Package 'DelayedRandomArray'

February 27, 2026

Version 1.18.0

Date 2024-07-05

Title Delayed Arrays of Random Values

Description Implements a DelayedArray of random values where the realization of the sampled values is delayed until they are needed.
Reproducible sampling within any subarray is achieved by chunking where each chunk is initialized with a different random seed and stream.
The usual distributions in the stats package are supported, along with scalar, vector and arrays for the parameters.

License GPL-3

Depends SparseArray (>= 1.5.15), DelayedArray (>= 0.31.6)

Imports methods, dqrng, Rcpp

LinkingTo dqrng, BH, Rcpp

Suggests testthat, knitr, BiocStyle, rmarkdown, Matrix

biocViews DataRepresentation

BugReports https://github.com/LTLA/DelayedRandomArray/issues

URL https://github.com/LTLA/DelayedRandomArray

SystemRequirements C++11

VignetteBuilder knitr

RoxygenNote 7.3.2

Encoding UTF-8

git_url https://git.bioconductor.org/packages/DelayedRandomArray

git_branch RELEASE_3_22

git_last_commit a060a75

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2026-02-27

Author Aaron Lun [aut, cre]

Maintainer Aaron Lun <infinite.monkeys.with.keyboards@gmail.com>

Contents

RandomArraySeed-class . . . 2
RandomBetaArray-class . . . 4
RandomBinomArray-class . . . 5
RandomCauchyArray-class . . . 6
RandomChisqArray-class . . . 7
RandomExpArray-class . . . 8
RandomFArray-class . . . 9
RandomGammaArray-class . . . 11
RandomGeomArray-class . . . 12
RandomHyperArray-class . . . 13
RandomLnormArray-class . . . 14
RandomLogisArray-class . . . 15
RandomNbinomArray-class . . . 16
RandomNormArray-class . . . 17
RandomPoisArray-class . . . 19
RandomTArray-class . . . 20
RandomUnifArray-class . . . 21
RandomWeibullArray-class . . . 22
RandomWilcoxArray-class . . . 23

Index **25**

`RandomArraySeed-class` *A DelayedArray seed supplying chunked random values*

Description

The RandomArraySeed is a DelayedArray seed that performs reproducible, on-demand sampling of randomly distributed values. Note that this is a virtual class; the intention is to define concrete subclasses corresponding to specific parameterized distributions.

Chunking dimensions

The array is conceptually partitioned into contiguous chunks of the same shape. The random values in each chunk are initialized with a different seed and stream via the PCG32 pseudo-random number generator (see the **dqrng** package for details). This design allows us to rapidly access any given subarray without having to do jump-aheads from the start of the stream.

The default chunking dimensions are set to the square root of the array dimensions - or 100, whichever is larger. This scheme provides decent though suboptimal performance along any dimension. If the access pattern is known beforehand, a better chunking scheme can often be chosen and passed to the `chunkdim` argument.

Note that changing the chunking dimensions will change the ordering of array values, even if the seeds are unchanged. This may be unexpected, given that chunking in real datasets will never change the data, only the performance of access operations. However, it is largely unavoidable in this context as the random number stream is rearranged within the array.

The `chunkdim(x)` method will return the chunk dimensions of a RandomArraySeed instance `x`. This will be used by the **DelayedArray** machinery to optimize block processing by extracting whole chunks where possible.

Implementing subclasses

To sample from a specific distribution, we can implement a concrete subclass of the RandomArraySeed. This is done by implementing methods for `sampleDistrFun` and `sampleDistrParam`.

In the code chunks below, `x` is an instance of a RandomArraySeed subclass:

- `sampleDistrFun(x)` returns a quantile function that accepts a vector of cumulative probabilities `p` and returns a numeric vector of quantiles. A typical example is `qnorm`, though similar functions from the **stats** package can also be used. The output vector should be the same length as `p`; any other distributional parameters should be recycled to the length of `p`.
- `sampleDistrParam(x)` returns a character vector specifying the names of the distributional parameters as slots of `x`. For example, for a subclass that samples from a normal distribution, this might be `"mean"` and `"sd"`. Each distributional parameter is expected to be numeric.

The `extract_array` method for the RandomArraySeed will automatically use both of the above methods to sample from the specified distribution. This is achieved by randomly sampling from a standard uniform distribution, treating the values as probabilities and converting them into quantiles.

Distributional parameters

Distributional parameters are passed to the relevant quantile function to obtain a random value from the desired distribution. Each parameter can be:

- A numeric scalar, which is used throughout the array.
- A numeric vector, which is recycled along the length of the array. This traverses the array along the first dimension, then the second, then the third, and so on; for matrices, this is equivalent to column-major ordering.
- A numeric array-like object of the same dimensions as `dim`, where each entry contains the parameter value for the corresponding entry of the output array. This can be another DelayedArray object.

Representing sparsity

For certain distributions, we may expect a large number of zeroes in the random output. We provide the *option* to treat the sampled values as being sparse, by setting `sparse=TRUE` in the constructors of the relevant subclasses. This is optional as most distributions will not yield sparse arrays for most of their possible parameter space.

When `sparse=TRUE`, the block processing machinery in **DelayedArray** will return a sparse array. This gives downstream applications the opportunity to use more efficient sparse algorithms when relevant. However, this option does not affect the sampling itself; the result is always the same as a dense array, just that the output is coerced into a SVT_SparseArray.

We can determine whether a RandomArraySeed `x` has a sparse interpretation with `is_sparse(x)`.

Author(s)

Aaron Lun

See Also

The RandomUnifArraySeed class, which implements sampling from a uniform distribution.

The RandomPoisArraySeed class, which implements sampling from a Poisson distribution.

RandomBetaArray-class *DelayedArray of random beta-distributed values*

Description

A DelayedArray subclass that performs on-the-fly sampling of beta-distributed values.

Usage

```
RandomBetaArraySeed(dim, shape1, shape2, ncp = 0, chunkdim = NULL)

## S4 method for signature 'RandomBetaArraySeed'
DelayedArray(seed)

RandomBetaArray(dim, shape1, shape2, ncp = 0, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
shape1, shape2, ncp	Numeric vector used as the argument of the same name in qbeta. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomBetaArraySeed object.

Value

All constructors return an instance of a RandomBetaArray object, containing random draws from a beta distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomBetaArraySeed(c(1e5, 1e5), shape1=1, shape2=10)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomBetaArraySeed(c(1e5, 1e5), shape1=runif(1e5), shape2=2)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
s1 <- rsparsematrix(1e5, 1e5, density=0.00001)
```

```
s1 <- abs(DelayedArray(s1)) + 1
X3 <- RandomBetaArraySeed(c(1e5, 1e5), shape1=s1, shape2=s1+1)
Y3 <- DelayedArray(X3)
Y3
```

RandomBinomArray-class

DelayedArray of random binomial values

Description

A DelayedArray subclass that performs on-the-fly sampling of binomial-distributed values.

Usage

```
RandomBinomArraySeed(dim, size, prob, chunkdim = NULL, sparse = FALSE)

## S4 method for signature 'RandomBinomArraySeed'
DelayedArray(seed)

RandomBinomArray(dim, size, prob, chunkdim = NULL, sparse = FALSE)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
size, prob	Numeric vector used as the argument of the same name in qbinom. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
sparse	Logical scalar indicating whether the sampled array should be treated as sparse.
seed	A RandomBinomArraySeed object.

Value

All constructors return an instance of a RandomBinomArray object, containing random draws from a binomial distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomBinomArraySeed(c(1e5, 1e5), size=10, prob=0.5)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomBinomArraySeed(c(1e5, 1e5), size=10,
    prob=runif(1e5, 0, 0.1), sparse=TRUE)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
size <- rsparsematrix(1e5, 1e5, density=0.00001)
size <- round(abs(DelayedArray(size)) * 10)
X3 <- RandomBinomArraySeed(c(1e5, 1e5), size=size, prob=0.5)
Y3 <- DelayedArray(X3)
Y3
```

RandomCauchyArray-class

DelayedArray of random Cauchy-distributed values

Description

A DelayedArray subclass that performs on-the-fly sampling of Cauchy-distributed values.

Usage

```
RandomCauchyArraySeed(dim, location = 0, scale = 1, chunkdim = NULL)

## S4 method for signature 'RandomCauchyArraySeed'
DelayedArray(seed)

RandomCauchyArray(dim, location = 0, scale = 1, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
location, scale	Numeric vector used as the argument of the same name in qcauchy. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomCauchyArraySeed object.

Value

All constructors return an instance of a RandomCauchyArray object, containing random draws from a Cauchy distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomCauchyArraySeed(c(1e5, 1e5))
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomCauchyArraySeed(c(1e5, 1e5), location=runif(1e5))
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
loc <- rsparsematrix(1e5, 1e5, density=0.00001)
X3 <- RandomCauchyArraySeed(c(1e5, 1e5), location=loc)
Y3 <- DelayedArray(X3)
Y3
```

RandomChisqArray-class

DelayedArray of random chi-squared-distributed values

Description

A DelayedArray subclass that performs on-the-fly sampling of chi-squared-distributed values.

Usage

```
RandomChisqArraySeed(dim, df, ncp = 0, chunkdim = NULL)

## S4 method for signature 'RandomChisqArraySeed'
DelayedArray(seed)

RandomChisqArray(dim, df, ncp = 0, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
df, ncp	Numeric vector used as the argument of the same name in qchisq. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomChisqArraySeed object.

Value

All constructors return an instance of a RandomChisqArray object, containing random draws from a chi-squared distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomChisqArraySeed(c(1e5, 1e5), df=5)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomChisqArraySeed(c(1e5, 1e5), df=runif(1e5)*20)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
df <- rsparsematrix(1e5, 1e5, density=0.00001)
df <- abs(DelayedArray(df) + 1) * 10
X3 <- RandomChisqArraySeed(c(1e5, 1e5), df=df)
Y3 <- DelayedArray(X3)
Y3
```

RandomExpArray-class *DelayedArray of random exponential values*

Description

A DelayedArray subclass that performs on-the-fly sampling of exponentially distributed values.

Usage

```
RandomExpArraySeed(dim, rate = 1, chunkdim = NULL)

## S4 method for signature 'RandomExpArraySeed'
DelayedArray(seed)

RandomExpArray(dim, rate = 1, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
rate	Numeric vector used as rate in qexp. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomExpArraySeed object.

Value

All constructors return an instance of a RandomExpArray object, containing random draws from a exponential distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomExpArraySeed(c(1e5, 1e5))
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomExpArraySeed(c(1e5, 1e5), rate=runif(1e5))
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
rate <- rsparsematrix(1e5, 1e5, density=0.00001)
rate <- abs(DelayedArray(rate)) + 1
X3 <- RandomExpArraySeed(c(1e5, 1e5), rate=rate)
Y3 <- DelayedArray(X3)
Y3
```

RandomFArray-class *DelayedArray of random F-distributed values*

Description

A DelayedArray subclass that performs on-the-fly sampling of F-distributed values.

Usage

```
RandomFArraySeed(dim, df1, df2, ncp, chunkdim = NULL)

## S4 method for signature 'RandomFArraySeed'
DelayedArray(seed)

RandomFArray(dim, df1, df2, ncp, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
df1, df2, ncp	Numeric vector used as the argument of the same name in qf. Alternatively, a numeric array-like object with the same dimensions as dim. If ncp is missing, a central F distribution is assumed.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomFArraySeed object.

Value

All constructors return an instance of a RandomFArray object, containing random draws from a exponential distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomFArraySeed(c(1e5, 1e5), df1=1, df2=10)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomFArraySeed(c(1e5, 1e5), df1=runif(1e5)*10, df2=10)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
ncp <- rsparsematrix(1e5, 1e5, density=0.00001)
ncp <- abs(DelayedArray(ncp)) + 1
X3 <- RandomFArraySeed(c(1e5, 1e5), df1=1, df2=10, ncp=ncp)
Y3 <- DelayedArray(X3)
Y3
```

RandomGammaArray-class *DelayedArray of random gamma-distributed values*

Description

A DelayedArray subclass that performs on-the-fly sampling of gamma-distributed values.

Usage

```
RandomGammaArraySeed(dim, shape, rate = 1, scale = 1/rate, chunkdim = NULL)

## S4 method for signature 'RandomGammaArraySeed'
DelayedArray(seed)

RandomGammaArray(dim, shape, rate = 1, scale = 1/rate, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
shape, rate, scale	Numeric vector used as the argument of the same name in qgamma. Alternatively, a numeric array-like object with the same dimensions as dim. If scale is explicitly supplied, rate is ignored.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomGammaArraySeed object.

Value

All constructors return an instance of a RandomGammaArray object, containing random draws from a gamma distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomGammaArraySeed(c(1e5, 1e5), shape=1, rate=10)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomGammaArraySeed(c(1e5, 1e5), shape=runif(1e5), rate=2)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
```

```
library(Matrix)
s1 <- rsparsematrix(1e5, 1e5, density=0.00001)
s1 <- abs(DelayedArray(s1)) + 1
X3 <- RandomGammaArraySeed(c(1e5, 1e5), shape=s1, rate=s1+1)
Y3 <- DelayedArray(X3)
Y3
```

RandomGeomArray-class *DelayedArray of random geometric-distributed values*

Description

A DelayedArray subclass that performs on-the-fly sampling of geometric-distributed values.

Usage

```
RandomGeomArraySeed(dim, prob, chunkdim = NULL, sparse = FALSE)

## S4 method for signature 'RandomGeomArraySeed'
DelayedArray(seed)

RandomGeomArray(dim, prob, chunkdim = NULL, sparse = FALSE)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
prob	Numeric vector used as prob in qgeom. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
sparse	Logical scalar indicating whether the sampled array should be treated as sparse.
seed	A RandomGeomArraySeed object.

Value

All constructors return an instance of a RandomGeomArray object, containing random draws from a geometric distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomGeomArraySeed(c(1e5, 1e5), prob=0.5)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomGeomArraySeed(c(1e5, 1e5), prob=runif(1e5, 0, 0.1), sparse=TRUE)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
prob <- RandomUnifArray(c(1e5, 1e5))
X3 <- RandomGeomArraySeed(c(1e5, 1e5), prob=prob)
Y3 <- DelayedArray(X3)
Y3
```

RandomHyperArray-class

DelayedArray of random hypergeometric-distributed values

Description

A DelayedArray subclass that performs on-the-fly sampling of hypergeometric-distributed values.

Usage

```
RandomHyperArraySeed(dim, m, n, k, chunkdim = NULL, sparse = FALSE)

## S4 method for signature 'RandomHyperArraySeed'
DelayedArray(seed)

RandomHyperArray(dim, m, n, k, chunkdim = NULL, sparse = FALSE)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
m, n, k	Numeric vector used as the argument of the same name in qhyper. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
sparse	Logical scalar indicating whether the sampled array should be treated as sparse.
seed	A RandomHyperArraySeed object.

Value

All constructors return an instance of a RandomHyperArray object, containing random draws from a hypergeometric distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomHyperArraySeed(c(1e5, 1e5), m=10, n=20, k=15)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomHyperArraySeed(c(1e5, 1e5), m=round(runif(1e5, 10, 20)),
    n=20, k=15, sparse=TRUE)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
m <- round(RandomUnifArray(c(1e5, 1e5), 10, 20))
X3 <- RandomHyperArraySeed(c(1e5, 1e5), m=m, n=50, k=20)
Y3 <- DelayedArray(X3)
Y3
```

RandomLnormArray-class

DelayedArray of random log-normal values

Description

A DelayedArray subclass that performs on-the-fly sampling of log-normally distributed values.

Usage

```
RandomLnormArraySeed(dim, meanlog = 0, sdlog = 1, chunkdim = NULL)

## S4 method for signature 'RandomLnormArraySeed'
DelayedArray(seed)

RandomLnormArray(dim, meanlog = 0, sdlog = 1, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
meanlog, sdlog	Numeric vector used as the argument of the same name in qlnorm. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomLnormArraySeed object.

Value

All constructors return an instance of a RandomLnormArray object, containing random draws from a log-normal distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomLnormArraySeed(c(1e5, 1e5))
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomLnormArraySeed(c(1e5, 1e5), meanlog=runif(1e5), sdlog=runif(1e5))
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
meanlog <- rsparsematrix(1e5, 1e5, density=0.00001)
X3 <- RandomLnormArraySeed(c(1e5, 1e5), meanlog=meanlog)
Y3 <- DelayedArray(X3)
Y3
```

RandomLogisArray-class

DelayedArray of random log-normal values

Description

A DelayedArray subclass that performs on-the-fly sampling of log-normally distributed values.

Usage

```
RandomLogisArraySeed(dim, location = 0, scale = 1, chunkdim = NULL)

## S4 method for signature 'RandomLogisArraySeed'
DelayedArray(seed)

RandomLogisArray(dim, location = 0, scale = 1, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
location, scale	Numeric vector used as the argument of the same name in qlogis. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomLogisArraySeed object.

Value

All constructors return an instance of a RandomLogisArray object, containing random draws from a log-normal distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomLogisArraySeed(c(1e5, 1e5))
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomLogisArraySeed(c(1e5, 1e5), location=runif(1e5), scale=runif(1e5))
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
location <- rsparsematrix(1e5, 1e5, density=0.00001)
X3 <- RandomLogisArraySeed(c(1e5, 1e5), location=location)
Y3 <- DelayedArray(X3)
Y3
```

RandomNbinomArray-class

DelayedArray of random negative binomial values

Description

A DelayedArray subclass that performs on-the-fly sampling of negative binomial-distributed values.

Usage

```
RandomNbinomArraySeed(
  dim,
  prob = prob,
  size = size,
  mu = mu,
  chunkdim = NULL,
  sparse = FALSE
)

## S4 method for signature 'RandomNbinomArraySeed'
DelayedArray(seed)

RandomNbinomArray(dim, prob, size, mu, chunkdim = NULL, sparse = FALSE)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
prob, size, mu	Numeric vector used as the argument of the same name in qnbinom. Alternatively, a numeric array-like object with the same dimensions as dim. Exactly one of prob or mu should be supplied.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
sparse	Logical scalar indicating whether the sampled array should be treated as sparse.
seed	A RandomNbinomArraySeed object.

Value

All constructors return an instance of a RandomNbinomArray object, containing random draws from a negative binomial distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomNbinomArraySeed(c(1e5, 1e5), size=10, mu=20)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomNbinomArraySeed(c(1e5, 1e5), size=10, mu=runif(1e5), sparse=TRUE)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
lambda <- rsparsematrix(1e5, 1e5, density=0.00001)
lambda <- abs(DelayedArray(lambda)) + 0.1
X3 <- RandomNbinomArraySeed(c(1e5, 1e5), size=1, mu=lambda)
Y3 <- DelayedArray(X3)
Y3
```

RandomNormArray-class *DelayedArray of random normal values*

Description

A DelayedArray subclass that performs on-the-fly sampling of normally distributed values.

Usage

```
RandomNormArraySeed(dim, mean = 0, sd = 1, chunkdim = NULL)

## S4 method for signature 'RandomNormArraySeed'
DelayedArray(seed)

RandomNormArray(dim, mean = 0, sd = 1, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
mean, sd	Numeric vector used as mean and sd, respectively, in qnorm. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomNormArraySeed object.

Value

All constructors return an instance of a RandomNormArray object, containing random draws from a normal distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomNormArraySeed(c(1e5, 1e5))
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomNormArraySeed(c(1e5, 1e5), mean=runif(1e5), sd=runif(1e5))
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
mean <- rsparsematrix(1e5, 1e5, density=0.00001)
X3 <- RandomNormArraySeed(c(1e5, 1e5), mean=mean)
Y3 <- DelayedArray(X3)
Y3
```

RandomPoisArray-class *DelayedArray of random Poisson values*

Description

A DelayedArray subclass that performs on-the-fly sampling of Poisson-distributed values.

Usage

```
RandomPoisArraySeed(dim, lambda, chunkdim = NULL, sparse = FALSE)

## S4 method for signature 'RandomPoisArraySeed'
DelayedArray(seed)

RandomPoisArray(dim, lambda, chunkdim = NULL, sparse = FALSE)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
lambda	Numeric vector used as lambda in qpois. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
sparse	Logical scalar indicating whether the sampled array should be treated as sparse.
seed	A RandomPoisArraySeed object.

Value

All constructors return an instance of a RandomPoisArray object, containing random draws from a Poisson distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomPoisArraySeed(c(1e5, 1e5), lambda=2)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomPoisArraySeed(c(1e5, 1e5), lambda=runif(1e5), sparse=TRUE)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
```

```
lambda <- rsparsematrix(1e5, 1e5, density=0.00001)
lambda <- abs(DelayedArray(lambda)) + 0.1
X3 <- RandomPoisArraySeed(c(1e5, 1e5), lambda=lambda)
Y3 <- DelayedArray(X3)
Y3
```

RandomTArray-class *DelayedArray of random F-distributed values*

Description

A DelayedArray subclass that performs on-the-fly sampling of F-distributed values.

Usage

```
RandomTArraySeed(dim, df, ncp, chunkdim = NULL)

## S4 method for signature 'RandomTArraySeed'
DelayedArray(seed)

RandomTArray(dim, df, ncp, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
df, ncp	Numeric vector used as the argument of the same name in qf. Alternatively, a numeric array-like object with the same dimensions as dim. If ncp is missing, a central T distribution is assumed.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomTArraySeed object.

Value

All constructors return an instance of a RandomTArray object, containing random draws from a exponential distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomTArraySeed(c(1e5, 1e5), df=10)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomTArraySeed(c(1e5, 1e5), df=sample(20, 1e5, replace=TRUE))
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
ncp <- rsparsematrix(1e5, 1e5, density=0.00001)
ncp <- abs(DelayedArray(ncp)) + 1
X3 <- RandomTArraySeed(c(1e5, 1e5), df=10, ncp=ncp)
Y3 <- DelayedArray(X3)
Y3
```

RandomUnifArray-class *DelayedArray of random uniform values*

Description

A DelayedArray subclass that performs on-the-fly sampling of uniformly distributed values.

Usage

```
RandomUnifArraySeed(dim, min = 0, max = 1, chunkdim = NULL)

## S4 method for signature 'RandomUnifArraySeed'
DelayedArray(seed)

RandomUnifArray(dim, min = 0, max = 1, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
min, max	Numeric vector used as min and max, respectively, in qunif. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomUnifArraySeed object.

Value

All constructors return an instance of a RandomUnifArray object, containing random draws from a uniform distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomUnifArraySeed(c(1e5, 1e5))
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomUnifArraySeed(c(1e5, 1e5), min=1:1e5, max=1:1e5*2)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
min <- rsparsematrix(1e5, 1e5, density=0.00001)
X3 <- RandomUnifArraySeed(c(1e5, 1e5), min=min, max=DelayedArray(min)+1)
Y3 <- DelayedArray(X3)
Y3
```

RandomWeibullArray-class

DelayedArray of random Weibull-distributed values

Description

A DelayedArray subclass that performs on-the-fly sampling of Weibull-distributed values.

Usage

```
RandomWeibullArraySeed(dim, shape, scale = 1, chunkdim = NULL)

## S4 method for signature 'RandomWeibullArraySeed'
DelayedArray(seed)

RandomWeibullArray(dim, shape, scale = 1, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
shape, scale	Numeric vector used as the argument of the same name in qweibull. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
seed	A RandomWeibullArraySeed object.

Value

All constructors return an instance of a RandomWeibullArray object, containing random draws from a Weibull distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomWeibullArraySeed(c(1e5, 1e5), shape=10)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomWeibullArraySeed(c(1e5, 1e5), shape=round(runif(1e5, 10, 20)))
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
shape <- round(RandomUnifArray(c(1e5, 1e5), 10, 20))
X3 <- RandomWeibullArraySeed(c(1e5, 1e5), shape=shape)
Y3 <- DelayedArray(X3)
Y3
```

RandomWilcoxArray-class

DelayedArray of random Wilcoxon-distributed values

Description

A DelayedArray subclass that performs on-the-fly sampling of Wilcoxon-distributed values.

Usage

```
RandomWilcoxArraySeed(dim, m, n, chunkdim = NULL, sparse = FALSE)

## S4 method for signature 'RandomWilcoxArraySeed'
DelayedArray(seed)

RandomWilcoxArray(dim, m, n, chunkdim = NULL)
```

Arguments

dim	Integer vector of positive length, specifying the dimensions of the array.
m, n	Numeric vector used as the argument of the same name in qwilcox. Alternatively, a numeric array-like object with the same dimensions as dim.
chunkdim	Integer vector of length equal to dim, containing the dimensions of each chunk.
sparse	Logical scalar indicating whether the sampled array should be treated as sparse.
seed	A RandomWilcoxArraySeed object.

Value

All constructors return an instance of a RandomWilcoxArray object, containing random draws from a Wilcox distribution with the specified parameters.

Author(s)

Aaron Lun

See Also

The RandomArraySeed class, for details on chunking and the distributional parameters.

Examples

```
X <- RandomWilcoxArraySeed(c(1e5, 1e5), m=10, n=20)
Y <- DelayedArray(X)
Y

# Fiddling with the distribution parameters:
X2 <- RandomWilcoxArraySeed(c(1e5, 1e5), m=round(runif(1e5, 10, 20)), n=20)
Y2 <- DelayedArray(X2)
Y2

# Using another array as input:
library(Matrix)
m <- round(RandomUnifArray(c(1e5, 1e5), 10, 20))
X3 <- RandomWilcoxArraySeed(c(1e5, 1e5), m=m, n=50)
Y3 <- DelayedArray(X3)
Y3
```

Index

chunkdim, *2*
chunkdim,RandomArraySeed-method *(RandomArraySeed-class)*, 2

DelayedArray, *2–9*, *11–17*, *19–23*
DelayedArray,RandomBetaArraySeed-method *(RandomBetaArray-class)*, 4
DelayedArray,RandomBinomArraySeed-method *(RandomBinomArray-class)*, 5
DelayedArray,RandomCauchyArraySeed-method *(RandomCauchyArray-class)*, 6
DelayedArray,RandomChisqArraySeed-method *(RandomChisqArray-class)*, 7
DelayedArray,RandomExpArraySeed-method *(RandomExpArray-class)*, 8
DelayedArray,RandomFArraySeed-method *(RandomFArray-class)*, 9
DelayedArray,RandomGammaArraySeed-method *(RandomGammaArray-class)*, 11
DelayedArray,RandomGeomArraySeed-method *(RandomGeomArray-class)*, 12
DelayedArray,RandomHyperArraySeed-method *(RandomHyperArray-class)*, 13
DelayedArray,RandomLnormArraySeed-method *(RandomLnormArray-class)*, 14
DelayedArray,RandomLogisArraySeed-method *(RandomLogisArray-class)*, 15
DelayedArray,RandomNbinomArraySeed-method *(RandomNbinomArray-class)*, 16
DelayedArray,RandomNormArraySeed-method *(RandomNormArray-class)*, 17
DelayedArray,RandomPoisArraySeed-method *(RandomPoisArray-class)*, 19
DelayedArray,RandomTArraySeed-method *(RandomTArray-class)*, 20
DelayedArray,RandomUnifArraySeed-method *(RandomUnifArray-class)*, 21
DelayedArray,RandomWeibullArraySeed-method *(RandomWeibullArray-class)*, 22
DelayedArray,RandomWilcoxArraySeed-method *(RandomWilcoxArray-class)*, 23

extract_array,RandomArraySeed-method *(RandomArraySeed-class)*, 2
extract_array,RandomBetaArraySeed-method *(RandomBetaArray-class)*, 4
extract_array,RandomChisqArraySeed-method *(RandomChisqArray-class)*, 7
extract_array,RandomFArraySeed-method *(RandomFArray-class)*, 9
extract_array,RandomNbinomArraySeed-method *(RandomNbinomArray-class)*, 16
extract_array,RandomTArraySeed-method *(RandomTArray-class)*, 20
extract_sparse_array,RandomArraySeed-method *(RandomArraySeed-class)*, 2

initialize,RandomArraySeed-method *(RandomArraySeed-class)*, 2
is_sparse,RandomArraySeed-method *(RandomArraySeed-class)*, 2

matrixClass,RandomBetaArray-method *(RandomBetaArray-class)*, 4
matrixClass,RandomBinomArray-method *(RandomBinomArray-class)*, 5
matrixClass,RandomCauchyArray-method *(RandomCauchyArray-class)*, 6
matrixClass,RandomChisqArray-method *(RandomChisqArray-class)*, 7
matrixClass,RandomExpArray-method *(RandomExpArray-class)*, 8
matrixClass,RandomFArray-method *(RandomFArray-class)*, 9
matrixClass,RandomGammaArray-method *(RandomGammaArray-class)*, 11
matrixClass,RandomGeomArray-method *(RandomGeomArray-class)*, 12
matrixClass,RandomHyperArray-method *(RandomHyperArray-class)*, 13
matrixClass,RandomLnormArray-method *(RandomLnormArray-class)*, 14
matrixClass,RandomLogisArray-method *(RandomLogisArray-class)*, 15
matrixClass,RandomNbinomArray-method *(RandomNbinomArray-class)*, 16
matrixClass,RandomNormArray-method *(RandomNormArray-class)*, 17

matrixClass,RandomPoisArray-method
(RandomPoisArray-class), 19
matrixClass,RandomTArray-method
(RandomTArray-class), 20
matrixClass,RandomUnifArray-method
(RandomUnifArray-class), 21
matrixClass,RandomWeibullArray-method
(RandomWeibullArray-class), 22
matrixClass,RandomWilcoxArray-method
(RandomWilcoxArray-class), 23

qbeta, *4*
qbinom, *5*
qcauchy, *6*
qchisq, *7*
qexp, *9*
qf, *10*, *20*
qgamma, *11*
qgeom, *12*
qhyper, *13*
qlnorm, *14*
qlogis, *15*
qnbinom, *17*
qnorm, *3*, *18*
qpois, *19*
qunif, *21*
qweibull, *22*
qwilcox, *23*

RandomArraySeed, *4, 5*, *7–12*, *14–20*, *22–24*
RandomArraySeed
(RandomArraySeed-class), 2
RandomArraySeed-class, 2
RandomBetaArray
(RandomBetaArray-class), 4
RandomBetaArray-class, 4
RandomBetaArraySeed
(RandomBetaArray-class), 4
RandomBetaArraySeed-class
(RandomBetaArray-class), 4
RandomBetaMatrix-class
(RandomBetaArray-class), 4
RandomBinomArray
(RandomBinomArray-class), 5
RandomBinomArray-class, 5
RandomBinomArraySeed
(RandomBinomArray-class), 5
RandomBinomArraySeed-class
(RandomBinomArray-class), 5
RandomBinomMatrix-class
(RandomBinomArray-class), 5
RandomCauchyArray
(RandomCauchyArray-class), 6
RandomCauchyArray-class, 6
RandomCauchyArraySeed
(RandomCauchyArray-class), 6
RandomCauchyArraySeed-class
(RandomCauchyArray-class), 6
RandomCauchyMatrix-class
(RandomCauchyArray-class), 6
RandomChisqArray
(RandomChisqArray-class), 7
RandomChisqArray-class, 7
RandomChisqArraySeed
(RandomChisqArray-class), 7
RandomChisqArraySeed-class
(RandomChisqArray-class), 7
RandomChisqMatrix-class
(RandomChisqArray-class), 7
RandomExpArray (RandomExpArray-class), 8
RandomExpArray-class, 8
RandomExpArraySeed
(RandomExpArray-class), 8
RandomExpArraySeed-class
(RandomExpArray-class), 8
RandomExpMatrix-class
(RandomExpArray-class), 8
RandomFArray (RandomFArray-class), 9
RandomFArray-class, 9
RandomFArraySeed (RandomFArray-class), 9
RandomFArraySeed-class
(RandomFArray-class), 9
RandomFMatrix-class
(RandomFArray-class), 9
RandomGammaArray
(RandomGammaArray-class), 11
RandomGammaArray-class, 11
RandomGammaArraySeed
(RandomGammaArray-class), 11
RandomGammaArraySeed-class
(RandomGammaArray-class), 11
RandomGammaMatrix-class
(RandomGammaArray-class), 11
RandomGeomArray
(RandomGeomArray-class), 12
RandomGeomArray-class, 12
RandomGeomArraySeed
(RandomGeomArray-class), 12
RandomGeomArraySeed-class
(RandomGeomArray-class), 12
RandomGeomMatrix-class
(RandomGeomArray-class), 12
RandomHyperArray
(RandomHyperArray-class), 13
RandomHyperArray-class, 13

RandomHyperArraySeed
(RandomHyperArray-class), 13
RandomHyperArraySeed-class
(RandomHyperArray-class), 13
RandomHyperMatrix-class
(RandomHyperArray-class), 13
RandomLnormArray
(RandomLnormArray-class), 14
RandomLnormArray-class, 14
RandomLnormArraySeed
(RandomLnormArray-class), 14
RandomLnormArraySeed-class
(RandomLnormArray-class), 14
RandomLnormMatrix-class
(RandomLnormArray-class), 14
RandomLogisArray
(RandomLogisArray-class), 15
RandomLogisArray-class, 15
RandomLogisArraySeed
(RandomLogisArray-class), 15
RandomLogisArraySeed-class
(RandomLogisArray-class), 15
RandomLogisMatrix-class
(RandomLogisArray-class), 15
RandomNbinomArray
(RandomNbinomArray-class), 16
RandomNbinomArray-class, 16
RandomNbinomArraySeed
(RandomNbinomArray-class), 16
RandomNbinomArraySeed-class
(RandomNbinomArray-class), 16
RandomNbinomMatrix-class
(RandomNbinomArray-class), 16
RandomNormArray
(RandomNormArray-class), 17
RandomNormArray-class, 17
RandomNormArraySeed
(RandomNormArray-class), 17
RandomNormArraySeed-class
(RandomNormArray-class), 17
RandomNormMatrix-class
(RandomNormArray-class), 17
RandomPoisArray
(RandomPoisArray-class), 19
RandomPoisArray-class, 19
RandomPoisArraySeed, *3*
RandomPoisArraySeed
(RandomPoisArray-class), 19
RandomPoisArraySeed-class
(RandomPoisArray-class), 19
RandomPoisMatrix-class
(RandomPoisArray-class), 19
RandomTArray (RandomTArray-class), 20
RandomTArray-class, 20
RandomTArraySeed (RandomTArray-class),
20
RandomTArraySeed-class
(RandomTArray-class), 20
RandomTMatrix-class
(RandomTArray-class), 20
RandomUnifArray
(RandomUnifArray-class), 21
RandomUnifArray-class, 21
RandomUnifArraySeed, *3*
RandomUnifArraySeed
(RandomUnifArray-class), 21
RandomUnifArraySeed-class
(RandomUnifArray-class), 21
RandomUnifMatrix-class
(RandomUnifArray-class), 21
RandomWeibullArray
(RandomWeibullArray-class), 22
RandomWeibullArray-class, 22
RandomWeibullArraySeed
(RandomWeibullArray-class), 22
RandomWeibullArraySeed-class
(RandomWeibullArray-class), 22
RandomWeibullMatrix-class
(RandomWeibullArray-class), 22
RandomWilcoxArray
(RandomWilcoxArray-class), 23
RandomWilcoxArray-class, 23
RandomWilcoxArraySeed
(RandomWilcoxArray-class), 23
RandomWilcoxArraySeed-class
(RandomWilcoxArray-class), 23
RandomWilcoxMatrix-class
(RandomWilcoxArray-class), 23

sampleDistrFun (RandomArraySeed-class),
2
sampleDistrFun,RandomBetaArraySeed-method
(RandomBetaArray-class), 4
sampleDistrFun,RandomBinomArraySeed-method
(RandomBinomArray-class), 5
sampleDistrFun,RandomCauchyArraySeed-method
(RandomCauchyArray-class), 6
sampleDistrFun,RandomChisqArraySeed-method
(RandomChisqArray-class), 7
sampleDistrFun,RandomExpArraySeed-method
(RandomExpArray-class), 8
sampleDistrFun,RandomFArraySeed-method
(RandomFArray-class), 9
sampleDistrFun,RandomGammaArraySeed-method
(RandomGammaArray-class), 11

sampleDistrFun,RandomGeomArraySeed-method
(RandomGeomArray-class), 12
sampleDistrFun,RandomHyperArraySeed-method
(RandomHyperArray-class), 13
sampleDistrFun,RandomLnormArraySeed-method
(RandomLnormArray-class), 14
sampleDistrFun,RandomLogisArraySeed-method
(RandomLogisArray-class), 15
sampleDistrFun,RandomNbinomArraySeed-method
(RandomNbinomArray-class), 16
sampleDistrFun,RandomNormArraySeed-method
(RandomNormArray-class), 17
sampleDistrFun,RandomPoisArraySeed-method
(RandomPoisArray-class), 19
sampleDistrFun,RandomTArraySeed-method
(RandomTArray-class), 20
sampleDistrFun,RandomUnifArraySeed-method
(RandomUnifArray-class), 21
sampleDistrFun,RandomWeibullArraySeed-method
(RandomWeibullArray-class), 22
sampleDistrFun,RandomWilcoxArraySeed-method
(RandomWilcoxArray-class), 23
sampleDistrParam
(RandomArraySeed-class), 2
sampleDistrParam,RandomBetaArraySeed-method
(RandomBetaArray-class), 4
sampleDistrParam,RandomBinomArraySeed-method
(RandomBinomArray-class), 5
sampleDistrParam,RandomCauchyArraySeed-method
(RandomCauchyArray-class), 6
sampleDistrParam,RandomChisqArraySeed-method
(RandomChisqArray-class), 7
sampleDistrParam,RandomExpArraySeed-method
(RandomExpArray-class), 8
sampleDistrParam,RandomFArraySeed-method
(RandomFArray-class), 9
sampleDistrParam,RandomGammaArraySeed-method
(RandomGammaArray-class), 11
sampleDistrParam,RandomGeomArraySeed-method
(RandomGeomArray-class), 12
sampleDistrParam,RandomHyperArraySeed-method
(RandomHyperArray-class), 13
sampleDistrParam,RandomLnormArraySeed-method
(RandomLnormArray-class), 14
sampleDistrParam,RandomLogisArraySeed-method
(RandomLogisArray-class), 15
sampleDistrParam,RandomNbinomArraySeed-method
(RandomNbinomArray-class), 16
sampleDistrParam,RandomNormArraySeed-method
(RandomNormArray-class), 17
sampleDistrParam,RandomPoisArraySeed-method
(RandomPoisArray-class), 19
sampleDistrParam,RandomTArraySeed-method
(RandomTArray-class), 20
sampleDistrParam,RandomUnifArraySeed-method
(RandomUnifArray-class), 21
sampleDistrParam,RandomWeibullArraySeed-method
(RandomWeibullArray-class), 22
sampleDistrParam,RandomWilcoxArraySeed-method
(RandomWilcoxArray-class), 23
show,RandomArraySeed-method
(RandomArraySeed-class), 2
SVT_SparseArray, *3*