-definite symmetric matrix for the covariance matrix
- `p_dimnames`: A character that represents the name of the dimension
- `tol`: tolerance (relative to largest variance) for numerical lack of positive-definiteness in `p_sigma`. 
- `empirical`: logical. If true, `mu` and `Sigma` specify the empirical not population mean and covariance matrix.

**Value**

An object of class DISTRIBUTION, MULTINORMAL

**Author(s)**

John J. Aponte

**See Also**

mvrnorm

**Examples**

```r
msigma <- matrix(c(1,0,0,1), ncol=2)
d1 <- new_MULTINORMAL(c(0,1), msigma)
rfunc(d1, 10)
```
NORMAL

Factory for a NORMAL distribution object

Description

Returns a NORMAL distribution object that produce random numbers from a normal distribution using the rnorm function

Usage

new_NORMAL(p_mean, p_sd, p_dimnames = "rvar")

Arguments

p_mean  A numeric that represents the mean value
p_sd    A numeric that represents the standard deviation
p_dimnames  A character that represents the name of the dimension

Value

An object of class DISTRIBUTION, NORMAL

Author(s)

John J. Aponte

Examples

myDistr <- new_NORMAL(0,1)
myDistr$rfunc(10)

omit_NA

Omit NA distributions from a list of distributions

Description

Omit NA distributions from a list of distributions

Usage

omit_NA(listdistr)

Arguments

listdistr  a list of DISTRIBUTION objects
plot.DISTRIBUTION

Value

the list without the NA_DISTRIBUTION

Author(s)

John J. Aponte

plot.DISTRIBUTION  plot of DISTRIBUTION objects

Description

Plot an histogram of the density of the distribution using random numbers from the distribution

Usage

## S3 method for class 'DISTRIBUTION'
plot(x, n = 10000, ...)

Arguments

x an object of class DISTRIBUTION
n number of observations
... other parameters to the hist function

Value

No return value. Side effect plot the histogram.

Examples

x <- new_NORMAL(0,1)
plot(x)
y <- new_DIRICHLET(c(10,20,70))
plot(x)
POISSON

Factory for a POISSON distribution using confidence intervals

Description

Returns an POISSON distribution object that produce random numbers from a Poisson distribution using the rpois function

Usage

new_POISSON(p_lambda, p_dimnames = "rvar")

Arguments

p_lambda | A numeric that represents the expected number of events
p_dimnames | A character that represents the name of the dimension

Value

An object of class DISTRIBUTION, POISSON

Author(s)

John J. Aponte

Examples

myDistr <- new_POISSON(5)
myDistr$rfunc(10)

restrict_environment

Build a new function with a smaller environment

Description

As standard feature, R include in the environment of a function all the variables that are available when the function is created. This, however is prompt to leak reference when you have a factory of function and they are created within a list.. it will include all the component of the list in the function environment. To prevent that, the random generator functions are encapsulated with a restricted environment where only the variables that the function requires to work are included

Usage

restrict_environment(f, ...)


rfunc

Arguments
  f  input function
  ... define the set of variables to be included as variable = value.

Value
  new function with a restricted environment

Author(s)
  John J. Aponte

Examples
  a = 0
  b = 1
  myfunc <- restrict_environment(
    function(n) {
      rnorm(meanvalue, sdvalue)
    },
    meanvalue = a, sdvalue = b)

  myfunc(10)
  ls(envir=environment(myfunc))

Description
  This is a generic method that calls the rfunc slot of the object

Usage
  rfunc(x, n)

Arguments
  x  an object
  n  the number of random samples

Value
  a matrix with as many rows as n and as many columns as dimensions have distribution

Author(s)
  John J. Aponte
## Default S3 method:
\texttt{rfunc(x, n)}

### Arguments

- \texttt{x} an object of class different from \texttt{DISTRIBUTION}
- \texttt{n} the number of random samples

### Value

No return value. Raise an error message.

### Author(s)

John J. Aponte
same_dimensions

Author(s)
John J. Aponte

Description
Check the dimensions of a list of distributions

Usage
same_dimensions(listdistr)

Arguments
listdistr a list of DISTRIBUTION objects

Value
return TRUE if all the dimensions are the same

set_seed

Modify a the seed of a Distribution object

Description
This create a new DISTRIBUTION object but with the specified seed

Usage
set_seed(distribution, seed)

Arguments
distribution a DISTRIBUTION object
seed the new seed

Value
a code DISTRIBUTION object of the same class

Author(s)
John J. Aponte
Summary of Distributions

Usage

```r
## S3 method for class 'DISTRIBUTION'
summary(object, n = 10000, ...)
```

Arguments

- `object`: object of class `DISTRIBUTION`
- `n`: the number of random samples from the distribution
- `...`: other parameters. Not used

Value

A data.frame with as many rows as dimensions had the distribution and with the following columns:

- distribution name
- varname name of the dimension
- oval value
- nsample number of random samples
- mean_ mean value of the sample
- sd_ standard deviation of the sample
- lci_ lower 95
- median_ median value of the sample
- uci_ upper 95

Note

The sample uses the seed saved in the object those it will provide the same values for an n value

Author(s)

John J. Aponte
TRIANGULAR Factory for a TRIANGULAR distribution object

Description

Returns an TRIANGULAR distribution object that produce random numbers from a triangular distribution using the rtriang function

Usage

new_TRIANGULAR(p_min, p_max, p_mode, p_dimnames = "rvar")

Arguments

- p_min A numeric that represents the lower limit
- p_max A numeric that represents the upper limit
- p_mode A numeric that represents the mode
- p_dimnames A character that represents the name of the dimension

Value

An object of class DISTRIBUTION, TRIANGULAR

Author(s)

John J. Aponte

Examples

myDistr <- new_TRIANGULAR(-1,1,0)
myDistr$rfunc(10)

TRUNCATED Factory for a TRUNCATED distribution object

Description

Returns an TRUNCATED distribution object that limits the values that are generated by the distribution to be in the limits p_min, p_max

Usage

new_TRUNCATED(p_distribution, p_min = -Inf, p_max = Inf)
UNIFORM

Arguments

p_distribution  An object of class DISTRIBUTION to truncate
p_min          A numeric that set the lower limit of the distribution
p_max          A numeric that set the upper limit of the distribution

Value

An object of class DISTRIBUTION, p_distribution$distribution, TRUNCATED

Note

The expected value of a truncated distribution could be very different from the expected value of the unrestricted distribution. Be careful as the oval field is not changed and may not represent any more the expected value of the distribution.

If the distribution is multidimensional, the limits will apply to all dimensions.

Author(s)

John J. Aponte

Examples

myDistr <- new_TRUNCATED(p_distribution = new_NORMAL(0,1), p_min = -1, p_max = 1)
myDistr$rfunc(10)

UNIFORM                        Factory for a UNIFORM distribution object

Description

Returns an UNIFORM distribution object that produce random numbers from a uniform distribution using the runif function

Usage

new_UNIFORM(p_min, p_max, p_dimnames = "rvar")

Arguments

p_min          A numeric that represents the lower limit
p_max          A numeric that represents the upper limit
p_dimnames     A character that represents the name of the dimension

Value

An object of class DISTRIBUTION, UNIFORM
**UNIFORM**

**Author(s)**

John J. Aponte

**Examples**

```r
myDistr <- new_UNIFORM(0,1)
myDistr$rfunc(10)
```
Index

* DISTRIBUTION
  DISTRIBUTION_factory, 15
  *.DISTRIBUTION (CONVOLUTION), 8
  +.DISTRIBUTION (CONVOLUTION), 8
  -.DISTRIBUTION (CONVOLUTION), 8
  /.DISTRIBUTION (CONVOLUTION), 8

add_total, 2
BETA, 3, 15
BINOMIAL, 4

cinqnum, 5
cinqnum.DIRAC, 6
cinqnum.DISTRIBUTION, 6
cinqnum.NA, 7
cinvsdistr, 7
CONVOLUTION, 8
CONVOLUTION_assoc, 9
CONVOLUTION_comb, 10

data.frame, 21, 30
DIRAC, 12, 15
DIRICHLET, 13, 15
DISCRETE, 13, 15
DISTRIBUTION, 2, 3, 5–11, 14, 15, 16, 19–25, 27–30
DISTRIBUTION_factory, 15
EXponential, 15, 16

fitbeta, 17, 18
fitbeta_ml (fitbeta), 17
fitDirichlet, 18, 19
fitdirichlet, 18
fitdist, 18

ggDISTRIBUTION, 19
ggplot, 19
ggplot2, 19

hist, 25
LOGNORMAL, 20
metadata, 20
mvnrnorm, 23

NA_DISTRIBUTION, 8, 15, 21, 25
new_BETA (BETA), 3
new_BETA_lci (BETA), 3
new_BETA_lci2 (BETA), 3
new_BINOMIAL (BINOMIAL), 4
new_CONVOLUTION (CONVOLUTION), 8
new_CONVOLUTION_assoc
  (CONVOLUTION_assoc), 9
new_CONVOLUTION_comb
  (CONVOLUTION_comb), 10
new_DIRAC (DIRAC), 12
new_DIRICHLET (DIRICHLET), 13
new_DISCRETE (DISCRETE), 13
new_DIVISION (CONVOLUTION), 8
new_DIVISION_assoc (CONVOLUTION_assoc), 9
new_DIVISION_comb
  (CONVOLUTION_comb), 10
new_EXPONENTIAL (EXPONENTIAL), 16
new_LOGNORMAL (LOGNORMAL), 20
new_MIXTURE, 22
new_MULTINORMAL, 23
new_MULTIPLICATION (CONVOLUTION), 8
new_MULTIPLICATION_assoc
  (CONVOLUTION_assoc), 9
new_MULTIPLICATION_comb
  (CONVOLUTION_comb), 10
new_NS (NA_DISTRIBUTION), 21
new_NORMAL (NORMAL), 24
new_POISSON (POISSON), 26
new_SUBTRACTION (CONVOLUTION), 8
new_SUBTRACTION_assoc
  (CONVOLUTION_assoc), 9
new_SUBTRACTION_comb
  (CONVOLUTION_comb), 10
new_SUM (CONVOLUTION), 8
new_SUM_assoc (CONVOLUTION_assoc), 9
new_SUM_comb (CONVOLUTION_comb), 10
new_TRIANGULAR (TRIANGULAR), 31
new_TRUNCATED (TRUNCATED), 31
new_UNIFORM (UNIFORM), 32
NORMAL, 15, 24
omit_NA, 24
plot.DISTRIBUTION, 25
POISSON, 15, 26
rbeta, 3
rbinom, 4
rdirichlet, 13
restrict_environment, 15, 26
rexp, 16
rfunc, 27
rfunc.default, 28
rfunc.DISTRIBUTION, 28
rlnorm, 20
rnorm, 24
rpois, 26
rtriang, 31
runif, 32
same_dimensions, 29
set_seed, 15, 29
summary.DISTRIBUTION, 30

TRIANGULAR, 15, 31
TRUNCATED, 15, 31

UNIFORM, 15, 32