Package ‘campsis’

July 1, 2024

Type Package
Title Generic PK/PD Simulation Platform CAMPSIS
Version 1.5.3
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 campsismod (>= 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, rmarkdown, roxygen2, rmarkdown, rlang, rmarkdown, rmarkdown, tidyverse, testthat, tictoc, vdiff
VignetteBuilder knitr
Encoding UTF-8
Language en-US
LazyData true
RoxygenNote 7.2.3
Collate 'global.R' 'utilities.R' 'time_utilities.R' 'check.R'
'generic.R' 'data.R' 'seed.R' 'distribution.R'
dataset_config.R' 'time_entry.R' 'occasion.R' 'occasions.R'

Contents

'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' 'dataset_summary.R' 'outfun.R'
hardware_settings.R' 'simulation_progress.R'
solver_settings.R' 'nocb_settings.R' 'declare_settings.R'
progress_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 2024-07-01 09:30:06 UTC

Contents

applyCompartmentCharacteristics ........................................... 5
Arm ................................................................. 5
arm-class ......................................................... 6
arms-class ....................................................... 6
BinomialDistribution ....................................................... 6
Bolus ................................................................. 7
bolus-class .......................................................... 7
Bootstrap ............................................................. 8
bootstrap-class ....................................................... 8
BootstrapDistribution ..................................................... 9
bootstrap_distribution-class ............................................. 9
camfis_handler ....................................................... 10
ConstantDistribution ..................................................... 10
constant_distribution-class ............................................ 10
convertTime .......................................................... 11
Covariate ............................................................ 11
covariate-class ...................................................... 12
covariates-class ...................................................... 12
Dataset ............................................................... 12
dataset-class ......................................................... 13
DatasetConfig ........................................................ 13
dataset_config-class ............................................... 14
days ................................................................. 14
Declare ............................................................... 15
declare_settings-class ................................................. 15
DiscreteDistribution ...................................................... 15
Observations ................................................................. 40
observations-class .................................................. 40
observations_set-class ............................................. 40
obsOnly ................................................................. 41
Occasion ....................................................................... 41
occasion-class .......................................................... 42
occasions-class ......................................................... 42
Outfun ....................................................................... 42
output_function-class ................................................. 43
ParameterDistribution .................................................. 43
PI .................................................................................. 44
Progress ....................................................................... 44
progress_settings-class .............................................. 45
protocol-class ........................................................... 45
retrieveParameterValue ............................................... 45
rxode_engine-class ..................................................... 46
sample ......................................................................... 46
scatterPlot ................................................................. 47
Scenario ....................................................................... 48
scenario-class ........................................................... 48
Scenarios ...................................................................... 49
scenarios-class ........................................................... 49
seconds ........................................................................ 49
setLabel ........................................................................ 50
setSubjects ................................................................. 50
Settings ........................................................................ 51
setupPlanDefault .......................................................... 51
setupPlanSequential ..................................................... 52
shadedPlot ................................................................. 52
simulate ..................................................................... 53
SimulationProgress ....................................................... 57
simulation_engine-class .............................................. 57
simulation_progress-class .......................................... 58
simulation_settings-class ............................................ 58
Solver ........................................................................... 59
solver_settings-class .................................................... 59
spaghettPlot ............................................................... 60
standardiseTime ........................................................... 60
TimeVaryingCovariate ................................................. 61
time_varying_covariate-class ..................................... 61
treatment-class .......................................................... 61
treatment_iov-class ..................................................... 62
treatment_iovs-class .................................................. 62
undefined_distribution-class ....................................... 62
UniformDistribution ..................................................... 63
VPC .............................................................................. 63
vpcPlot ........................................................................ 64
weeks .......................................................................... 64
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

Create a treatment arm.

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

Arm class.

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.

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 
)

**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

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
**Description**

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

**Usage**

```r
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
### 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**

```r
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
**convertTime**

Convert numeric time vector based on the provided units.

**Description**

Convert numeric time vector based on the provided units.

**Usage**

convertTime(x, from, to)

**Arguments**

- **x** numeric time vector
- **from** unit of x, single character value
- **to** destination unit, single character value

**Value**

numeric vector with the converted times

---

**Covariate**

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  

**Description**  
Covariate class.

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

covariates-class  

**Description**  
Covariates class.

Dataset  

**Description**  
Create a dataset.

**Usage**  

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

**Arguments**
- `subjects` number of subjects in the default arm

**Value**  
a dataset
**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.

**Description**

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,
  timeUnitDataset = "hour",
  timeUnitExport = "hour"
)
```

**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
- **timeUnitDataset** unit of time in dataset, character (’hour’ by default)
- **timeUnitExport** unit of time in export, character (’hour’ by default)

**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
time_unit_dataset  unit of time in dataset, character ('hour' by default)
time_unit_export  unit of time in export, character ('hour' by default)

days  Convert days to hours.

Description

Convert days to hours.

Usage

days(x)

Arguments

x  numeric vector in days

Value

numeric vector in hours
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

declare_settings-class

Declare settings class.

Description

Declare settings class.

Slots

variables uninitialized variables to be declared, only needed with mrgsolve

DiscreteDistribution

Discrete distribution.

Description

Discrete distribution.

Usage

DiscreteDistribution(x, prob, replace = TRUE)
DoseAdaptation

Arguments

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

Value

\- a discrete distribution

\textit{distribution-class} \quad \textit{Distribution class. See this class as an interface.}

Description

Distribution class. See this class as an interface.

\textit{DoseAdaptation} \quad \textit{Create a dose adaptation.}

Description

Create a dose adaptation.

Usage

\texttt{DoseAdaptation(formula, compartments = integer(0))}

Arguments

\- \textit{formula} \quad \text{formula to apply, single character string, e.g. "AMT*WT"}
\- \textit{compartments} \quad \text{compartments 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.

**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.

**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

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
**Event class**

*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'.

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**

Generate IIV matrix for the given Campsis model.

**Usage**

```r
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**

```r
generateIIV_(omega, n)
```

**Arguments**

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

**Value**

IIV data frame
getAvailableTimeUnits  

Return the list of available time units.

**Description**

Return the list of available time units.

**Usage**

getAvailableTimeUnits()

**Value**

character vector

getCovariates  

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

**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)

---
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)

## 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**  
*Get all occasions.*

**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**  
*Get seed for dataset export.*

**Description**  
Get seed for dataset export.

**Usage**  
getSeedForDatasetExport(seed, progress)

**Arguments**  
- seed: original seed
- progress: simulation progress
getSeedForParametersSampling

**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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataset</td>
<td>Campsis dataset to export</td>
</tr>
<tr>
<td>hardware</td>
<td>hardware configuration</td>
</tr>
</tbody>
</table>

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)

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)
getTimeVaryingCovariates

Get all time-varying covariates.

Arguments

object any object

Value

numeric vector with all unique times, sorted

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
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**

*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

**hours**

*Convert hours to hours (do nothing).*

**Description**

Convert hours to hours (do nothing).

**Usage**

`hours(x)`

**Arguments**

- `x`: numeric vector in hours

**Value**

numeric vector in hours
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

*Infusion class.*

**Description**

Infusion class.

**Slots**

- *duration* infusion duration, distribution
- *rate* infusion rate, distribution

internal_settings-class

*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

IOV

*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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>colname</td>
<td>name of the column that will be output in dataset</td>
</tr>
<tr>
<td>distribution</td>
<td>distribution</td>
</tr>
<tr>
<td>doseNumbers</td>
<td>dose numbers, if provided, IOV is generated at these doses only. By default, IOV is generated for all doses.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>arm</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>dataset</td>
</tr>
</tbody>
</table>
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

minutes  
Convert minutes to hours.

Description
Convert minutes to hours.

Usage
minutes(x)

Arguments
x  numeric vector in minutes

Value
numeric vector in hours
```r
months

Convert pharma months (1 month = 4 weeks) to hours.

Description
Convert pharma months (1 month = 4 weeks) to hours.

Usage
months(x)

Arguments
x numeric vector in months

Value
numeric vector in hours
```

```r
mrgsolve_engine-class mrgsolve engine class.

Description
mrgsolve engine class.
```

```r
nhanes

NHANES database (demographics and body measure data combined, from 2017-2018).

Description
NHANES database (demographics and body measure data combined, from 2017-2018).

Usage
nhanes
```
Format

data frame

BS_ID  Original identifier
SEX   Sex: 1 for males, 2 for females
AGE   Age in years
BW    Body weight in kg
BMI   Body mass index
HT    Height in cm

Source


NOCB

Create NOCB settings.

Description

Create NOCB settings.

Usage

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

Arguments

enable     enable/disable next-observation carried backward mode (NOCB), default value is TRUE for mrgsolve, FALSE for RxODE
variables  variable names subject to NOCB behavior (see vignette for more info)

Value

NOCB settings
NOCB settings class.

Description

NOCB settings class.

Slots

- enable: enable/disable next-observation carried backward mode (NOCB), default value is TRUE for mrgsolve, FALSE for RxODE
- variables: variable names subject to NOCB behavior (see vignette for more info)

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.

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.
Filter CAMPSIS output on observation rows.

**Usage**

```r
ox
```

**Arguments**

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

**Value**

a data frame with the observation rows

---

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**

```r
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.

Outfun

Create a new output function

Description

Create a new output function

Usage

Outfun(
  fun = function(x, ...) {
    x
  },
  args = list(),
  packages = NULL,
  level = "scenario"
)

Arguments

fun  function or purrr-style lambda formula, first argument ’x’ must be the results
args  extra arguments, named list
packages  packages that must be loaded to execute the given function, character vector
level  either ’scenario’ or ’replicate’. Default is ’scenario’.
Value

an output function

---

**output_function-class**  
*Output function class.*

---

**Description**

Output function class.

**Slots**

- **fun**  
  function or purrr-style lambda formula, first argument 'x' must be the results

- **args**  
  extra arguments, named list

- **packages**  
  packages that must be loaded to execute the given function, character vector

- **level**  
  either 'scenario' or 'replicate'. Default is 'scenario'.

---

**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**

```r
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**  
```r
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

---

**Progress**  
*Create progress settings.*

**Description**  
Create progress settings.

**Usage**  
```r
Progress(tick_slice = TRUE)
```

**Arguments**  
- `tick_slice`: tick() is called after each simulated slice, default is TRUE. In some cases, when the number of subjects per slice is low, it may be useful disable this flag, to improve performance issues.

**Value**  
progress settings
**progress_settings-class**

*Progress settings class.*

**Description**

Progress settings class.

**Slots**

*tick_slice*  
tick() is called after each simulated slice, default is TRUE. In some cases, when the number of subjects per slice is low, it may be useful disable this flag, to improve performance issues.

**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
Sample generic object.

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

---

**scatterPlot**

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

**Description**

Scatter plot (or X vs Y plot).

**Usage**

```r
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

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

---

**scenarios-class**

Scenarios class.

**Description**

Scenarios class.

---

**seconds**

Convert seconds to hours.

**Description**

Convert seconds to hours.

**Usage**

seconds(x)

**Arguments**

x numeric vector in seconds

**Value**

numeric vector in hours
setLabel  
Set the label.

Description
Set the label.

Usage
setLabel(object, x)

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>any object that has a label</td>
</tr>
<tr>
<td>x</td>
<td>the new label</td>
</tr>
</tbody>
</table>

Value
the updated object

setSubjects  
Set the number of subjects.

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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>any object</td>
</tr>
<tr>
<td>x</td>
<td>the new number of subjects</td>
</tr>
</tbody>
</table>
**Settings**

Create advanced simulation settings.

**Description**

Create advanced simulation settings.

**Usage**

```r
Settings(...) 
```

**Arguments**

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

```r
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
setupPlanSequential()

Value
nothing

shadedPlot  
Shaded plot (or prediction interval plot).

Description
Shaded plot (or prediction interval plot).

Usage
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**

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

```r
## S4 method for signature
## 'campsis_model',
## dataset,
## character,
## events,
## scenarios,
## `function`,
## character,
## output_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,
## character,
## events,
## scenarios,
## `function`,
## character,
## output_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,
## output_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,
## output_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,
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       an output function to apply on the simulation results. Type ?Outfun for more info.
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.

**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.

**Description**

Simulation settings class.

**Slots**

- **hardware**: hardware settings object
- **solver**: solver settings object
- **nocb**: NOCB settings object
- **declare**: declare settings (mrgsolve only)
- **progress**: progress settings
- **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.
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
spaghettiPlot(x, output, colour = NULL)

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

Value
plot

standardiseTime  Standardise time to hours.

Description
Standardise time to hours.

Usage
standardiseTime(x, unit)

Arguments
x numeric time vector
unit unit of x, single character value
**TimeVaryingCovariate**

**Value**

numeric vector with the times converted to hours

---

**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

---

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

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.)

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

weeks

Convert weeks to hours.

Description

Convert weeks to hours.

Usage

weeks(x)

Arguments

x numeric vector in weeks

Value

numeric vector in hours
years

Convert pharma years (1 year = 12*4 weeks) to hours.

Description
Convert pharma years (1 year = 12*4 weeks) to hours.

Usage
years(x)

Arguments
x  numeric vector in years

Value
numeric vector in hours
Index

*datasets
  nhanes, 37
applyCompartmentCharacteristics, 5
  Arm, 5
  arm-class, 6
  arms-class, 6
BinomialDistribution, 6
  Bolus, 7
  bolus-class, 7
Bootstrap, 8
  bootstrap-class, 8
  bootstrap_distribution-class, 9
BootstrapDistribution, 9
  campsis_handler, 10
  constant_distribution-class, 10
  ConstantDistribution, 10
  convertTime, 11
  Covariate, 11
  covariate-class, 12
  covariates-class, 12
Dataset, 12
  dataset-class, 13
  dataset_config-class, 14
DatasetConfig, 13
  days, 14
Declare, 15
  declare_settings-class, 15
DiscreteDistribution, 15
  distribution-class, 16
  dose_adaptation-class, 17
  dose_adaptations-class, 17
DoseAdaptation, 16
  dosingOnly, 17
EtaDistribution, 18
Event, 18
  event-class, 19
  event_covariate-class, 20
  EventCovariate, 19
  Events, 20
  events-class, 20
  fixed_covariate-class, 21
  fixed_distribution-class, 21
  FixedDistribution, 21
  function_distribution-class, 22
  FunctionDistribution, 22
  generateIIV, 23
  generateIIV_, 23
  getAvailableTimeUnits, 24
  getCovariates, 24
  getCogvariate,arm-method (getCovariates), 24
  getCogvariate,arms-method (getCovariates), 24
  getCogvariate,covariates-method (getCovariates), 24
  getCovariates,dataset-method (getCovariates), 24
  getEventCogvariate, 25
  getEventCogvariate,arm-method (getEventCogvariate), 25
  getEventCogvariate,arms-method (getEventCogvariate), 25
  getEventCogvariate,covariates-method (getEventCogvariate), 25
  getFixedCogvariate, 25
  getFixedCogvariate,arm-method (getFixedCogvariate), 25
  getFixedCogvariate,arms-method (getFixedCogvariate), 25
  getFixedCogvariate,covariates-method (getFixedCogvariate), 25
INDEX

getFixedCovariates, dataset-method (getFixedCovariates), 25
getIOVs, 26
getIOVs, arm-method (getIOVs), 26
getIOVs, arms-method (getIOVs), 26
getIOVs, dataset-method (getIOVs), 26
getOccasions, 27
getOccasions, arm-method (getOccasions), 27
getOccasions, arms-method (getOccasions), 27
getOccasions, dataset-method (getOccasions), 27
getSeedForDatasetExport, 27
getSeedForIteration, 28
getSeedForParametersSampling, 28
getSplittingConfiguration, 29
getTimes, 29
getTimes, arm-method (getTimes), 29
getTimes, arms-method (getTimes), 29
getTimes, dataset-method (getTimes), 29
getTimes, events-method (getTimes), 29
getTimes, observations_set-method (getTimes), 29
g getTimeVaryingCovariates, 30
g getTimeVaryingCovariates, arm-method (getTimeVaryingCovariates), 30
g getTimeVaryingCovariates, arms-method (getTimeVaryingCovariates), 30
g getTimeVaryingCovariates, covariates-method (getTimeVaryingCovariates), 30
g getTimeVaryingCovariates, dataset-method (getTimeVaryingCovariates), 30

Hardware, 31
hardware_settings-class, 32
hours, 32

Infusion, 33
infusion-class, 34
internal_settings-class, 34
IOV, 34

length, arm-method, 35
length, dataset-method, 35
LogNormalDistribution, 36

minutes, 36
months, 37

mrgsolve_engine-class, 37
nhanes, 37
NOCB, 38
nocb_settings-class, 39
NormalDistribution, 39

Observations, 40
observations-class, 40
observations_set-class, 40
obsOnly, 41
Occasion, 41
occasion-class, 42
occasions-class, 42
Outfun, 42
output_function-class, 43

ParameterDistribution, 43
PI, 44
Progress, 44
progress_settings-class, 45
protocol-class, 45

retrieveParameterValue, 45
rxode_engine-class, 46

sample, 46
sample, bolus, integer-method (sample), 46
sample, bootstrap, integer-method (sample), 46
sample, bootstrap_distribution, integer-method (sample), 46
sample, campsis_model, integer-method (sample), 46
sample, constant_distribution, integer-method (sample), 46
sample, covariate, integer-method (sample), 46
sample, fixed_distribution, integer-method (sample), 46
sample, function_distribution, integer-method (sample), 46
sample, infusion, integer-method (sample), 46
sample, observations, integer-method (sample), 46
scatterPlot, 47
Scenario, 48
scenario-class, 48
Scenarios, 49
scenarios-class, 49
seconds, 49
setLabel, 50
setLabel, arm, character-method
  (setLabel), 50
setSubjects, 50
setSubjects, arm, integer-method
  (setSubjects), 50
setSubjects, dataset, integer-method
  (setSubjects), 50
Settings, 51
setupPlanDefault, 51
setupPlanSequential, 52
shadedPlot, 52
simulate, 53
  simulate, campsis_model, data.frame, character, events, scenarios, function, character, output_function, integer,
  (simulate), 53
  simulate, campsis_model, dataset, character, events, scenarios, function, character, output_function, integer,
  (simulate), 53
  simulate, campsis_model, tbl_df, character, events, scenarios, function, character, output_function, integer,
  (simulate), 53
  simulate, campsis_model, tbl_df, mrgsolve_engine, events, scenarios, function, character, output_function, integer,
  (simulate), 53
  simulate, campsis_model, tbl_df, rxode_engine, events, scenarios, function, character, output_function, integer,
  (simulate), 53
simulation_engine-class, 57
simulation_progress-class, 58
simulation_settings-class, 58
SimulationProgress, 57
Solver, 59
solver_settings-class, 59
spaghettiPlot, 60
standardiseTime, 60
time_varying_covariate-class, 61
TimeVaryingCovariate, 61
treatment-class, 61
treatment_iov-class, 62
treatment_iovs-class, 62
undefined_distribution-class, 62
UniformDistribution, 63
VPC, 63
vpcPlot, 64
weeks, 64
years, 65