

Package ‘campsis’

October 12, 2022

Type Package

Title Generic PK/PD Simulation Platform CAMPSIS

Version 1.3.0

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)

URL <https://github.com/Calvagone/campsis>, <https://calvagone.github.io/>

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

Depends campsismod, R (>= 4.0.0)

Imports assertthat, digest, dplyr, ggplot2, MASS, methods, plyr, progress, purrr, rlang, stats, tibble, tidy

Suggests bookdown, devtools, gridExtra, knitr, mrgsolve, pkgdown, rmarkdown, roxygen2, RxODE, rxode2, stringr, testthat

VignetteBuilder knitr

Encoding UTF-8

Language en-US

RoxygenNote 7.1.2

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'

'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' 'simulation_progress.R' 'dataset_summary.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 2022-06-01 10:20:07 UTC

R topics documented:

applyCompartmentCharacteristics	4
Arm	5
arm-class	5
arms-class	6
Bolus	6
bolus-class	7
Bootstrap	7
bootstrap-class	8
BootstrapDistribution	8
bootstrap_distribution-class	9
ConstantDistribution	9
constant_distribution-class	9
Covariate	10
covariate-class	10
covariates-class	10
Dataset	11
dataset-class	11
DatasetConfig	12
dataset_config-class	12
DiscreteDistribution	13
distribution-class	13
DoseAdaptation	13
dose_adaptation-class	14
dose_adaptations-class	14
dosingOnly	14
EtaDistribution	15
Event	15
event-class	16
EventCovariate	16
Events	17
events-class	17

event_covariate-class	17
FixedDistribution	18
fixed_covariate-class	18
fixed_distribution-class	18
FunctionDistribution	19
function_distribution-class	19
generateIIV	20
getCovariates	20
getEventCovariates	21
getFixedCovariates	21
getIOVs	22
getOccasions	23
getSeedForDatasetExport	23
getSeedForIteration	24
getSeedForParametersSampling	24
getTimes	25
getTimeVaryingCovariates	26
Infusion	26
infusion-class	27
IOV	28
length,arm-method	28
length,dataset-method	29
LogNormalDistribution	29
mrgsolve_engine-class	30
NormalDistribution	30
Observations	30
observations-class	31
observations_set-class	31
obsOnly	31
Occasion	32
occasion-class	32
occasions-class	32
ParameterDistribution	33
PI	33
protocol-class	34
retrieveParameterValue	34
rxode_engine-class	34
sample	35
Scenario	36
scenario-class	36
Scenarios	37
scenarios-class	37
setLabel	37
setSubjects	38
shadedPlot	38
simulate	39
SimulationProgress	43
simulation_engine-class	43

simulation_progress-class	44
spaghettiPlot	44
TimeVaryingCovariate	45
time_varying_covariate-class	45
treatment-class	45
treatment_iov-class	46
treatment_iovs-class	46
undefined_distribution-class	46
UniformDistribution	47
VPC	47
vpcPlot	48

Index	49
--------------	-----------

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)
```

Arguments

table	current dataset
properties	compartment properties from model

Value

updated dataset

Arm	<i>Create a treatment arm.</i>
-----	--------------------------------

Description

Create a treatment arm.

Usage

```
Arm(id = as.integer(NA), subjects = 1, label = as.character(NA))
```

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

arm-class	<i>Arm class.</i>
-----------	-------------------

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.

Description

Arms class.

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
)
```

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	<i>Bolus class.</i>
-------------	---------------------

Description

Bolus class.

Bootstrap	<i>Create a bootstrap object.</i>
-----------	-----------------------------------

Description

Create a bootstrap object.

Usage

```
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

bootstrap-class	<i>Bootstrap class.</i>
-----------------	-------------------------

Description

Bootstrap class.

Slots

data data frame to be bootstrapped. Column 'BS_ID' is mandatory and corresponds to the original row ID from the bootstrap. It must be numeric and unique. Other columns are covariates to be bootstrapped (row by row).

replacement values can be reused or not, logical

random values are drawn randomly, logical

export_id tell CAMPSIS if 'BS_ID' must be exported into the dataset, logical

BootstrapDistribution	<i>Create a bootstrap distribution. During function sampling, CAMPSIS will generate values depending on the given data and arguments.</i>
-----------------------	---

Description

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)
```

Arguments

data values to draw, numeric vector

replacement values can be reused or not, logical

random values are drawn randomly, logical

Value

a bootstrap distribution

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

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)

constant_distribution-class
Constant distribution class.

Description

Constant distribution class.

Slots

value covariate value, single numeric value

Covariate	<i>Create a non time-varying (fixed) covariate.</i>
-----------	---

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	<i>Covariate class.</i>
-----------------	-------------------------

Description

Covariate class.

Slots

name	covariate name, single character value
distribution	covariate distribution

covariates-class	<i>Covariates class.</i>
------------------	--------------------------

Description

Covariates class.

Dataset	<i>Create a dataset.</i>
---------	--------------------------

Description

Create a dataset.

Usage

```
Dataset(subjects = NULL)
```

Arguments

subjects number of subjects in the default arm

Value

a dataset

dataset-class	<i>Dataset class.</i>
---------------	-----------------------

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	<i>Create a dataset configuration. This configuration allows CAMPSIS to know which are the default depot and observed compartments.</i>
---------------	---

Description

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

Usage

```
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	<i>Dataset configuration class.</i>
----------------------	-------------------------------------

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

DiscreteDistribution *Discrete distribution.*

Description

Discrete distribution.

Usage

```
DiscreteDistribution(x, prob, replace = TRUE)
```

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

distribution-class *Distribution class. See this class as an interface.*

Description

Distribution class. See this class as an interface.

DoseAdaptation *Create a dose adaptation.*

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

dosingOnly(x)

Arguments

x data frame, CAMPSIS output

Value

a data frame with the dosing rows

EtaDistribution	<i>Create an ETA distribution. The resulting distribution is a normal distribution, with mean=0 and sd=sqrt(OMEGA).</i>
-----------------	---

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	<i>Create an interruption event.</i>
-------	--------------------------------------

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

event-class	<i>Event class.</i>
-------------	---------------------

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	<i>Create an event covariate. These covariates can be modified further in interruption events.</i>
----------------	--

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	<i>Create a list of interruption events.</i>
--------	--

Description

Create a list of interruption events.

Usage

Events()

Value

a events object

events-class	<i>Events class.</i>
--------------	----------------------

Description

Events class.

event_covariate-class	<i>Event covariate class.</i>
-----------------------	-------------------------------

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")).*

Description

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

function_distribution-class
Function distribution class.

Description

Function distribution class.

Slots

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

generateIIV	<i>Generate IIV.</i>
-------------	----------------------

Description

Generate IIV.

Usage

```
generateIIV(omega, n)
```

Arguments

omega	omega matrix
n	number of subjects

Value

IIV data frame

getCovariates	<i>Get all covariates (fixed / time-varying / event covariates).</i>
---------------	--

Description

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

Usage

```
getCovariates(object)

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

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

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

## 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

```
getEventCovariates(object)

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

## S4 method for signature 'arm'
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	<i>Get all IOV objects.</i>
---------	-----------------------------

Description

Get all IOV objects.

Usage

```

getIOVs(object)

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

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

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

```

Arguments

object any object

Value

all IOV's from object

getOccasions	<i>Get all occasions.</i>
--------------	---------------------------

Description

Get all occasions.

Usage

```
getOccasions(object)

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

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

## S4 method for signature 'dataset'
getOccasions(object)
```

Arguments

object any object

Value

all occasions from object

getSeedForDatasetExport	<i>Get seed for dataset export.</i>
-------------------------	-------------------------------------

Description

Get seed for dataset export.

Usage

```
getSeedForDatasetExport(seed, replicate, iterations)
```

Arguments

seed	original seed
replicate	the current replicate number
iterations	total number of iterations

Value

the seed value used to export the dataset

getSeedForIteration *Get seed for iteration.*

Description

Get seed for iteration.

Usage

```
getSeedForIteration(seed, replicate, iterations, iteration)
```

Arguments

seed	original seed
replicate	the current replicate number
iterations	total number of iterations
iteration	current iteration number

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

getTimes *Get all distinct times for the specified object.*

Description

Get all distinct times for the specified object.

Usage

```
getTimes(object)

## S4 method for signature 'observations_set'
getTimes(object)

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

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

## S4 method for signature 'events'
getTimes(object)

## S4 method for signature 'dataset'
getTimes(object)
```

Arguments

object any object

Value

numeric vector with all unique times, sorted

getTimeVaryingCovariates
Get all time-varying covariates.

Description

Get all time-varying covariates.

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

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	<i>Infusion class.</i>
----------------	------------------------

Description

Infusion class.

Slots

duration	infusion duration, distribution
rate	infusion rate, distribution

IOV	<i>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.</i>
-----	--

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	<i>Return the number of subjects contained in this arm.</i>
-------------------	---

Description

Return the number of subjects contained in this arm.

Usage

```
## 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

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

Arguments

x dataset

Value

a number

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.*

Description

mrgsolve engine class.

NormalDistribution *Create a normal distribution.*

Description

Create a normal distribution.

Usage

NormalDistribution(mean, sd)

Arguments

mean	mean value of distribution
sd	standard deviation of distribution

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

times	observation times, numeric vector
compartment	compartment index, integer

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	<i>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.</i>
----------	---

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	<i>Occasion class.</i>
----------------	------------------------

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	<i>Occasions class.</i>
-----------------	-------------------------

Description

Occasions class.

ParameterDistribution *Create a parameter distribution. The resulting distribution is a log-normal distribution, with meanlog=log(THETA) and sdlog=sqrt(OMEGA).*

Description

Create a parameter distribution. The resulting distribution is a log-normal distribution, with meanlog=log(THETA) and sdlog=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.*

Description

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 CAMPSIS should use rxode2 (field set to TRUE) or RxODE (field set to FALSE). Default is TRUE.

sample	<i>Sample generic object.</i>
--------	-------------------------------

Description

Sample generic object.

Usage

```
sample(object, n, ...)  
  
## S4 method for signature 'constant_distribution,integer'  
sample(object, n)  
  
## S4 method for signature 'fixed_distribution,integer'  
sample(object, n)  
  
## S4 method for signature 'function_distribution,integer'  
sample(object, n)  
  
## S4 method for signature 'bootstrap_distribution,integer'  
sample(object, n)  
  
## S4 method for signature 'bolus,integer'  
sample(object, n, ...)  
  
## S4 method for signature 'infusion,integer'  
sample(object, n, ...)  
  
## S4 method for signature 'observations,integer'  
sample(object, n, ...)  
  
## S4 method for signature 'covariate,integer'  
sample(object, n)  
  
## S4 method for signature 'bootstrap,integer'  
sample(object, n)  
  
## S4 method for signature 'campsis_model,integer'  
sample(object, n)
```

Arguments

object	generic object
n	number of samples required
...	extra arguments

Value

sampling result

Scenario	<i>Create an scenario.</i>
----------	----------------------------

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	<i>Scenario class.</i>
----------------	------------------------

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	<i>Create a list of scenarios.</i>
-----------	------------------------------------

Description

Create a list of scenarios.

Usage

```
Scenarios()
```

Value

a scenarios object

scenarios-class	<i>Scenarios class.</i>
-----------------	-------------------------

Description

Scenarios class.

setLabel	<i>Set the label.</i>
----------	-----------------------

Description

Set the label.

Usage

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

```
## S4 method for signature 'arm,character'  
setLabel(object, x)
```

Arguments

object	any object that has a label
x	the new label

Value

the updated object

setSubjects	<i>Set the number of subjects.</i>
-------------	------------------------------------

Description

Set the number of subjects.

Usage

```
setSubjects(object, x)

## S4 method for signature 'arm,integer'
setSubjects(object, x)

## S4 method for signature 'dataset,integer'
setSubjects(object, x)
```

Arguments

object	any object
x	the new number of subjects

Value

the updated object

shadedPlot	<i>Shaded plot (or prediction interval plot).</i>
------------	---

Description

Shaded plot (or prediction interval plot).

Usage

```
shadedPlot(x, output, scenarios = NULL, level = 0.9, alpha = 0.25)
```

Arguments

x	data frame
output	variable to show
scenarios	scenarios
level	PI level, default is 0.9 (90% PI)
alpha	alpha parameter (transparency) given to geom_ribbon

Value

a ggplot object

simulate	<i>Simulate function.</i>
----------	---------------------------

Description

Simulate function.

Usage

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

```
    scenarios = NULL,
    tablefun = NULL,
    outvars = NULL,
    outfun = NULL,
    seed = NULL,
    replicates = 1,
    nocb = NULL,
    dosing = FALSE,
    ...
)

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

## S4 method for signature
## 'campsis_model,
## data.frame,
## character,
## events,
## scenarios,
## `function`,
## character,
```



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

```

)

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

```

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
nocb	next-observation carried backward mode (NOCB), default value is TRUE for mrgsolve, FALSE for RxODE
dosing	output dosing information, default is FALSE
...	optional arguments like 'declare' and 'nocbvars'

Value

dataframe with all results

SimulationProgress *Create a simulation progress object.*

Description

Create a simulation progress object.

Usage

```
SimulationProgress(replicates = 1, scenarios = 1)
```

Arguments

replicates total number of replicates to simulate
scenarios total number of scenarios to simulate

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

spaghettiPlot

Spaghetti plot.

Description

Spaghetti plot.

Usage

```
spaghettiPlot(x, output, scenarios = NULL)
```

Arguments

x	data frame
output	variable to show
scenarios	scenarios

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

```
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

treatment_iovs-class *Treatment IOV's class.*

Description

Treatment IOV's class.

undefined_distribution-class

Undefined distribution class. This type of object is automatically created in method toExplicitDistribution() when the user does not provide a concrete distribution. This is because S4 objects do not accept NULL values.

Description

Undefined distribution class. This type of object is automatically created in method toExplicitDistribution() when the user does not provide a concrete distribution. This is because S4 objects do not accept NULL values.

UniformDistribution *Create an uniform distribution.*

Description

Create an uniform distribution.

Usage

```
UniformDistribution(min, max)
```

Arguments

min	min value
max	max value

Value

an uniform distribution

VPC	<i>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</i>
-----	--

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	<i>VPC plot.</i>
---------	------------------

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

Bolus, 6
bolus-class, 7
Bootstrap, 7
bootstrap-class, 8
bootstrap_distribution-class, 9
BootstrapDistribution, 8

constant_distribution-class, 9
ConstantDistribution, 9
Covariate, 10
covariate-class, 10
covariates-class, 10

Dataset, 11
dataset-class, 11
dataset_config-class, 12
DatasetConfig, 12
DiscreteDistribution, 13
distribution-class, 13
dose_adaptation-class, 14
dose_adaptations-class, 14
DoseAdaptation, 13
dosingOnly, 14

EtaDistribution, 15
Event, 15
event-class, 16
event_covariate-class, 17
EventCovariate, 16
Events, 17
events-class, 17

fixed_covariate-class, 18
fixed_distribution-class, 18
FixedDistribution, 18
function_distribution-class, 19
FunctionDistribution, 19

generateIIV, 20
getCovariates, 20
getCovariates, arm-method
(getCovariates), 20
getCovariates, arms-method
(getCovariates), 20
getCovariates, covariates-method
(getCovariates), 20
getCovariates, dataset-method
(getCovariates), 20
getEventCovariates, 21
getEventCovariates, arm-method
(getEventCovariates), 21
getEventCovariates, arms-method
(getEventCovariates), 21
getEventCovariates, covariates-method
(getEventCovariates), 21
getEventCovariates, dataset-method
(getEventCovariates), 21
getFixedCovariates, 21
getFixedCovariates, arm-method
(getFixedCovariates), 21
getFixedCovariates, arms-method
(getFixedCovariates), 21
getFixedCovariates, covariates-method
(getFixedCovariates), 21
getFixedCovariates, dataset-method
(getFixedCovariates), 21
getIOVs, 22
getIOVs, arm-method (getIOVs), 22
getIOVs, arms-method (getIOVs), 22
getIOVs, dataset-method (getIOVs), 22
getOccasions, 23
getOccasions, arm-method (getOccasions),
23
getOccasions, arms-method
(getOccasions), 23

- getOccasions, dataset-method
(getOccasions), 23
- getSeedForDatasetExport, 23
- getSeedForIteration, 24
- getSeedForParametersSampling, 24
- getTimes, 25
- getTimes, arm-method (getTimes), 25
- getTimes, arms-method (getTimes), 25
- getTimes, dataset-method (getTimes), 25
- getTimes, events-method (getTimes), 25
- getTimes, observations_set-method
(getTimes), 25
- getTimeVaryingCovariates, 26
- getTimeVaryingCovariates, arm-method
(getTimeVaryingCovariates), 26
- getTimeVaryingCovariates, arms-method
(getTimeVaryingCovariates), 26
- getTimeVaryingCovariates, covariates-method
(getTimeVaryingCovariates), 26
- getTimeVaryingCovariates, dataset-method
(getTimeVaryingCovariates), 26

- Infusion, 26
- infusion-class, 27
- IOV, 28

- length, arm-method, 28
- length, dataset-method, 29
- LogNormalDistribution, 29

- mrgsolve_engine-class, 30

- NormalDistribution, 30

- Observations, 30
- observations-class, 31
- observations_set-class, 31
- obsOnly, 31
- Occasion, 32
- occasion-class, 32
- occasions-class, 32

- ParameterDistribution, 33
- PI, 33
- protocol-class, 34

- retrieveParameterValue, 34
- rxode_engine-class, 34

- sample, 35
- sample, bolus, integer-method (sample), 35
- sample, bootstrap, integer-method
(sample), 35
- sample, bootstrap_distribution, integer-method
(sample), 35
- sample, campsis_model, integer-method
(sample), 35
- sample, constant_distribution, integer-method
(sample), 35
- sample, covariate, integer-method
(sample), 35
- sample, fixed_distribution, integer-method
(sample), 35
- sample, function_distribution, integer-method
(sample), 35
- sample, infusion, integer-method
(sample), 35
- sample, observations, integer-method
(sample), 35
- Scenario, 36
- scenario-class, 36
- Scenarios, 37
- scenarios-class, 37
- setLabel, 37
- setLabel, arm, character-method
(setLabel), 37
- setSubjects, 38
- setSubjects, arm, integer-method
(setSubjects), 38
- setSubjects, dataset, integer-method
(setSubjects), 38
- shadedPlot, 38
- simulate, 39
- simulate, campsis_model, data.frame, character, events, scenarios,
(simulate), 39
- simulate, campsis_model, dataset, character, events, scenarios,
(simulate), 39
- simulate, campsis_model, tbl_df, character, events, scenarios, f
(simulate), 39
- simulate, campsis_model, tbl_df, mrgsolve_engine, events, scena
(simulate), 39
- simulate, campsis_model, tbl_df, rxode_engine, events, scenario
(simulate), 39
- simulation_engine-class, 43
- simulation_progress-class, 44
- SimulationProgress, 43
- spaghettiPlot, 44

- time_varying_covariate-class, 45

TimeVaryingCovariate, [45](#)

treatment-class, [45](#)

treatment_iov-class, [46](#)

treatment_iovs-class, [46](#)

undefined_distribution-class, [46](#)

UniformDistribution, [47](#)

VPC, [47](#)

vpcPlot, [48](#)