Package ‘rv’

October 14, 2022

Title Simulation-Based Random Variable Objects
Version 2.3.5
Maintainer Jemma Stachelek <stachel2@msu.edu>
URL https://github.com/jsta/rv
BugReports https://github.com/jsta/rv/issues
Description Implements a simulation-based random variable class and a suite of methods for extracting parts of random vectors, calculating extremes of random vectors, and generating random vectors under a variety of distributions following Kerman and Gelman (2007) <doi:10.1007/s11222-007-9020-4>.
Depends R(>= 2.15.1), stats, utils, grDevices, graphics
License GPL-2
Encoding UTF-8
RoxygenNote 7.1.2
Suggests knitr, rmarkdown, testthat
VignetteBuilder knitr
NeedsCompilation no
Author Jouni Kerman [aut].
Jemma Stachelek [ctb, cre]
Repository CRAN
Date/Publication 2022-02-12 23:10:06 UTC

R topics documented:

rv-package .................................................. 4
abline.rv .................................................. 4
aperm.rv .................................................. 5
apply.rv .................................................. 6
as.double.rv ............................................. 7
as.integer.rv ............................................ 8
as.list.rv ............................................... 9
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>as.logical.rv</td>
<td>10</td>
</tr>
<tr>
<td>as.rv.bugs</td>
<td>11</td>
</tr>
<tr>
<td>as.rv.stanfit</td>
<td>12</td>
</tr>
<tr>
<td>as.vector.rv</td>
<td>12</td>
</tr>
<tr>
<td>cbind.rv</td>
<td>13</td>
</tr>
<tr>
<td>cc</td>
<td>14</td>
</tr>
<tr>
<td>detachrv</td>
<td>15</td>
</tr>
<tr>
<td>Extract-rv</td>
<td>16</td>
</tr>
<tr>
<td>Extremes-rv</td>
<td>17</td>
</tr>
<tr>
<td>hist_rv</td>
<td>18</td>
</tr>
<tr>
<td>is.constant</td>
<td>19</td>
</tr>
<tr>
<td>is.fuzzy</td>
<td>20</td>
</tr>
<tr>
<td>is.na.rv</td>
<td>21</td>
</tr>
<tr>
<td>ivplot</td>
<td>22</td>
</tr>
<tr>
<td>lines.rv</td>
<td>23</td>
</tr>
<tr>
<td>Math.rv</td>
<td>24</td>
</tr>
<tr>
<td>matmult</td>
<td>26</td>
</tr>
<tr>
<td>mean.rv</td>
<td>27</td>
</tr>
<tr>
<td>median.rv</td>
<td>28</td>
</tr>
<tr>
<td>mlplot</td>
<td>29</td>
</tr>
<tr>
<td>numeric_rv</td>
<td>31</td>
</tr>
<tr>
<td>outer.rv</td>
<td>32</td>
</tr>
<tr>
<td>plot.rv</td>
<td>33</td>
</tr>
<tr>
<td>points.rv</td>
<td>34</td>
</tr>
<tr>
<td>posterior</td>
<td>36</td>
</tr>
<tr>
<td>postsim</td>
<td>37</td>
</tr>
<tr>
<td>print.rv</td>
<td>38</td>
</tr>
<tr>
<td>print.rvfactor</td>
<td>39</td>
</tr>
<tr>
<td>quantile.rv</td>
<td>40</td>
</tr>
<tr>
<td>range.rv</td>
<td>41</td>
</tr>
<tr>
<td>rep.rv</td>
<td>42</td>
</tr>
<tr>
<td>rv</td>
<td>43</td>
</tr>
<tr>
<td>rvattr</td>
<td>44</td>
</tr>
<tr>
<td>rvbern</td>
<td>45</td>
</tr>
<tr>
<td>rvbeta</td>
<td>46</td>
</tr>
<tr>
<td>rvbinom</td>
<td>47</td>
</tr>
<tr>
<td>rvboot</td>
<td>48</td>
</tr>
<tr>
<td>rvcat</td>
<td>49</td>
</tr>
<tr>
<td>rvcauchy</td>
<td>50</td>
</tr>
<tr>
<td>rvchisq</td>
<td>51</td>
</tr>
<tr>
<td>rvcie</td>
<td>52</td>
</tr>
<tr>
<td>rvconst</td>
<td>53</td>
</tr>
<tr>
<td>rvcov</td>
<td>54</td>
</tr>
<tr>
<td>rvcut</td>
<td>55</td>
</tr>
<tr>
<td>rvdens</td>
<td>56</td>
</tr>
<tr>
<td>rvdirichlet</td>
<td>57</td>
</tr>
<tr>
<td>rvdiscrete</td>
<td>58</td>
</tr>
<tr>
<td>rvempirical</td>
<td>59</td>
</tr>
</tbody>
</table>
topics documented:

rvexp .................................................. 60
rvgamma ................................................ 61
rvhist .................................................. 62
rvifelse ............................................... 62
rvinvchisq .......................................... 63
rvmapply .............................................. 64
rvmatch ............................................... 65
rvmatrix ............................................... 67
rvmax .................................................. 68
rvmean ................................................ 68
rvmeanunif ........................................... 69
rvmin .................................................. 70
rvmultinom .......................................... 71
rvnchains ............................................ 72
rvneff ............................................... 73
rvnorm ............................................... 74
rvnsims .............................................. 75
rvpar .................................................. 76
rvpermuat ........................................... 77
rvpois ............................................... 78
rvpredict ............................................ 79
rvquantile .......................................... 80
rvrangle .............................................. 81
rvRhat ............................................... 82
rvsample ............................................. 82
rvsmapply ........................................... 83
rvsims ............................................... 84
rv ...................................................... 86
rvunif ............................................... 87
rvvar .................................................. 87
simapply ............................................. 88
sims ................................................... 89
solve.rv ............................................. 91
sort.rv ............................................... 92
splitbyname ......................................... 93
summaries ............................................ 94
unlistrv ............................................. 95
%*in*% .............................................. 96

Index 97
rv-package  Simulation-based Random Variable Objects

Description

‘rv’ implements a simulation-based random variable object class.

Details

Please refer to the vignette: vignette("rv") for details.

Package: rv
Version: 2.3.0
Date: 2013-05-18
 Namespace: rv
Depends: R(>= 2.10.0), methods, utils, grDevices, graphics
License: GPL-2

Author(s)

Jouni Kerman <jouni@kerman.com> Package built on Sat May 18 22:47:25 CEST 2013

References


See also vignette("rv").

abline.rv  Add (Random) Straight Lines to a Plot

Description

abline.rv, with random arguments (i.e. arguments of which at least one is an rv object), plots a sample of lines corresponding to of simulations of rv object x. If the arguments are all numeric (none is an rv object), the function call is passed on to abline.

Usage

abline.rv(a = NULL, b = NULL, h = NULL, v = NULL, ...)
aperm.rv

Arguments

a  intercept
b  slope
h  y-value(s) horizontal line(s)
v  x-value(s) horizontal line(s)
... further arguments passed to abline

Details

This is a version of abline that accepts random variable objects for the arguments a, b, h, or v.
The number of lines is determined by rvpar("line.sample"), default 20.
See the original help page in package `graphics`.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

## Not run:
demo("rvexample1")
## End(Not run)

aperm.rv Random Array Transposition

Description

Transpose a random array by permuting its dimensions and optionally resizing it.

Usage

## S3 method for class 'rv'
aperm(a, perm, ...)
apply.rv

Arguments

- `a` the random matrix to be transposed
- `perm` the subscript permutation vector. See the manual page for the generic method `aperm`.
- `...` further arguments passed to `aperm`

Details

This is the `rv`-compatible version of the function `aperm`. It first applies

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

`aperm`

Examples

```r
x <- rvarray(rvnorm(24), dim=c(2,3,4))
print(aperm(x))
```

---

**apply.rv**

Apply Functions over Margins of Random Arrays

Description

The `rv`-compatible version of `apply`

Usage

```r
apply.rv(X, MARGIN, FUN, ...)
```

Arguments

- `X` a random array
- `MARGIN` subscripts.
- `FUN` function.
- `...` optional arguments to `FUN`. 

Details

This is the rv-compatible version of the function apply.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

apply

Examples

```r
## Not run:
x <- rvmatrix(rvnorm(12), nrow=3, ncol=4)
print(apply.rv(x, 1, sum))
## End(Not run)
```

### Coercing Random Vectors to Real-valued

**Description**

Coerces random vector objects into double-valued ones.

**Usage**

```r
## S3 method for class 'rv'
as.double(x, ...)
```

**Arguments**

- `x`: an rv object
- `...`: other arguments

**Details**

`as.double` coerces an `rv` object into double-valued one. In effect, the function `as.double` is applied to all simulations.
Author(s)
Jouni Kerman <jouni@kerman.com>

References
See also vignette("rv").

Examples

```r
x <- as.logical(rvbern(prob=0.5))
print(x)
print(as.double(x))
```

---

as.integer.rv  Integer Random vectors

Description
Coerces a random variable to an integer-valued (discrete) one

Usage
```r
## S3 method for class 'rv'
as.integer(x, ...)
```

Arguments
- `x`  an rv object
- `...`  Further arguments passed on

Details
In effect, the function `as.integer` is applied to all simulations.

Note
`is.integer(x)` returns TRUE if and only if each component of `x` is integer-valued (each simulation vector is of type 'integer').

Author(s)
Jouni Kerman <jouni@kerman.com>
as.list.rv

References
See also vignette("rv").

See Also
as.logical.rv.

Examples

```r
x <- rpois(lambda=3)  # some integer-valued random variable
print(x)
is.integer(x)  # FALSE, because by default x is 'double'!
x <- as.integer(x)  # coerce to integer
is.integer(x)  # TRUE
print(x)  # Shows also the 'min' and 'max' columns
```

Description
as.list.rv coerces a given rv object into a list.

Usage
```r
## S3 method for class 'rv'
as.list(x, ...)
```

Arguments

- `x` an rv object
- `...` arguments passed on to other methods

Details
Each component of the argument is extracted into a component of an enclosing list, which is returned.

Author(s)
Jouni Kerman <jouni@kerman.com>
References
See also vignette("rv").

Examples

```r
x <- rnorm(10)
L <- as.list(x)
```

---

**as.logical.rv**

**Logical Random vectors**

**Description**

Coerces a random variable to a logical-valued one (Bernoulli r.v.)

**Usage**

```r
## S3 method for class 'rv'
as.logical(x, ...)
```

**Arguments**

- `x` an `rv` object
- `...` Further arguments passed on

**Details**

In effect, the function `as.logical` is applied to all simulations.

**Note**

`is.logical(x)` returns TRUE if and only if each component of `x` is logical-valued (i.e. TRUE/FALSE).

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

See also vignette("rv").
Examples

```r
x <- rvbern(prob=0.5)  # some 0/1 valued random variable
prient(x)
is.logical(x)  # FALSE, because by default x is 'double'
x <- as.logical(x)  # coerce to logical; all zeros become FALSE, ones become TRUE
is.logical(x)  # TRUE
print(x)  # Shows the expectations and not the quantiles
```

---

### as.rv.bugs

Coerce a bugs object into Random Variable Objects

as.rv.bugs coerces an R2WinBUGS object to a list of rv objects or to a named rv object (vector).

**Usage**

```r
## S3 method for class 'bugs'
as.rv(x, list. = TRUE, ...)
```

**Arguments**

- `x`: a bugs (R2WinBUGS) object
- `list.`: logical; return a list of rv objects instead of a single rv object (vector)?
- `...`: (ignored)

**Details**

as.rvsummary.bugs works similarly but coerces the resulting rv objects into rvsummary objects.

**Value**

If list.=TRUE, a named list of random vectors or a named random vector, otherwise a random vector. (Usually one would prefer a list.)

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**


See also vignette("rv").
as.rv.stanfit  \hspace{1cm} \textit{Convert simulations generated by Stan to a list of rv objects}

\section*{Description}

Convert simulations generated by Stan to a list of \textit{rv} objects.

\section*{Usage}

\begin{verbatim}
## S3 method for class 'stanfit'
as.rv(x, list. = TRUE, ...)
\end{verbatim}

\section*{Arguments}

\begin{itemize}
  \item \texttt{x} \hspace{1cm} A 'stanfit' object
  \item \texttt{list.} \hspace{1cm} logical; return a list of \textit{rv} objects instead of a single \textit{rv} object (vector)?
  \item \texttt{...} \hspace{1cm} (ignored)
\end{itemize}

\section*{Value}

A list of \textit{rv} objects, with the names set for each \textit{rv} object.

\section*{Author(s)}

J Kerman

\section*{References}

Stan: \url{https://mc-stan.org/}

as.vector.rv  \hspace{1cm} \textit{Coerce an rv object}

\section*{Description}

\texttt{as.vector.rv} coerces a given \textit{rv} object into a vector; matrices lose their dimension attributes, but \textit{rv} objects stay as \textit{rv} objects (since they are considered to be "vectors").

\section*{Usage}

\begin{verbatim}
## S3 method for class 'rv'
as.vector(x, mode = "any")
\end{verbatim}
cbind.rv

Arguments

- `x`: an object
- `mode`: (currently not used)

Details

The `as.vector.rv` function removes the dimension attribute and returns the `rv` object. Needed for compatibility with code that uses `as.vector`.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also `vignette("rv")`.

Examples

```r
x <- rvmatrix(rvnorm(10), 2, 5)
as.vector(x)
```

**Description**

Combines random vectors by columns (`cbind.rv`) or rows (`rbind.rv`).

**Usage**

```r
## S3 method for class 'rv'
cbind(..., deparse.level = 1)
```

**Arguments**

- `...`: vectors or matrices, can be `rv` objects
- `deparse.level`: (passed on to `cbind`)

**Details**

See `cbind` and `rbind` for details.
cc

Combine values in an rv object

Description
Concatenates random vectors.

Usage
cc(..., recursive = FALSE)
## S3 method for class 'rv'
c(..., recursive = FALSE)

Arguments
... objects to be concatenated. Can be a mixture of constants and rv objects.
recursive logical. If recursive = TRUE, the function recursively descends through lists (and pairlists) combining all their elements into a vector.

Details
NOTE: recursive has not yet been tested.

cc is a function that works for both non-rv and other vectors. To make code compatible for both constant vectors and rv objects, one can use cc instead of c.

Author(s)
Jouni Kerman <jouni@kerman.com>
References


See also vignette("rv").

Examples

```r
x <- rvnorm(2)
y <- rvbern(2, prob=0.5)
z <- c(x, y)
print(z)
z1 <- cc(1, z)
z2 <- c(as.rv(1), z)
z3 <- c(as.rv(1), z)
print(z1)
print(z2)
print(z3)
```

detachrv

Detach the rv package

Description

detachrv detaches the rv package and restores the original functions in base, graphics and stats packages.

Usage

detachrv()

Details

Currently equivalent to detach("package:rv").

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").
Extract-rv

Extract or Replace Parts of a Random Vector

Description

Bracket slice and assignment methods adapted for random vectors and arrays. The assignment function impute<- is compatible with both non-rv and rv objects (rv, rvsummary, and rvfactor objects). To write universal code that works both atomic and rv objects, use impute(x, ...) <- value instead of x[...] <- value.

NOTE. x will NOT be automatically coerced into an rv object.

value may be an rv object or a regular numeric object.

Extracting rv objects works the same way as extracting components of a numerical vector or array. The return value is always an object of class 'rv'. Type ?Extract for details.

Note: the index arguments (i, j, etc.) must be constants, but this may change in the future.

%Note: the index arguments (i, j, etc.) may be %themselves random variables, however they will be coerced %into integers, as one would expect.

Arguments

x object from which to extract element(s) or in which to replace element(s).

... indices specifying elements to extract or replace.

value typically an array-like R object of a similar class as x.

drop For matrices and arrays. If TRUE the result is coerced to the lowest possible dimension (see the examples). This only works for extracting elements, not for the replacement.

Value

A random variable (an rv object).

References


See also vignette("rv").
Examples

```r
x <- rnorm(1)
y <- (1:5)
## Not run:
y[2] <- x  ## Will not work
## End(Not run)
impute(y, 2) <- x
```

### Maxima and Minima of Random Variables

**Description**

Returns the maxima and minima of the components of a random vector.

- `rvmin` applies the function `min` to each component of the argument `x`. Missing values are removed.
- `rvmax` applies the function `max` to each component of the argument `x`. Missing values are removed.
- `rvrange` applies the function `range` to each component of the argument `x`. Missing values are removed.

`min.rv` returns the minimum of the random vector, returning thus one random variable. Similarly, `max.rv` returns the maximum of a vector.

`pmin.rv` and `pmax.rv` returns the componentwise minima or maxima of several random vectors or constants, yielding thus a random vector of the same length.

**Arguments**

- `x` an `rv` or `rvsummary` object
- `na.rm` remove missing values?
- `...` one or more `rv` objects or numeric objects

**Value**

A numeric vector of the same dimension as `x`.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**


See also vignette("rv").
See Also

rvmedian, rvmean.

Examples

```r
x <- rvpois(10, lambda=3)
rvmin(x)
rvmax(x)
rvrange(x)
```

---

**hist.rv**  
*Histogram of a random vector*

**Description**

hist.rv shows a grid of histograms generated from random draws of the random vector argument.

**Usage**

```r
## S3 method for class 'rv'
hist(
  x,
  grid = c(4, 5),
  xlim = x.range,
  main = paste(xname, "simulation"),
  freq = FALSE,
  ...)
```

**Arguments**

- `x`: an object
- `grid`: a vector of two numbers, indicating the size of the grid to plot the histograms
- `xlim`: x limits
- `main`: main title
- `freq`: logical; if FALSE, plots as probability density, as it should.
- `...`: Other arguments passed on to hist

**Author(s)**

Jouni Kerman <jouni@kerman.com>
is.constant

References

See also vignette("rv").

Examples

```r
## Not run:
x <- rnorm(30)
hist(x)
## End(Not run)
```

is.constant

Constant Vectors

Description

Functions to coerce or test for non-random objects.

Usage

```r
is.constant(x)
as.constant(x)
```

Arguments

x
an object, random variable (rv) or not

Details

is.constant returns TRUE for each component of the argument object if there is only one simulation (that is, the variable is "constant").
Note: rv objects that merely have variance zero are not therefore necessarily "true" constants.
as.constant coerces rv or rvsummary objects into constant strings; NA is returned for component that is not random.

Value

a logical vector (not rv), TRUE if a component is constant w.p. 1

Author(s)

Jouni Kerman <jouni@kerman.com>
is.fuzzy

References


See also vignette("rv").

Examples

```r
is.constant(1)  # TRUE
is.constant(as.rv(1))  # TRUE
setnsims(200)
x <- rvbern(prob=0.001)
all(sims(x)==0)  # most probably true
is.constant(x)  # always FALSE
x <- rvnorm(3)
x[1] <- 1
as.constant(x)  # 1, NA, NA
all(is.random(x) & is.na(as.constant(x)))  # always TRUE
```

is.fuzzy

Fuzziness

Description

Tests whether an object is "fuzzy", i.e. a logical random scalar that has probability strictly between zero and one (not strictly true nor strictly false).

Usage

```r
is.fuzzy(x)
```

Arguments

x an object, random or constant

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").
Examples

```r
x <- as.logical(rvbern(1, 0.4)) # a logical random variable
is.fuzzy(x) # TRUE, since x is logical and not constant
is.fuzzy(x<2) # FALSE, since x is less than 2 with probability one
is.fuzzy(rvnorm(1)) # FALSE, since it's not a probability
is.fuzzy(TRUE) # FALSE, since TRUE is strictly TRUE
is.fuzzy(1) # FALSE, since 1 is not a logical variable
```

is.na.rv

**Missing Data Indicators**

**Description**

is.na.rv returns the distribution (random variable) of the indicator function of missing data. rv.all.na returns TRUE if all components of the argument vector are completely missing. rv.any.na returns TRUE if any component of the argument vector has missing values.

**Usage**

```r
## S3 method for class 'rv'
is.na(x)
```

**Arguments**

- `x` an `rv` object

**Details**

Internally, is.na.rv applies the function is.na to each simulation of each component of the argument vector.

**Value**

is.na.rv returns a "Bernoulli" random vector of the same length and dimension as those of x. rv.all.na and rv.any.na return TRUE or FALSE (single value).

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**


See also vignette("rv").
Examples

```r
x <- trunc(rvnorm(1))
y <- !(x==0 & NA) # TRUE if x!=0
x <- y*x
is.na(x) # 69%: Pr(-1<Z<1)
is.logical(is.na(x)) # TRUE
rv.any.na(x) # TRUE
rv.all.na(x) # FALSE
```

ivplot

Interval plot

Description

Create a plot based on a data frame providing endpoints of intervals, colors, line weights etc.

Usage

```r
ivplot(
X,
name = "",
file.name = "",
split = NULL,
Intervals = NULL,
xlim,
left.margin = 3,
x.ticks = NULL,
exp.labels = FALSE,
xlab = "",
title = "",
top.axis = FALSE,
use_color = TRUE,
vline = NULL,
device = "X11",
size = c(297, 210)/25.4/2,
font.family = "Courier",
cex.label = NULL,
...
)
```

Arguments

- **X** A data frame providing data for creating one interval per row. See details below.
- **name** Name of file to produce
- **file.name** Name of file to produce
lines.rv

Add Connected (Random) Line Segments to a Plot

Description

Adds a sample of line segments randomly drawn from the joint distribution of \((x, y)\).

Usage

```r
## S3 method for class 'rv'
lines(x, y, type = "l", ...)```

split Name of column by which to divide the plot into groups.

Intervals A list defining what intervals or dots to output per each row.

xlim Numeric vector of length 2. Limits for the horizontal axis.

left.margin Scalar. Size of left margin. If labels take too much space, increase this (default is 3)

x.ticks Numeric vector.

exp.labels Logical. Use log scale? Then print numeric values at x-ticks in the original (exponentiated) scale

xlab Character.

title Character; title.

top.axis Logical. Print top axis?

use_color Logical. Use color in plot or black?

vline Scalar. Plot vertical line (will be plotted before intervals are

device Character. To which device to output?

size Numeric vector of length 2. Size of plot: vertical and horizontal sizes in inches.

font.family Character. Font family (sans (Helvetica), serif (Times), mono (Courier), ...)

cex.label number, a factor to shrink the 'cex' of the labels, between 0 and 1

... Other arguments passed to plot

Details

...

Value

The file name that was output; as a side effect a plot (a pdf file if device="pdf").

Author(s)

J Kerman
Arguments

- **x, y**: coordinate vectors of points to join
- **type**: character indicating the type of plotting, currently 'l' and 'p' are the only possibilities
- **...**: further arguments passed to `points`

Details

The size of the sample (number of segments drawn) is determined by `rvpar(line.sample)`.

`lines.rv` is implemented as part of `points.rv`.

See `points.rv` for details of the parameters.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

```r
## Not run:
x <- as.rv(1:10)
y <- rvnorm(mean=x)
par(mfrow=c(2,2))
plot(x, y, type="b", main="Intervals and random lines", rvcol="blue", col="gray")
plot(x, y, type="l", main="Only random lines", col="gray")
plot(x, E(y), type="b", main="Means, connected by a constant line", col="gray")
plot(x, rvmedian(y), type="b", pch=19, main="Median & middle 95 pc CI band", col="darkgray")
lines(rvquantile(y, 0.025), col="gray")
lines(rvquantile(y, 1-0.025), col="gray")

## End(Not run)
```

Description

Mathematical functions and operators adapted to work with random variable (rv) objects.
Usage

## S3 method for class 'rv'
Math(x, ...)

## S3 method for class 'rv'
Ops(e1, e2 = NULL)

Arguments

x       object
...     further arguments passed to or from other methods
e1      object
e2      object

Details

The operator method preserves the names of the longer vector (or those of the first if the lengths match).

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

x <- rnorm(10)
-x
names(x) <- paste("x[", seq_along(x), "]", sep="")
x + 1:10
1:2 + x
cumsum(x)
cumprod(exp(x))
Random Matrix Multiplication

Description

Multiplies two random matrices, if they are conformable. If one argument is a vector, it will be coerced to either a row or column matrix to make the two arguments conformable. If both are vectors it will return the inner product.

Usage

```
\texttt{\textbackslash{}%^{}.rv}(x, y)
```

```
x \texttt{%**%} y
```

Arguments

x, y numeric or complex matrices or vectors.

Details

Optimized internally for the case of random matrix multiplied by a constant column vector.

Value

The (distribution of the) matrix product. Use \texttt{drop} to get rid of dimensions which have only one level.

References


See also vignette("rv").

See Also

\texttt{matrix}, \texttt{Ops}, \texttt{diag}.

Examples

```
x <- 1:4
(z <- x %**% x)  # scalar ("inner") product (1 x 1 matrix)
drop(z)          # as scalar
```

```
y <- diag(x)
z <- matrix(1:12, ncol = 3, nrow = 4)
y %**% z
```
mean.rv

\[
\text{mean.rv} \quad \text{Distribution of the Arithmetic Mean of a Random Vector}
\]

Description

mean.rv computes the distribution of the arithmetic average of its argument rv object.

Usage

```r
## S3 method for class 'rv'
mean(x, ...)
```

Arguments

- `x`: an object
- `...`: further arguments passed to or from other methods

Details

mean gives the distribution (that is, a random variable object) of the statistic \[
\frac{1}{n} \sum_{i=1}^{n} x_i \quad (\text{sum}(x)/\text{length}(x)).
\]

In particular, mean(x) of a random vector x of length one is equal to x as it would be in the case of numerical x.

To find the expectation of a random vector x (that is, the individual means of random components in a vector), use rvmean(x) (same as E(x) and Pr(x)).

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See Also

rvmean
Examples

```r
cy <- rnorm(10, mean=0, sd=1)
m1 <- mean(y)
m2 <- rnorm(1, mean=0, sd=1/sqrt(10))
print(c(m1, m2)) # should have the same distribution
```

---

**median.rv**

*Distribution of the Sample Median*

**Description**

Compute the distribution sample median of the vector of values given as its argument.

**Usage**

```r
## S3 method for class 'rv'
median(x, na.rm = FALSE, ...)
```

**Arguments**

- `x`: a randomv vector containing the components whose distribution of the median value is to be computed.
- `na.rm`: a logical value indicating whether NA values should be stripped before the computation proceeds.
- `...`: further arguments passed to `median`

**References**


See also vignette("rv").

**See Also**

`rvmedian` for the componentwise medians. `quantile` for general quantiles.

**Examples**

```r
x <- rnorm(10) ## A random vector of length 10.
median(x) # A random scalar (vector of length 1).
rvmedian(x) # A numeric vector of length 10.
```
mlplot

Horizontal interval plot of components of a random vector

Description

mlplot plots the scalar components as of the given random array or vector as horizontal intervals, grouped by row.

Usage

mlplot(X, ...)

## Default S3 method:
mlplot(
  X,
  y.center = TRUE,
  y.shift = 0,
  y.map = NULL,
  mar = par("mar"),
  left.margin = 3,
  vline = NULL,
  top.axis = TRUE,
  exp.labels = FALSE,
  x.ticks = NULL,
  axes = NULL,
  xlim = NULL,
  ylim = NULL,
  xlab = deparse(substitute(X)),
  ylab = NULL,
  las = NULL,
  add = FALSE,
  ...
)

Arguments

X 
a random array or vector

... 
进一步 arguments passed to plot and points

y.center 
center the intervals nicely at each y-coordinate?

y.shift 
add this amount to each y coordinate of an interval

y.map 
onoptional function to compute the y-coordinates, given X

mar 
the margins of the plot

left.margin 
offset to add to the left margin of the plot (to add space for the labels)

vline 
if numeric, plot vertical lines at these (horizontal) coordinates

top.axis 
(logical) plot the top axis?
exp.labels (logical) if the original scale is logarithmic, label ticks in original (exp) scale?
x.ticks positions for the ticks of the x-axis
axes (logical) plot the axes at all?
xlim x limits
ylim y limits
xlab x label
ylab not used (instead of labels, the row names are shown)
las the style of axis labels, see par
add (logical) add the intervals to an existing plot?

Details

mlplot plots the scalar components of a vector or an array (2 or 3-dimensional) vertically (up to down) so that a component of a vector or a row of a matrix is plotted at vertical points 1...nrow(x). An 'mlplot' of a vector implements a "forest plot."

Scalars on the same row are plotted closely together. The positioning of the scalars within a row are controlled by the arguments y.center, y.shift, y.map. These do not need to be set for the default plot; if two arrays or vectors are plotted over on top of each other (using add=TRUE) then you should probably change y.shift which controls the vertical position of the array elements.

See demo(mlplot) for a detailed

To change the color of the random components of the vector, use rvcol. Typically this is of the same length as X, giving the color ‘theme’ for each component.

If X is a 3-dimensional array, mlplot is called repeatedly for each 2-dimensional array X[, ,k] for each k.

X may also be a fixed numeric object.

NAs (or random scalars with 100\mlplot is still experimental.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

## Not run:
# You can run this complete example by typing demo("mlplot")
n.rows <- 4; n.cols <- 5; n <- (n.rows*n.cols)
# Draw some fixed numbers
mu.true <- rnorm(1:n.rows, mean=1:n.rows, sd=1)
sigma.true <- 1
theta <- rvmatrix(rvnorm(n=n.cols, mean=mu.true, sd=sigma.true), nrow=n.rows)
  
  col.labels <- paste("Time", 1:n.cols, sep=":")
row.labels <- paste("Unit", 1:n.rows, sep=":")
dimnames(theta) <- list(row.labels, col.labels)

par(mfrow=c(2,2))
mlplot(theta, main="theta")
abline(v=0, lty="dotted")
mlplot(t(theta), main="theta transposed")
abline(v=0, lty="dotted")
row.sd <- apply.rv(theta, 1, sd.rv)
col.sd <- apply.rv(theta, 2, sd.rv)
x.max <- max(rvquantile(c(row.sd, col.sd), 0.99))
mlplot(row.sd, xlim=c(0, x.max), main="theta: within-row sd for each unit")
abline(v=0)
mlplot(col.sd, xlim=c(0, x.max), main="theta: between-row sd for each time point")
abline(v=0)

## End(Not run)

---

**numeric_rv**  
*Numeric Random Vectors*

**Description**

Creates or coerces rv objects of type "numeric".

**Usage**

```r
## S3 method for class 'rv'
is.numeric(x)

## S3 method for class 'rv'
as.numeric(x, ...)

## S3 method for class 'rvfactor'
as.numeric(x, ...)
```

**Arguments**

- `x` an rv object to be coerced or tested.
- `...` further arguments passed to or from other methods.
Details

\texttt{is.numeric(x)} returns \texttt{TRUE} if and only if each component of \(x\) is numeric-valued.

\texttt{as.numeric.rv} coerces an \texttt{rv} object into numeric-valued one. In effect, the function \texttt{as.numeric} is applied to all simulations.

Random factors are not numeric (just as non-random factors aren't).

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

\texttt{numeric}.

Examples

\begin{verbatim}
x <- as.logical(rvbern(1,0.5)) \# Bernoulli \texttt{rv}
is.numeric(x) \# FALSE
x <- as.numeric(x) \# coerce to numeric; all TRUEs become ones, FALSEs zeros
is.numeric(x) \# TRUE
\end{verbatim}

---

\texttt{outer.rv} \hspace{1cm} \textit{Outer Product of Random Arrays}

Description

\texttt{outer.rv}

Usage

\texttt{outer.rv(X, Y = NULL, FUN = "\*", \ldots)}

Arguments

\begin{description}
\item[X] First argument for function FUN
\item[Y] Second argument for function FUN; if missing, \(X\) is used instead
\item[FUN] a function to use on the outer products; a character string or a function
\item[\ldots] optional arguments to be passed to FUN
\end{description}
Details

Implements the outer product for random arrays.

Note. outer is not a generic function; thus outer(x) will not work if x is an rv object. You must write outer.rv(x) explicitly.

See the function outer for further details.

Value

A random array.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

```r
#
```
Details

If a component $x$ is fixed and the corresponding component of $y$ is random, the resulting 'point' is a vertical uncertainty ('credible') interval. *NOTE.* You must call `plot.rv` explicitly to obtain this behavior.

If a component $y$ is fixed and the corresponding component of $x$ is random, the resulting 'point' is a horizontal uncertainty ('credible') interval.

If a component of $x$ and the corresponding component of $y$ is random, the resulting 'point' is a scatterplot of simulations from the joint distribution of $\text{code}(x,y)$.

Compatible with objects of class 'rvsummary'.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

`mlplot`

Examples

```r
x <- as.rv(1:30)
y <- rvnorm(mean=x, sd=1)
## Not run: plot(x, y)
## Not run: plot(y, x)
## Not run: plot(y)
y <- as.rvsummary(x)
## Not run: plot(x, y)
## Not run: plot(y, x)
## Not run: plot(y)
```

---

**points.rv**  
*Add Points and Intervals to a Plot*

**Description**

Draw a sequence of points or uncertainty intervals at specified (fixed) x-coordinates.
Usage

## S3 method for class 'rv'
points(
  x,
  y = NULL,
  type = "p",
  xlim = NULL,
  ylim = NULL,
  rvlwd = rvpar("rvlwd"),
  rvcol = rvpar("rvcol"),
  rvpoint = rvpar("rvpoint"),
  rvlex = rvpar("rvlex"),
  ...
)

Arguments

x x-coordinates
y y-coordinates
type character indicating the type of plotting
xlim x-limits (optional)
ylim y-limits (optional)
rvlwd line width of the thin interval
rvcol colors for the intervals
rvpoint character vector of length 3, indicating intervals (points) to print
rvlex factor to multiply rvlwd with, to get the thicker interval
... further arguments passed to points

Details

Each point with a fixed coordinate and a random coordinate is plotted as an interval. If lines are plotted (type="l" or type="b"), the result is a random draw of lines connecting the coordinates. See lines.rv for details on how to set the sample size of the random draw.

Each interval consists of a maximum of three components. (1) a dot (2) thick interval (3) thin interval. Typically the dot marks the mean or the median; the thin and the thick intervals show a shorter and a longer middle uncertainty interval. The appearance of these intervals can be controlled using the parameters rvlwd, rvpoint, rvcol, and rvlex.

rvlwd sets the line width of the thin interval; rvlex sets the factor to multiply rvlwd to get the line width of the thicker interval.

points attempts to color the intervals and the dot using the color given as rvcol. The basic name of the color should be given, e.g. "red" or "blue". The thin line is colored using the basic color, the thick line is colored using a darker hue (numbered '2', e.g. "red2") and the dot is colored using the darkest hue (numbered '3', e.g. "red3"). That is, for example. if rvcol="red", the color scheme generated for the dot, the thick line, and the thin line, respectively, are c("red3", "red2", "red").
Special color themes: the default `rvcol` color scheme is called "default" and yields the color scheme `c("grey20", "grey40", "grey60")`. Other special color themes: "grey", "lightgrey", "darkgrey". (The spellings 'gray' and 'grey' are interchangeable).

The parameter `rvpoint` is a character vector of length 3, with the first component indicating what to plot as a dot (possible values: "mean", "median"), the second component indicating what to plot as a "thick interval" (possible values: "n component indicating what to plot as a "thin interval". Default: `c("mean", "50%", "95%")`). If you wish only to plot the mean and the 95% `rvpoint=c("mean", "95%").

The color `col` is used for plotting fully fixed dots (both x and y coordinates fixed) and lines (fixed and random lines – see `lines.rv`).

NOTE. This parameterization is yet experimental, and may change.

It is possible to have both x and y random, but this code is not yet fully functional.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

```r
x <- as.rv(1:10)
y <- rvnorm(mean=x)
par(mfrow=c(2,2))
plot(x, y, main="Fixed x-coordinate")
plot(y, x, main="Fixed y-coordinate")
plot(x, y, lwd=4, rvcol="red", main="Color and line width changed")
plot(x, y, type="b", main="Intervals and random lines", rvcol="blue", col="gray")

## Not run:
# Don't use the rv-only parameters when plotting fixed vectors.
plot(x, E(y), rvcol="blue", col="gray")
plot(x, E(y), rvcol="blue", col="gray")

## End(Not run)
```

### posterior

**Generate Posterior Simulations**

**Description**

Generate posterior simulations for a given fitted linear or general linear model, assuming the standard "noninformative" priors on the unknowns.
Usage

posterior(obj, ...)

Arguments

obj   an object
...
   further arguments

Value

A (named) list of random vectors. For example, the `lm` method returns a list with components `sigma` (the residual s.d.) and `beta`, the regression coefficients.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

```r
## Not run:
x <- 1:20
y <- rnorm(length(x), mean=x, sd=10)
print(summary(fit <- lm(y ~ x)))
bayes.estimates <- posterior(fit)
## End(Not run)
```

Description

*DEFUNCT.* Use posterior instead.

Usage

postsim(fit)
Arguments

fit an lm or glm object

Details

Generate posterior simulations for a given fitted linear or general linear model, assuming the standard "noninformative" priors on the unknowns.

Value

A (named) random vector for each fitted coefficient.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

print.rv

Print Distribution Summary of a Random Variable

Description

Prints a summary of the random variable object.

Usage

## S3 method for class 'rv'
print(x, digits = rvpar("print.digits"), ...)

Arguments

x an rv object
digits minimal number of significant digits
... further arguments passed to or from other methods

Details

Invokes first the summary method of the object, then prints the result.

Author(s)

Jouni Kerman <jouni@kerman.com>
References


See also vignette("rv").

See Also

rvfactor

Examples

```r
print(rvnorm(mean = rvnorm(1)))
```

---

Description

Creates or tests objects of type "rvfactor".

Usage

```r
## S3 method for class 'rvfactor'
print(x, all.levels = FALSE, ...)
rvfactor(x, ...)
## Default S3 method:
rvfactor(x, levels = NULL, ...)
```

Arguments

- `x`: object to be coerced or tested.
- `all.levels`: logical; whether to print all levels or not (see below for details)
- `...`: other arguments
- `levels`: factor levels (labels for the levels)

Details

Internally random factors are integer-valued just like regular factors in R.

The number of levels to print when `all.levels==FALSE` can be set by `rvpar(max.levels=...)`. By default this is set to 10.
quantile.rv

Value

rvfactor: an rvfactor object.
is.rvfactor: TRUE or FALSE.
as.rv.rvfactor: an rv object.
as.rvfactor.rv: an rvfactor object.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

# Probabilities of each integer of trunc(Z) where Z ~ N(0,1) ?
x <- rnorm(1)
rvfactor(trunc(x))
rvfactor(x>0)
rvfactor(rvpois(1, lambda=0.5))

quantile.rv                  Distribution of a Quantile of a Random Vector

Description

quantile.rv returns the distribution of the quantile of a random vector (as a random variable).

Usage

## S3 method for class 'rv'
quantile(x, ...)

Arguments

x an object
...

Value

A random vector (rv object) with components giving the distribution of the desired quantiles.
Note
quantile.rv does not return the simulated quantiles of the quantiles of the argument x. This is done by rvquantile.

Author(s)
Jouni Kerman <jouni@kerman.com>

References
See also vignette("rv").

Examples
```r
x <- rnorm(30)
quantile(x)
```

---

**range.rv**

Distribution of the Range of a Random Vector

Description
range.rv returns a 2-component random vector containing the distributions of the minimum and the maximum values of all the given arguments.

Usage
```r
## S3 method for class 'rv'
range(..., na.rm = FALSE, finite = FALSE)
```

Arguments
- `...` further arguments passed to or from other methods
- `na.rm` logical, indicating if NAs should be omitted
- `finite` logical, indicating if all non-finite elements should be omitted

Details
This is the rv-compatible version of the function range.

Author(s)
Jouni Kerman <jouni@kerman.com>
References


See also vignette("rv").

See Also

quantile.rv

Examples

```r
x <- rvnorm(mean=1:10, sd=1)
print(range(x))
print(quantile(x, c(0,1)))
```

rep.rv

Replicate Elements of Random Vectors

Description

Transpose a random array by permuting its dimensions and optionally resizing it.

Usage

```r
## S3 method for class 'rv'
rep(x, times, ...)
```

Arguments

- `x`: a random vector to be replicated
- `times`: number of replications
- `...`: further arguments passed to `rep`

Details

This is the `rv`-compatible version of the function `rep`.

Since `rep` is not a generic function, the whole name `rep.rv` must be specified when calling the function when `x` is an 'rv' object.

Author(s)

Jouni Kerman <jouni@kerman.com>
References

See also vignette("rv").

See Also
rep

Examples

```r
print(rep(rvnorm(1), times=4))
```

---

**rv**

*Random Vectors*

**Description**

Creates or tests for objects of type "rv".

**Usage**

```r
rv(length = 0)

is.rv(x)
```

**Arguments**

- `length`: desired length.
- `x`: object to be coerced or tested.

**Details**

- `rv` creates a random vector of the specified length. The elements of the vector are all equal to NA.
- `is.rv` returns TRUE if its argument is a rv object, FALSE otherwise.
- `as.rv` attempts to coerce its argument to the random vector (rv) type.
- `is.random` returns TRUE or FALSE for each component of the argument vector, depending on whether the component is a random variable object.
- `is.rvobj` tests whether its argument object is either of class rv or of class rvssummary.
- `as.rvobj` coerces its argument object to rv unless the object is an rv object (is.rvobj(x) is TRUE).

**Value**

An rv object of desired length, with the single simulation value NA.
Note

rv objects are internally lists with the class attribute set to “rv”. The number of simulations in rv objects is set by setnsims. This is by default set to 2500.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

For a short version of the paper, view the vignette by vignette("rv").

Examples

```r
x <- rv(1)
```

---

rvattr  Attributes of Random Variables

Description

rvattr

Usage

rvattr(x, attrib = NULL)

rvattr(x, attrib = NULL, by.name = FALSE) <- value

Arguments

- `x`: an object
- `attrib`: name of the attribute
- `by.name`: logical; attempt matching of attributes by name?
- `value`: vector of values to set; can be a list or an atomic vector
Details

If by \texttt{name=TRUE}, the values within the list \texttt{value} are matched by their name (e.g. \texttt{`theta[1]`}) if possible. Matching by \texttt{NA} or the empty string in a name is not possible.

Otherwise, the list is matched by position; in this case, the length of \texttt{value} must be equal to that of \texttt{x}.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also \texttt{vignette("rv")}.

Examples

##

```r
rvbern(n = 1, prob, logical = FALSE)
```

###

<table>
<thead>
<tr>
<th>\texttt{rvbern}</th>
<th>Generate a Random Vector from a Bernoulli Sampling Model</th>
</tr>
</thead>
</table>

Description

\texttt{rvbern} generates a random vector where each simulation comes from a Bernoulli sampling distribution.

Usage

```
rvbern(n = 1, prob, logical = FALSE)
```

Arguments

- \texttt{n} : number of random scalars to draw
- \texttt{prob} : probability of "success"; may be a random vector itself
- \texttt{logical} : logical; return a logical random variable instead

Details

\texttt{rvbern} is a special case of \texttt{rvbinom} with the argument \texttt{size=1}.

If \texttt{logical} is \texttt{TRUE}, the function returns a logical random variable which has \texttt{TRUE} for 1, \texttt{FALSE} for 0. (The printed summary of this object is slightly different from a regular continuous numeric random variable.)
Value

A random vector (an rv object) of length n.

Note

The resulting vector will not be independent and identically distributed Bernoulli unless prob is a fixed number.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

```r
rvbern(2, prob=0.5)
rvbinom(2, size=1, prob=0.5) # Equivalent
print(rvbern(1, 0.5, logical=TRUE)) # won't show the quantiles
print(as.logical(rvbern(1, 0.5))) # equivalent
```

---

rvbeta

Generate Random Vectors from a Beta Sampling Model

Description

rvbeta generates a random vector from the beta sampling model;

Usage

```r
rvbeta(n = 1, shape1, shape2)
```

Arguments

<table>
<thead>
<tr>
<th>n</th>
<th>integer, number of random variables to generate</th>
</tr>
</thead>
<tbody>
<tr>
<td>shape1</td>
<td>positive number or rv, 1st shape parameter</td>
</tr>
<tr>
<td>shape2</td>
<td>positive number or rv, 2nd shape parameter</td>
</tr>
</tbody>
</table>

Details

rvnbeta(n, a, b) ("neutral" Beta distribution) is equivalent to rvbeta(n, 1/3+a, 1/3+b).
rvbinom

Generate Random Variables from a Binomial Sampling Model

Description

Generates a random vector from a binomial sampling model.

Usage

rvbinom(n = 1, size, prob)

Arguments

- `n` integer, number of random variables to generate
- `size` integer or integer-valued rv: the number of trials (size of each sample)
- `prob` prior probability of success of each trial (may be constant or an rv object)

Details

rvbinom generates a random vector with given length, the distribution for size and the distribution for the probability of success.

Value

An rv object.

Author(s)

Jouni Kerman <jouni@kerman.com>
References


See also vignette("rv").

Examples

```r
## Not run:

s <- 1 + rvpois(1, lambda=3) # A prior distribution on the 'size' parameter.
srvbinom(1, size=s, prob=0.5) # The 'size' is random.
p <- rvbinom(1, 10, prob=0.5)/10 # Prior probability of success.
srvbinom(1, size=10, prob=p) # Now the probability is random.
srvbinom(1, size=s, prob=p) # Both the size and the probability are random.

## End(Not run)
```

rvboot

Generate a Random Vector from an Empirical Distribution

Description

rvboot generates a random vector of the same length as data from the empirical distribution of the data.

Usage

rvboot(data)

Arguments

data A vector of constants

Details

rvboot

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").
Examples

```r
y <- rnorm(30) # Some data: 30 draws from standard normal.
x <- rvboot(y) # A random vector of length 30 (each component has the same distribution)
print(mean(x)) # Bootstrap estimate of the mean.
print(sd.rv(x)) # Bootstrap estimate of the sd.
# rvinci(mean(x), 0) # Hypothesis test: mean of x is zero (at 5% level) FALSE => reject.
```

---

rvcat  

### Generate Categorical Random Variables

#### Description

Generates a random factor (i.e. a categorical random variable), given the probabilities of each category and their corresponding labels.

#### Usage

```r
rvcat(n = 1, prob, levels = NULL)
```

#### Arguments

- `n`: integer, number of random variables to generate
- `prob`: vector of probabilities of successes of each trial (may be constant or an rv object)
- `levels`: (character) labels for the categories

#### Details

The length of `prob` determines the number of bins.
The vector `prob` will be normalized to have sum 1.

#### Value

A random factor of length `length(prob)`.

#### Author(s)

Jouni Kerman <jouni@kerman.com>

#### References

See also vignette("rv").
rvcauchy

Generate Random Variables from a Cauchy Sampling Model

Description
Random vector generation for the Cauchy distribution.

Usage
rvcauchy(n = 1, location = 0, scale = 1)

Arguments
- n: integer: number of variables to generate
- location: location parameter (may be random)
- scale: scale parameter (may be random)

Details
For details on the Cauchy distribution, see Cauchy. See also rvt; Cauchy is a special case of the t-distribution with 1 degree of freedom, and therefore rvcauchy(n, location, scale) is equivalent to rvt(n, mu, scale, df=1).

Value
A random vector (rv object) of length n.

Author(s)
Jouni Kerman <jouni@kerman.com>

References
See also vignette("rv").
Generate Random Variables from a Chi-Square Sampling Model

Description

Generates a random vector from a chi-square sampling model.

Usage

rvchisq(n = 1, df, ncp = 0)

Arguments

- `n`: number of variables to generate
- `df`: integer, degrees of freedom, may be random
- `ncp`: non-centrality parameter, may be random

Details

If any of the arguments are random, the resulting simulations may have non-Poisson marginal distributions.

Value

A random vector (rv object) of length `n`.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

rvchisq(1, df = 3)
rvci

Credible (Uncertainty) Intervals for Random Scalars

Description

Computes credible (uncertainty) intervals for a given vector, given quantiles or the size of the middle interval

Usage

rvci(obj, interval = 0.95, one.sided = FALSE, left = TRUE)

Arguments

obj random scalar or vector
interval size of the middle interval or the quantile range of the interval
one.sided logical, FALSE if two-sided interval is desired
left logical, indicating if the left one-sided interval is desired

Details

If interval is of length two or more, the return value will be the quantiles given by range(interval).

Value

For two-sided intervals, an array of numbers of dimension c(2,length(x)), for one-sided intervals, a vector of the same length as x.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

rvci(rvnorm(1), interval=0.683) # Should be about c(-1,1).
Description

Coerces a given vector of constants into a random vector with 1 simulation in each component.

Usage

rvconst(n = 1, x = 0)

Arguments

n integer: number of variables to generate
x a vector of constants

Details

Coerces a given vector of constants into a random vector with 1 simulation in each component.

Value

A random vector (rv object) of length n.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

x <- rvconst(x=1:3)
c(x, 4)
Description

rvcov

Usage

rvcov(x, y = NULL, ...)

Arguments

x  a random vector
y  (optional) a random vector
... further arguments passed to or from other methods

Details

rvcov

Value

A covariance matrix.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

x <- rnorm(mean=1:3)
y <- rnorm(mean=2:4)
rvcov(x,y)
rvcov(x,x)
**rvcut**

*Convert Numeric to Random Factor*

**Description**

Convert implements the `cut` function using random variables.

**Usage**

```r
rvcut(x, ...)
```

**Arguments**

- **x**
  - a plain or a random vector which is to be converted to a factor by cutting.
- **...**
  - arguments passed to the function `cut`.

**Value**

A random factor.

**Author(s)**

Jouni Kerman &lt;jouni@kerman.com&gt;

**References**


See also `vignette("rv")`.

**See Also**

`rvfactor`, `cut`.

**Examples**

```r
rvcut(rvnorm(1), breaks=c(-Inf,-2,-1,0,1,2,Inf))
```
Description

rvdens generates a random vector where each simulation comes from a Bernoulli sampling distribution.

Usage

rvdens(n = 1, FUN, range, unitprecision = 10, ...)

Arguments

n number of random scalars to draw
FUN density function
range range to discretize over
unitprecision how many points per unit length
... other arguments passed on to FUN

Value

A random vector (an rv object) of length n.

Note

The resulting vector will not be independent and identically distributed Bernoulli unless prob is a fixed number.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

x <- rvdens(FUN=stats:::dnorm, range=c(-5, 5), unitprecision=10)
y <- rnorm(1) ## Should be close to x
rvdirichlet  Generate Random Variables from a Dirichlet Sampling Model

Description
Generates random variables from a Dirichlet sampling model.

Usage
rvdirichlet(n = 1, alpha)

Arguments
n integer: number of vectors to generate
alpha the parameter vector; may be random

Details
The Dirichlet distribution is a generalization of the Beta distribution. (If alpha is of length two, rvdirichlet draws from the Beta model.)

Value
A random vector (rv object) of length n.

Author(s)
Jouni Kerman <jouni@kerman.com>

References
See also vignette("rv").

Examples
## Not run:

a <- rvdirichlet(1, alpha=c(6, 3, 1)) #
sum(a) # one with probability 1

## End(Not run)
rvdiscrete

Generate Random Vectors from a Discrete Sampling Model

Description
Generates random variables from a discrete distribution (from a finite population with replacement).

Usage
rvdiscrete(n = 1, x, prob = NULL)

Arguments
n: integer: number of scalars to generate
x: values of the distribution
prob: probabilities (optional, default: all equal)

Details
Computes a random vector of length n, consisting of identically distributed discrete random scalars with the discrete distribution with values x and corresponding probabilities prob. If prob is not given, all values are considered equally distributed.

Author(s)
Jouni Kerman <jouni@kerman.com>

References
See also vignette("rv")

Examples
## Not run:
# 8 people draw a number each from 1..10 with replacement.
# What is the probability that the highest number of the eight is "10"?
# 8 iid variables from the discrete uniform 1:10.
# What is the probability that the person with the 3rd smallest number
# has at least "3"?
# End(Not run)
rvemprirical

Generate a Random Vector from an Empirical Distribution

Description

rvemprirical generates a random vector of the same length as data from the empirical distribution of the data.

Usage

rvemprirical(n, data)

Arguments

n Number of i.i.d. rv components to generate
data Data (constants)

Details

rvemprirical

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

y <- c(1.0, 1.2, 3, 1.1, 0.8, 0.9) ## Some data
x <- rvemprirical(4, data=y)
rvexp

Generate Random Vectors from an Exponential Sampling Model

Description

rvexp

Usage

rvexp(n = 1, rate = 1)

Arguments

n integer: number of variables to generate
rate prior distribution for the rate parameter (constant or random)

Details

rvexp

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

y <- rvexp(1, rate=rvexp(1)) # What marginal distribution does y have now?
**rvgamma**  
*Generate Random Variables from a Gamma Sampling Model*

**Description**

Generates random variables from a Gamma sampling model.

**Usage**

\[ \texttt{rvgamma(n = 1, shape, rate = 1, scale = 1/rate)} \]

**Arguments**

- **n**: integer: number of variables to generate
- **shape**: shape parameter, may be a rv
- **rate**: rate parameter, may be a rv
- **scale**: inverse of rate, may be specified optionally instead of rate

**Details**

\[ \texttt{rvngamma(n, shape, rate)} \text{ is equivalent to } \texttt{rvgamma(n, 1/3 + shape, rate)}. \]

**Value**

A random vector (rv object).

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**


See also vignette("rv").

**Examples**

\[ \text{round(rvmedian(rvngamma(n=1, shape=1:10, rate=1)), 1)} \] ## close to 1:10
**rvhist**

*Histogram of Distributions of Components of a Random Vector*

**Description**

rvhist shows a grid of histograms of simulations of the components of a random vector.

**Usage**

```
rvhist(x, ...)
```

**Arguments**

- `x`: an rv object
- `...`: further arguments passed to the function `hist`

**Details**

Outputs a histogram using the `hist` function with the option `freq=FALSE`. This can be overridden by specifying the argument `freq` or `prob`. See the function `hist` for details.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**


See also vignette("rv").

---

**rvifelse**

*Conditional Random Element Selection*

**Description**

rvifelse is the rv-compatible version of the function ifelse.

**Usage**

```
rvifelse(test, yes, no)
```

**Arguments**

- `test`: an object which can be coerced to logical mode.
- `yes`: return values for true elements of test
- `no`: return joint simulations and not simulations from each component separately
Details

`rvifelse` returns a *random* value with the same shape as `test` which is filled with random or constant elements selected from either *yes* or *no*, depending on whether the random draw in an element of `test` is `TRUE` or `FALSE`.

Value

A *numeric* array of dimensions `size` times `length(x)`.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also `vignette("rv")`.

See Also

`ifelse`.

---

### rvinvchisq

**Generate Random Variables from a Inverse-Chi-Square Sampling Model**

**Description**

`rvinvchisq`

**Usage**

`rvinvchisq(n = 1, df, scale = 1)`

**Arguments**

- `n` integer: number of variables to generate
- `df` degrees of freedom (may be random)
- `scale` scale parameter (may be random)

**Details**

`rvinvchisq`

**Value**

A random vector (rv object).
Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

rvinvchisq(df=3, scale=2)

Description

rvmapply is the rv-compatible version of mapply. It repeats the function FUN for each joint draw of the random (or constant) arguments, while allowing vectorizing.

Usage

rvmapply(
  FUN,
  ..., 
  MoreArgs = NULL,
  SIMPLIFY = FALSE,
  USE.NAMES = TRUE,
  SAMPLESIZE = NULL
)

Arguments

FUN the function to apply to the simulations of X.

... further arguments to FUN, possibly random vectors or array.

MoreArgs Other args passed to FUN ‘as is’ (must not be rv objects unless the function already accepts them)

SIMPLIFY logical; see mapply for details

USE.NAMES logical; see mapply for details

SAMPLESIZE if specified, takes a (joint) sample of the simulations and processes only them.
rvmatch

Details

rvmapply applies a given function to each simulation (vector or array) of the given random vectors, returning a the results as a random vector or array.

The dimensions of each joint draw are preserved. For an example, see solve, that returns the distribution of the inverse of a random matrix.

Usually used in functions that implement an 'rv'-compatible routine.
For an example of a function that uses SAMPLESIZE, abline.

Value

Depends on FUN; a random vector or array if FUN is numeric.

Note

If the function (FUN) has an argument “FUN”, it must be specified within the list supplied to MoreArgs.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

mapply, simapply

---

rvmatch Generate a Random Vector from a Bernoulli Sampling Model

Description

rvmatch returns a (random) vector of the positions of (first) matches of its first argument in its second.

Usage

rvmatch(x, table, nomatch = NA_integer_, incomparables = NULL)
Arguments

x random vector, regular atomic vector, or NULL: the values to be matched.
table random vector, regular atomic vector, or NULL: the values to be matched against.
nomatch the value to be returned in the case when no match is found. Note that the value is coerced to integer.
incomparables a vector of values that cannot be matched. Any value in x matching a value in this vector is assigned the nomatch value. For historical reasons, FALSE is equivalent to NULL.

Details

%*in*% is a binary operator (analogous in its operation to %in%) which returns a logical (random) vector indicating if there is a match or not for its left operand.

Value

A random vector (an rv object) of the same length as x.
rvmatch returns an integer-valued vector.
%*in*% returns a logical-valued vector.
Both functions are compatible with regular atomic vectors.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

x <- rvmprirical(5, 1:10)
z <- rvmatch(x, table=1:3, nomatch=0L)
**rvmatrix**

**Matrices and Arrays of Random Vectors**

**Description**

Arrange a given random vector into a matrix or array form.

**Usage**

```r
rvmatrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
```

```r
rvarray(data = NA, dim = length(data), dimnames = NULL)
```

**Arguments**

- `data`: an optional data vector.
- `nrow`: the desired number of rows.
- `ncol`: the desired number of columns.
- `byrow`: logical. If `FALSE` (the default) the matrix is filled by columns, otherwise the matrix is filled by rows.
- `dimnames`: A dimnames attribute for the matrix: a list of length 2 giving the row and column names respectively.
- `dim`: the dim attribute for the array to be created, that is a vector of length one or more giving the maximal indices in each dimension.

**Details**

These are 'rv' compatible versions of the functions `matrix` and `array`.

The function `rvmatrix` generates the random variable matrix via an `rvarray` call.

The `rvarray` function calls first `array` to set the dimensions of the argument `data` and then coerces the resulting array object to an 'rv' object.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**


See also vignette("rv").

**See Also**

To plot random matrices, see `mlplot`. 
Examples

```r
n.rows <- 3; n.cols <- 4; n <- (n.rows*n.cols)
mu.true <- rnorm(1:n.rows, mean=1:n.rows, sd=1)
theta <- rvmatrix(rvnorm(n=n.cols, mean=mu.true, sd=0.5), nrow=n.rows)
col.labels <- paste("Time", 1:n.cols, sep=";")
row.labels <- paste("Unit", 1:n.rows, sep=";")
dimnames(theta) <- list(row.labels, col.labels)
print(theta)
print(E(theta))
```

---

**rvmax**

*Get the max values of an rv object*

**Description**

Get the max values of an rv object

**Usage**

```r
rvmax(x)
```

**Arguments**

- `x` an rv or rvsummary object

---

**rvmean**

*Expectation of a Random Variable*

**Description**

`rvmean`

**Usage**

```r
rvmean(x)
```

**Arguments**

- `x` an rv object
Details

`rvmean` computes the means of the simulations of all individual components of a random vector (rv) object.

E is an alias for `rvmean`, standing for "Expectation."

Pr is another alias for `rvmean`, standing for "Probability of"; suggested to be used when the argument is a logical statement involving random variables (that is, a description of an event such as $x>0$ or $x>y$). Then $Pr(x>0)$ gives the probability of the event "$x>0". The statement $x>0$ returns a Bernoulli (indicator) random variable object (having 1/0 or TRUE/FALSE values) and the expectation of such variable is just the probability of the event where the indicator is one.

Value

A numerical vector with the same dimension as x.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

`mean.rv`: distribution of the arithmetic mean of a vector; `rvmin`, `rvmax`, `rvmedian`, `link{rvvar}`, `rvsd`.

Examples

```r
x <- rvnorm(mean=(1:10)/5, sd=1)
rvmean(x) # means of the 10 components
E(x) # same as rvmean(x)
Pr(x>1) # probabilities that each component is >1.
```

---

**rvmeanunif**

The distribution of the mean of uniform random variables

Description

The distribution of the mean of uniform random variables with each of them in the interval (-1, 1), then scaled and shifted.
Usage
rvmeanunif(n = 1, mode = 0, scale = 1, df)

Arguments
- **n**: Length of the vector to output
- **mode**: Mode (center) of the distribution
- **scale**: Scale (half-width) of the distribution around the mode
- **df**: "degrees of freedom": number of independent components to average

Details
Assuming that all inputs are constants, each generated variable has a mode (center) at mode, constrained between \((-\text{scale}, \text{scale})\).

The shape becomes more and more bell-shaped (Normal) as the number of the independent variables in the sum (mean) increases.

The case of \(df=2\) (mean of two variables) is the special case of the symmetric triangular distribution in the range

Value
A random vector of length \(n\).

Author(s)
J Kerman

Examples
```r
x <- rvtriang(1)
y <- rvmeanunif(df=2)  ## same distribution as that of x
```

---

**rvmin**

*Get the min values of an rv object*

Description
Get the min values of an rv object

Usage
rvmin(x)

Arguments
- **x**: an rv or rvssummary object
Generate Random Variables from a Multinomial Sampling Model

Description
Generates a random vector from a multinomial sampling model.

Usage
rvmultinom(n = 1, size = 1, prob)

Arguments
- **n**: integer, number of random variables to generate
- **size**: integer or integer-valued rv: the number of trials (size of each sample)
- **prob**: vector (of length at least 3) prior probabilities of successes of each trial (may be constant or an rv object)

Details
The length of `prob` determines the number of bins.
The vector `prob` will be normalized to have sum 1.
If `length(prob)` is two, `rvbinom` is called instead.
NOTE. Case of random `n` or `size` or `prob` — not yet optimized for speed.

Value
A random array of dimensions `length(prob)` times `n`.

Author(s)
Jouni Kerman <jouni@kerman.com>

References
See also vignette("rv").

Examples
```r
y <- rvmultinom(n = 3, size = 1, prob = c(0.20, 0.30, 0.50))
```
Description

Retrieves the number of mcmc chains in each components of the argument.

Usage

rvnchains(x)

Arguments

x an rv object (supposed to be generated by a MCMC process)

Details

Assumes that the rv object was generated by a MCMC process. Umacs and R2WinBUGS are compatible.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

as.rv.bugs

Examples

#
Description

Retrieves the number of effective draws in each component of the argument.

Usage

rvneff(x)

Arguments

x an rv object

Details

The number of effective draws is supposed to be saved by the simulation generating program (e.g. WinBUGS via R2WinBUGS).

Value

A numeric object of the same length as the argument x.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

#
rvnorm

Generate Random Variables from a Gaussian (Normal) Sampling Model

Description
Generates a random vector from a Gaussian sampling model.

Usage
rvnorm(n = 1, mean = 0, sd = 1, var = NULL, precision)

Arguments
- **n**: integer: number of variables to generate.
- **mean**: mean, may be a rv
- **sd**: standard deviation; scalar or vector (constant or rv, not matrix)
- **var**: variance, can be given instead of sd. Scalar, vector, or matrix.
- **precision**: inverse variance or variance matrix, may be given instead of sd or var

Value
An rv object of length n times the length of the mean vector.
If mean is a vector, a vector is returned: n refers to how many vectors or scalars are replicated.

Note
If any of the arguments are random, the resulting simulations may have non-normal marginal distributions; for example, if an inverse-chi-squared scalar rv var and zero mean is given, the resulting rv will have a t-distribution.

Author(s)
Jouni Kerman <jouni@kerman.com>

References
See also vignette("rv").

Examples
```r
x <- rvnorm(mean=1:10, sd=1:10) # A vector of length 10.
Sigma <- diag(1:10)
y <- rvnorm(mean=1:10, var=Sigma)
```
Description

rvnsims returns the number of simulations stored in each component of its argument; setnsims sets the default number of simulations; getnsims retrieves the default number of simulations.

Usage

rvnsims(x)

Arguments

x
an rv object.

Details

If the argument is a non-rv numeric vector, rvnsims returns 1 (corresponding to a ‘constant’) for each component.

The minimum number of default simulations is 2.

Value

rvnsims: a vector of integers.
setnsims: previously set default number of simulations.
getnsims: (integer) currently set default number of simulations.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

#  rvnsims(1.23)       # 1
x <- rvnorm(1)        # an rv
rvnsims(x)            # equal to setnsims()
rvnsims(x)==nrow(sims(x)) # TRUE
rvnsims(x)==getnsims() # TRUE
```r
setnsims(1000)    # set n.sims to 1000
n.sims <- setnsims(10000)  # s is now 1000
print(getnsims())     # prints 10000
setnsims(n.sims)      # restore the number of simulations back to 1000
```

---

### Description

Sets or retrieves parameters of the `rv` package.

### Usage

```r
rvpar(...)  
```

### Arguments

... arguments in tag = value form, or a list or character vector of tagged values. The available tags are described below.

### Details

- **rvcol** color of a random point (interval), such as 'red' or 'blue'
- **rvlex** middle interval expansion factor
- **rvlwd** line weight of a random interval
- **print.digits** number of digits to show in the summaries
- **rvpoint** what to output when plotting a random point; default `list("95\%", "50\%", "mean")`
- **point.sample** number of points to plot when plotting a rv-rv scatterplot. Default `400`.
- **line.sample** number of lines to draw when plotting a random sample of lines (see `abline`). Default `20`.
- **summary.dimnames** logical; output dimnames in the summary of an `rv` object? Default `TRUE`.
- **summary.quantiles.numeric** vector of quantiles to compute for the summary of a numeric `rv` object.
- **summary.quantiles.integer** vector of quantiles to compute for the summary of an integer-valued `rv` object. By default contains 0 and 1 (for the min and max values).

### Value

In the case of a single tag query, the requested value.

In the case of multiple tag query, a list of requested values.

### Author(s)

Jouni Kerman <jouni@kerman.com>
rvpermut

References

See also vignette("rv").

Examples

rvpar()$rvcol
rvpar("rvcol")

rvpermut(data, prob = NULL)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>a fixed numeric vector</td>
</tr>
<tr>
<td>prob</td>
<td>optional probabilities for the components in data</td>
</tr>
</tbody>
</table>

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

x <- rvpermut(1:10)
rvpois

Generate Random Vectors from a Poisson Sampling Model

Description

Generates random variables from a Poisson sampling model.

Usage

rvpois(n = 1, lambda)

Arguments

n integer: number of variables to generate
lambda a vector of (positive) mean parameters; (may be random)

Note

If any of the arguments are random, the resulting simulations may have non-Poisson marginal distributions.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

x <- rvpois(lambda=10) # A Poisson rv with mean 10
lbd <- rvchisq(1,1)    # Some positive rv
y <- rvpois(lambda=lbd) # Not a Poisson rv, although each simulation is a draw from Poisson.
rvpredict  Generate predictions from models

Description
Performs predictions (in the form of rv objects) from models based on given covariates.

Usage
rvpredict(object, ...)

## S3 method for class 'lm'
rvpredict(object, newdata, ...)

Arguments

object  An object representing a statistical model fit.

...  Arguments passed to and from other methods.

newdata  A data frame with new covariates to be used in the predictions. The column names of the data frame must match those in the model matrix (although order may be arbitrary). If omitted, the model matrix is used instead; the resulting predictions are then the replications of the data. Note: this can be an rv object to incorporate extra uncertainty into predictions.

Details
The lm method generates predictions of the outcome variable. The posterior coefficient estimates (the "intercept" and the "betas") are estimated in a Bayesian framework by posterior(object); the coefficients are multiplied by newdata (if omitted, the model covariate matrix is used instead) to obtain the predicted model mean; lastly, the outcomes are predicted from the Normal sampling model, taking into account the sampling variability along with the uncertainty in the estimation of the standard deviation ("sigma").

The covariate matrix newdata can be an rv, representing additional uncertainty in the covariates.

Value
For the lm method, a vector as long as there are rows in the data frame newdata.

Author(s)

J Kerman
Examples

```r
## Create some fake data
n <- 10
## Some covariates
set.seed(1)
X <- data.frame(x1=rnorm(n, mean=0), x2=rpois(n, 10) - 10)
y.mean <- (1.0 + 2.0 * X$x1 + 3.0 * X$x2)
y <- rnorm(n, y.mean, sd=1.5) # n random numbers
D <- cbind(data.frame(y=y), X)
## Regression model fit
obj <- lm(y ~ x1 + x2, data=D)
## Bayesian estimates
posterior(obj)
## Replications
y.rep <- rvpredict(obj)
## Predictions at the mean of the covariates
X.pred <- data.frame(x1=mean(X$x1), x2=mean(X$x2))
y.pred <- rvpredict(obj, newdata=X.pred)
## Plot predictions
plot(y.rep, D$x1)
points(D$y, D$x1, col="red")
## 'Perturb' (add uncertainty to) covariate x1
X.pred2 <- X
X.pred2$x1 <- rnorm(n=n, mean=X.pred2$x1, sd=sd(X.pred2$x1))
y.pred2 <- rvpredict(obj, newdata=X.pred2)
```

---

### rvquantile

#### Componentwise Quantiles of Random Variables

**Description**

Computes componentwise quantiles of random vectors or arrays.

**Usage**

```
rvquantile(x, ...)  
```

```
## S3 method for class 'rv'
rvquantile(
  x,
  probs = c(0.025, 0.1, 0.25, 0.5, 0.75, 0.9, 0.975),
  ignoreInf = FALSE,
  ...
)
```
rvrange

Arguments

x an object
... further arguments passed to quantile
probs numeric vector of probabilities with values in [0,1]
ignoreInf ignore infinite values

Details

rvquantile applies the quantile function to each column of sims(x).
rvmedian applies median to the each column of sims(x).

Value

A numeric vector of quantiles.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

Examples

x <- rvnorm(3)
rvquantile(x)
rvquantile(x, probs=c(0, 0.01, 0.99, 1))
rvmedian(x)

rvr

rvarange

Get the value range of an rv object

Description

Get the value range of an rv object

Usage

rvarange(x)

Arguments

x an rv or rvsummary object
**rvRhat**  
*R-hat Convergence Diagnostic*

**Description**

Retrieves the R-hat convergence diagnostic for each component of the argument

**Usage**

rvRhat(x)

**Arguments**

- **x**: an object

**Details**

The R-hat values are assumed to be saved as attributes. If they are not available, NA will be returned. R-hat is computed by programs such as Umacs and R2WinBUGS.

**Value**

Vector of numbers, NA if R-hat is not available.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**


See also vignette("rv").

---

**rvsample**  
*Draw a Sample from the Simulation Matrix of a Random Variable*

**Description**

Draws a sample of desired size from each component of a given random variable x.

**Usage**

e = rvsample(x, size = 1, jointly = TRUE, reject.na = FALSE)
rvsimapply

Arguments

- **x**: an object
- **size**: size of the sample
- **jointly**: return joint simulations and not simulations from each component separately
- **reject.na**: reject each draw that contains an NA

Details

Samples (with replacement) from the distribution of the random variable object. In effect it samples from the rows of the simulation matrix `sims(x)`.

Value

A *numeric* array of dimensions `size` times `length(x)`.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

```R
#
```

---

**rvsimapply** 

*Apply a Function to Columns of the Matrix of Simulation of a Random Vector*

Description

`rvsimapply`

Usage

`rvsimapply(x, FUN, ...)`

Arguments

- **x**: an object
- **FUN**: an R function object
- **...**: further arguments passed to the function `FUN`
Details

`rvsimapply` applies a given function to the rows of the simulation matrix of the given random vector.

If the function is to be applied to rows of the simulation matrix, use `simapply` or `rvmapply` instead. Usually used in functions that implement an 'rv'-compatible routine.

Value

A numeric vector or array.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also `vignette("rv")`.

Examples

```r
#
```

---

**rvsims**

Create Random Vectors from Simulation Draws

Description

`rvsims` takes a vector, matrix, or list (`sims`) containing simulations, and returns a random vector (an object of type 'rv')

Usage

```r
rvsims(sims, n.sims = getnsims(), permute = FALSE)
```

Arguments

- `sims`: an array of simulations (1, or 2-dimensional) or a list
- `n.sims`: number of simulations to save
- `permute`: logical, indicate if scramble the simulations
Details

If `sims` is a plain numeric vector, this is interpreted to be equivalent to a one-dimensional array, containing simulations for one single random variable.

If the array `sims` is one-dimensional, this is interpreted to be equivalent to a two-dimensional array with 1 column.

If `sims` is two-dimensional, the columns are supposed to contain simulations for one or more several random variables.

If `sims` is a list, the numeric vectors are recursively combined to a list of random vectors: each component of the list is supposed to be containing one (joint) draw from some distribution—this may be a list.

If `permute` is `TRUE`, the simulations are scrambled, i.e. the joint draws are permuted randomly.

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

```r
## x and y have the same distributions but not the same simulations:
setnsims(n.sims <- 200L)
y <- rnorm(1)
x1 <- rvsims(rnorm(n.sims))
##
s <- sims(x1)
z <- array(s) ## One-dimensional array
x2 <- rvsims(z) ## Same as
##
identical(x1, x2) ## TRUE
##
s <- t(array(rnorm(n.sims * 2, mean=c(0, 10)), dim=c(2, n.sims)))
x3 <- rvsims(s)
identical(2L, length(x3)) ## TRUE
```
Generate Random Variables from a Student-t Sampling Model

**Description**

Generates a random variable from a Student-t sampling model.

**Usage**

```r
rvt(n = 1, mu = 0, scale = 1, df, ncp, Sigma)
```

**Arguments**

- `n`: integer, number of scalars to generate
- `mu`: location, may be a `rv`
- `scale`: scale, may be a `rv`
- `df`: degrees of freedom, may be a `rv`
- `ncp`: non-centrality parameter
- `Sigma`: (optional) scaling matrix for multivariate generation

**Details**

This function generates both univariate (independent and identically distributed) Student-t random variables and multivariate Student-t distributed vectors (with a given scaling matrix).

For details of the parameters, see the entry on `mvt` in the `mvtnorm` package.

**Note**

If any of the arguments are random, the resulting simulations may have non-t marginal distributions.

**Author(s)**

Jouni Kerman &lt;jouni@kerman.com&gt;

**References**


See also vignette("rv").

**Examples**

```r
df <- 3
x <- rvt(n=1, df=df)
y <- rnorm(1)/sqrt(rvchisq(1, df=df)/df) # Same distribution as above
print(c(x,y))
```
rvunif  

Generate Random Vectors from a Uniform Sampling Model

Description
Generates random variables from a Uniform sampling model.

Usage
rvunif(n = 1, min = 0, max = 1)

Arguments
n integer: number of scalars to generate
min lower limit of the distribution, (may be random)
max upper limit of the distribution, (may be random)

Author(s)
Jouni Kerman <jouni@kerman.com>

References
See also vignette("rv").

Examples
#
y <- rvunif(1, min=rvunif(1)-1, rvunif(1)+1) # What marginal distribution does y have now?

rvvar  

Variances of Components of Random Vectors

Description
Computes variances of the simulations of components of a random vector of array.

Usage
rvvar(x)
Arguments

x an object

Details

rvvar computes the means of the simulations of all individual components of a random vector (rv) object.
That is, rvvar applies the function var to the vector of simulations of each component of x, thus computing "columnwise" variances of the matrix of simulations of x.
rvsd applies the function sd to the vector of simulations of each component of x, thus computing "columnwise" standard deviations of the matrix of simulations of x.

Value

A numeric vector or array (of the same dimension as that of x)

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

See Also

rvmin, rvmax, rvmedian, rvsd.

Examples

x <- rnorm(mean=0, var=1:10)
rvvar(x)
rvsd(x)

---

simapply Apply a Function to Rows of Simulations of Random Vectors

Description

simapply applies a given function FUN to each row of the simulation matrix, returning an rv object.

Usage

simapply(x, FUN, ...)

---
Arguments

x       a random vector.
FUN     a function.
...     further arguments passed to FUN.

Details

simapply applies a given function to the rows of the simulation matrix of the given random vector. If the function accepts arrays, use rvmapply instead.

If the function is to be applied to each component of the random vector separately (such as in rvmean), use rvsimapply instead.

Usually used in functions that implement an 'rv'-compatible numeric function.

Value

An rv object, representing the distribution of FUN(x,...).

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

#
x <- rnorm(10)
simapply(x, mean) # Same result as that of mean(x).

sims

Retrieve the Simulations of Random Vectors

Description

Returns the simulation matrix for the random variable object x.
Usage

sims(x, ...)

## S3 method for class 'rvsummary'
sims(x, dimensions = FALSE, ...)

## S3 method for class 'rv'
sims(x, dimensions = FALSE, n.sims = getnsims(), ...)

Arguments

x a random variable object
...
arguments passed on
dimensions logical, try to preserve the dimensions of x
n.sims (optional) number of simulations

Details

sims returns the matrix of simulations for a given random variable object x.
The first index of the matrix indicates the number of the simulation draw ("simulations are in rows").

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

Examples

setnsims(n.sims=2500)
x <- rvnorm(24)
dim(x) <- c(2,3,4)
dim(sims(x)) # 2500x24
dim(sims(x, dimensions=TRUE)) # 2500x2x3x4
Description

solve.rv

Usage

## S3 method for class 'rv'
solve(a, b, ...)

Arguments

a  a square random vector containing the coefficients of the linear system
b  a square random vector giving the right-hand side(s) of the linear system
... further arguments passed to solve

Details

solve.rv is the rv-object compatible version of the function solve.
For details of the function, see solve.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

See Also
	solve
Description

sort.rv computes the distribution of the order statistics of a random vector.

Usage

## S3 method for class 'rv'
sort(x, ...)

Arguments

x a random vector
...
... further arguments passed to sort.rv

Details

The result is the distribution of the order statistic of the given vector \( x \): that is, the \texttt{sort} function is applied to each row of the matrix of simulations of \( x \) (sims(\( x \))) and returned then in random vector form.

See \texttt{sort} for further details of the function \texttt{sort}.

Value

An \texttt{rv} object of the same length as \( x \).

Author(s)

Jouni Kerman <jouni@kerman.com>

References


See also vignette("rv").

See Also

\texttt{sort}
Description

`splitbyname` is a utility function that splits the given vector based on the names of the components and returns a named list of arrays and vectors.

Usage

```r
splitbyname(x)
```

Arguments

- `x` a vector or a list with the name attributes set

Details

The names are supposed to be of the format `name[index]`, for example `alpha[1,1]`, `beta[1]`, etc.
A name without brackets is equivalent to a name with `[1]`.
The dimension attribute will not be set in case of vectors.

Value

A list of arrays and vectors. Missing entries in the arrays and vectors are filled in with `NA`s.

Author(s)

Jouni Kerman <jouni@kerman.com>

Examples

```r
x <- structure(c(1,3), names=c("x[1,1]", "x[3,3]"))
splitbyname(x) # yields a list containing a 3x3 matrix
```
Random Vector Summaries

Description
rvsummary is a class of objects that hold the summary information on each scalar component of a random variable (quantiles, mean, sd, number of simulations etc.)

Usage
as.rvsummary(x, ...)

is.rvsummary(x)

## S3 method for class 'data.frame'
as.rvsummary(x, quantiles = rvpar("summary.quantiles.numeric"), ...)

## S3 method for class 'rvsummary_rvfactor'
print(x, all.levels = FALSE, ...)

Arguments
x object to be coerced or tested
...
quantiles quantiles to calculate and store in the object
all.levels logical; whether to print all levels or not (see below for details)

Details
The rvsummary class provides a means to store a concise representation of the marginal posterior distributions of the vector components. By default, the 201 quantiles

\[0, 0.005, 0.01, 0.015, \ldots, 0.990, 0.995, 1\]

are saved for each vector component in an rvsummary object.

is.rvsummary tests whether the object is an rvsummary object; as.rvsummary coerces a random vector object to a rvsummary object.
as.data.frame is another way to obtain the data frame that is produced by the summary method.

A data frame that has the format of an rv summary can be coerced into an rvsummary; if quantiles are not specified within the data frame, quantiles from the Normal distribution are filled in, if the mean and s.d. are given.

Therefore, the following (generic) functions work with rvsummary objects: rvmean, rvsd, rvvar, rvquantile, rnsims, sims, and consequently any 'rv-only' function that depends only on these functions will work; e.g. is.constant, which depends only on rvnsims.
The method `is.double` is provided for compatibility reasons; this is needed in a function called by `plot.rvsummary`

The arithmetic operators and mathematical functions will not work with `rvsummary` objects.

The `sims` method returns the quantiles.

**Value**

An object of class `rvsummary` and of subclass `rvsummary_numeric`, `rvsummary_integer`, `rvsummary_logical`, or `rvsummary_rvfactor`.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**


See also vignette("rv").

**See Also**

`rvfactor`

**Examples**

```r
x <- rvnorm(mean=1:12)
sx <- as.rvsummary(x)
print(sx) # prints the summary of the rvsummary object
length(sx) # 12
dim(sx) # NULL
dim(sx) <- c(3,4) #
dimnames(sx) <- list(1:3, 1:4)
names(sx) <- 1:12 #
print(sx) # prints the names and dimnames as well
```

---

**unlistrv**

Flatten Lists Containing rv Objects

**Description**

Given a list structure `x`, `unlistrv` simplifies it to produce a vector which contains all the atomic components (`containing rv objects`) which occur in `x`.

**Usage**

`unlistrv(x, recursive = TRUE, use.names = TRUE)`
Arguments

- **x**: An R object, typically a list or vector (containing rv objects)
- **recursive**: logical. Should unlisting be applied to list components of x?
- **use.names**: logical. Should names be preserved? (now fixed to TRUE)

Details

This is the rv-compatible version of the function `unlist`.
Since `unlist` is not a generic function, the whole name `unlistrv` must be specified when calling the function when x is an 'rv' object.

Author(s)

Jouni Kerman <jouni@kerman.com>

References

See also vignette("rv").

See Also

- `unlist`

Examples

```r
x <- list(a=rvnorm(2), b=rvnorm(3))
print(unlistrv(x))
```

%*in*%

Test if in set

Description

Test if in set

Usage

```r
x %*in*% y
```

Arguments

- **x**: random vector, regular atomic vector, or NULL: the values to be matched.
- **y**: random vector, regular atomic vector, or NULL: the values to be matched against.
Index

!-rv (Math.rv), 24
* aplot
  abline.rv, 4
  lines.rv, 23
  plot.rv, 33
  points.rv, 34
* arith
  matmult, 26
* array
  matmult, 26
* classes
  as.double.rv, 7
  as.integer.rv, 8
  as.list.rv, 9
  as.logical.rv, 10
  as.rv.bugs, 11
  as.vector.rv, 12
  cbind.rv, 13
  cc, 14
  detach.rv, 15
  Extract-rv, 16
  Extremes-rv, 17
  hist.rv, 18
  is.constant, 19
  is.fuzzy, 20
  is.na.rv, 21
  Math.rv, 24
  mean.rv, 27
  median.rv, 28
  mlplot, 29
  numeric_rv, 31
  outer.rv, 32
  posterior, 36
  postsim, 37
  print.rv, 38
  print.rvfactor, 39
  quantile.rv, 40
  range.rv, 41
  rv, 43
  rv-package, 4
  rvattr, 44
  rvbern, 45
  rvbeta, 46
  rvbinom, 47
  rvboot, 48
  rvcat, 49
  rvcauchy, 50
  rvchisq, 51
  rvci, 52
  rvconst, 53
  rvcov, 54
  rvcut, 55
  rvdens, 56
  rvdirichlet, 57
  rvdiscrete, 58
  rvempirical, 59
  rvexp, 60
  rvgamma, 61
  rvhist, 62
  rvifelse, 62
  rvinvchisq, 63
  rvmatch, 65
  rvmatrix, 67
  rvmean, 68
  rvmultinom, 71
  rvchains, 72
  rvneff, 73
  rvnorm, 74
  rvnsims, 75
  rvpar, 76
  rvpermut, 77
  rvpois, 78
  rvquantile, 80
  rvRhat, 82
  rvsample, 82
  rvsimapply, 83
  rvstats, 84
  rvts, 86
rvunif, 87
rvvar, 87
sims, 89
solve.rv, 91
summaries, 94
* dist
   rvmeanunif, 69
* hplot
   ivplot, 22
* manip
   aperm.rv, 5
   apply.rv, 6
   as.rv.stanfit, 12
   rep.rv, 42
   rvmapply, 64
   simapply, 88
   sort.rv, 92
   splitbyname, 93
   unlistrv, 95
* models
   rvpredict, 79
   [.rv (Extract-rv), 16
   [.rvfactor (Extract-rv), 16
   [.rvsummary (Extract-rv), 16
   [<-.rv (Extract-rv), 16
   [.rvsummary (Extract-rv), 16
   %**% (matmult), 26
   %**.rv (matmult), 26
   %in%, 96
   abline, 5, 65
   abline.rv, 4
   aperm, 6
   aperm.rv, 5
   apply, 7
   apply.rv, 6
   array, 67
   as.constant (is.constant), 19
   as.data.frame.rvsummary (summaries), 94
   as.double.rv, 7
   as.double.rvsummary (summaries), 94
   as.integer.rv, 8
   as.list.rv, 9
   as.logical.rv, 9, 10
   as.matrix.rv(rvmatrix), 67
   as.numeric.rv (numeric_rv), 31
   as.numeric.rvfactor (numeric_rv), 31
   as.rv (rv), 43
   as.rv.bugs, 11, 72
   as.rv.rvfactor (print.rvfactor), 39
   as.rv.stanfit, 12
   as.rvfactor (print.rvfactor), 39
   as.rvobj (rv), 43
   as.rvsummary (summaries), 94
   as.rvsummary.bugs (as.rv.bugs), 11
   as.vector.rv, 12
   c.rv (cc), 14
   c.rvsummary (cc), 14
   Cauchy, 50
   cbind, 13
   cbind.rv, 13
   cc, 14
   cummax.rv (Math.rv), 24
   cummin.rv (Math.rv), 24
   cumprod.rv (Math.rv), 24
   cumsum.rv (Math.rv), 24
   cut, 55
   detachrv, 15
   diag, 26
   drop, 26
   E (rvmean), 68
   Extract-rv, 16
   Extremes-rv, 17
   fuzzy (is.fuzzy), 20
   getnsims (rvnsims), 75
   hist, 18
   hist.rv, 18
   ifelse, 63
   impute<- (Extract-rv), 16
   is.constant, 19
   is.fuzzy, 20
   is.matrix.rv (rvmatrix), 67
   is.na.rv, 21
   is.numeric.rv (numeric_rv), 31
   is.numeric.rvfactor (numeric_rv), 31
   is.random (rv), 43
   is.rv (rv), 43
   is.rvfactor (print.rvfactor), 39
   is.rvobj (rv), 43
   is.rvsummary (summaries), 94
   ivplot, 22
lines.rv, 23, 35, 36
mapply, 64, 65
Math.rv, 24
Math.rvsim(Math.rv), 24
matmult, 26
matrix, 26, 67
max.rv(Extremes-rv), 17
mean.rv, 27, 69
median.rv, 28
min.rv(Extremes-rv), 17
mlplot, 29, 34, 67
NA, 41
numeric, 32
numeric.rv (numeric_rv), 31
numeric_rv, 31
Ops, 26
Ops.rv (Math.rv), 24
Ops.rvsim(Math.rv), 24
outer.rv, 32
par, 30
plot.rv, 33
plot.rvsummary (plot.rv), 33
pmax.rv(Extremes-rv), 17
pmin.rv(Extremes-rv), 17
points.rv, 24, 34
posterior, 36, 37
postsim, 37
Pr (rvmean), 68
print.rv, 38
print.rvfactor, 39
print.rvsummary (summaries), 94
print.rvsummary_rvfactor (summaries), 94
quantile, 28
quantile.rv, 40, 42
range, 41
range.rv, 41
rbind, 13
rbind.rv (cbind.rv), 13
rep, 42, 43
rep.rv, 42
rv, 43
rv-package, 4
rv.all.na(is.na.rv), 21
rv.any.na(is.na.rv), 21
rvarray (rvmatrix), 67
rvatr, 44
rvattr<-(rvattr), 44
rvbern, 45
rvbeta, 46
rvbinom, 47
rvboot, 48
rvcat, 49
rvcauchy, 50
rvchisq, 51
rvci, 52
rvconst, 53
rvcov, 54
rvcut, 55
rvedens, 56
rvedirichlet, 57
rvdiscrete, 58
rvempirical, 59
rvexp, 60
rvfactor, 39, 50, 55, 95
rvfactor (print.rvfactor), 39
rvgamma, 61
rvhist, 62
rvifelse, 62
rvinvchisq, 63
rvmapply, 64, 84, 89
rvmatch, 65
rvmatrix, 67
rvmax, 68, 69, 88
rvmean, 18, 27, 68, 89
rvmeanunif, 69
rvmedian, 18, 28, 69, 88
rvmedian (rvquantile), 80
rvmin, 69, 70, 88
rvmultinom, 71
rvnbeta (rvbeta), 46
rvnchains, 72
rvneff, 73
rvngamma (rvgamma), 61
rvnorm, 74
rvnchisq, 75
rvpar, 76
rvpermuted, 77
rvpois, 78
rvpredict, 79
rvpredict, 79
rvquantile, 41, 80
rvrange, 81
rvrange, 82
rvRhat, 82
rvsample, 82
rvsd, 69, 88
rvsd(rvvar), 87
rvsimapply, 83, 89
rvsims, 84
rvsummary(summaries), 94
rvtriang(rvmeanunif), 69
rvunif, 87
rvvar, 87
rvVectorize(rvmapply), 64

setnsims, 44
setnsims(rvnsims), 75
simapply, 65, 84, 88
sims, 89
solve, 65, 91
solve.rv, 91
sort, 92
sort.rv, 92
splitbyname, 93
summaries, 94

unlist, 96
unlistrv, 95