Package ‘campsis’

October 13, 2023

Type Package

Title Generic PK/PD Simulation Platform CAMPSIS

Version 1.5.0

Description A generic, easy-to-use and intuitive pharmacokinetic/pharmacodynamic (PK/PD) simulation platform based on R packages ‘rxode2’, ‘RxODE’ and ‘mrgsolve’. CAMPSIS provides an abstraction layer over the underlying processes of writing a PK/PD model, assembling a custom dataset and running a simulation. CAMPSIS has a strong dependency to the R package ‘campsismod’, which allows to read/write a model from/to files and adapt it further on the fly in the R environment. Package ‘campsis’ allows the user to assemble a dataset in an intuitive manner. Once the user’s dataset is ready, the package is in charge of preparing the simulation, calling ‘rxode2’, ‘RxODE’ or ‘mrgsolve’ (at the user’s choice) and returning the results, for the given model, dataset and desired simulation settings.

License GPL (= 3)


BugReports https://github.com/Calvagone/campsis/issues

Depends campismod (= 1.1.0), R (= 4.0.0)

Imports assertthat, digest, dplyr, ggplot2, furrr, future, MASS, methods, plyr, progressr, purrr, rlang, stats, tibble, tidyr

Suggests bookdown, devtools, gridExtra, knitr, mrgsolve, pkgdown, markdown, roxygen2, rxode2, stringr, testthat, tictoc, vdiffr

VignetteBuilder knitr

Encoding UTF-8

Language en-US

RoxygenNote 7.2.3

Collate 'global.R' 'utilities.R' 'check.R' 'generic.R' 'seed.R'
  'distribution.R' 'dataset_config.R' 'time_entry.R' 'occasion.R'
  'occasions.R' 'treatment_iov.R' 'treatment_iovs.R'
  'dose_adaptation.R' 'dose_adaptations.R' 'treatment_entry.R'
R topics documented:

'treatment.R' 'observations.R' 'observations_set.R'
'covariate.R' 'covariates.R' 'bootstrap.R' 'protocol.R' 'arm.R'
'arms.R' 'event.R' 'events.R' 'scenario.R' 'scenarios.R'
'simulation_engine.R' 'dataset.R' 'parameter_uncertainty.R'
'event_logic.R' 'dataset_summary.R' 'hardware_settings.R'
'simulation_progress.R' 'solver_settings.R' 'noCB_settings.R'
'declare_settings.R' 'internal_settings.R'
'simulation_settings.R' 'plan_setup.R' 'simulate_preprocess.R'
'simulate.R' 'results_processing.R' 'default_plot.R'

NeedsCompilation no

Author  Nicolas Luyckx [aut, cre]
Maintainer Nicolas Luyckx <nicolas.luyckx@calvagone.com>
Repository CRAN
Date/Publication 2023-10-13 13:40:02 UTC

R topics documented:

applyCompartmentCharacteristics ........................................ 4
Arm ................................................................. 5
arm-class .......................................................... 5
arms-class ......................................................... 6
BinomialDistribution .................................................. 6
Bolus ................................................................. 6
bolus-class ........................................................ 7
Bootstrap ............................................................ 7
bootstrap-class ...................................................... 8
BootstrapDistribution ................................................. 8
bootstrap_distribution-class ......................................... 9
campsis_handler ........................................................ 9
ConstantDistribution ................................................ 10
constant_distribution-class ......................................... 10
Covariate ............................................................. 11
covariate-class ...................................................... 11
covariates-class ..................................................... 11
Dataset ............................................................... 12
dataset-class ........................................................ 12
DatasetConfig ........................................................ 13
dataset_config-class ................................................ 13
Declare .............................................................. 13
declare_settings-class ............................................. 14
discrete ............................................................. 14
discrete-class ......................................................... 15
DoseAdaptation ....................................................... 15
dose_adaptation-class .............................................. 16
dose_adaptations-class ............................................. 16
dosingOnly ........................................................... 16
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>EtaDistribution</td>
<td>17</td>
</tr>
<tr>
<td>Event</td>
<td>17</td>
</tr>
<tr>
<td>event-class</td>
<td>18</td>
</tr>
<tr>
<td>EventCovariate</td>
<td>18</td>
</tr>
<tr>
<td>Events</td>
<td>19</td>
</tr>
<tr>
<td>events-class</td>
<td>19</td>
</tr>
<tr>
<td>event_covariate-class</td>
<td>19</td>
</tr>
<tr>
<td>FixedDistribution</td>
<td>20</td>
</tr>
<tr>
<td>fixed_covariate-class</td>
<td>20</td>
</tr>
<tr>
<td>fixed_distribution-class</td>
<td>20</td>
</tr>
<tr>
<td>FunctionDistribution</td>
<td>21</td>
</tr>
<tr>
<td>function_distribution-class</td>
<td>21</td>
</tr>
<tr>
<td>generateIV</td>
<td>22</td>
</tr>
<tr>
<td>generateIV_</td>
<td>22</td>
</tr>
<tr>
<td>getCovariates</td>
<td>23</td>
</tr>
<tr>
<td>getEventCovariates</td>
<td>23</td>
</tr>
<tr>
<td>getFixedCovariates</td>
<td>24</td>
</tr>
<tr>
<td>getIOVs</td>
<td>25</td>
</tr>
<tr>
<td>getOccasions</td>
<td>25</td>
</tr>
<tr>
<td>getSeedForDatasetExport</td>
<td>26</td>
</tr>
<tr>
<td>getSeedForIteration</td>
<td>26</td>
</tr>
<tr>
<td>getSeedForParametersSampling</td>
<td>27</td>
</tr>
<tr>
<td>getSplittingConfiguration</td>
<td>27</td>
</tr>
<tr>
<td>getTimes</td>
<td>28</td>
</tr>
<tr>
<td>getTimeVaryingCovariates</td>
<td>28</td>
</tr>
<tr>
<td>Hardware</td>
<td>29</td>
</tr>
<tr>
<td>hardware_settings-class</td>
<td>30</td>
</tr>
<tr>
<td>Infusion</td>
<td>31</td>
</tr>
<tr>
<td>infusion-class</td>
<td>32</td>
</tr>
<tr>
<td>internal_settings-class</td>
<td>32</td>
</tr>
<tr>
<td>IOV</td>
<td>32</td>
</tr>
<tr>
<td>length.arm-method</td>
<td>33</td>
</tr>
<tr>
<td>length.dataset-method</td>
<td>33</td>
</tr>
<tr>
<td>LogNormalDistribution</td>
<td>34</td>
</tr>
<tr>
<td>mrgsolve_engine-class</td>
<td>34</td>
</tr>
<tr>
<td>NOCB</td>
<td>35</td>
</tr>
<tr>
<td>noch_settings-class</td>
<td>35</td>
</tr>
<tr>
<td>NormalDistribution</td>
<td>36</td>
</tr>
<tr>
<td>Observations</td>
<td>36</td>
</tr>
<tr>
<td>observations-class</td>
<td>37</td>
</tr>
<tr>
<td>observations_set-class</td>
<td>37</td>
</tr>
<tr>
<td>obsOnly</td>
<td>37</td>
</tr>
<tr>
<td>Occasion</td>
<td>38</td>
</tr>
<tr>
<td>occasion-class</td>
<td>38</td>
</tr>
<tr>
<td>occasions-class</td>
<td>38</td>
</tr>
<tr>
<td>ParameterDistribution</td>
<td>39</td>
</tr>
<tr>
<td>PI</td>
<td>39</td>
</tr>
<tr>
<td>protocol-class</td>
<td>40</td>
</tr>
</tbody>
</table>
applyCompartmentCharacteristics

Apply compartment characteristics from model. In practice, only compartment infusion duration needs to be applied.

Description

Apply compartment characteristics from model. In practice, only compartment infusion duration needs to be applied.

Usage

applyCompartmentCharacteristics(table, properties)
Create a treatment arm.

**Arguments**

- `id` unique identifier for this arm (available through dataset), integer. If NA (default), this identifier is auto-incremented.
- `subjects` number of subjects in arm, integer
- `label` arm label, single character string. If set, this label will be output in the ARM column of CAMPSIS instead of the identifier.

**Value**

an arm

**Description**

Arm class.

**Slots**

- `id` arm unique ID, integer
- `subjects` number of subjects in arm, integer
- `label` arm label, single character string
- `protocol` protocol
- `covariates` covariates
- `bootstrap` covariates to be bootstrapped
arms-class  

Arms class.

BinomialDistribution  

Binomial distribution.

Description

Binomial distribution.

Usage

BinomialDistribution(trials, prob)

Arguments

trials  
  number of Bernoulli trials per observation (=subject), integer

prob  
  probability of success for each trial

Value

a binomial distribution

Bolus  

Create one or several bolus(es).

Description

Create one or several bolus(es).

Usage

Bolus(
  time,  
  amount,  
  compartment = NA,  
  f = NULL,  
  lag = NULL,  
  ii = NULL,  
  addl = NULL  
)
bolus-class

Arguments

- **time**: treatment time(s), numeric value or vector. First treatment time if used together with ii and addl.
- **amount**: amount to give as bolus, single numeric value
- **compartment**: compartment index, single integer value
- **f**: fraction of dose amount, distribution
- **lag**: dose lag time, distribution
- **ii**: interdose interval, requires argument 'time' to be a single numeric value
- **addl**: number of additional doses, requires argument 'time' to be a single integer value

Value

a single bolus or a list of boluses

bolus-class  

Bolus class.

Description

Bolus class.

Bootstrap

Create a bootstrap object.

Description

Create a bootstrap object.

Usage

```r
Bootstrap(
  data,
  id = "BS_ID",
  replacement = FALSE,
  random = FALSE,
  export_id = FALSE
)
```
Arguments

- **data**: data frame to be bootstrapped. It must have a unique identifier column named according to the specified argument 'id' (default value is 'BS_ID'). Other columns are covariates to bootstrap. They must all be numeric. Whatever the configuration of the bootstrap, these covariates are always read row by row and belong to a same individual.

- **id**: unique identifier column name in data

- **replacement**: values can be reused or not when drawn, logical

- **random**: values are drawn randomly, logical

- **export_id**: tell CAMPSIS if the identifier 'BS_ID' must be output or not, logical

Value

- a bootstrap object

---

**BootstrapDistribution**

Create a bootstrap distribution. During function sampling, CAMPSIS will generate values depending on the given data and arguments.

**Usage**

BootstrapDistribution(data, replacement = FALSE, random = FALSE)
### bootstrap_distribution-class

*Bootstrap distribution class.*

#### Description

Bootstrap distribution class.

#### Slots

- **data**: values to draw, numeric vector
- **replacement**: values can be reused or not, logical
- **random**: values are drawn randomly, logical

---

**campsis_handler**

*Suggested Campsis handler for showing the progress bar.*

#### Description

Suggested Campsis handler for showing the progress bar.

#### Usage

```
campsis_handler()
```

#### Value

a progressr handler list
ConstantDistribution  

Create a constant distribution. Its value will be constant across all generated samples.

**Description**

Create a constant distribution. Its value will be constant across all generated samples.

**Usage**

ConstantDistribution(value)

**Arguments**

value  

covariate value, single numeric value

**Value**

a constant distribution (same value for all samples)

**Slots**

value  

covariate value, single numeric value
Create a non time-varying (fixed) covariate.

**Description**
Create a non time-varying (fixed) covariate.

**Usage**
Covariate(name, distribution)

**Arguments**
- name: covariate name, single character value
- distribution: covariate distribution

**Value**
a fixed covariate

### covariate-class
*Covariate class.*

**Description**
Covariate class.

**Slots**
- name: covariate name, single character value
- distribution: covariate distribution

### covariates-class
*Covariates class.*

**Description**
Covariates class.
Dataset: Create a dataset.

Description

Create a dataset.

Usage

Dataset(subjects = NULL)

Arguments

subjects: number of subjects in the default arm

Value

a dataset

dataset-class: Dataset class.

Description

Dataset class.

Slots

arms: a list of treatment arms
config: dataset configuration for export
iiv: data frame containing the inter-individual variability (all ETAS) for the export
DatasetConfig

Create a dataset configuration. This configuration allows CAMPSIS to know which are the default depot and observed compartments.

Usage

```r
DatasetConfig(
    defDepotCmt = 1,
    defObsCmt = 1,
    exportTSLD = FALSE,
    exportTDOS = FALSE
)
```

Arguments

defDepotCmt default depot compartment, integer
defObsCmt default observation compartment, integer
exportTSLD export column TSLD (time since last dose), logical
exportTDOS export column TDOS (time of last dose), logical

Value

a dataset configuration

dataset_config-class Dataset configuration class.

Description

Dataset configuration class.

Slots

def_depot_cmt default depot compartment, integer
def_obs_cmt default observation compartment, integer
export_tsld export column TSLD, logical
export_tdos export column TDOS, logical
Declare

Create declare settings.

Description

Create declare settings.

Usage

Declare(variables = character(0))

Arguments

variables uninitialized variables to be declared, only needed with mrgsolve

Value

Declare settings

DiscreteDistribution

Discrete distribution.

Description

Discrete distribution.

Usage

DiscreteDistribution(x, prob, replace = TRUE)
distribution-class

Arguments

  x  vector of one or more integers from which to choose
  prob  a vector of probability weights for obtaining the elements of the vector being sampled
  replace  should sampling be with replacement, default is TRUE

Value

  a discrete distribution

DoseAdaptation

Description

  Create a dose adaptation.

Usage

  DoseAdaptation(formula, compartments = integer(0))

Arguments

  formula  formula to apply, single character string, e.g. "AMT*WT"
  compartments  compartment numbers where the formula needs to be applied, integer vector. Default is integer(0) (formula applied on all compartments)

Value

  a fixed covariate
dose_adaptation-class  

Dose adaptation class.

Description

Dose adaptation class.

Slots

- **formula**: formula to apply, single character string, e.g. "AMT*WT"
- **compartments**: compartment numbers where the formula needs to be applied

---

dose_adaptations-class  

Dose adaptations class.

Description

Dose adaptations class.

---

dosingOnly  

Filter CAMPSIS output on dosing rows.

Description

Filter CAMPSIS output on dosing rows.

Usage

```r
dosingOnly(x)
```

Arguments

- **x**: data frame, CAMPSIS output

Value

a data frame with the dosing rows
EtaDistribution

Create an ETA distribution. The resulting distribution is a normal distribution, with mean=0 and sd=sqrt(OMEGA).

Description

Create an ETA distribution. The resulting distribution is a normal distribution, with mean=0 and sd=sqrt(OMEGA).

Usage

EtaDistribution(model, omega)

Arguments

- model : model
- omega : corresponding THETA name, character

Value

an ETA distribution

Event

Create an interruption event.

Description

Create an interruption event.

Usage

Event(name = NULL, times, fun, debug = FALSE)

Arguments

- name : event name, character value
- times : interruption times, numeric vector
- fun : event function to apply at each interruption
- debug : output the variables that were changed through this event

Value

an event definition
**Description**

Event class.

**Slots**

- **name**: event name, character value
- **times**: interruption times, numeric vector
- **fun**: event function to apply at each interruption
- **debug**: output the variables that were changed through this event

---

**EventCovariate**

Create an event covariate. These covariates can be modified further in interruption events.

**Description**

Create an event covariate. These covariates can be modified further in interruption events.

**Usage**

`EventCovariate(name, distribution)`

**Arguments**

- **name**: covariate name, character
- **distribution**: covariate distribution at time 0

**Value**

a time-varying covariate
Events

Create a list of interruption events.

Description

Create a list of interruption events.

Usage

Events()

Value

a events object

events-class

Events class.

Description

Events class.

event_covariate-class

Event covariate class.

Description

Event covariate class.
FixedDistribution

Create a fixed distribution. Each sample will be assigned a fixed value coming from vector 'values'.

Description

Create a fixed distribution. Each sample will be assigned a fixed value coming from vector 'values'.

Usage

FixedDistribution(values)

Arguments

values covariate values, numeric vector (1 value per sample)

Value

a fixed distribution (1 value per sample)

fixed_covariate-class

Fixed covariate class.

Description

Fixed covariate class.

fixed_distribution-class

Fixed distribution class.

Description

Fixed distribution class.

Slots

values covariate values, numeric vector (1 value per sample)
FunctionDistribution

Create a function distribution. During distribution sampling, the provided function will be responsible for generating values for each sample. If first argument of this function is not the size (n), please tell which argument corresponds to the size 'n' (e.g. list(size="n")).

Usage

FunctionDistribution(fun, args)

Arguments

fun function name, character (e.g. 'rnorm')
args list of arguments (e.g list(mean=70, sd=10))

Value

a function distribution

Description

Function distribution class.

Slots

fun function name, character (e.g. 'rnorm')
args list of arguments (e.g list(mean=70, sd=10))
**generateIIV**

*Generate IIV matrix for the given Campsis model.*

**Description**

Generate IIV matrix for the given Campsis model.

**Usage**

`generateIIV(model, n, offset = 0)`

**Arguments**

- `model` : Campsis model
- `n` : number of subjects
- `offset` : if specified, resulting ID will be ID + offset

**Value**

IIV data frame with ID column

---

**generateIIV_**

*Generate IIV matrix for the given OMEGA matrix.*

**Description**

Generate IIV matrix for the given OMEGA matrix.

**Usage**

`generateIIV_(omega, n)`

**Arguments**

- `omega` : omega matrix
- `n` : number of subjects

**Value**

IIV data frame
**getCovariates**

Get all covariates (fixed / time-varying / event covariates).

**Description**

Get all covariates (fixed / time-varying / event covariates).

**Usage**

```r
getCovariates(object)
```

```r
## S4 method for signature 'covariates'
getCovariates(object)
```

```r
## S4 method for signature 'arm'
getCovariates(object)
```

```r
## S4 method for signature 'arms'
getCovariates(object)
```

```r
## S4 method for signature 'dataset'
getCovariates(object)
```

**Arguments**

- **object**
  - any object

**Value**

- all covariates from object

---

**getEventCovariates**

Get all event-related covariates.

**Description**

Get all event-related covariates.

**Usage**

```r
getEventCovariates(object)
```

```r
## S4 method for signature 'covariates'
getEventCovariates(object)
```

```r
## S4 method for signature 'arm'
getEventCovariates(object)
```
getEventCovariates(object)

## S4 method for signature 'arms'
getEventCovariates(object)

## S4 method for signature 'dataset'
getEventCovariates(object)

Arguments

object any object

Value

all event-related covariates from object

getFixedCovariates Get all fixed covariates.

Description

Get all fixed covariates.

Usage

getFixedCovariates(object)

## S4 method for signature 'covariates'
getFixedCovariates(object)

## S4 method for signature 'arm'
getFixedCovariates(object)

## S4 method for signature 'arms'
getFixedCovariates(object)

## S4 method for signature 'dataset'
getFixedCovariates(object)

Arguments

object any object

Value

all fixed covariates from object
getIOVs

Get all IOV objects.

Description
Get all IOV objects.

Usage
getIOVs(object)

Arguments
object any object

Value
all IOV's from object

getOccasions
Get all occasions.

Description
Get all occasions.

Usage
getOccasions(object)

Arguments
getOccasions(object)

Arguments
object any object
getSeedForIteration

Arguments

object any object

Value

all occasions from object

getSeedForDatasetExport

Get seed for dataset export.

Description

Get seed for dataset export.

Usage

getSeedForDatasetExport(seed, progress)

Arguments

seed original seed
progress simulation progress

Value

the seed value used to export the dataset

getSeedForIteration

Get seed for iteration.

Description

Get seed for iteration.

Usage

getSeedForIteration(seed, progress)

Arguments

seed original seed
progress simulation progress

Value

the seed value to be used for the given replicate number and iteration
**getSeedForParametersSampling**

*Get seed for parameter uncertainty sampling.*

**Description**

Get seed for parameter uncertainty sampling.

**Usage**

```
getSeedForParametersSampling(seed)
```

**Arguments**

- **seed**
  - original seed

**Value**

the seed value used to sample parameter uncertainty

---

**getSplittingConfiguration**

*Get splitting configuration for parallel export.*

**Description**

Get splitting configuration for parallel export.

**Usage**

```
getSplittingConfiguration(dataset, hardware)
```

**Arguments**

- **dataset**
  - Campsis dataset to export
- **hardware**
  - hardware configuration

**Value**

splitting configuration list (if 'parallel_dataset' is enabled) or NA (if 'parallel_dataset' disabled or if the length of the dataset is less than the dataset export slice size)
getTimeVaryingCovariates

Get all time-varying covariates.

Description
Get all time-varying covariates.

Usage
getTimes(object)

Arguments
object any object

Value
numeric vector with all unique times, sorted
**Usage**

getTimeVaryingCovariates(object)

## S4 method for signature 'covariates'
getTimeVaryingCovariates(object)

## S4 method for signature 'arm'
getTimeVaryingCovariates(object)

## S4 method for signature 'arms'
getTimeVaryingCovariates(object)

## S4 method for signature 'dataset'
getTimeVaryingCovariates(object)

**Arguments**

object any object

**Value**

all time-varying covariates from object

---

**Hardware**

*Create hardware settings.*

**Description**

Create hardware settings.

**Usage**

Hardware(
    cpu = 1,
    replicate_parallel = FALSE,
    scenario_parallel = FALSE,
    slice_parallel = FALSE,
    slice_size = NULL,
    dataset_parallel = FALSE,
    dataset_slice_size = 500,
    auto_setup_plan = NULL
)
Arguments

- `cpu` number of CPU cores to use, default is 1
- `replicate_parallel` enable parallel computing for replicates, default is FALSE
- `scenario_parallel` enable parallel computing for scenarios, default is FALSE
- `slice_parallel` enable parallel computing for slices, default is FALSE
- `slice_size` number of subjects per simulated slice, default is NULL (auto-configured by Campsis depending on the specified engine)
- `dataset_parallel` enable parallelisation when exporting dataset into a table, default is FALSE
- `dataset_slice_size` dataset slice size when exporting subjects to a table, default is 500. Only applicable if ‘dataset_parallel’ is enabled.
- `auto_setup_plan` auto-setup plan with the library future, if not set (i.e. =NULL), plan will be setup automatically if the number of CPU’s > 1.

Value

hardware settings

---

Hardware settings class.

Description

Hardware settings class.

Slots

- `cpu` number of CPU cores to use, default is 1
- `replicate_parallel` enable parallel computing for replicates, default is FALSE
- `scenario_parallel` enable parallel computing for scenarios, default is FALSE
- `slice_parallel` enable parallel computing for slices, default is FALSE
- `slice_size` number of subjects per simulated slice, default is NULL (auto-configured by Campsis depending on the specified engine)
- `dataset_parallel` enable parallelisation when exporting dataset into a table, default is FALSE
- `dataset_slice_size` dataset slice size when exporting subjects to a table, default is 500. Only applicable if ‘dataset_parallel’ is enabled.
- `auto_setup_plan` auto-setup plan with the library future, default is FALSE
Infusion

Create one or several infusion(s).

Description

Create one or several infusion(s).

Usage

Infusion(
  time,
  amount,
  compartment = NA,
  f = NULL,
  lag = NULL,
  duration = NULL,
  rate = NULL,
  ii = NULL,
  addl = NULL
)

Arguments

time treatment time(s), numeric value or vector. First treatment time if used together with ii and addl.
amount total amount to infuse, numeric
compartment compartment index, integer
f fraction of infusion amount, distribution
lag infusion lag time, distribution
duration infusion duration, distribution
rate infusion rate, distribution
ii interdose interval, requires argument 'time' to be a single numeric value
addl number of additional doses, requires argument 'time' to be a single integer value

Value

a single infusion or a list of infusions.
Infusion class.

Slots

duration infusion duration, distribution
rate infusion rate, distribution

Internal settings class (transient object from the simulation settings).

Description

Internal settings class (transient object from the simulation settings).

Slots

dataset_summary dataset summary
progress simulation progress
iterations list of event iterations

Define inter-occasion variability (IOV) into the dataset. A new variable of name 'colname' will be output into the dataset and will vary at each dose number according to the given distribution.

Description

Define inter-occasion variability (IOV) into the dataset. A new variable of name 'colname' will be output into the dataset and will vary at each dose number according to the given distribution.

Usage

IOV(colname, distribution, doseNumbers = NULL)
Arguments

- colname: name of the column that will be output in dataset
- distribution: distribution
- doseNumbers: dose numbers, if provided, IOV is generated at these doses only. By default, IOV is generated for all doses.

Value

An IOV object

---

**length,arm-method**  
Return the number of subjects contained in this arm.

Description

Return the number of subjects contained in this arm.

Usage

```r
## S4 method for signature 'arm'
length(x)
```

Arguments

- x: arm

Value

A number

---

**length,dataset-method**  
Return the number of subjects contained in this dataset.

Description

Return the number of subjects contained in this dataset.

Usage

```r
## S4 method for signature 'dataset'
length(x)
```

Arguments

- x: dataset
LogNormalDistribution  
Create a log normal distribution.

Description

Create a log normal distribution.

Usage

LogNormalDistribution(meanlog, sdlog)

Arguments

meanlog  mean value of distribution in log domain
sdlog  standard deviation of distribution in log domain

Value

a log normal distribution

mrgsolve_engine-class  
mrgsolve engine class.
Create NOCB settings.

Usage

NOCB(enable = NULL, variables = character(0))

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable</td>
<td>enable/disable next-observation carried backward mode (NOCB), default value is TRUE for mrgsolve, FALSE for RxODE</td>
</tr>
<tr>
<td>variables</td>
<td>variable names subject to NOCB behavior (see vignette for more info)</td>
</tr>
</tbody>
</table>

Value

NOCB settings

nocb_settings-class  NOCB settings class.

Description

NOCB settings class.

Slots

<table>
<thead>
<tr>
<th>Slot</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable</td>
<td>enable/disable next-observation carried backward mode (NOCB), default value is TRUE for mrgsolve, FALSE for RxODE</td>
</tr>
<tr>
<td>variables</td>
<td>variable names subject to NOCB behavior (see vignette for more info)</td>
</tr>
</tbody>
</table>
NormalDistribution

Create a normal distribution.

Description

Create a normal distribution.

Usage

NormalDistribution(mean, sd)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>mean value of distribution</td>
</tr>
<tr>
<td>sd</td>
<td>standard deviation of distribution</td>
</tr>
</tbody>
</table>

Value

a normal distribution

Observations

Create an observations list. Please note that the provided 'times' will automatically be sorted. Duplicated times will be removed.

Description

Create an observations list. Please note that the provided 'times' will automatically be sorted. Duplicated times will be removed.

Usage

Observations(times, compartment = NA)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>times</td>
<td>observation times, numeric vector</td>
</tr>
<tr>
<td>compartment</td>
<td>compartment index, integer</td>
</tr>
</tbody>
</table>

Value

an observations list
observations-class

Observations class.

Description
Observations class.

Slots

- `times` observation times, numeric vector
- `compartment` compartment index, integer
- `dv` observed values, numeric vector (FOR EXTERNAL USE)

observations_set-class

Observations set class.

Description
Observations set class.

obsOnly

Filter CAMPSIS output on observation rows.

Description
Filter CAMPSIS output on observation rows.

Usage

`obsOnly(x)`

Arguments

- `x` data frame, CAMPSIS output

Value

a data frame with the observation rows
Occasion

*Define a new occasion. Occasions are defined by mapping occasion values to dose numbers. A new column will automatically be created in the exported dataset.*

**Description**

Define a new occasion. Occasions are defined by mapping occasion values to dose numbers. A new column will automatically be created in the exported dataset.

**Usage**

Occasion(colname, values, doseNumbers)

**Arguments**

- **colname**: name of the column that will be output in dataset
- **values**: the occasion numbers, any integer vector
- **doseNumbers**: the related dose numbers, any integer vector of same length as 'values'

**Value**

occasion object

---

**occasion-class** _Occasion class._

**Description**

Occasion class.

**Slots**

- **colname**: single character value representing the column name related to this occasion
- **values**: occasion values, integer vector, same length as dose_numbers
- **dose_numbers**: associated dose numbers, integer vector, same length as values

---

**occasions-class** _Occasions class._

**Description**

Occasions class.
ParameterDistribution Create a parameter distribution. The resulting distribution is a log-normal distribution, with mean_log = log(THETA) and sd_log = sqrt(OMEGA).

Description

Create a parameter distribution. The resulting distribution is a log-normal distribution, with mean_log = log(THETA) and sd_log = sqrt(OMEGA).

Usage

ParameterDistribution(model, theta, omega = NULL)

Arguments

- model: model
- theta: corresponding THETA name, character
- omega: corresponding OMEGA name, character, NULL if not defined

Value

a parameter distribution

PI Compute the prediction interval summary over time.

Description

Compute the prediction interval summary over time.

Usage

PI(x, output, scenarios = NULL, level = 0.9, gather = TRUE)

Arguments

- x: data frame
- output: variable to show, character value
- scenarios: scenarios, character vector, NULL is default
- level: PI level, default is 0.9 (90% PI)
- gather: FALSE: med, low & up columns, TRUE: metric column

Value

a summary table
**protocol-class**  
*Protocol class.*

**retrieveParameterValue**  
*Retrieve the parameter value (standardized) for the specified parameter name.*

**Description**  
Retrieve the parameter value (standardized) for the specified parameter name.

**Usage**  
```
retrieveParameterValue(model, paramName, default = NULL, mandatory = FALSE)
```

**Arguments**
- `model`: model
- `paramName`: parameter name
- `default`: default value if not found
- `mandatory`: must be in model or not

**Value**  
the standardized parameter value or the given default value if not found

**rxode_engine-class**  
*RxODE/rxode2 engine class.*

**Description**  
RxODE/rxode2 engine class.

**Slots**
- `rxode2` logical field to indicate if CAMPIS should use rxode2 (field set to TRUE) or RxODE (field set to FALSE). Default is TRUE.
Sample generic object.

Usage

```r
sample(object, n, ...)
```

## S4 method for signature 'constant_distribution,integer'
```r
sample(object, n)
```

## S4 method for signature 'fixed_distribution,integer'
```r
sample(object, n)
```

## S4 method for signature 'function_distribution,integer'
```r
sample(object, n)
```

## S4 method for signature 'bootstrap_distribution,integer'
```r
sample(object, n)
```

## S4 method for signature 'bolus,integer'
```r
sample(object, n, ...)
```

## S4 method for signature 'infusion,integer'
```r
sample(object, n, ...)
```

## S4 method for signature 'observations,integer'
```r
sample(object, n, ...)
```

## S4 method for signature 'covariate,integer'
```r
sample(object, n)
```

## S4 method for signature 'bootstrap,integer'
```r
sample(object, n)
```

## S4 method for signature 'campsis_model,integer'
```r
sample(object, n)
```

Arguments

- `object`: generic object
- `n`: number of samples required
- `...`: extra arguments
Scenario

**Value**
sampling result

---

**scatterPlot**  
*Scatter plot (or X vs Y plot)*.

---

**Description**
Scatter plot (or X vs Y plot).

**Usage**
```
scatterPlot(x, output, colour = NULL, time = NULL)
```

**Arguments**
- `x`: data frame
- `output`: the 2 variables to show, character vector
- `colour`: variable(s) to colour
- `time`: the time to look at those 2 variables, if NULL, min time is used (usually 0)

**Value**
a ggplot object

---

**Scenario**  
*Create an scenario.*

---

**Description**
Create an scenario.

**Usage**
```
Scenario(name = NULL, model = NULL, dataset = NULL)
```

**Arguments**
- `name`: scenario name, single character string
- `model`: either a CAMPSIS model, a function or lambda-style formula
- `dataset`: either a CAMPSIS dataset, a function or lambda-style formula

**Value**
a new scenario
Scenario class

Description

Scenario class.

Slots

name  scenario name, single character string
model either a CAMPSIS model, a function or lambda-style formula
dataset either a CAMPSIS dataset, a function or lambda-style formula

Scenarios

Create a list of scenarios.

Description

Create a list of scenarios.

Usage

Scenarios()

Value

a scenarios object
### setLabel

*Set the label.*

**Description**
Set the label.

**Usage**

```r
setLabel(object, x)
```

#### S4 method for signature 'arm,character'

```r
setLabel(object, x)
```

**Arguments**

- **object**: any object that has a label
- **x**: the new label

**Value**

the updated object

### setSubjects

*Set the number of subjects.*

**Description**
Set the number of subjects.

**Usage**

```r
setSubjects(object, x)
```

#### S4 method for signature 'arm,integer'

```r
setSubjects(object, x)
```

#### S4 method for signature 'dataset,integer'

```r
setSubjects(object, x)
```

**Arguments**

- **object**: any object
- **x**: the new number of subjects
Value

the updated object

---

Settings

Create advanced simulation settings.

Description

Create advanced simulation settings.

Usage

Settings(...)  

Arguments

...  

any user-required settings: see ?Hardware, ?Solver, ?NOCB or ?Declare settings

Value

advanced simulation settings

---

setupPlanDefault  

Setup default plan for the given simulation or hardware settings. This plan will prioritise the distribution of workers in the following order: 1) Replicates (if 'replicate_parallel' is enabled) 2) Scenarios (if 'scenario_parallel' is enabled) 3) Dataset export / slices (if 'dataset_export' or 'slice_parallel' is enabled)

Description

Setup default plan for the given simulation or hardware settings. This plan will prioritise the distribution of workers in the following order: 1) Replicates (if 'replicate_parallel' is enabled) 2) Scenarios (if 'scenario_parallel' is enabled) 3) Dataset export / slices (if 'dataset_export' or 'slice_parallel' is enabled)

Usage

setupPlanDefault(object)

Arguments

object  

simulation or hardware settings

Value

nothing
setupPlanSequential  
*Setup plan as sequential (i.e. no parallelisation).*

**Description**

Setup plan as sequential (i.e. no parallelisation).

**Usage**

```r
setupPlanSequential()
```

**Value**

nothing

shadedPlot  
*Shaded plot (or prediction interval plot).*

**Description**

Shaded plot (or prediction interval plot).

**Usage**

```r
shadedPlot(
  x,
  output,
  colour = NULL,
  strat_extra = NULL,
  level = 0.9,
  alpha = 0.25
)
```

**Arguments**

- `x`  
data frame
- `output`  
variable to show
- `colour`  
variable(s) to colour
- `strat_extra`  
variable(s) to stratify, but not to colour (useful for use with facet_wrap)
- `level`  
PI level, default is 0.9 (90% PI)
- `alpha`  
alpha parameter (transparency) given to geom_ribbon

**Value**

a ggplot object
simulate

Simulate function.

Description

Simulate function.

Usage

```
simulate(
  model,
  dataset,
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = 1,
  dosing = FALSE,
  settings = NULL
)
```

## S4 method for signature `simulate'##

```
simulate(
  model,
  dataset,
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = 1,
  dosing = FALSE,
  settings = NULL
)
```

## S4 method for signature `simulate'

```
simulate(
  model,
  dataset,
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = 1,
  dosing = FALSE,
  settings = NULL
)
replicates = 1,
dosing = FALSE,
    settings = NULL
  )

## S4 method for signature
## 'campsis_model,'
## tbl_df,
## character,
## events,
## scenarios,
## 'function',
## character,
## 'function',
## integer,
## integer,
## logical,
## simulation_settings'
simulate(
  model,
  dataset,
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = 1,
  dosing = FALSE,
  settings = NULL
)

## S4 method for signature
## 'campsis_model,'
## data.frame,
## character,
## events,
## scenarios,
## 'function',
## character,
## 'function',
## integer,
## integer,
## logical,
## simulation_settings'
simulate(
  model,
simulate
dataset,
dest = NULL,
events = NULL,
scenarios = NULL,
tablefun = NULL,
outvars = NULL,
outfun = NULL,
seed = NULL,
replicates = 1,
dosing = FALSE,
settings = NULL
)

## S4 method for signature
## 'campsis_model,
## tbl_df,
## rxode_engine,
## events,
## scenarios,
## 'function',
## character,
## 'function',
## integer,
## integer,
## logical,
## simulation_settings'
simulate(
  model,
  dataset,
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = 1,
  dosing = FALSE,
  settings = NULL
)

## S4 method for signature
## 'campsis_model,
## tbl_df,
## mrgsolve_engine,
## events,
## scenarios,
## 'function',
## character,
## character,
## character,
## character,
## character,
## character,
## character,
## character,
## character,
## simulate

```r
simulate(
  model,
  dataset,
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = 1,
  dosing = FALSE,
  settings = NULL
)
```

### Arguments

- **model**: generic CAMPSIS model
- **dataset**: CAMPSIS dataset or 2-dimensional table
- **dest**: destination simulation engine, default is 'RxODE'
- **events**: interruption events
- **scenarios**: list of scenarios to be simulated
- **tablefun**: function or lambda formula to apply on exported 2-dimensional dataset
- **outvars**: variables to output in resulting dataframe
- **outfun**: function or lambda formula to apply on resulting dataframe after each replicate
- **seed**: seed value
- **replicates**: number of replicates, default is 1
- **dosing**: output dosing information, default is FALSE
- **settings**: advanced simulation settings

### Value

dataframe with all results
SimulationProgress

Create a simulation progress object.

Description

Create a simulation progress object.

Usage

SimulationProgress(
  replicates = 1,
  scenarios = 1,
  progressor = NULL,
  hardware = NULL
)

Arguments

- replicates: total number of replicates to simulate
- scenarios: total number of scenarios to simulate
- progressor: progressr progressor
- hardware: hardware settings

Value

- a progress bar

simulation_engine-class

Simulation engine class.

Description

Simulation engine class.
simulation_progress-class

Simulation progress class.

Description
Simulation progress class.

Arguments
replicates  total number of replicates to simulate
scenarios  total number of scenarios to simulate
iterations  total number of iterations to simulate
slices  total number of slices to simulate
replicate  current replicate number being simulated
scenario  current scenario number being simulated
iteration  current iteration number being simulated
slice  current slice number being simulated
progressor  progressr progressor
hardware  hardware settings

simulation_settings-class

Simulation settings class.

Description
Simulation settings class.

Slots
hardware  hardware settings object
solver  solver settings object
nocb  NOCB settings object
declare  declare settings (mrgsolve only)
internal  internal settings
Create solver settings.

Usage

Solver(
  atol = 1e-08,
  rtol = 1e-08,
  hmax = NA,
  maxsteps = 70000L,
  method = "liblsoda"
)

Arguments

atol  absolute solver tolerance, default is 1e-08
rtol  relative solver tolerance, default is 1e-08
hmax  limit how big a solver step can be, default is NA
maxsteps  max steps between 2 integration times (e.g. when observations records are far apart), default is 70000
method  solver method, for RxODE/rxode2 only: 'liblsoda' (default), 'lsoda', 'dop853', 'indLin'. Mrgsolve's method is always 'lsoda'.

Value

solver settings

Description

Solver settings class. See ?mrgsolve::update. See ?rxode2::rxSolve.
**TimeVaryingCovariate**

**Slots**
- **atol** absolute solver tolerance, default is 1e-08
- **rtol** relative solver tolerance, default is 1e-08
- **hmax** limit how big a solver step can be, default is NA
- **maxsteps** max steps between 2 integration times (e.g. when observations records are far apart), default is 70000
- **method** solver method, for RxODE/rxode2 only: 'liblsoda' (default), 'lsoda', 'dop853', 'indLin'. Mrgsolve’s method is always 'lsoda'.

**spaghettiPlot**

*Spaghetti plot.*

**Description**
Spaghetti plot.

**Usage**

```r
spaghettiPlot(x, output, colour = NULL)
```

**Arguments**
- **x** data frame
- **output** variable to show
- **colour** variable(s) to colour

**Value**
plot

---

**TimeVaryingCovariate**

*Create a time-varying covariate. This covariate will be implemented using EVID=2 rows in the exported dataset and will not use interruption events.*

**Description**
Create a time-varying covariate. This covariate will be implemented using EVID=2 rows in the exported dataset and will not use interruption events.

**Usage**

```r
TimeVaryingCovariate(name, table)
```
Arguments

name       covariate name, character

- **table**
  - data.frame, must contain the mandatory columns 'TIME' and 'VALUE'. An 'ID' column may also be specified. In that case, ID’s between 1 and the max number of subjects in the dataset/arm can be used. All ID’s must have a VALUE defined for TIME 0.

Value

- a time-varying covariate

---

**time_varying_covariate-class**

*Time-varying covariate class.*

---

**Description**

Time-varying covariate class.

---

**treatment-class**

*Treatment class.*

---

**Description**

Treatment class.

---

**treatment_iov-class**

*Treatment IOV class.*

---

**Description**

Treatment IOV class.

---

**Slots**

- colname  name of the column that will be output in dataset
- distribution  distribution
- dose_numbers  associated dose numbers, integer vector, same length as values
UniformDistribution

Description

Create an uniform distribution.

Usage

UniformDistribution(min, max)

Arguments

- min: min value
- max: max value

Value

an uniform distribution
VPC

Compute the VPC summary. Input data frame must contain the following columns: - replicate: replicate number - low: low percentile value in replicate (and in scenario if present) - med: median value in replicate (and in scenario if present) - up: up percentile value in replicate (and in scenario if present) - any scenario column

Description

Compute the VPC summary. Input data frame must contain the following columns: - replicate: replicate number - low: low percentile value in replicate (and in scenario if present) - med: median value in replicate (and in scenario if present) - up: up percentile value in replicate (and in scenario if present) - any scenario column

Usage

VPC(x, scenarios = NULL, level = 0.9)

Arguments

x data frame
scenarios scenarios, character vector, NULL is default
level PI level, default is 0.9 (90% PI)

Value

VPC summary with columns TIME, <scenarios> and all combinations of low, med, up (i.e. low_low, low_med, low_up, etc.)

vpcPlot VPC plot.

Description

VPC plot.

Usage

vpcPlot(x, scenarios = NULL, level = 0.9, alpha = 0.15)

Arguments

x data frame, output of CAMPSIS with replicates
scenarios scenarios, character vector, NULL is default
level PI level, default is 0.9 (90% PI)
alpha alpha parameter (transparency) given to geom_ribbon
Value

a ggplot object
Index

applyCompartmentCharacteristics, 4
Arm, 5
arm-class, 5
arms-class, 6
BinomialDistribution, 6
Bolus, 6
bolus-class, 7
Bootstrap, 7
bootstrap-class, 8
bootstrap_distribution-class, 9
BootstrapDistribution, 8
camquis_handler, 9
constant_distribution-class, 10
ConstantDistribution, 10
Covariate, 11
covariate-class, 11
covariates-class, 11
dataset-class, 12
dataset_config-class, 13
DatasetConfig, 13
Declare, 14
declare_settings-class, 14
DiscreteDistribution, 14
distribution-class, 15
dose_adaptation-class, 16
dose_adaptations-class, 16
DoseAdaptation, 15
dosingOnly, 16
EtaDistribution, 17
Event, 17
event-class, 18
event_covariate-class, 19
EventCovariate, 18
Events, 19
events-class, 19
fixed_covariate-class, 20
fixed_distribution-class, 20
FixedDistribution, 20
function_distribution-class, 21
FunctionDistribution, 21
generateIIV, 22
generateIIV_, 22
getCovariates, 23
getCovariates, arm-method
getCovariates, arms-method
getCovariates, covariates-method
getCovariates, dataset-method
getEventCovariates, 23
getEventCovariates, arm-method
getEventCovariates, arms-method
getEventCovariates, covariates-method
getEventCovariates, dataset-method
getFixedCovariates, 24
getFixedCovariates, arm-method
getFixedCovariates, arms-method
getFixedCovariates, covariates-method
getFixedCovariates, dataset-method
getIOVs, 25
getIOVs, arm-method
getIOVs, arms-method
getIOVs, dataset-method
getOccasions, 25
getOccurrences,arm-method (getOccurrences), 25
getOccurrences,arms-method (getOccurrences), 25
getOccurrences,dataset-method (getOccurrences), 25
getSeedForDatasetExport, 26
getSeedForIteration, 26
getSeedForParametersSampling, 27
getSplittingConfiguration, 27
getTimes, 28
getTimes,arm-method (getTimes), 28
getTimes,arms-method (getTimes), 28
getTimes,dataset-method (getTimes), 28
getTimes,events-method (getTimes), 28
getTimes,observations_set-method (getTimes), 28
gendTimeVaryingCovariates, 28
gendTimeVaryingCovariates,arm-method (getTimeVaryingCovariates), 28
gendTimeVaryingCovariates,arms-method (getTimeVaryingCovariates), 28
gendTimeVaryingCovariates,covariates-method (getTimeVaryingCovariates), 28
gendTimeVaryingCovariates,dataset-method (getTimeVaryingCovariates), 28

Hardware, 29
hardware_settings-class, 30

Infusion, 31
infusion-class, 32
internal_settings-class, 32
IOV, 32

length,arm-method, 33
length,dataset-method, 33
LogNormalDistribution, 34

mrgsolve_engine-class, 34

NOCB, 35
nocb_settings-class, 35
NormalDistribution, 36

Observations, 36
observations-class, 37
observations_set-class, 37
obsOnly, 37
Occasion, 38

occasion-class, 38
occasions-class, 38

ParameterDistribution, 39
PI, 39
protocol-class, 40

retrieveParameterValue, 40
rxode_engine-class, 40

sample, 41
sample,bolus,integer-method (sample), 41
sample,bootstrap,integer-method (sample), 41
sample,bootstrap_distribution,integer-method (sample), 41
sample,campsis_model,integer-method (sample), 41
sample,constant_distribution,integer-method (sample), 41
sample,covariate,integer-method (sample), 41
sample,fixed_distribution,integer-method (sample), 41
sample,function_distribution,integer-method (sample), 41
sample,infusion,integer-method (sample), 41
sample,observations,integer-method (sample), 41
scatterPlot, 42
Scenario, 42
scenario-class, 43
Scenarios, 43
scenarios-class, 43
setLabel, 44
setLabel,arm,character-method (setLabel), 44
setSubjects, 44
setSubjects,arm,integer-method (setSubjects), 44
setSubjects,dataset,integer-method (setSubjects), 44

Settings, 45
setupPlanDefault, 45
setupPlanSequential, 46
shadedPlot, 46
simulate, 47
simulate, campsis_model, data.frame, character, events, scenarios, function, character, function, integer, integer
(simulate), 47
simulate, campsis_model, dataset, character, events, scenarios, function, character, function, integer, integer, logical
(simulate), 47
simulate, campsis_model, tbl_df, character, events, scenarios, function, character, function, integer, integer, logical
(simulate), 47
simulate, campsis_model, tbl_df, mrgsolve_engine, events, scenarios, function, character, function, integer, integer, logical
(simulate), 47
simulate, campsis_model, tbl_df, rxode_engine, events, scenarios, function, character, function, integer, integer, logical
(simulate), 47
simulation_engine-class, 51
simulation_progress-class, 52
simulation_settings-class, 52
SimulationProgress, 51
Solver, 53
solver_settings-class, 53
spaghettiPlot, 54
time_varying_covariate-class, 55
TimeVaryingCovariate, 54
treatment-class, 55
treatment_iov-class, 55
treatment_iovs-class, 56
undefined_distribution-class, 56
UniformDistribution, 56
VPC, 57
vpcPlot, 57