

# Package ‘rv’

January 27, 2012

**Title** Simulation-based random variable object class in R

**Version** 2.0.0

**Date** 2012-01-27

**Author** Jouni Kerman <jouni@kerman.com>

**Maintainer** Jouni Kerman <jouni@kerman.com>

**Description** Simulation-based random variable object class

**Namespace** rv

**Depends** R(>= 2.10.0), stats, utils, grDevices, graphics

**License** GPL-2

## R topics documented:

|                         |    |
|-------------------------|----|
| rv-package . . . . .    | 3  |
| abline.rv . . . . .     | 4  |
| aperm.rv . . . . .      | 5  |
| apply.rv . . . . .      | 6  |
| as.double.rv . . . . .  | 7  |
| as.integer.rv . . . . . | 8  |
| as.list.rv . . . . .    | 9  |
| as.logical.rv . . . . . | 9  |
| as.rv.bugs . . . . .    | 10 |
| as.vector.rv . . . . .  | 11 |
| c . . . . .             | 12 |
| cbind.rv . . . . .      | 13 |
| constant . . . . .      | 13 |
| detachrv . . . . .      | 14 |
| Extract.rv . . . . .    | 15 |
| Extremes-rv . . . . .   | 16 |
| fuzzy . . . . .         | 17 |
| hist.rv . . . . .       | 18 |
| is.na.rv . . . . .      | 19 |
| lines.rv . . . . .      | 20 |
| Math.rv . . . . .       | 21 |
| matmult.rv . . . . .    | 22 |

|                       |    |
|-----------------------|----|
| mean.rv . . . . .     | 23 |
| median.rv . . . . .   | 24 |
| mlplot . . . . .      | 24 |
| numeric.rv . . . . .  | 27 |
| outer.rv . . . . .    | 28 |
| plot.rv . . . . .     | 29 |
| points.rv . . . . .   | 30 |
| posterior . . . . .   | 32 |
| postsim . . . . .     | 33 |
| print.rv . . . . .    | 33 |
| quantile.rv . . . . . | 34 |
| range.rv . . . . .    | 35 |
| rep.rv . . . . .      | 36 |
| rv . . . . .          | 37 |
| rvarray . . . . .     | 38 |
| rvattach . . . . .    | 39 |
| rvattr . . . . .      | 40 |
| rvbern . . . . .      | 41 |
| rvbeta . . . . .      | 42 |
| rvbinom . . . . .     | 43 |
| rvboot . . . . .      | 44 |
| rvcat . . . . .       | 45 |
| rvcauchy . . . . .    | 46 |
| rvchisq . . . . .     | 46 |
| rvci . . . . .        | 47 |
| rvconst . . . . .     | 48 |
| rvcov . . . . .       | 49 |
| rvcut . . . . .       | 50 |
| rvdens . . . . .      | 51 |
| rvdirichlet . . . . . | 52 |
| rvdiscrete . . . . .  | 53 |
| rvexp . . . . .       | 54 |
| rvfactor . . . . .    | 54 |
| rvgamma . . . . .     | 55 |
| rvhist . . . . .      | 56 |
| rvifelse . . . . .    | 57 |
| rvinvchisq . . . . .  | 58 |
| rvmapply . . . . .    | 59 |
| rvmean . . . . .      | 60 |
| rvmeanunif . . . . .  | 61 |
| rvmultinom . . . . .  | 62 |
| rvnchains . . . . .   | 63 |
| rvneff . . . . .      | 63 |
| rvnorm . . . . .      | 64 |
| rvnsims . . . . .     | 65 |
| rvpar . . . . .       | 66 |
| rvpermut . . . . .    | 67 |
| rvpois . . . . .      | 68 |
| rvpredict . . . . .   | 69 |
| rvquantile . . . . .  | 70 |
| rvRhat . . . . .      | 71 |
| rvsample . . . . .    | 72 |

|                       |    |
|-----------------------|----|
| rvsimapply . . . . .  | 73 |
| rvsims . . . . .      | 74 |
| rvsummary . . . . .   | 75 |
| rvt . . . . .         | 76 |
| rvunif . . . . .      | 77 |
| rvvar . . . . .       | 78 |
| simapply . . . . .    | 79 |
| sims . . . . .        | 80 |
| solve.rv . . . . .    | 81 |
| sort.rv . . . . .     | 82 |
| splitbyname . . . . . | 83 |
| unlist.rv . . . . .   | 83 |

|              |           |
|--------------|-----------|
| <b>Index</b> | <b>85</b> |
|--------------|-----------|

---

|            |   |
|------------|---|
| rv-package | <i>Simulation-based Random Variable Objects</i> |
|------------|---|

---

## Description

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

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

## Details

|            |   |
|------------|---|
| Package:   | rv  |
| Version:   | 2.0.0   |
| Date:      | 2012-01-27  |
| Namespace: | rv  |
| Depends:   | R(>= 2.10.0), methods, utils, grDevices, graphics |
| License:   | GPL-2   |

## Author(s)

Jouni Kerman <jouni@kerman.com> Package built on Fri Jan 27 01:10:32 CET 2012

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

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 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, ...)
```

## Arguments

|     |  |
|-----|--|
| a   | intercept  |
| b   | slope  |
| h   | y-value(s) horizontal line(s)                      |
| v   | x-value(s) horizontal line(s)                      |
| ... | further arguments passed to <a href="#">abline</a> |

## 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

## Examples

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

## End(Not run)
```

## Description

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

## Usage

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

## Arguments

|      |   |
|------|---|
| a    | the random matrix to be transposed  |
| perm | the subscript permutation vector. See the manual page for the generic method <code>aperm</code> . |
| ...  | further arguments passed to <code>aperm</code>  |

## Details

This is the rv-compatible version of the function [aperm](#). It first applies

## Author(s)

Jouni Kerman <jouni@kerman.com>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[aperm](#)

## Examples

```
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

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

## Arguments

|                     |  |
|---------------------|--|
| <code>X</code>      | a random array                           |
| <code>MARGIN</code> | subscripts.                              |
| <code>FUN</code>    | function.                                |
| <code>...</code>    | optional arguments to <code>FUN</code> . |

## Details

This is the rv-compatible version of the function [apply](#).

## Author(s)

Jouni Kerman <jouni@kerman.com>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[apply](#)

## Examples

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

## End(Not run)
```

---

as.double.rv*Coercing Random Vectors to Real-valued*

---

## Description

Coerces random vector objects into double-valued ones.

## Usage

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

as.real is a synonym.

## Author(s)

Jouni Kerman <jouni@kerman.com>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

## Examples

```
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

```
## 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>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[as.logical.rv](#).

## Examples

```
x <- rvpois(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
```



---

`as.list.rv`*Coerce a random vector object to a list*

---

**Description**

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

**Usage**

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

**Arguments**

|                  |                                      |
|------------------|--------------------------------------|
| <code>x</code>   | an <code>rv</code> object            |
| <code>...</code> | 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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
x <- rvmnorm(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**

```
## 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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
x <- rvbern(prob=0.5) # some 0/1 valued random variable
print(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*


---

**Description**

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

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

**Usage**

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

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

---

|              |                            |
|--------------|----------------------------|
| as.vector.rv | <i>Coerce an rv object</i> |
|--------------|----------------------------|

---

**Description**

`as.vector.rv` coerces a given `rv` object into a vector; matrices lose their dimension attributes, but `rv` objects stay as `rv` objects (since they are considered to be “vectors”).

**Usage**

```
## S3 method for class 'rv'
as.vector(x, mode="any")
```

**Arguments**

|                   |                      |
|-------------------|----------------------|
| <code>x</code>    | an object            |
| <code>mode</code> | (currently not used) |

**Details**

`as.vector.rv` 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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

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

c

*Concatenation of random vectors***Description**

Concatenates random vectors.

**Usage**

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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
x <- rnorm(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)
```

---

cbind.rv

Combine random vectors by columns or rows

---

## Description

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

## Usage

```
cbind.rv(..., deparse.level = 1)
rbind.rv(..., deparse.level = 1)
```

## Arguments

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

## Details

See [cbind](#) and [rbind](#) for details.

## Author(s)

Jouni Kerman <jouni@kerman.com>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## Examples

```
x <- rvmnorm(10)
y <- rvmnorm(10)
cbind.rv(x, y)
rbind.rv(x, y)
```

---

constant

Constant Vectors

---

## Description

Functions to coerce or test for non-random objects.

**Usage**

```
is.constant(x)
as.constant(x)
## S3 method for class 'rv'
as.constant(x)
## S3 method for class 'rvsummary'
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.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
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
```

---

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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
## Not run:
library(rv)
detachrv()

## End(Not run)
```

---

Extract.rv

*Extract or Replace Parts of a Random Vector*

---

**Description**

Bracket slice and assignment methods adapted for random vectors and arrays.

**Usage**

```
\bsl{}method\{[\]\}{rv\}(x, ... , drop = TRUE)
\bsl{}method\{[\]\}{rvsummary\}(x, ... , drop = TRUE)
\bsl{}method\{[<-\}\}{rv\}(x, ... , drop = TRUE)
\bsl{}method\{[<-\}\}{rvsummary\}(x, ... , drop = TRUE)
```

**Details**

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.

**Value**

A random variable (an rv object).

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
#
```

---

Extremes-rv

*Maxima and Minima of Random Variables*

---

**Description**

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

**Usage**

```
rvmin(x)
rvmax(x)
rvrange(x)
## S3 method for class 'rv'
min(..., na.rm=FALSE)
## S3 method for class 'rv'
max(..., na.rm=FALSE)
## S3 method for class 'rv'
pmin(..., na.rm=FALSE)
## S3 method for class 'rv'
pmax(..., na.rm=FALSE)
```

**Arguments**

|                    |   |
|--------------------|---|
| <code>x</code>     | an rv or rvsummary object                 |
| <code>na.rm</code> | remove missing values?                    |
| <code>...</code>   | one or more rv objects or numeric objects |

**Details**

`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.



**Value**

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

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

[rvmedian](#), [rvmean](#).

**Examples**

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

---

|       |                  |
|-------|------------------|
| fuzzy | <i>Fuzziness</i> |
|-------|------------------|

---

**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**

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

**Arguments**

`x` an object, random or constant

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
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
```

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**

```
## 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 <a href="#">hist</a>                               |

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
x <- rvnorm(30)
hist(x)
```

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**

```
## S3 method for class 'rv'
is.na(x)
rv.all.na(x)
rv.any.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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

**Examples**

```
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
```

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**

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

**Arguments**

|                   |   |
|-------------------|---|
| <code>x, y</code> | coordinate vectors of points to join  |
| <code>type</code> | character indicating the type of plotting, currently 'l' and 'p' are the only possibilities |
| <code>...</code>  | 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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
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")
```

## 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)
```

## Arguments

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

## 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## Examples

```
x <- rvmnorm(10)  
-x  
names(x) <- paste("x[", seq_along(x), "]", sep="")  
x + 1:10  
1:2 + x  
cumsum(x)  
cumprod(exp(x))
```

## 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

```
## S3 method for class 'rv'
x %**% y
x %**% 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 [drop](#) to get rid of dimensions which have only one level.

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[matrix](#), [Ops](#), [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
y %**% x
x %**% z
```

---

mean.rv*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

```
## 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  $\frac{1}{n} \sum_{i=1}^n x_i$  (sum(x)/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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. Statistics and Computing 17:3, 235-244.

See also vignette("rv").

## See Also

[rvmean](#)

## Examples

```
y <- rvmnorm(10, mean=0, sd=1)  
m1 <- mean(y)  
m2 <- rvmnorm(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

```
## 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.     |

### References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

### See Also

[rvmedian](#) for the componentwise medians. [quantile](#) for general quantiles.

### Examples

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

---

mplot

*Horizontal interval plot of components of a random vector*


---

### Description

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



**Usage**

```

mplot(X, ...)
## S3 method for class 'rvsummary'

## Default S3 method:
mplotmplot(X, y.center = TRUE, y.shift = 0, y.map = NULL, mar =
  par("mar"), left.margin = 3, 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, ...)
## Default S3 method:
mplot(X, y.center = TRUE, y.shift = 0, y.map = NULL, mar =
  par("mar"), left.margin = 3, 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**

|                          |  |
|--------------------------|--|
| <code>X</code>           | a random array or vector   |
| <code>y.center</code>    | center the intervals nicely at each y-coordinate?                                    |
| <code>y.shift</code>     | add this amount to each y coordinate of an interval                                  |
| <code>y.map</code>       | optional function to compute the y-coordinates, given <code>X</code>                 |
| <code>mar</code>         | the margins of the plot  |
| <code>left.margin</code> | offset to add to the left margin of the plot (to add space for the labels)           |
| <code>top.axis</code>    | (logical) plot the top axis?   |
| <code>exp.labels</code>  | (logical) if the original scale is logarithmic, label ticks in original (exp) scale? |
| <code>x.ticks</code>     | positions for the ticks of the x-axis  |
| <code>axes</code>        | (logical) plot the axes at all?  |
| <code>xlim</code>        | x limits   |
| <code>ylim</code>        | y limits   |
| <code>las</code>         | the style of axis labels, see <a href="#">par</a>                                    |
| <code>add</code>         | (logical) add the intervals to an existing plot?                                     |
| <code>xlab</code>        | x label  |
| <code>ylab</code>        | not used (instead of labels, the row names are shown)                                |
| <code>...</code>         | further arguments passed to plot and points  |

**Details**

`mplot` 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 'mplot' 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% NA) are not plotted.

`mlplot` is still experimental.

### Author(s)

Jouni Kerman <jouni@kerman.com>

### References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

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 <- rmatrix(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)
```

## Description

Creates or coerces rv objects of type "numeric".

## Usage

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

`is.numeric(x)` returns TRUE if and only if *each* component of `x` is numeric-valued.

`as.numeric.rv` coerces an rv object into numeric-valued one. In effect, the function `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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[numeric.](#)

## Examples

```
x <- as.logical(rvbern(1,0.5)) # Bernoulli rv  
is.numeric(x)                 # FALSE  
x <- as.numeric(x)            # coerce to numeric; all TRUEs become ones, FALSEs zeros  
is.numeric(x)                 # TRUE
```

---

`outer.rv`*Outer Product of Random Arrays*

---

**Description**`outer.rv`**Usage**`outer.rv(X, Y=NULL, FUN="*", ...)`**Arguments**

|                  |   |
|------------------|---|
| <code>X</code>   | First argument for function FUN   |
| <code>Y</code>   | Second argument for function FUN; if missing, X is used instead           |
| <code>FUN</code> | a function to use on the outer products; a character string or a function |
| <code>...</code> | optional arguments to be passed to FUN                                    |

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
#
```

**Description**

Draw a "random scatter plot" or random points as horizontal or vertical intervals.

**Usage**

```
## S3 method for class 'rv'  
plot(x, y, ...)  
## S3 method for class 'rvsummary'  
plot(x, y, ...)
```

**Arguments**

|     |                                   |
|-----|-----------------------------------|
| x   | an rv object                      |
| y   | random or fixed vector            |
| ... | other arguments passed on to plot |

**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 `code(x,y)`.

Compatible with objects of class 'rvsummary'.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

[mplot](#)

## Examples

```
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                            |
| rvcol   | colors for the intervals   |
| xlim    | x-limits (optional)  |
| ylim    | y-limits (optional)  |
| rvlwd   | line width of the thin interval                                      |
| 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 the second 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% interval, use rvpoint=c("mean", NA, "95%") or rvpoint=c("mean", "95%", NA).

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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

## Examples

```
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, ...)
## S3 method for class 'lm'
posterior(obj, ...)
## S3 method for class 'glm'
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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
## 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)
```



---

|         |   |
|---------|---|
| postsim | <i>Generate Posterior Simulations for lm or glm Objects (defunct)</i> |
|---------|---|

---

**Description**

*DEFUNCT.* Use `posterior` instead.

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

**Usage**

```
postsim(fit)
```

**Arguments**

|     |                     |
|-----|---------------------|
| fit | an lm or glm object |
|-----|---------------------|

**Value**

A (named) random vector for each fitted coefficient.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

---

|          |  |
|----------|--|
| print.rv | <i>Print Distribution Summary of a Random Variable</i> |
|----------|--|

---

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

[summary.rv](#), [rvfactor](#)

**Examples**

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

---

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**

|                  |   |
|------------------|---|
| <code>x</code>   | an object   |
| <code>...</code> | further arguments passed to or from other methods |

**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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## Examples

```
x <- rvmnorm(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

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

## Arguments

|                     |  |
|---------------------|--|
| <code>...</code>    | further arguments passed to or from other methods                |
| <code>na.rm</code>  | logical, indicating if <a href="#">NAs</a> should be omitted     |
| <code>finite</code> | 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[quantile.rv](#)

## Examples

```
x <- rvmnorm(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

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

## Arguments

|                    |  |
|--------------------|--|
| <code>x</code>     | a random vector to be replicated             |
| <code>times</code> | number of replications                       |
| <code>...</code>   | further arguments passed to <code>rep</code> |

## 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[rep](#)

## Examples

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

---

|    |                       |
|----|-----------------------|
| rv | <i>Random Vectors</i> |
|----|-----------------------|

---

**Description**

Creates or tests for objects of type “rv”.

**Usage**

```
rv(length = 0)
as.rv(x, ...)
is.rv(x)
is.random(x)
as.rvobj(x)
is.rvobj(x)
```

**Arguments**

|        |  |
|--------|--|
| length | desired length.                                    |
| x      | object to be coerced or tested.                    |
| ...    | further arguments passed to or from other methods. |

**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 rvsummary.

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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

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

**Examples**

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

---

rvarray

*Matrices and Arrays of Random Vectors*


---

**Description**

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

These are 'rv' compatible versions of the functions [matrix](#) and [array](#).

**Usage**

```
rvarray(data = NA, dim = length(data), dimnames = NULL)
rvmatrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
## S3 method for class 'rv'
is.matrix(x)
## S3 method for class 'rv'
as.matrix(x, ...)
```

**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. |
| ...      | arguments passed to other methods   |
| x        | an R object.  |

**Details**

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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

To plot random matrices, see `mlplot`.

**Examples**

```
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))
```

---

rvattach

---

*Attach components of rv objects to search path*


---

**Description**

`rattach` works like `attach` but removes objects from `.GlobalEnv` that have conflicting names; optionally can also *impute* vector components *by name* into the object that is in `.GlobalEnv` (copying the imputed object into the attached list. `rvattach` first splits a named vector into a list, and then uses `rattach` to attach them to the search path.

**Usage**

```
rattach(what, name=deparse(substitute(what)), overwrite=TRUE, impute=FALSE, ...)
rvattach(what, name=deparse(substitute(what)), overwrite=TRUE, impute=FALSE, ...)
```

**Arguments**

|                        |   |
|------------------------|---|
| <code>what</code>      | an rv object to be split up into a list of sub-vectors and attached   |
| <code>name</code>      | name to use for the attached database. All databases with the exact same name in the search path are removed first.   |
| <code>overwrite</code> | If TRUE, objects with identical names in the workspace ( <code>.GlobalEnv</code> ) that are masking objects in the database to be attached will be deleted.   |
| <code>impute</code>    | If TRUE, the components of each sub-vector are imputed into the object with identical name in the workspace and saved into the list. The object in the workspace is not affected (unless <code>overwrite=TRUE</code> ). |
| <code>...</code>       | further arguments passed to <code>attach</code>   |

## Details

rvattach takes an rv object, splits it up into a list of sub-vectors (using `split.rv`) by their name attributes and attaches the list to the search path. For example, if the object to be processed has names `x[1]` and `y[3]`, this will be split into a list of two sub-vectors, `x` and `y`, and then attached.

If the option `impute` is `TRUE`, the corresponding variables with the same names are merged with the ones that are found in `.GlobalEnv`, and copied into the database.

If the option `overwrite` is `TRUE`, the objects with identical names in the Workspace are deleted.

All databases with the exact same name in the search path are removed first.

## Author(s)

Jouni Kerman <jouni@kerman.com>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## Examples

```
# Trying rattach:
x <- rvmnorm(9)
names(x) <- paste("theta[", 1+seq_along(x), "]", sep="")
L <- list(x=x)
x <- 1:10
rattach(L, impute=TRUE) # attaches a list called 'L' containing an object 'x'
print(x)
detach("L")
exists("x") # FALSE
# Trying rvattach:
rvattach(L$x) # attaches a list 'rvattach' containing an object 'theta'
print(theta)
rvattach() # Detaches the list 'rvattach'
exists("theta") # FALSE
```

---

rvattr

*Attributes of Random Variables*

---

## Description

rvattr

## Usage

```
rvattr(x, attrib=NULL)
rvattr(x, attrib=NULL) <- value
```



**Arguments**

|        |                       |
|--------|-----------------------|
| x      | an object             |
| attrib | name of the attribute |
| value  | list of values to set |

**Details**

The values within the list value are matched by their name (e.g. 'theta[1]') if possible; if both x and value do not have names, the list is matched by position.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
#
```

---

rvbern

---

*Generate a Random Vector from a Bernoulli Sampling Model*


---

**Description**

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

**Usage**

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

**Arguments**

|         |   |
|---------|---|
| n       | number of random scalars to draw                        |
| prob    | probability of "success"; may be a random vector itself |
| logical | logical; return a logical random variable instead       |

**Details**

rvbern is a special case of `rvbinom` with the argument `size=1`.

If `logical` is `TRUE`, the function returns a logical random variable which has `TRUE` for 1, `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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

**Examples**

```
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;

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

**Usage**

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

**Arguments**

|        |   |
|--------|---|
| n      | integer, number of random variables to generate |
| shape1 | positive number or rv, 1st shape parameter      |
| shape2 | positive number or rv, 2nd shape parameter      |

**Author(s)**

Jouni Kerman <jouni@kerman.com>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## Examples

```
n <- 12          # sample size
y <- (0:(n-1))   # observations
a <- b <- 1/3    # the neutral beta prior
rvbeta(1, shape1=a+y, shape2=b+n-y)
rvnbeta(1, shape1=y, shape2=n-y)
```

---

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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```

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

```

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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

**Examples**

```

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.

```

---

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**

```
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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

**See Also**

[rvfactor](#)

**Examples**

```
rvcat(1, prob=c(0.5, 0.3, 0.2)) # default levels: 1, 2, 3
rvcat(1, prob=c(5, 3, 2)) # same as above
p <- rmdirichlet(1, alpha=c(0.7, 0.3)) # prior probabilities
rvcat(1, prob=p, levels=c("Group 1", "Group 2"))
```

---

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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

---

rvchisq

*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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
#
```

---

|      |  |
|------|--|
| rvci | <i>Credible (Uncertainty) Intervals for Random Scalars</i> |
|------|--|

---

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

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

---

rvconst

*Random Vector with a Point-Mass Distribution*


---

**Description**

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

**Usage**

```
rvconst(n=1, x)
```

**Arguments**

|                |  |
|----------------|--|
| <code>n</code> | integer: number of variables to generate |
| <code>x</code> | 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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.



**Examples**

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

rvcov

*Covariance Between Components of Random Vectors***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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
x <- rvmnorm(mean=1:3)
y <- rvmnorm(mean=2:4)
rvcov(x,y)
rvcov(x,x)
```

---

|       |   |
|-------|---|
| rvcut | <i>Convert Numeric to Random Factor</i> |
|-------|---|

---

## Description

Convert implements the ‘cut’ function using random variables.

## Usage

```
rvcut(x, ...)  
## S3 method for class 'rv'  
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 <a href="#">cut</a> .                      |

## Value

A random factor.

## Author(s)

Jouni Kerman <jouni@kerman.com>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[rvfactor](#), [cut](#).

## Examples

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

---

rvdens*Sample from an arbitrary density function using grid approximation*

---

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
x <- rvdens(FUN=stats::dnorm, range=c(-5, 5), unitprecision=10)
y <- rnorm(1) ## Should be close to x
```

---

|             |  |
|-------------|--|
| rmdirichlet | <i>Generate Random Variables from a Dirichlet Sampling Model</i> |
|-------------|--|

---

**Description**

Generates random variables from a Dirichlet sampling model.

**Usage**

```
rmdirichlet(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, rmdirichlet draws from the Beta model.)

**Value**

A random vector (rv object) of length n.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

**Examples**

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

---

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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## Examples

```
# 8 people draw a number each from 1..10 with replacement.
# What is the probability that the highest number of the eight is "10"?
u <- rvdiscrete(n=8, x=1:10) # 8 iid variables from the discrete uniform 1:10.
Pr(max(u)==10)
# What is the probability that the person with the 3rd smallest number
# has at least "3"?
s <- sort(u) # order distribution
Pr(s[3]>=3)
```

---

|       |   |
|-------|---|
| rvexp | <i>Generate Random Vectors from an Exponential Sampling Model</i> |
|-------|---|

---

**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 &lt;jouni@kerman.com&gt;

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
y <- rvexp(1, rate=rvexp(1)) # What marginal distribution does y have now?
```

---

|          |  |
|----------|--|
| rvfactor | <i>Categorical Random Variables (Random Factors)</i> |
|----------|--|

---

**Description**

Creates or tests for objects of type “rvfactor”.

**Usage**

```
rvfactor(x, ...)
## S3 method for class 'rv'
rvfactor(x, levels=NULL, ...)
is.rvfactor(x)
## S3 method for class 'rvfactor'
as.rv(x, ...)
as.rvfactor(x, ...)
## S3 method for class 'rvfactor'
print(x, all.levels=FALSE, ...)
```

**Arguments**

|                         |   |
|-------------------------|---|
| <code>x</code>          | object to be coerced or tested.                                     |
| <code>levels</code>     | factor levels (labels for the levels)                               |
| <code>all.levels</code> | logical; whether to print all levels or not (see below for details) |
| <code>...</code>        | other arguments   |

**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.

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

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

---

rvgamma

---

*Generate Random Variables from a Gamma Sampling Model*


---

**Description**

Generates random variables from a Gamma sampling model.

**Usage**

```
rvgamma(n=1, shape, rate = 1, scale = 1/rate)
rvngamma(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**

`rvngamma(n, shape, rate)` is equivalent to `rvgamma(n, 1/3 + shape, rate)`.

**Value**

A random vector (rv object).

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
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 <code>hist</code> |

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

[ifelse](#).

---

|            |   |
|------------|---|
| rvinvchisq | <i>Generate Random Variables from a Inverse-Chi-Square Sampling Model</i> |
|------------|---|

---

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
rvinvchisq(df=3, scale=2)
```

---

rvmapply

*Apply a function to multiple random vector arguments*


---

## 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)
rvVectorize(FUN, vectorize.args = arg.names, SIMPLIFY = FALSE, USE.NAMES = TRUE, SAMPLESIZE=NULL)
```

## Arguments

|                |  |
|----------------|--|
| FUN            | the function to apply to the simulations of X.   |
| MoreArgs       | Other args passed to FUN ‘as is’ (must not be rv objects unless the function already accepts them) |
| USE.NAMES      | logical; see <a href="#">mapply</a> for details  |
| SIMPLIFY       | logical; see <a href="#">mapply</a> for details  |
| SAMPLESIZE     | if specified, takes a (joint) sample of the simulations and processes only them.                   |
| vectorize.args | a character vector of arguments which should be vectorized. Defaults to all arguments to FUN.      |
| ...            | further arguments to FUN, possibly random vectors or array.  |

## 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## See Also

[mapply](#), [simapply](#)

---

|        |   |
|--------|---|
| rvmean | <i>Expectation of a Random Variable</i> |
|--------|---|

---

## Description

rvmean

## Usage

```
rvmean(x)
E(x)
Pr(X)
```

## Arguments

|   |                     |
|---|---------------------|
| x | an rv object        |
| X | a logical 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  $\text{Pr}(x > 0)$  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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

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

**Examples**

```
x <- rvmnorm(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  $(-1, 1)$ .

**Usage**

```
rvtriang(n = 1, mode, scale)
rvmeanunif(n=1, mode, scale, 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  $(-scale, scale)$ .

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**

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

---

`rvmultinom`*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**

|                   |  |
|-------------------|--|
| <code>n</code>    | integer, number of random variables to generate  |
| <code>size</code> | integer or integer-valued rv: the number of trials (size of each sample)                                       |
| <code>prob</code> | 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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

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

---

|           |  |
|-----------|--|
| rvnchains | <i>Number of Markov Chains Used to Generate Simulations of a Random Vector</i> |
|-----------|--|

---

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

[as.rv.bugs](#)

**Examples**

```
#
```

---

|        |   |
|--------|---|
| rvneff | <i>Number of Effective Draws in Each Component of a Random Variable</i> |
|--------|---|

---

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.  
See also `vignette("rv")`.

**Examples**

```
#
```

---

|        |  |
|--------|--|
| rvnorm | <i>Generate Random Variables from a Gaussian (Normal) Sampling Model</i> |
|--------|--|

---

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

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

---

rvnsims

*Number of simulations stored in each component of an rv object*

---

**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)
setnsims(n.sims)
getnsims()
```

**Arguments**

|                     |  |
|---------------------|--|
| <code>x</code>      | an rv object.                                      |
| <code>n.sims</code> | default number of simulations; must be at least 2. |

**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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
#
rvnsims(1.23)           # 1
x <- rvmnorm(1)         # an rv
rvnsims(x)              # equal to setnsims()
rvnsims(x)==nrow(sims(x)) # TRUE
rvnsims(x)==getnsims()  # TRUE
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
```

---

rvpar

---

*Set or Query Parameters of the 'rv' Package*


---

**Description**

Sets or retrieves parameters of the rv package.

**Usage**

```
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>

### References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

### Examples

```
rvpar()$rvcol
rvpar("rvcol")
```

---

 rvpermut

---

*Random Vectors with a Permutation Distribution*


---

### Description

Generates a random vector with each component having a permutation distribution based on the given (fixed) data vector.

### Usage

```
rvpermut(data, prob=NULL)
```

### Arguments

|                   |  |
|-------------------|--|
| <code>data</code> | a fixed numeric vector   |
| <code>prob</code> | optional probabilities for the components in <code>data</code> |

### Author(s)

Jouni Kerman <jouni@kerman.com>

### References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

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**

|                      |   |
|----------------------|---|
| <code>object</code>  | An object representing a statistical model fit.   |
| <code>newdata</code> | 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 <i>replications</i> of the data. <i>Note</i> : this can be an <i>rv</i> object to incorporate extra uncertainty into predictions. |
| <code>...</code>     | Arguments passed to and from other methods.   |

**Details**

The `lm` method generates predictions of the outcome variable. The posterior coefficient estimates (the “intercept” and the “betas”) are estimated Bayesianly 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**

```
## 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.rv(D$x1, y.rep)
points(D$x1, D$y, col="red")
## 'Perturb' (add uncertainty to) covariate x1
X.pred2 <- X
X.pred2$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, 0.10, 0.25, 0.50, 0.75, 0.90, 0.975), ignoreInf=FALSE, ...)
## S3 method for class 'rvsummary'
rvquantile(x, probs=c(0.025, 0.10, 0.25, 0.50, 0.75, 0.90, 0.975), ...)
rvmedian(x)

```

**Arguments**

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

**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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## Examples

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

---

|        |                                     |
|--------|-------------------------------------|
| rvRhat | <i>R-hat Convergence Diagnostic</i> |
|--------|-------------------------------------|

---

## 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

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**

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

**Arguments**

|                        |   |
|------------------------|---|
| <code>x</code>         | an object   |
| <code>size</code>      | size of the sample  |
| <code>jointly</code>   | return joint simulations and not simulations from each component separately |
| <code>reject.na</code> | 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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**Examples**

```
#
```



---

|            |   |
|------------|---|
| rvsimapply | <i>Apply a Function to Columns of the Matrix of Simulation of a Random Vector</i> |
|------------|---|

---

## 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 [rvmapapply](#) 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

## Examples

```
#
```

---

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

```
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

A vector is interpreted to contain simulations for one single random variable.

If sims is two-dimensional, the columns are supposed to contain simulations for 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

## Examples

```
# x and y have the same distributions but not the same simulations:
setnsims(200)
y <- rnorm(1)
x <- rvsims(rnorm(getnsims()))
#
```

**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**

```
is.rvsummary(x)
as.rvsummary(x, ...)
## S3 method for class 'rv'
as.rvsummary(x, quantiles = (0:200/200), ...)
## S3 method for class 'rvsummary'
as.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, ...)
```

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

**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) |
| ...        | further arguments passed to or from other methods.                  |

**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, ..., 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.

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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

### See Also

[rvfactor](#)

### Examples

```
x <- rvmnorm(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
```

---

rvt

*Generate Random Variables from a Student-t Sampling Model*

---

### Description

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

### Usage

```
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                                    |
| ncp   | non-centrality parameter                              |
| df    | degrees of freedom, may be a rv                       |
| 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 <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. Statistics and Computing 17:3, 235-244.

See also vignette("rv").

**Examples**

```
df <- 3
x <- rvt(n=1, df=df)
y <- rvmnorm(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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

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)
## S3 method for class 'rv'
rvvar(x)
## S3 method for class 'rvsummary'
rvvar(x)
rvsd(x)
## S3 method for class 'rv'
rvsd(x)
## S3 method for class 'rvsummary'
rvsd(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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

`rvmin`, `rvmax`, `rvmedian`, `rvsd`.

**Examples**

```
x <- rvmnorm(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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

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

## Arguments

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

## 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

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.



**Examples**

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

solve.rv

*Random Vectors***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**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

[solve](#)

**Examples**

```
#
```

---

sort.rv*Distribution of Order Statistics of a Random Vector*

---

## 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 sort function is applied to each *row* of the matrix of simulations of x (sims(x)) and returned then in random vector form.

See [sort](#) for further details of the function sort.

## Value

An rv object of the same length as x.

## Author(s)

Jouni Kerman <jouni@kerman.com>

## References

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also vignette("rv").

## See Also

[sort](#)

## Examples

```
#
```

splitbyname

*Split a vector based on the names of the components***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**

```
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 NAs.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**Examples**

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

unlist.rv

*Flatten Lists Containing rv Objects***Description**

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

**Usage**

```
## S3 method for class 'rv'
unlist(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 `unlist.rv` must be specified when calling the function when `x` is an 'rv' object.

**Author(s)**

Jouni Kerman <jouni@kerman.com>

**References**

Kerman, J. and Gelman, A. (2007). Manipulating and Summarizing Posterior Simulations Using Random Variable Objects. *Statistics and Computing* 17:3, 235-244.

See also `vignette("rv")`.

**See Also**

[unlist](#)

**Examples**

```
x <- list(a=rvnorm(2), b=rvnorm(3))
print(unlist.rv(x))
```

# Index

!.rv (Math.rv), 21

\*Topic **aplot**

- abline.rv, 4
- lines.rv, 20
- plot.rv, 29
- points.rv, 30

\*Topic **arith**

- matmult.rv, 22

\*Topic **array**

- matmult.rv, 22

\*Topic **classes**

- as.double.rv, 7
- as.integer.rv, 8
- as.list.rv, 9
- as.logical.rv, 9
- as.rv.bugs, 10
- as.vector.rv, 11
- c, 12
- cbind.rv, 13
- constant, 13
- detachrv, 14
- Extract.rv, 15
- Extremes-rv, 16
- fuzzy, 17
- hist.rv, 18
- is.na.rv, 19
- Math.rv, 21
- mean.rv, 23
- median.rv, 24
- mlplot, 24
- numeric.rv, 27
- outer.rv, 28
- posterior, 32
- postsim, 33
- print.rv, 33
- quantile.rv, 34
- range.rv, 35
- rv, 37
- rv-package, 3
- rvarray, 38
- rvattach, 39
- rvattr, 40
- rvbern, 41
- rvbeta, 42
- rvbinom, 43
- rvboot, 44
- rvcat, 45
- rvcauchy, 46
- rvchisq, 46
- rvci, 47
- rvconst, 48
- rvcov, 49
- rvcut, 50
- rvdens, 51
- rvdirichlet, 52
- rvdiscrete, 53
- rvexp, 54
- rvfactor, 54
- rvgamma, 55
- rvhist, 56
- rvifelse, 57
- rvinvchisq, 58
- rvmean, 60
- rvmultinom, 62
- rvnchains, 63
- rvneff, 63
- rvnorm, 64
- rvnsims, 65
- rvpar, 66
- rvpermut, 67
- rvpois, 68
- rvquantile, 70
- rvRhat, 71
- rvsample, 72
- rvsimapply, 73
- rvsims, 74
- rvsummary, 75
- rvt, 76
- rvunif, 77
- rvvar, 78
- sims, 80
- solve.rv, 81

\*Topic **dist**

- rvmeanunif, 61

\*Topic **manip**

- aperm.rv, 5

- apply.rv, 6
  - rep.rv, 36
  - rvmapply, 59
  - simapply, 79
  - sort.rv, 82
  - splitbyname, 83
  - unlist.rv, 83
- \*Topic **models**
  - rvpredict, 69
- [.rv (Extract.rv), 15
- [.rvsummary (Extract.rv), 15
- [<-.rv (Extract.rv), 15
- [<-.rvsummary (Extract.rv), 15
- %\*\*% (matmult.rv), 22
- %\*%.rv (matmult.rv), 22
- abline, 4, 59
- abline.rv, 4
- aperm, 5
- aperm.rv, 5
- apply, 6
- apply.rv, 6
- array, 38
- as.constant (constant), 13
- as.double.rv, 7
- as.double.rvsummary (rvsummary), 75
- as.integer.rv, 8
- as.list.rv, 9
- as.logical.rv, 8, 9
- as.matrix.rv (rvarray), 38
- as.numeric.rv (numeric.rv), 27
- as.numeric.rvfactor (numeric.rv), 27
- as.real.rv (as.double.rv), 7
- as.rv (rv), 37
- as.rv.bugs, 10, 63
- as.rv.rvfactor (rvfactor), 54
- as.rvfactor (rvfactor), 54
- as.rvobj (rv), 37
- as.rvsummary (rvsummary), 75
- as.rvsummary.bugs (as.rv.bugs), 10
- as.vector.rv, 11
- c, 12
- Cauchy, 46
- cbind, 13
- cbind.rv, 13
- cc (c), 12
- constant, 13
- cummax.rv (Math.rv), 21
- cummin.rv (Math.rv), 21
- cumprod.rv (Math.rv), 21
- cumsum.rv (Math.rv), 21
- cut, 50
- detachrv, 14
- diag, 22
- drop, 22
- E (rvmean), 60
- Extract.rv, 15
- Extremes-rv, 16
- fuzzy, 17
- getnsims (rvnsims), 65
- hist, 18
- hist.rv, 18
- ifelse, 57
- is.constant (constant), 13
- is.fuzzy (fuzzy), 17
- is.matrix.rv (rvarray), 38
- is.na.rv, 19
- is.numeric.rv (numeric.rv), 27
- is.numeric.rvfactor (numeric.rv), 27
- is.random (rv), 37
- is.rv (rv), 37
- is.rvfactor (rvfactor), 54
- is.rvobj (rv), 37
- is.rvsummary (rvsummary), 75
- lines.rv, 20, 30, 31
- mapply, 59, 60
- Math.rv, 21
- Math.rvsim (Math.rv), 21
- matmult.rv, 22
- matrix, 22, 38
- max.rv (Extremes-rv), 16
- mean.rv, 23, 61
- median.rv, 24
- min.rv (Extremes-rv), 16
- mlplot, 24, 29, 39
- NA, 35
- numeric, 27
- numeric.rv, 27
- Ops, 22
- Ops.rv (Math.rv), 21
- Ops.rvsim (Math.rv), 21
- outer.rv, 28
- par, 25
- plot.rv, 29
- plot.rvsummary (plot.rv), 29
- pmax.rv (Extremes-rv), 16

pmin.rv (Extremes-rv), 16  
 points.rv, 20, 30  
 posterior, 32, 33  
 postsim, 33  
 Pr (rvmean), 60  
 print.rv, 33  
 print.rvfactor (rvfactor), 54  
 print.rvsummary (rvsummary), 75  
 print.rvsummary\_rvfactor (rvsummary), 75  
  
 quantile, 24  
 quantile.rv, 34, 35  
  
 range, 35  
 range.rv, 35  
 rattach (rvattach), 39  
 rbind, 13  
 rbind.rv (cbind.rv), 13  
 rep, 36  
 rep.rv, 36  
 rv, 37  
 rv-package, 3  
 rv.all.na (is.na.rv), 19  
 rv.any.na (is.na.rv), 19  
 rvarray, 38  
 rvattach, 39  
 rvattr, 40  
 rvattr<- (rvattr), 40  
 rvbern, 41  
 rvbeta, 42  
 rvbinom, 43  
 rvboot, 44  
 rvcat, 45  
 rvcauchy, 46  
 rvchisq, 46  
 rvci, 47  
 rvconst, 48  
 rvcov, 49  
 rvcut, 50  
 rvdens, 51  
 rvidirichlet, 52  
 rvidiscrete, 53  
 rvexp, 54  
 rvfactor, 34, 45, 50, 54, 76  
 rvgamma, 55  
 rvhist, 56  
 rvifelse, 57  
 rvinvchisq, 58  
 rvmapapply, 59, 73, 79  
 rvmatrix (rvarray), 38  
 rvmax, 61, 79  
 rvmax (Extremes-rv), 16  
 rvmean, 17, 23, 60, 79  
 rvmeanunif, 61  
 rvmedian, 17, 24, 61, 79  
 rvmedian (rvquantile), 70  
 rvmin, 61, 79  
 rvmin (Extremes-rv), 16  
 rvmultinom, 62  
 rvnbeta (rvbeta), 42  
 rvnchains, 63  
 rvneff, 63  
 rvngamma (rvgamma), 55  
 rvnorm, 64  
 rvnsims, 65  
 rvpar, 66  
 rvpermut, 67  
 rvpois, 68  
 rvpredict, 69  
 rvquantile, 34, 70  
 rvrange (Extremes-rv), 16  
 rvRhat, 71  
 rvsample, 72  
 rvsd, 61, 79  
 rvsd (rvvar), 78  
 rvsimapply, 73, 79  
 rvsims, 74  
 rvsummary, 75  
 rvt, 46, 76  
 rvtriang (rvmeanunif), 61  
 rvunif, 77  
 rvvar, 78  
 rvVectorize (rvmapapply), 59  
  
 setnsims, 37  
 setnsims (rvnsims), 65  
 simapply, 60, 73, 79  
 sims, 80  
 solve, 59, 81  
 solve.rv, 81  
 sort, 82  
 sort.rv, 82  
 splitbyname, 83  
 summary.rv, 34  
  
 unlist, 84  
 unlist.rv, 83