Package 'randPack'

July 21, 2025

Title Randomization routines for Clinical Trials

Description A suite of classes and functions for randomizing patients in clinical trials.

Version 1.54.0

Author Vincent Carey <stvjc@channing.harvard.edu> and Robert Gentleman

Maintainer Robert Gentleman <rgentlem@gmail.com>

Imports Biobase

Depends methods

License Artistic 2.0

biocViews StatisticalMethod

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

git_branch RELEASE_3_21

git_last_commit a37f283

git_last_commit_date 2025-04-15

Repository Bioconductor 3.21

Date/Publication 2025-07-20

Contents

`ClinicalExperiment-class`
Class "ClinicalExperiment"

Description

A class that defines the parameters for a clinical trial.

Objects from the Class

Objects can be created by calls of the form `new("ClinicalExperiment", ...)`.

Slots

`name`**:** A string naming the experiment.

`factors`**:** A named list, the names are the names of the factors that will be used for randomization. The elements of the list are the names of the factor levels.

`treatments`**:** A named integer vector indicating the relative frequency of treatment assignments that are desired. The names are the treatment names.

`strataFun`**:** A function that can be applied to instances of `PatientData` to obtain the strata that the patient is to be entered on.

`randomization`**:** A list of the randomizers to be used. There is one for each strata.

`patientIDs`**:** An instance of the `PatientID` class that is used to assign patient IDs to the individuals as they are randomized into the trial.

Methods

show `signature(object = "ClinicalExperiment")`: a print method.

randomization<- `signature(object = "ClinicalExperiment", value="list")`: binds a list of lists of randomizer descriptions to a clinical experiment instance

Description

Basically the ClinicalExperiment class should contain the static information that describes a clinical trial and in particular how patients will be randomized to treatments in that trial.

There is a name for the trial, then a list of the factors that can be used for randomization. We have yet to decide on how to handle continuous variables (these can be used in some randomization schemes). We also have not yet included strata.

Most of the accessors are simple functions, not methods.

Note

Similar to the classes defined by N. Balasubraminian.

Author(s)

R. Gentleman

See Also

`treatmentNames`, `numberOfTreatments`, `factorNames`

Examples

```
showClass("ClinicalExperiment")
data(CT1)
CT1@Experiment
```

ClinicalTrial-class *Class "ClinicalTrial"*

Description

container for clinical trial randomization infrastructure and patient data

Objects from the Class

Objects can be created by calls of the form `new("ClinicalTrial", ...)`.

Slots

`Experiment`: Object of class `"ClinicalExperiment"`.

`PatientData`: Object of class `"environment"`.

`Randomizers`: Object of class `"list"`, one randomizer for each strata.

Methods

No methods defined with class "ClinicalTrial" in the signature.

Author(s)

RG and VC

See Also

`ClinicalExperiment-class`

Examples

```
data(CT1)
CT1
```

createTrial *A function to create instances of the ClinicalTrial class.*

Description

This function should be used in preference to calls to new to create instances of the ClinicalTrials class.

Usage

```
createTrial(CExp, seed)
```

Arguments

CExp	An instance of the ClinicalExperiment class.
seed	A seed for the random number generator.

Details

To ensure proper initialization of the different components this function should be called when creating new instances of the ClinicalTrial class.

Value

An instance of the ClinicalTrial class.

Author(s)

R. Gentleman

See Also

ClinicalExperiment-class

Examples

```
#define the available treatments and their relative allocations
trts = c( A = 3L, B = 4L, C = 1L)
#describe the permuted block design
pbdesc = new("PermutedBlockDesc", treatments = trts, type="PermutedBlock",
    numBlocks=4L)

pIDs = new("PatientID", strata="Test", start=100L, stop=200L)

#define the ClinicalExperiment
CE1 = new("ClinicalExperiment",
   name="My first experiment",
   treatments = trts,
   factors = list( F1 = c("A", "B", "C"), F2 = c("t1", "t2")),
```

```
   randomization = list(pbdesc),
   patientIDs = pIDs
)


#now create the instance of the trial
CT1 = createTrial(CE1, seed=301)
```

CT1 *demonstration data for use with randPack*

Description

demonstration data for use with randPack

Usage

```
data(CT1)
data(SampleData)
data(alltabs)
data(pD1)
data(sco)
```

Format

The format for CT1 is is: Formal class 'ClinicalTrial' [package "randPack"] with 3 slots
..@ Experiment :Formal class 'ClinicalExperiment' [package "randPack"] with 6 slots
.. .. ..@ name : chr "My first experiment"
.. .. ..@ factors :List of 2
.. ..$ F1: chr [1:3] "A" "B" "C"
.. ..$ F2: chr [1:2] "t1" "t2"
.. .. ..@ treatments : Named int [1:3] 3 4 1
.. ..- attr(*, "names")= chr [1:3] "A" "B" "C"
.. .. ..@ randomization:List of 2
.. ..$ Center1:List of 1
.. ..$:Formal class 'PermutedBlockDesc' [package "randPack"] with 3 slots
.. ..@ numBlocks : int 4
.. ..@ treatments: Named int [1:3] 3 4 1
.. ..- attr(*, "names")= chr [1:3] "A" "B" "C"
.. ..@ type : chr "PermutedBlock"
.. ..$ Center2:List of 1
.. ..$:Formal class 'RandomDesc' [package "randPack"] with 3 slots
.. ..@ numPatients: int 100
.. ..@ treatments : Named int [1:3] 3 4 1
.. ..- attr(*, "names")= chr [1:3] "A" "B" "C"
.. ..@ type : chr "Random"
.. .. ..@ strataFun :function (pDesc)

.. ..- attr(*, "source")= chr "function(pDesc) pDesc@strata"
.. .. ..@ patientIDs :Formal class 'PatientID' [package "randPack"] with 3 slots
.. ..@ strata: chr [1:2] "Center1" "Center2"
.. ..@ start : int [1:2] 1000 2000
.. ..@ stop : int [1:2] 1150 2150
..@ PatientData:<environment: 0x101c86590>
..@ Randomizers:List of 2
.. ..$ Center1:List of 1
.. .. ..$:Formal class 'PermutedBlock' [package "randPack"] with 3 slots
.. ..@ name : chr "Center1"
.. ..@ treatmentTable: Named int [1:3] 3 4 1
.. ..- attr(*, "names")= chr [1:3] "A" "B" "C"
.. ..@ stateVariables:<environment: 0x101c867c0>
.. ..$ Center2:List of 1
.. .. ..$:Formal class 'Random' [package "randPack"] with 3 slots
.. ..@ name : chr "Center2"
.. ..@ treatmentTable: Named int [1:3] 3 4 1
.. ..- attr(*, "names")= chr [1:3] "A" "B" "C"
.. ..@ stateVariables:<environment: 0x101c86a60>

Details

various items used in examples or vignette have been serialized for convenience.

Examples

```
data(CT1)
## maybe str(CT1) ; plot(CT1) ...
```

`getEnrolleeInfo` *Returns a named list, data frames by strata giving all covariate information along with allocations*

Description

return data frames by strata giving all covariate information along with allocations

Usage

```
getEnrolleeInfo(x)
```

Arguments

x	An instance of the `ClinicalTrial-class`

Value

A list of data frames, one per stratum, for all current enrollees. The first column is named `name` and gives the patient name, the last column is named `alloc` and gives the treatment allocation for that patient.

Author(s)

RG and VC

Examples

```
data(CT1)
getEnrolleeInfo(CT1)
```

---

getTreatment *Computes random allocation to treatment of a patient in a trial*

Description

Computes random allocation to treatment of a patient in a trial according to the randomizers etc. defined.

Usage

```
getTreatment(cTrial, pData)
```

Arguments

`cTrial`	An instance of `ClinicalTrial-class`.
`pData`	An instance of `PatientData-class`.

Author(s)

RG and VC

See Also

`help`

Examples

```
data(pD1)
data(CT1)
getEnrolleeInfo(CT1)
getTreatment(CT1, pD1)
getEnrolleeInfo(CT1)
```

makeRandomizer *A function to create specific instances of randomizers.*

Description

This function should be used to create all randomizers. Direct calls to new should not be used as some coordination etc is needed to create coherent and complete randomizers. All of the information about the randomizer should be in the type argument which must be an instance of the RandomizerDesc class.

Usage

```
makeRandomizer(name, type, seed)
```

Arguments

name	The name of the randomizer.
type	An instance of the RandomizerDesc class.
seed	The initial seed for the randomizer.

Details

None right now.

Value

An instance of the Randomizer class.

Author(s)

RG and VC

See Also

RandomizerDesc-class

`minimizePocSim` *use the Pocock-Simon or Taves algorithm for computing covariate-adaptive 'minimization' allocations for a clinical trial*

Description

use the Pocock-Simon or Taves algorithm for computing covariate-adaptive 'minimization' allocations for a clinical trial

Usage

```
minimizePocSim(df, features, trtvec, obsdf, trttab, f = function(x, y) sum(abs(x + 1 - y)))
minimizeTaves(df, features, trtvec, obsdf, trttab)
```

Arguments

`df`	a data frame with columns corresponding to covariates rows corresponding to subjects
`features`	character vector of covariates to use
`trtvec`	vector of assignments made so far
`obsdf`	data frame for incoming observation, with values for all components enumerated in `features`
`trttab`	table of treatment ratios
`f`	score that determines impending allocation

Details

These functions are generally not called directly. See the vignette; if supplied as the method slot of a MinimizationDesc object the appropriate data are assembled as arguments to these functions.

Value

a treatment code

Examples

```
new("MinimizationDesc", treatments=c(A=1L, B=1L), method=minimizePocSim,
   type="Minimization", featuresInUse="sex")
```

PatientData-class *Class "PatientData"*

Description

A class designed to hold patient data when randomizing participants to a clinical trial.

Objects from the Class

Objects can be created by calls of the form `new("PatientData", ...)`.

Slots

`name`: That patients name, a length one `character` vector.

`covariates`: A named list, with one element for each variable.

`date`: An instance of the `Date` class.

`patientID`: A length one character vector containing the assigned patient ID.

`strata`: A length one character vector describing the strata the patient will be randomized to.

Methods

No methods defined with class "PatientData" in the signature.

Description

This class is used to hold the internal representation of patient data. We chose to use a named list to hold the covariate information as it might be relatively free form, with different centers (strata) providing different levels of detail. Internal code will need to be careful to handle this properly (and it doesn't currently).

The `patientID` slot will hold the assigned patient ID, usually this is provided by the software during randomization, but there are situations (eg cross-over trials, or trials with specific requirements where this might be pre-assigned) [current implementation does not handle this case].

Author(s)

RG and VC

Examples

```
showClass("PatientData")
```

PatientID-class *Class "PatientID"*

Description

A class to represent ID strings that will be assigned to patients as they are randomized in a clinical trial.

Objects from the Class

Objects can be created by calls of the form `new("PatientID", ...)`.

Slots

`strata`**:** A vector of strata identifiers.

`start`**:** The starting indices for each strata.

`stop`**:** The largest value for each strata.

Methods

No methods defined with class "PatientID" in the signature.

Description

Generally patient IDs are assigned at the time a patient is randomized. The scheme used here is very simple, disjoint sets of integers are used for the different strata.

Once all patiend IDs have been allocated further attempts to randomize patients on that strata should fail.

Author(s)

RG and VC

Examples

```
pIDs = new("PatientID",
           strata = c("Center1", "Center2"),
           start = c(1000L, 2000L),
           stop = c(1150L, 2150L)
)
validPID(pIDs)
```

`Randomizer-class` *Class "Randomizer" and "RandomizerDesc" plus their subclasses.*

Description

These classes form the infrastructure that can be used to create different randomizers for clinical trials or other similar treatment allocation experiments.

Objects from the Class

Both "Randomizer and "RandomizerDesc" are virtual classes and no objects may be created from them. Their various subclasses, "ForcedAlloc", "Random", "PermutedBlock" and "Urn" can be instantiated.

Slots

For "RandomizerDesc":

`treatments`**:** A named integer vector. The names correspond to treatment names, the integers are relative allocations.

`type`**:** The name of the randomizer, this will be set internally.

For "Randomizer":

`name`**:** The name of the randomizer.

`treatmentTable`**:** A named integer vector. The names correspond to treatment names and the integer values to relative allocations.

`stateVariables`**:** An environment that is used to hold any variables that need to retain state information.

Methods

coerce `signature(from = "Randomizer", to = "Minimization")`: ...

Author(s)

RG and VC

Examples

```
showClass("Randomizer")
```

SampleData *Randomly generated data representing a potential patient cohort.*

Description

The data were generated using the script in `inst/scripts/makePats.R` and are intended for testing components of the package.

Usage

```
data(SampleData)
```

Format

A data frame with 78 observations on the following 7 variables.

`name` The patients name.

`sex` The sex of the patient, `M` or `F`.

`age` The age of the patient.

`date` The date they were entered into the trial.

`strata` Which center they were entered by, `Center1` or `Center2`.

`trt` A treatment label, either `A` or `B`.

`surv` Survival time, if treatment is `A` then exponential with a mean of 10, for `B` exponential with mean 12.

Details

Clearly the randomizer will assign the patients to treatments, so the indicator here is irrelevant, but it seemed useful to have some difference in survival, and to keep track of it.

Examples

```
data(SampleData)
SampleData[1:10,]
```

simPats *A function to simulate patient covariate data.*

Description

The function allows for a relatively simple description of covariates for patients in a clinical trial and then the subsequent simulation of a cohort of the necessary size.

Usage

```
simPats(npat, factList)
```

Arguments

npat	The number of patients to simulate.
factList	A list containing either functions that will be called or named vectors containing relative proportions, for factors.

Value

A data.frame with npat rows and one column for each variable given in the factList.

Author(s)

RG and VC

Examples

```
coh1 =  list(center=c(C1=.4, C2=.2, C3=.1, C4=.3),
             sex=c(Male=.5, Female=.5),
             age = function(x) runif(x, min=50, max=70))
simDat = simPats(100, coh1)
simDat[1:5,]
table(simDat[,1])
```

treatmentNames *Accessor functions for the ClinicalExperiment class.*

Description

These functions should be used to access the information in a instance of the ClinicalExperiment-class class.

Usage

```
treatmentNames(object)
numberOfTreatments(object)
treatmentFactors(object)
factorNames(object)
numberOfFactorLevels(object)
```

Arguments

object	An instance of the ClinicalExperiment class.

Details

These functions provide the names of treatments, the names of factors, and the levels of the factors.

Value

In all cases the information indicated by the name of the function is returned, in an appropriate form.

Author(s)

R. Gentleman

Examples

```
data(CT1)
treatmentNames(CT1@Experiment)
```

Index