

# Package ‘bdlp’

July 22, 2025

**Version** 0.9-2

**Date** 2021-01-03

**Title** Transparent and Reproducible Artificial Data Generation

**Depends** R (>= 3.0.0), graphics

**Imports** GenOrd, MultiOrd, stringdist, rgl, RSQLite, MASS, DBI,  
methods, grDevices, stats, utils

**Description** The main function generateDataset() processes a user-supplied .R file that contains metadata parameters in order to generate actual data. The metadata parameters have to be structured in the form of metadata objects, the format of which is outlined in the package vignette. This approach allows to generate artificial data in a transparent and reproducible manner.

**License** GPL-2

**LazyLoad** yes

**Author** Rainer Dangl [aut, cre]

**Maintainer** Rainer Dangl <rainer.dangl@bildung.gv.at>

**RoxygenNote** 7.1.1

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2021-01-10 15:10:05 UTC

## Contents

addCluster . . . . .	2
checkSetup . . . . .	3
createFileskeleton . . . . .	3
deleteCluster . . . . .	4
generateData . . . . .	5
generateData,metadata.binary-method . . . . .	5
generateData,metadata.functional-method . . . . .	6

generateData,metadata.metric-method . . . . .	6
generateData,metadata.ordinal-method . . . . .	7
generateData,metadata.randomstring-method . . . . .	7
generateDatabase . . . . .	8
getRandomstrings . . . . .	9
initializeObject . . . . .	9
metadata.binary-class . . . . .	10
metadata.functional-class . . . . .	10
metadata.general-class . . . . .	11
metadata.metric-class . . . . .	11
metadata.ordinal-class . . . . .	11
metadata.randomstring-class . . . . .	11
plot3dMetadata . . . . .	12
plot3dMetadata,metadata.metric-method . . . . .	12
plotMetadata . . . . .	13
plotMetadata,metadata.binary-method . . . . .	13
plotMetadata,metadata.functional-method . . . . .	14
plotMetadata,metadata.metric-method . . . . .	14
plotMetadata,metadata.ordinal-method . . . . .	15
sampleGrid . . . . .	15
saveSetup . . . . .	16
summarizeSetup . . . . .	17
<b>Index</b>	<b>18</b>

---

addCluster	<i>Add an empty cluster to a metadata object</i>
------------	--

---

### Description

Add an empty cluster to a metadata object

### Usage

```
addCluster(m)
```

### Arguments

m	A metadata object
---	-------------------

### Value

A metadata object with an empty additional cluster

**Examples**

```
require(MASS)
m <- new("metadata.metric",
        clusters = list(c1 = list(n = 25, mu = c(4,5), Sigma=diag(1,2)),
                        c2 = list(n = 25, mu = c(-1,-2), Sigma=diag(1,2))),
        genfunc = mvrnorm)
m2 <- addCluster(m)
```

---

checkSetup	<i>Performs various consistency checks on a setup file</i>
------------	--

---

**Description**

Performs various consistency checks on a setup file

**Usage**

```
checkSetup(file)
```

**Arguments**

file	A .R file with a new simulation setup
------	---------------------------------------

---

createFileskeleton	<i>Create a new setup file template</i>
--------------------	---

---

**Description**

Create a new setup file template

**Usage**

```
createFileskeleton(
  newname,
  mail,
  inst,
  author,
  type = c("metric", "functional", "ordinal", "binary", "randomstring", "wordnet"),
  infotable = NULL,
  ref = "Unpublished",
  codefile = F
)
```

**Arguments**

newname	The name of the new setup (and subsequently the file name)
mail	The contact e-mail address of the author
inst	The institution of the author
author	The full name of the author
type	The data type of this setup
infotable	The setup summary table
ref	The reference to the publication where the setup was used
codefile	If functions that are needed for the data generation of the setup are stored in some other .R file, the path can be supplied

---

deleteCluster	<i>Delete a cluster from a metadata object</i>
---------------	--

---

**Description**

Delete a cluster from a metadata object

**Usage**

```
deleteCluster(m, clnumber)
```

**Arguments**

m	A metadata object
clnumber	The cluster to delete

**Value**

A metadata object

**Examples**

```
require(MASS)
m <- new("metadata.metric",
        clusters = list(c1 = list(n = 25, mu = c(4,5), Sigma=diag(1,2)),
                       c2 = list(n = 25, mu = c(-1,-2), Sigma=diag(1,2))),
        genfunc = mvrnorm)
m2 <- deleteCluster(m, 2)
```

---

generateData	<i>Generate a dataset from a metadata object</i>
--------------	--

---

**Description**

Generate a dataset from a metadata object

**Usage**

```
generateData(m)
```

**Arguments**

m                    A metadata object

**Value**

A dataset as specified by the metadata object

**Examples**

```
require(MASS)
m <- new("metadata.metric",
        clusters = list(c1 = list(n = 25, mu = c(4,5), Sigma=diag(1,2)),
                       c2 = list(n = 25, mu = c(-1,-2), Sigma=diag(1,2))),
        genfunc = mvrnorm)
generateData(m)
```

---

generateData,metadata.binary-method	<i>Generate a dataset from a metadata object</i>
-------------------------------------	--

---

**Description**

Generate a dataset from a metadata object

**Usage**

```
## S4 method for signature 'metadata.binary'
generateData(m)
```

**Arguments**

m                    A metadata object

**Value**

A dataset as specified by the metadata object

---

`generateData,metadata.functional-method`

*Generate a dataset from a metadata object*

---

**Description**

Generate a dataset from a metadata object

**Usage**

```
## S4 method for signature 'metadata.functional'  
generateData(m)
```

**Arguments**

`m`                    A metadata object

**Value**

A dataset as specified by the metadata object

---

`generateData,metadata.metric-method`

*Generate a dataset from a metadata object*

---

**Description**

Generate a dataset from a metadata object

**Usage**

```
## S4 method for signature 'metadata.metric'  
generateData(m)
```

**Arguments**

`m`                    A metadata object

**Value**

A dataset as specified by the metadata object

---

`generateData,metadata.ordinal-method`

*Generate a dataset from a metadata object*

---

**Description**

Generate a dataset from a metadata object

**Usage**

```
## S4 method for signature 'metadata.ordinal'  
generateData(m)
```

**Arguments**

m                    A metadata object

**Value**

A dataset as specified by the metadata object

---

`generateData,metadata.randomstring-method`

*Generate a dataset from a metadata object*

---

**Description**

Generate a dataset from a metadata object

**Usage**

```
## S4 method for signature 'metadata.randomstring'  
generateData(m)
```

**Arguments**

m                    A metadata object

**Value**

A dataset as specified by the metadata object

---

generateDatabase	<i>Generates a number of datasets from one metadata scenario</i>
------------------	--

---

### Description

Generates a number of datasets from one metadata scenario

### Usage

```
generateDatabase(
  name = NULL,
  setnr = NULL,
  draws = 1,
  seedinfo = list(100, paste(R.version$major, R.version$minor, sep = "."), RNGkind()),
  metaseedinfo = list(100, paste(R.version$major, R.version$minor, sep = "."),
    RNGkind()),
  file = NULL,
  seedincrement = 1
)
```

### Arguments

name	The path to the setup file
setnr	The metadata scenario, as taken from the info table
draws	The number of datasets that are drawn from the metadata scenario
seedinfo	The random number generator seed parameters
metaseedinfo	If necessary, a separate set of random number generator parameters for the metadata (e.g. cluster centers)
file	A custom file name for the output database. Defaults to the pattern setup-name_setnr_seed.db
seedincrement	The random number seed will by default increase by 1 for each draw from the base seed given in seedinfo unless specified otherwise here

### Value

An SQLite database that contains the desired number of data sets drawn from a certain metadata scenario

### Examples

```
## Not run:
source(system.file("dangl2014.R", package="bdlp"))
generateDatabase(name="dangl2014.R", setnr=1, draws=10)
unlink("dangl2014_set_1_seed_100.db")

## End(Not run)
```



---

getRandomstrings	<i>Generates random strings</i>
------------------	---------------------------------

---

**Description**

Generates random strings

**Usage**

```
getRandomstrings(  
  center = NULL,  
  maxdist = NULL,  
  length = nchar(center),  
  n = 1,  
  method = "lv"  
)
```

**Arguments**

center	Reference string, i.e. the cluster center
maxdist	The maximum allowed string distance
length	The length of the string
n	Number of strings to be generated
method	The string distance method used to calculate the string, defaults to Levensthein distance

**Value**

A character string

**Examples**

```
getRandomstrings(center="hello", maxdist = 2, n = 5)
```

---

initializeObject	<i>Initialize a new metadata object</i>
------------------	---

---

**Description**

Initialize a new metadata object

**Usage**

```
initializeObject(
  type,
  k,
  genfunc,
  seed = list(100, paste(R.version$major, R.version$minor, sep = "."), RNGkind())
)
```

**Arguments**

type	The data type for the new object
k	Number of clusters
genfunc	The distribution function for data generation
seed	The random number seed parameters for the data generation

**Value**

A metadata object

**Examples**

```
require(MASS)
initializeObject(type = "metric", k = 3, genfunc = mvrnorm)
```

---

metadata.binary-class *A class that represents a metadata object for binary data*

---

**Description**

A class that represents a metadata object for binary data

---

metadata.functional-class  
*A class that represents a metadata object for functional data*

---

**Description**

A class that represents a metadata object for functional data

---

metadata.general-class

*A class to represent a metadata object*

---

**Description**

A class to represent a metadata object

**Fields**

clusters A list of cluster information

genfunc A string specifying a distribution for the random numbers

seedinfo A list with the parameters for the random number generator

---

metadata.metric-class *A class that represents a metadata object for metric data*

---

**Description**

A class that represents a metadata object for metric data

**Fields**

standardization If standardization is needed, function can be supplied

---

metadata.ordinal-class

*A class that represents a metadata object for ordinal data*

---

**Description**

A class that represents a metadata object for ordinal data

---

metadata.randomstring-class

*A class that represents a metadata object for string data*

---

**Description**

A class that represents a metadata object for string data

---

plot3dMetadata      *3d plot of a metric metadata object*

---

**Description**

3d plot of a metric metadata object

**Usage**

```
plot3dMetadata(m)
```

**Arguments**

m                      A metadata object (for metric data)

**Value**

A 3d plot using function plot3d from package rgl

**Examples**

```
require(MASS)
m <- new("metadata.metric",
        clusters = list(c1 = list(n = 25, mu = c(4,5,4), Sigma=diag(1,3)),
                       c2 = list(n = 25, mu = c(-1,-2,-2), Sigma=diag(1,3))),
        genfunc = mvrnorm)
plot3dMetadata(m)
```

---

plot3dMetadata,metadata.metric-method  
*3d plot of a metric metadata object*

---

**Description**

3d plot of a metric metadata object

**Usage**

```
## S4 method for signature 'metadata.metric'
plot3dMetadata(m)
```

**Arguments**

m                      A metadata object (for metric data)

**Value**

A 3d plot using function plot3d from package rgl

---

plotMetadata	<i>Plot a metadata object</i>
--------------	-------------------------------

---

**Description**

Plot a metadata object

**Usage**

```
plotMetadata(m)
```

**Arguments**

m                    A metadata object

**Value**

A plot, created by generating an instance of the dataset from the metadata object

**Examples**

```
require(MASS)
m <- new("metadata.metric",
        clusters = list(c1 = list(n = 25, mu = c(4,5), Sigma=diag(1,2)),
                       c2 = list(n = 25, mu = c(-1,-2), Sigma=diag(1,2))),
        genfunc = mvrnorm)
plotMetadata(m)
```

---

plotMetadata,metadata.binary-method	<i>Plot a metadata object</i>
-------------------------------------	-------------------------------

---

**Description**

Plot a metadata object

**Usage**

```
## S4 method for signature 'metadata.binary'
plotMetadata(m)
```

**Arguments**

m                    A metadata object

**Value**

A plot, created by generating an instance of the dataset from the metadata object

---

`plotMetadata,metadata.functional-method`  
*Plot a metadata object*

---

**Description**

Plot a metadata object

**Usage**

```
## S4 method for signature 'metadata.functional'  
plotMetadata(m)
```

**Arguments**

`m`                    A metadata object

**Value**

A plot, created by generating an instance of the dataset from the metadata object

---

`plotMetadata,metadata.metric-method`  
*Plot a metadata object*

---

**Description**

Plot a metadata object

**Usage**

```
## S4 method for signature 'metadata.metric'  
plotMetadata(m)
```

**Arguments**

`m`                    A metadata object

**Value**

A plot, created by generating an instance of the dataset from the metadata object

---

```
plotMetadata,metadata.ordinal-method
      Plot a metadata object
```

---

**Description**

Plot a metadata object

**Usage**

```
## S4 method for signature 'metadata.ordinal'
plotMetadata(m)
```

**Arguments**

m                    A metadata object

**Value**

A plot, created by generating an instance of the dataset from the metadata object

---

```
sampleGrid                    Sample grid points for functional data
```

---

**Description**

Sample grid points for functional data

**Usage**

```
sampleGrid(total_n, minT, maxT, granularity, regular = FALSE)
```

**Arguments**

total_n	Number of Observations
minT	Minimum number of time points sampled
maxT	Maximum number of time points sampled
granularity	Number of possible time points in total
regular	If TRUE, maxT time points are sampled at the same time points for each function

**Value**

A binary matrix indicating whether the function should be evaluated at a given time point

**Examples**

```
sampleGrid(total_n = 10, minT = 4, maxT = 10, granularity = 20)
```

---

```
saveSetup
```

---

*Saves a list of metadata objects to a new setup file*

---

**Description**

Saves a list of metadata objects to a new setup file

**Usage**

```
saveSetup(
  name,
  author,
  mail,
  inst,
  cit = "Unpublished",
  objects,
  table,
  seedinfo = list(100, paste(R.version$major, R.version$minor, sep = "."), RNGkind()),
  metaseedinfo = list(100, paste(R.version$major, R.version$minor, sep = "."),
    RNGkind()),
  custom_funcs = NULL,
  custom_name = NULL
)
```

**Arguments**

name	The name of the new setup (and thus also the filename)
author	Full name of the author
mail	Contact e-mail address of the author
inst	Institution of the author
cit	Reference to the publication where the setup was used, defaults to unpublished
objects	List of metadata objects
table	Info table for the setup
seedinfo	Random number generator parameters for the data sets
metaseedinfo	Random number generator parameters for the metadata
custom_funcs	Custom functions that are needed to generate the meta(data)
custom_name	Custom filename that deviates from the authorYear format

**Value**

A .R file that can be processed by `create.dataset`



**Examples**

```
require(MASS)
a = new("metadata.metric",
      clusters = list(c1 = list(n = 25, mu = c(4,5), Sigma=diag(1,2)),
                    c2 = list(n = 25, mu = c(-1,-2), Sigma=diag(1,2))),
      genfunc = mvrnorm)
b = new("metadata.metric",
      clusters = list(c1 = list(n = 44, mu = c(1,2), Sigma=diag(1,2)),
                    c2 = list(n = 66, mu = c(-5,-6), Sigma=diag(1,2))),
      genfunc = mvrnorm)

## Not run:
saveSetup(name="doe2002.R", author="John Dow", mail="john.doe@edu.com",
         inst="Example University", cit="Simple Data, pp. 23-24", objects=list(a, b),
         table=data.frame(n = c(50, 110), k = c(2,2), shape = c("spherical", "spherical")))
unlink("doe2002.R")

## End(Not run)
```

---

summarizeSetup	<i>Returns the setup summary</i>
----------------	----------------------------------

---

**Description**

Returns the setup summary

**Usage**

```
summarizeSetup(name)
```

**Arguments**

name	The name of the setup
------	-----------------------

**Value**

The summary table of name

# Index

[addCluster](#), [2](#)

[checkSetup](#), [3](#)  
[createFileskeleton](#), [3](#)

[deleteCluster](#), [4](#)

[generateData](#), [5](#)  
[generateData,metadata.binary-method](#), [5](#)  
[generateData,metadata.functional-method](#),  
[6](#)  
[generateData,metadata.metric-method](#), [6](#)  
[generateData,metadata.ordinal-method](#),  
[7](#)  
[generateData,metadata.randomstring-method](#),  
[7](#)  
[generateDatabase](#), [8](#)  
[getRandomstrings](#), [9](#)

[initializeObject](#), [9](#)

[metadata.binary-class](#), [10](#)  
[metadata.functional-class](#), [10](#)  
[metadata.general-class](#), [11](#)  
[metadata.metric-class](#), [11](#)  
[metadata.ordinal-class](#), [11](#)  
[metadata.randomstring-class](#), [11](#)

[plot3dMetadata](#), [12](#)  
[plot3dMetadata,metadata.metric-method](#),  
[12](#)  
[plotMetadata](#), [13](#)  
[plotMetadata,metadata.binary-method](#),  
[13](#)  
[plotMetadata,metadata.functional-method](#),  
[14](#)  
[plotMetadata,metadata.metric-method](#),  
[14](#)  
[plotMetadata,metadata.ordinal-method](#),  
[15](#)

[sampleGrid](#), [15](#)  
[saveSetup](#), [16](#)  
[summarizeSetup](#), [17](#)