Package 'campsis'

October 19, 2026

Type Package

Title Generic PK/PD Simulation Platform Campsis

Version 1.9.3

Maintainer Nicolas Luyckx <nicolas.luyckx@lynxsoft.be>

Description A generic, easy-to-use and intuitive pharmacokinetic/pharmacodynamic (PK/PD) simulation platform based on the R packages 'rxode2' and 'mrgsolve'. Campsis provides an abstraction layer over the underlying processes of defining a PK/PD model, assembling a custom dataset and running a simulation. The package has a strong dependency on the R package 'campsismod', which allows models to be read from and written to files, including through a JSON-based interface, and to be adapted further on the fly in the R environment. In addition, 'campsis' allows users to assemble datasets in an intuitive manner, including via a JSON-based interface to import Campsis datasets defined using formal JSON schemas distributed with the package. Once the dataset is ready, the package prepares the simulation, calls 'rxode2' or 'mrgsolve' (at the user's choice), and returns the results for the given model, dataset and desired simulation settings. The package itself is licensed under the GPL (>= 3); the JSON schema files shipped in inst/extdata are licensed separately under the Creative Commons Attribution 4.0 International (CC BY 4.0).

License GPL (>= 3)

URL https://github.com/campsisverse/campsis, https://campsisverse.github.io/, https://campsisverse.github.io/campsis.doc/

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

Depends campsismod, R (>= 4.1.0)

Imports assertthat, digest, dplyr, furrr, future, ggplot2, jsonlite, jsonvalidate, lifecycle, MASS, methods, progressr, purrr, rlang, stats, tibble, tidyr, vctrs

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

VignetteBuilder knitr

Config/roxygen2/version 8.1.0

Encoding UTF-8

Language en-US

LazyData true

Collate 'all_classes.R' 'global.R' 'utilities.R' 'time_utilities.R' 'check.R' 'generic.R' 'data.R' 'seed.R' 'distribution.R' 'dataset_config.R' 'time_vector.R' 'time_entry.R' 'repeated_schedule.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' 'simulation_engine.R' 'dataset.R' 'scenario_action.R' 'scenario_actions.R' 'scenario.R' 'scenarios.R' 'event_logic.R' 'dataset_summary.R' 'outfun_core.R' 'outfun.R' 'outfuns.R' 'default_settings.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' 'campsis_metadata.R' 'plan_setup.R' 'json_interface.R' 'simulate_preprocess.R' 'simulate.R' 'default_plot.R' 'campsis_plot_helpers.R' 'spaghetti_plot.R' 'shaded_plot.R' 'scatter_plot.R' 'vpc_plot.R' 'deprecated_methods.R'

NeedsCompilation no

Author Nicolas Luyckx [aut, cre], Calvagone [cph]

Repository CRAN

Date/Publication 2026-10-19 13:10:13 UTC

Contents

apply_action 6
apply_compartment_characteristics 7
apply_outfun 7
Arm 8
arm-class 9
arms-class 9
as.numeric,time_sequence-method 9
as.numeric,time_vector-method 10
BinomialDistribution 10

binomial_distribution-class . . . 11
Bolus . . . 11
bolus-class . . . 12
bolus_wrapper-class . . . 12
Bootstrap . . . 12
bootstrap-class . . . 13
BootstrapDistribution . . . 13
bootstrap_distribution-class . . . 14
campsis_handler . . . 14
campsis_metadata-class . . . 14
campsis_tbl-class . . . 15
compute_pi . . . 15
compute_stats . . . 15
ConstantDistribution . . . 16
constant_distribution-class . . . 16
convert_time . . . 17
Covariate . . . 17
covariate-class . . . 18
covariates-class . . . 18
CyclicSchedule . . . 18
cyclic_schedule-class . . . 19
Dataset . . . 19
dataset-class . . . 19
DatasetConfig . . . 20
dataset_config-class . . . 20
days . . . 21
Declare . . . 21
declare_settings-class . . . 22
DefaultOutfun . . . 22
DefaultSettings . . . 22
default_outfun-class . . . 23
default_settings-class . . . 23
DiscreteDistribution . . . 24
discrete_distribution-class . . . 24
distribution-class . . . 24
DoseAdaptation . . . 25
dose_adaptation-class . . . 25
dose_adaptations-class . . . 25
DosingSchedule . . . 26
dosing_only . . . 26
dosing_schedule-class . . . 26
EtaDistribution . . . 27
Event . . . 27
event-class . . . 28
EventCovariate . . . 28
Events . . . 29
events-class . . . 29
event_covariate-class . . . 29

FixedDistribution . . . 30
fixed_covariate-class . . . 30
fixed_distribution-class . . . 30
FunctionDistribution . . . 31
function_distribution-class . . . 31
generate_iiv . . . 32
generate_iiv_ . . . 32
get_available_time_units . . . 33
get_campsis_option . . . 33
get_covariates . . . 34
get_event_covariates . . . 34
get_fixed_covariates . . . 35
get_iovs . . . 36
get_occasions . . . 36
get_splitting_configuration . . . 37
get_times . . . 37
get_time_varying_covariates . . . 38
Hardware . . . 39
hardware_settings-class . . . 40
hours . . . 41
Infusion . . . 41
infusion-class . . . 42
infusion_wrapper-class . . . 42
internal_settings-class . . . 43
IOV . . . 43
iov-class . . . 44
length,arm-method . . . 44
length,cyclic_schedule-method . . . 45
length,dataset-method . . . 45
length,repeat_at_schedule-method . . . 46
LogNormalDistribution . . . 46
lognormal_distribution-class . . . 47
minutes . . . 47
months . . . 47
mrgsolve_engine-class . . . 48
NCATableOutfun . . . 48
nca_table_outfun-class . . . 48
nhanes . . . 49
NOCB . . . 49
nocb_settings-class . . . 50
NormalDistribution . . . 50
normal_distribution-class . . . 50
Observations . . . 51
observations-class . . . 51
observations_set-class . . . 51
obs_only . . . 52
Occasion . . . 52
occasion-class . . . 53

occasions-class . . . 53
Outfun . . . 53
outfun-class . . . 54
Outfuns . . . 54
outfuns-class . . . 55
ParameterDistribution . . . 55
PIOutfun . . . 55
pi_campsis_tbl-class . . . 56
pi_outfun-class . . . 56
Progress . . . 57
progress_settings-class . . . 57
protocol-class . . . 57
RepeatAtSchedule . . . 58
repeated_schedule-class . . . 58
repeat_at_schedule-class . . . 58
repeat_schedule . . . 59
ReplaceAction . . . 59
replace_action-class . . . 60
retrieve_parameter_value . . . 60
rxode_engine-class . . . 60
sample . . . 61
scatter_plot . . . 62
scatter_plot.std_campsis_tbl . . . 62
Scenario . . . 63
scenario-class . . . 64
Scenarios . . . 64
scenarios-class . . . 64
scenario_action-class . . . 65
scenario_actions-class . . . 65
seconds . . . 65
Settings . . . 66
setup_plan_default . . . 66
setup_plan_sequential . . . 67
set_label . . . 67
set_subjects . . . 68
shaded_plot . . . 68
shaded_plot.std_campsis_tbl . . . 69
simulate . . . 70
SimulationProgress . . . 74
simulation_engine-class . . . 75
simulation_progress-class . . . 75
simulation_settings-class . . . 76
Solver . . . 76
solver_settings-class . . . 77
spaghetti_plot . . . 77
spaghetti_plot.std_campsis_tbl . . . 78
standardise_time . . . 79
StatsOutfun . . . 79

stats_campsis_tbl-class 80
stats_outfun-class 80
std_campsis_tbl-class 80
TimeSequence 81
TimeVaryingCovariate 81
TimeVector 82
time_sequence-class 82
time_varying_covariate-class 82
time_vector-class 82
treatment-class 83
treatment_iovs-class 83
undefined_distribution-class 83
undefined_schedule-class 83
UniformDistribution 84
uniform_distribution-class 84
unwrap_treatment 84
update_addl 85
update_amount 86
update_ii 87
update_repeat 88
vpc_plot 89
vpc_plot.pi_campsis_tbl 90
vpc_plot.std_campsis_tbl 91
weeks 92
years 93

Index **94**

`apply_action` *Apply some action on the given object.*

Description

Apply some action on the given object.

Usage

```
apply_action(object, action)

## S4 method for signature 'campsis_model,replace_action'
apply_action(object, action)

## S4 method for signature 'dataset,replace_action'
apply_action(object, action)
```

Arguments

`object`	any object
`action`	action to apply

Value

updated object

`apply_compartment_characteristics` *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

```
apply_compartment_characteristics(table, properties)
```

Arguments

`table`	current dataset
`properties`	compartment properties from model

Value

updated dataset

`apply_outfun` *Apply output function(s) on the given Campsis results.*

Description

Apply output function(s) on the given Campsis results.

Usage

```
apply_outfun(x, outfun, level = NULL, ...)

## S4 method for signature 'outfun'
apply_outfun(x, outfun, level = NULL, ...)

## S4 method for signature 'outfuns'
apply_outfun(x, outfun, level = NULL, ...)
```

Arguments

x	Campsis simulation results
outfun	output function(s), an ‘outfun‘ or ‘outfuns‘ object
level	level at which the output function is applied, ’replicate’ by default
...	extra arguments transmitted automatically by Campsis (e.g. ‘replicate‘ for the replicate number)

Value

the updated Campsis results

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 trough 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.

as.numeric,time_sequence-method *Time sequence to numeric vector.*

Description

Time sequence to numeric vector.

Usage

```
## S4 method for signature 'time_sequence'
as.numeric(x)
```

Arguments

`x`	time sequence object

Value

a numeric vector

as.numeric,time_vector-method *Time vector to numeric vector.*

Description

Time vector to numeric vector.

Usage

```
## S4 method for signature 'time_vector'
as.numeric(x)
```

Arguments

x	time vector object

Value

a numeric vector

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

binomial_distribution-class *Binomial distribution class.*

Description

Binomial distribution class.

Bolus *Create one or several bolus(es).*

Description

Create one or several bolus(es).

Usage

```
Bolus(
  time,
  amount,
  compartment = NULL,
  f = NULL,
  lag = NULL,
  ii = NULL,
  addl = NULL,
  wrap = TRUE,
  ref = NULL,
  rep = 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 or name to give the bolus(es). A vector of integers or names can be used for a complex model administration.
f	fraction of dose amount, list of distributions (one per compartment)
lag	dose lag time, list of distributions (one per compartment)
ii	inter-dose interval, requires argument 'time' to be a single numeric value
addl	number of additional doses, requires argument 'time' to be a single integer value
wrap	if TRUE, the bolus wrapper will be stored as is in the dataset, otherwise, it will be split into a list of boluses distinct in time. Default is TRUE.
ref	any reference name used to identify this bolus, single character value
rep	repeat the base dosing schedule several times, a 'repeated schedule' object is expected. Default is NULL (no repetition).

Value

a single bolus or a list of boluses

`bolus-class` *Bolus class.*

Description

Bolus class.

`bolus_wrapper-class` *Bolus wrapper class.*

Description

Bolus wrapper 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

`bootstrap-class` *Bootstrap class.*

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

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

`campsis_handler` *Suggested Campsis handler for showing the progress bar.*

Description

Suggested Campsis handler for showing the progress bar.

Usage

```
campsis_handler()
```

Value

a progressr handler list

`campsis_metadata-class` *Campsis metadata class.*

Description

Campsis metadata class.

Slots

`dataset` Campsis dataset object
`dest` destination simulation engine
`scenarios` Campsis scenarios object
`outvars` character vector of output variable names
`outfun` collection of output functions
`replicates` number of replicates

campsis_tbl-class *Campsis table class (see this class as an interface)*

Description

Campsis table class (see this class as an interface)

compute_pi *Compute the prediction interval summary over time.*

Description

Compute the prediction interval summary over time.

Usage

```
compute_pi(x, variable, strata = get_default_strata(), level = 0.9)
```

Arguments

x	data frame
variable	variable(s) used to compute the prediction interval, character vector.
strata	named vector with the strata to use, default is c(SCENARIO="all", ARM="all").
level	PI level, default is 0.9 (90% PI)

Value

a summary table

compute_stats *Compute generic statistics over time.*

Description

Compute generic statistics over time.

Usage

```
compute_stats(
  x,
  variable,
  strata = get_default_strata(),
  stats = c("p5", "median", "p95")
)
```

Arguments

x	data frame
variable	variable(s) used to compute the statistics, character vector.
strata	named vector with the strata to use, default is c(SCENARIO="all", ARM="all").
stats	character vector of statistics to compute. Supported: "median", "mean", or percentiles like "p5", "p95", "p2.5", "p97.5".

Value

a summary table in long format

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

convert_time *Convert numeric time vector based on the provided units.*

Description

Convert numeric time vector based on the provided units.

Usage

```
convert_time(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 *Covariate class.*

Description

Covariate class.

Slots

`name` covariate name, single character value

`distribution` covariate distribution

covariates-class *Covariates class.*

Description

Covariates class.

CyclicSchedule *Cyclic schedule constructor.*

Description

Cyclic schedule constructor.

Usage

```
CyclicSchedule(duration, repetitions)
```

Arguments

`duration`	duration of the cycle, numeric value
`repetitions`	number of additional repetitions to the base pattern, integer value

Value

a cyclic schedule

cyclic_schedule-class *Cyclic schedule class.*

Description

Cyclic schedule class.

Slots

`duration` duration of the cycle, numeric value

`repetitions` number of additional repetitions to the base pattern, integer value

Dataset *Create a dataset.*

Description

Create a dataset.

Usage

```
Dataset(subjects = NULL, label = as.character(NA), json = NULL)
```

Arguments

`subjects`	number of subjects in the default arm
`label`	label of the default arm, NA by default
`json`	path to JSON dataset file or JSON content in string form

Value

a dataset

dataset-class *Dataset class.*

Description

Dataset class.

Slots

`arms` a list of treatment arms

`config` dataset configuration for export

`iiv` data frame containing the inter-individual variability (all ETAS) for the export

`DatasetConfig` *Create a dataset configuration. This configuration allows Campsis to know which are the default depot and observed compartments.*

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,
  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

DefaultOutfun *Default output function (identity function).*

Description

Default output function (identity function).

Usage

```
DefaultOutfun()
```

Value

an output function that returns the Campsis results as is.

DefaultSettings *Create default settings.*

Description

Create default settings.

Usage

```
DefaultSettings(
  engine = NULL,
  seed = NULL,
  replicates = 1L,
  outvars = character(),
  outfuns = Outfuns(),
  disabled_variabilities = character(),
  dosing = FALSE
)
```

Arguments

engine	Simulation engine: 'rxode2' or 'mrgsolve'. Default is NULL (unspecified). If NULL, 'mrgsolve' is used first (if installed), followed by 'rxode2' (if installed).
seed	random seed number, integer (or NULL for auto-generated seed)
replicates	number of replicates, integer. Default is 1.
outvars	output variables, character vector
outfuns	output functions, outfuns object
disabled_variabilities	variabilities to disable in the simulation, character vector
dosing	output dosing information, logical

Value

default settings

---

default_outfun-class *Default output function class.*

Description

Default output function class.

---

default_settings-class *Default settings class.*

Description

Default settings class.

Slots

engine Simulation engine: 'rxode2' or 'mrgsolve'. Default is NULL (unspecified). If NULL, 'mrgsolve' is used first (if installed), followed by 'rxode2' (if installed).

seed random seed number, integer

replicates number of replicates, integer

outvars output variables, character vector

outfuns output functions, outfuns object

disabled_variabilities variabilities to disable in the simulation, character vector

dosing output dosing information, 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

discrete_distribution-class *Discrete distribution class.*

Description

Discrete distribution class.

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 = NULL)
```

Arguments

formula	formula to apply, single character string, e.g. "AMT*WT"
compartments	compartment indexes or names where the formula needs to be applied, integer or character vector. Default is NULL (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.

DosingSchedule *Dosing schedule constructor.*

Description

Dosing schedule constructor.

Usage

```
DosingSchedule()
```

Value

a dosing schedule (schedule repeated at dose times)

dosing_only *Filter Campsis output on dosing rows.*

Description

Filter Campsis output on dosing rows.

Usage

```
dosing_only(x)
```

Arguments

x	data frame, Campsis output

Value

a data frame with the dosing rows

dosing_schedule-class *Dosing schedule class.*

Description

Dosing schedule class.

`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` *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'.*

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))

generate_iiv *Generate IIV matrix for the given Campsis model.*

Description

Generate IIV matrix for the given Campsis model.

Usage

```
generate_iiv(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

generate_iiv_ *Generate IIV matrix for the given OMEGA matrix.*

Description

Generate IIV matrix for the given OMEGA matrix.

Usage

```
generate_iiv_(omega, n)
```

Arguments

omega	omega matrix
n	number of subjects

Value

IIV data frame

get_available_time_units *Return the list of available time units.*

Description

Return the list of available time units.

Usage

```
get_available_time_units()
```

Value

character vector

get_campsis_option *Get Campsis option logic.*

Description

Get Campsis option logic.

Usage

```
get_campsis_option(name, default)
```

Arguments

name	option to search
default	default value if option not found

Value

option value

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

Description

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

Usage

```
get_covariates(object)

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

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

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

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

Arguments

object	any object

Value

all covariates from object

get_event_covariates *Get all event-related covariates.*

Description

Get all event-related covariates.

Usage

```
get_event_covariates(object)

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

## S4 method for signature 'arm'
```

```
get_event_covariates(object)

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

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

Arguments

object	any object

Value

all event-related covariates from object

`get_fixed_covariates` *Get all fixed covariates.*

Description

Get all fixed covariates.

Usage

```
get_fixed_covariates(object)

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

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

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

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

Arguments

object	any object

Value

all fixed covariates from object

get_iovs *Get all IOV objects.*

Description

Get all IOV objects.

Usage

```
get_iovs(object)

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

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

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

Arguments

object	any object

Value

all IOV's from object

get_occasions *Get all occasions.*

Description

Get all occasions.

Usage

```
get_occasions(object)

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

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

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

Arguments

object	any object

Value

all occasions from object

---

get_splitting_configuration *Get splitting configuration for parallel export.*

Description

Get splitting configuration for parallel export.

Usage

```
get_splitting_configuration(dataset, hardware)
```

Arguments

dataset	Campsis dataset to export
hardware	hardware configuration

Value

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

---

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

Description

Get all distinct times for the specified object.

Usage

```
get_times(object, ...)

## S4 method for signature 'treatment'
get_times(object, unwrap = TRUE)

## S4 method for signature 'observations'
get_times(object, doseTimes = NULL)

## S4 method for signature 'observations_set'
get_times(object, doseTimes = NULL)

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

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

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

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

Arguments

object	any object
...	extra arguments like ‘doseTimes‘ in observations or ‘unwrap‘ in treatment
unwrap	unwrap treatment before accessing the times, default value is TRUE
doseTimes	times of the doses, only needed if a [DosingSchedule()] is referred to

Value

numeric vector with all unique times, sorted

---

get_time_varying_covariates

Get all time-varying covariates.

Description

Get all time-varying covariates.

Usage

```
get_time_varying_covariates(object)

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

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

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

## S4 method for signature 'dataset'
get_time_varying_covariates(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 = NULL,
  f = NULL,
  lag = NULL,
  duration = NULL,
  rate = NULL,
  ii = NULL,
  addl = NULL,
  wrap = TRUE,
  ref = NULL,
  rep = NULL
)
```

Arguments

time	treatment time(s), numeric value or vector. First treatment time if used together with ii and addl.
amount	amount to infuse, single numeric value
compartment	compartment index or name to give the infusion(s). A vector of integers or names can be used for a complex model administration.
f	fraction of infusion amount, list of distributions (one per compartment)
lag	infusion lag time, , list of distributions (one per compartment)
duration	infusion duration, list of distributions (one per compartment)
rate	infusion rate, list of distributions (one per compartment)
ii	inter-dose interval, requires argument 'time' to be a single numeric value
addl	number of additional doses, requires argument 'time' to be a single integer value
wrap	if TRUE, the infusion wrapper will be stored as is in the dataset, otherwise, it will be split into a list of infusions distinct in time. Default is TRUE.
ref	any reference name used to identify this infusion, single character value
rep	repeat the base dosing schedule several times, a 'repeated schedule' object is expected. Default is NULL (no repetition).

Value

a single infusion or a list of infusions.

---

infusion-class *Infusion class.*

Description

Infusion class.

Slots

`duration` infusion duration, distribution list

`rate` infusion rate, distribution list

---

infusion_wrapper-class *Infusion wrapper class.*

Description

Infusion wrapper class.

`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 = NULL, dose_numbers = NULL, omega_ref = NULL)
```

Arguments

`colname`	name of the column that will be output in dataset
`distribution`	distribution
`dose_numbers`	dose numbers, if provided, IOV is generated at these doses only. By default, IOV is generated for all doses.
`omega_ref`	name of the OMEGA (e.g. 'IOV' without the 'OMEGA_')

Value

an IOV object

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

`omega_ref` name of the OMEGA (e.g. 'IOV' without the 'OMEGA_')

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

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,cyclic_schedule-method

Return the number of repetition cycles.

Description

Return the number of repetition cycles.

Usage

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

Arguments

x	schedule object

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

`length,repeat_at_schedule-method`
Return the number of repetition cycles.

Description

Return the number of repetition cycles.

Usage

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

Arguments

`x`	schedule object

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

lognormal_distribution-class *Log normal distribution class.*

Description

Log normal distribution class.

minutes *Convert minutes to hours.*

Description

Convert minutes to hours.

Usage

```
minutes(x)
```

Arguments

x	numeric vector in minutes

Value

numeric vector in hours

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

mrgsolve_engine-class *mrgsolve engine class.*

Description

mrgsolve engine class.

NCATableOutfun *Create a NCA table output function*

Description

Create a NCA table output function

Usage

```
NCATableOutfun(table, export_type = "summary", name = "default_nca_table")
```

Arguments

table	NCA table from campsisnca, object, path to JSON, or JSON element
export_type	type of export, 'summary', 'summary_wide', 'summary_pretty', 'individual' or 'individual_wide'
name	name of the output function. Default is 'default_nca_table'.

Value

a stats_outfun object

nca_table_outfun-class *NCA table output function class.*

Description

NCA table output function class.

Slots

table Campsisnca table object

export_type type of export, 'summary', 'summary_wide', 'summary_pretty', 'individual' or 'individual_wide'

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

https://wwwn.cdc.gov/Nchs/Nhanes/2017-2018/DEMO_J.XPT

https://wwwn.cdc.gov/Nchs/Nhanes/2017-2018/BMX_J.XPT

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` *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

`normal_distribution-class`
Normal distribution class.

Description

Normal distribution class.

`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, rep = NULL)
```

Arguments

`times`	observation times, numeric vector
`compartment`	compartment index (integer) or name (character)
`rep`	repetition schedule

Value

an observations list

`observations-class` *Observations class.*

Description

Observations class.

Slots

`times` any object that implements

`compartment` compartment index (integer) or name (character)

`dv` observed values, numeric vector (FOR EXTERNAL USE)

`rep` repetition schedule

`observations_set-class` *Observations set class.*

Description

Observations set class.

obs_only *Filter Campsis output on observation rows.*

Description

Filter Campsis output on observation rows.

Usage

```
obs_only(x)
```

Arguments

x	data frame, Campsis output

Value

a data frame with the observation rows

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

Description

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

Usage

```
Occasion(colname, values, dose_numbers)
```

Arguments

colname	name of the column that will be output in dataset
values	the occasion numbers, any integer vector
dose_numbers	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 = "replicate",
  name = "custom"
)
```

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	where to apply the output function, only 'replicate' is allowed since Campsis v1.9.0
name	name of the output function. Default is 'custom'.

Value

an output function

---

outfun-class *Output function class.*

Description

Output function class.

Slots

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

name name of the output function

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'.

cls resulting S3 class(es) of the Campsis output

outfun_as_arg logical, whether to pass the outfun object as an argument to the function

---

Outfuns *Create a collection of output functions.*

Description

Create a collection of output functions.

Usage

```
Outfuns()
```

Value

an output_functions object

outfuns-class *Output functions class (i.e. a collection of output functions).*

Description

Output functions class (i.e. a collection of output functions).

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

PIOutfun *Create a prediction interval output function*

Description

Create a prediction interval output function

Usage

```
PIOutfun(
  variable,
  strata = get_default_strata(),
  level = 0.9,
 name = sprintf("pi_%s_%i%%", paste0(variable, collapse = "_"), round(level * 100))
)
```

Arguments

variable	variable(s) used to compute the prediction interval, character vector
strata	named vector with the strata to use, default is c(SCENARIO="all", ARM="all")
level	PI level, default is 0.9 (90% PI)
name	name of the output function. Default is 'pi_<variable>_<level>pc'.

Value

a pi_outfun object

---

pi_campsis_tbl-class *Prediction Interval (PI) Campsis table class*

Description

Prediction Interval (PI) Campsis table class

---

pi_outfun-class *Prediction interval output function class.*

Description

Prediction interval output function class.

Slots

variable variable(s) used to compute the prediction interval, character vector

strata named vector with the strata to use

pi_level PI level, default is 0.9 (90% PI)

Progress *Create progress settings.*

Description

Create progress settings.

Usage

```
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.

RepeatAtSchedule *'Repeat at' schedule constructor. Note that the time 0 for the base pattern will be added by default if not provided.*

Description

'Repeat at' schedule constructor. Note that the time 0 for the base pattern will be added by default if not provided.

Usage

```
RepeatAtSchedule(times)
```

Arguments

times	times at which the original schedule must be repeated, numeric vector

Value

a 'repeat-at' schedule

repeated_schedule-class *Repeated schedule class. See this class as an interface.*

Description

Repeated schedule class. See this class as an interface.

repeat_at_schedule-class *'Repeat at' schedule class.*

Description

'Repeat at' schedule class.

Slots

times	times at which the event is repeated, numeric vector

repeat_schedule *Repeat schedule.*

Description

Repeat schedule.

Usage

```
repeat_schedule(x, schedule)

## S4 method for signature 'numeric,cyclic_schedule'
repeat_schedule(x, schedule)

## S4 method for signature 'numeric,repeat_at_schedule'
repeat_schedule(x, schedule)

## S4 method for signature 'numeric,undefined_schedule'
repeat_schedule(x, schedule)
```

Arguments

x	object to repeat the schedule
schedule	initial times vector

Value

resulting times vector

ReplaceAction *Create a replace action.*

Description

Create a replace action.

Usage

```
ReplaceAction(object)
```

Arguments

object	replacement object

Value

a replace action

`replace_action-class` *Replace action class.*

Description

Replace action class.

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

Description

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

Usage

```
retrieve_parameter_value(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` *rxode2 engine class.*

Description

rxode2 engine class.

`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)
```

Arguments

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

Value

sampling result

scatter_plot *Scatter plot (S3 generic).*

Description

Scatter plot (S3 generic).

Usage

```
scatter_plot(x, ...)
```

Arguments

x	a Campsis output object
...	additional arguments passed to the method

Value

a ggplot object

scatter_plot.std_campsis_tbl *Scatter plot for standard Campsis simulation output.*

Description

Plots one or two variables against each other at a given time point, from a `std_campsis_tbl` (the default output of `simulate()` when no custom `outfun` is used).

Usage

```
## S3 method for class 'std_campsis_tbl'
scatter_plot(x, variable = "CONC", colour = "auto", time = NULL, ...)
```

Arguments

x	a `std_campsis_tbl` object
variable	character vector of length 1 or 2 giving the column name(s) to plot. Defaults to `"CONC"`. An informative error is raised when any column is absent.
colour	stratification for point colour. One of `"auto"`, `NULL`, or a character vector of column names. See Details.
time	numeric vector of time point(s) to filter to before plotting. When `NULL` (default) the minimum time in the data is used (typically time 0), which is useful for plotting individual parameters. Forwarded to `scatterPlot`.
...	additional arguments (currently unused, reserved for future use)

Details

When a single variable is supplied it is plotted on the x-axis with y fixed to 0, producing a 1-D strip chart. When two variables are supplied the first is mapped to x and the second to y.

Colour stratification is resolved as follows:

- `"auto"` (default): `ARM` is added when more than one distinct arm is detected in the data; `SCENARIO` is added when more than one distinct scenario is detected.
- `NULL`: no colour mapping.
- character vector: explicit column name(s) to colour by (forwarded directly to `scatterPlot`).

Value

a ggplot object

See Also

`scatterPlot`

`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

`actions` list of actions to apply

`enabled` is the scenario enabled in the simulation, logical value

Scenarios *Create a list of scenarios.*

Description

Create a list of scenarios.

Usage

```
Scenarios(json = NULL)
```

Arguments

`json`	path to JSON scenarios file or JSON content in string form

Value

a scenarios object

scenarios-class *Scenarios class.*

Description

Scenarios class.

scenario_action-class *Scenario action class.*

Description

Scenario action class.

scenario_actions-class *Scenario actions class.*

Description

Scenario actions 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

Settings — *Create advanced simulation settings.*

Description

Create advanced simulation settings.

Usage

```
Settings(..., json = NULL)
```

Arguments

...	any user-required settings: see ?Hardware, ?Solver, ?NOCB, ?Declare, ?Progress or ?AutoReplicationSettings
json	path to JSON settings file or JSON content in string form

Value

advanced simulation settings

setup_plan_default — *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

```
setup_plan_default(object)
```

Arguments

object	simulation or hardware settings

Value

nothing

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

Description

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

Usage

```
setup_plan_sequential()
```

Value

nothing

set_label *Set the label.*

Description

Set the label.

Usage

```
set_label(object, x)

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

## S4 method for signature 'dataset,character'
set_label(object, x)
```

Arguments

object	any object that has a label
x	the new label

Value

the updated object

set_subjects *Set the number of subjects.*

Description

Set the number of subjects.

Usage

```
set_subjects(object, x)

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

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

Arguments

object	any object
x	the new number of subjects

Value

the updated object

shaded_plot *Shaded plot / prediction interval plot (S3 generic).*

Description

Shaded plot / prediction interval plot (S3 generic).

Usage

```
shaded_plot(x, ...)
```

Arguments

x	a Campsis output object
...	additional arguments passed to the method

Value

a ggplot object

shaded_plot.std_campsis_tbl *Shaded plot for standard Campsis simulation output.*

Description

Plots a median line with a prediction interval ribbon from a `std_campsis_tbl` (the default output of `simulate()` when no custom `outfun` is used).

Usage

```
## S3 method for class 'std_campsis_tbl'
shaded_plot(
  x,
  variable = "CONC",
  colour = "auto",
  strat_extra = NULL,
  level = 0.9,
  alpha = 0.25,
  ...
)
```

Arguments

`x`	a `std_campsis_tbl` object
`variable`	name of the column to summarise on the y-axis. Defaults to `"CONC"`. An informative error is raised when the column is absent.
`colour`	stratification for line and ribbon colour. One of `"auto"`, `NULL`, or a character vector of column names. See Details.
`strat_extra`	additional column name(s) used for stratification in the prediction interval computation but *not* mapped to colour — useful when combined with `facet_wrap()`. Forwarded to `shadedPlot`. Default is `NULL`.
`level`	prediction interval level. Default is `0.90` (90% PI).
`alpha`	transparency of the ribbon. Default is `0.25`.
`...`	additional arguments (currently unused, reserved for future use)

Details

Colour stratification is resolved as follows:

- `"auto"` (default): `ARM` is added when more than one distinct arm is detected in the data; `SCENARIO` is added when more than one distinct scenario is detected.
- `NULL`: no colour mapping.
- character vector: explicit column name(s) to colour/fill by (forwarded directly to `shadedPlot`).

Value

a ggplot object

See Also

shadedPlot

`simulate` *Simulate function.*

Description

Simulate function.

Usage

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

## S4 method for signature
## 'replicated_campsis_model,
##   dataset,
##   character,
##   events,
##   scenarios,
##   function,
##   character,
##   outfuns,
##   integer,
##   integer,
##   logical,
##   simulation_settings'
simulate(
  model,
  dataset,
```

```
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = NULL,
  dosing = FALSE,
  settings = NULL
)

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

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

```
##   outfuns,
##   integer,
##   integer,
##   logical,
##   simulation_settings'
simulate(
  model,
  dataset,
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = NULL,
  dosing = FALSE,
  settings = NULL
)

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

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

## S4 method for signature
## 'campsis_model,
##   tbl_df,
##   mrgsolve_engine,
##   events,
##   scenarios,
##   function,
##   character,
##   outfuns,
##   integer,
##   integer,
##   logical,
##   simulation_settings'
simulate(
  model,
  dataset,
  dest = NULL,
  events = NULL,
  scenarios = NULL,
  tablefun = NULL,
```

```
  outvars = NULL,
  outfun = NULL,
  seed = NULL,
  replicates = NULL,
  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 *Simulation progress class.*

Description

Simulation progress class.

Slots

replicates total number of replicates to simulate

scenarios total number of scenarios to simulate

iterations total number of iterations to simulate

slices total number of slices to simulate

replicate current replicate number being simulated

scenario current scenario number being simulated

iteration current iteration number being simulated

slice current slice number being simulated

progressor progressr progressor

hardware hardware settings

simulation_settings-class *Simulation settings class.*

Description

Simulation settings class.

Slots

`default` default settings of the simulate method

`hardware` hardware settings object

`solver` solver settings object

`nocb` NOCB settings object

`declare` declare settings (mrgsolve only)

`progress` progress settings

`replication` replication settings

`internal` internal settings

Solver *Create solver settings.*

Description

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 rxode2 only: 'liblsoda' (default), 'lsoda', 'dop853', 'indLin'. Mrgsolve's method is always 'lsoda'.

Value

solver settings

`solver_settings-class` *Solver settings class. See ?mrgsolve::update. See ?rxode2::rxSolve.*

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 rxode2 only: 'liblsoda' (default), 'lsoda', 'dop853', 'indLin'. Mrgsolve's method is always 'lsoda'.

`spaghetti_plot` *Spaghetti plot (S3 generic).*

Description

Spaghetti plot (S3 generic).

Usage

```
spaghetti_plot(x, ...)
```

Arguments

`x`	a Campsis output object
`...`	additional arguments passed to the method

Value

a ggplot object

`spaghetti_plot.std_campsis_tbl` *Spaghetti plot for standard Campsis simulation output.*

Description

Plots individual-level time profiles from a `std_campsis_tbl` (the default output of `simulate()` when no custom `outfun` is used).

Usage

```
## S3 method for class 'std_campsis_tbl'
spaghetti_plot(x, variable = "CONC", colour = "auto", ...)
```

Arguments

`x`	a `std_campsis_tbl` object
`variable`	name of the column to plot on the y-axis. Defaults to `"CONC"`. An informative error is raised when the column is absent.
`colour`	stratification for line colour. One of `"auto"`, `NULL`, or a character vector of column names. See Details.
`...`	additional arguments (currently unused, reserved for future use)

Details

Colour stratification is resolved as follows:

- `"auto"` (default): `ARM` is added when more than one distinct arm is detected in the data; `SCENARIO` is added when more than one distinct scenario is detected.
- `NULL`: no colour mapping.
- character vector: explicit column name(s) to colour by (forwarded directly to `spaghettiPlot`).

Value

a ggplot object

See Also

`spaghettiPlot`

standardise_time *Standardise time to hours.*

Description

Standardise time to hours.

Usage

```
standardise_time(x, unit)
```

Arguments

x	numeric time vector
unit	unit of x, single character value

Value

numeric vector with the times converted to hours

StatsOutfun *Create a statistics output function*

Description

Create a statistics output function

Usage

```
StatsOutfun(
  variable,
  strata = get_default_strata(),
  stats = c("p5", "median", "p95"),
  name = sprintf("stats_%s", paste0(variable, collapse = "_"))
)
```

Arguments

variable	variable(s) used to compute the statistics, character vector
strata	named vector with the strata to use, default is c(SCENARIO="all", ARM="all")
stats	character vector of statistics to compute. Supported: "median", "mean", or percentiles like "p5", "p95", "p2.5", "p97.5". Default is c("p5", "median", "p95").
name	name of the output function. Default is 'stats_<variable>'.

Value

a stats_outfun object

stats_campsis_tbl-class *Statistics Campsis table class*

Description

Statistics Campsis table class

stats_outfun-class *Statistics output function class.*

Description

Statistics output function class.

Slots

`variable` variable(s) used to compute the statistics, character vector

`strata` named vector with the strata to use

`stats` character vector of statistics to compute

std_campsis_tbl-class *Standard Campsis table class*

Description

Standard Campsis table class

`TimeSequence` *Instantiate a new time sequence.*

Description

Instantiate a new time sequence.

Usage

```
TimeSequence(start, end, by)
```

Arguments

`start`	starting value
`end`	maximal value
`by`	increment of the sequence

Value

a sequence

`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

TimeVector *Instantiate a new time vector*

Description

Instantiate a new time vector

Usage

```
TimeVector(x)
```

Arguments

x	time vector, numeric

Value

a time vector

time_sequence-class *Time sequence class.*

Description

Time sequence class.

time_varying_covariate-class *Time-varying covariate class.*

Description

Time-varying covariate class.

time_vector-class *Time vector class.*

Description

Time vector class.

`treatment-class` *Treatment class.*

Description

Treatment class.

`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 to_explicit_distribution() 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 to_explicit_distribution() when the user does not provide a concrete distribution. This is because S4 objects do not accept NULL values.

`undefined_schedule-class`
Undefined schedule class.

Description

Undefined schedule class.

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

uniform_distribution-class *Uniform distribution class.*

Description

Uniform distribution class.

unwrap_treatment *Unwrap treatment.*

Description

Unwrap treatment.

Usage

```
unwrap_treatment(object)

## S4 method for signature 'bolus'
unwrap_treatment(object)

## S4 method for signature 'infusion'
unwrap_treatment(object)

## S4 method for signature 'bolus_wrapper'
unwrap_treatment(object)

## S4 method for signature 'infusion_wrapper'
unwrap_treatment(object)

## S4 method for signature 'treatment'
unwrap_treatment(object)

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

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

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

Arguments

object	any object

Value

updated object

---

update_addl *Update the number of additional doses (ADDL).*

Description

Update the number of additional doses (ADDL).

Usage

```
update_addl(object, addl, ref = NULL)

## S4 method for signature 'bolus_wrapper,integer,character'
```

```
update_addl(object, addl, ref = NULL)

## S4 method for signature 'infusion_wrapper,integer,character'
update_addl(object, addl, ref = NULL)

## S4 method for signature 'bolus,integer,character'
update_addl(object, addl, ref = NULL)

## S4 method for signature 'infusion,integer,character'
update_addl(object, addl, ref = NULL)

## S4 method for signature 'treatment,integer,character'
update_addl(object, addl, ref = NULL)

## S4 method for signature 'arm,integer,character'
update_addl(object, addl, ref = NULL)

## S4 method for signature 'arms,integer,character'
update_addl(object, addl, ref = NULL)

## S4 method for signature 'dataset,integer,character'
update_addl(object, addl, ref = NULL)
```

Arguments

object	generic object
addl	new number of additional doses
ref	reference treatment name

Value

updated object

`update_amount` *Update amount.*

Description

Update amount.

Usage

```
update_amount(object, amount, ref)

## S4 method for signature 'bolus,numeric,character'
update_amount(object, amount, ref)
```

```
## S4 method for signature 'infusion,numeric,character'
update_amount(object, amount, ref)

## S4 method for signature 'bolus_wrapper,numeric,character'
update_amount(object, amount, ref)

## S4 method for signature 'infusion_wrapper,numeric,character'
update_amount(object, amount, ref)

## S4 method for signature 'treatment,numeric,character'
update_amount(object, amount, ref)

## S4 method for signature 'arm,numeric,character'
update_amount(object, amount, ref)

## S4 method for signature 'arms,numeric,character'
update_amount(object, amount, ref)

## S4 method for signature 'dataset,numeric,character'
update_amount(object, amount, ref)
```

Arguments

object	generic object
amount	new amount
ref	reference treatment name

Value

updated object

`update_ii` *Update the inter-dose interval (II).*

Description

Update the inter-dose interval (II).

Usage

```
update_ii(object, ii, ref = NULL)

## S4 method for signature 'bolus_wrapper,numeric,character'
update_ii(object, ii, ref = NULL)

## S4 method for signature 'infusion_wrapper,numeric,character'
update_ii(object, ii, ref = NULL)
```

```
## S4 method for signature 'bolus,numeric,character'
update_ii(object, ii, ref = NULL)

## S4 method for signature 'infusion,numeric,character'
update_ii(object, ii, ref = NULL)

## S4 method for signature 'treatment,numeric,character'
update_ii(object, ii, ref = NULL)

## S4 method for signature 'arm,numeric,character'
update_ii(object, ii, ref = NULL)

## S4 method for signature 'arms,numeric,character'
update_ii(object, ii, ref = NULL)

## S4 method for signature 'dataset,numeric,character'
update_ii(object, ii, ref = NULL)
```

Arguments

object	generic object
ii	new inter-dose interval
ref	reference treatment name

Value

updated object

`update_repeat` *Update the repeat field (argument 'rep' in Bolus and Infusion constructors).*

Description

Update the repeat field (argument 'rep' in Bolus and Infusion constructors).

Usage

```
update_repeat(object, rep, ref = NULL)

## S4 method for signature 'bolus_wrapper,repeated_schedule,character'
update_repeat(object, rep, ref = NULL)

## S4 method for signature 'infusion_wrapper,repeated_schedule,character'
update_repeat(object, rep, ref = NULL)
```

```
## S4 method for signature 'bolus,repeated_schedule,character'
update_repeat(object, rep, ref = NULL)

## S4 method for signature 'infusion,repeated_schedule,character'
update_repeat(object, rep, ref = NULL)

## S4 method for signature 'treatment,repeated_schedule,character'
update_repeat(object, rep, ref = NULL)

## S4 method for signature 'arm,repeated_schedule,character'
update_repeat(object, rep, ref = NULL)

## S4 method for signature 'arms,repeated_schedule,character'
update_repeat(object, rep, ref = NULL)

## S4 method for signature 'dataset,repeated_schedule,character'
update_repeat(object, rep, ref = NULL)
```

Arguments

object	generic object
rep	repeated dosing schedule (definition) object
ref	reference treatment name

Value

updated object

`vpc_plot` *VPC plot (S3 generic).*

Description

VPC plot (S3 generic).

Usage

```
vpc_plot(x, ...)
```

Arguments

x	a Campsis output object
...	additional arguments passed to the method

Value

a ggplot object

`vpc_plot.pi_campsis_tbl` *VPC plot for prediction-interval Campsis output.*

Description

Plots VPC ribbons (a confidence interval around the median, lower and upper percentiles) from a `pi_campsis_tbl`, i.e. the output of `simulate()` with multiple replicates and a `PIOutfun`. Such data already contains the per-replicate prediction interval in long format (columns `replicate`, `TIME`, `metric` with values `"low"` / `"med"` / `"up"` and `value`). This method only needs to compute the confidence interval around each percentile across replicates.

Usage

```
## S3 method for class 'pi_campsis_tbl'
vpc_plot(x, strata = "auto", level = 0.9, alpha = 0.15, facet = TRUE, ...)
```

Arguments

`x`	a `pi_campsis_tbl` object with a `replicate` column
`strata`	stratification for the VPC. One of `"auto"`, `NULL`, or a named character vector of length 1. See Details.
`level`	confidence interval level for the ribbons around each percentile. Default is `0.90` (90% CI).
`alpha`	transparency of the ribbons. Default is `0.15`.
`facet`	how the stratification variable is displayed when present. `TRUE` (default) draws one panel per stratum; `FALSE` overlays the strata in a single panel and maps the stratum to the ribbon fill colour. Forwarded to `vpcPlot`.
`...`	additional arguments (currently unused, reserved for future use)

Details

Stratification is resolved as follows:

- `"auto"` (default): `ARM` is used when more than one distinct arm is detected; otherwise `SCENARIO` is used when more than one distinct scenario is detected; otherwise `NULL` (no stratification). Only one stratification variable is applied because `vpcPlot` supports at most one.
- `NULL`: no stratification.
- named character vector: passed directly to `vpcPlot` (e.g. `c(ARM = "all")` or `c(SCENARIO = "all")`).

Value

a ggplot object

See Also

vpcPlot, PIOutfun

---

vpc_plot.std_campsis_tbl *VPC plot for standard Campsis simulation output.*

Description

Plots VPC ribbons (a confidence interval around the median, lower and upper percentiles) from a std_campsis_tbl, i.e. the default output of simulate() with multiple replicates and no custom outfun (DefaultOutfun). In this case the data contains all unsummarised individual profiles, so the per-replicate prediction interval is computed first (the lower/median/upper percentiles of variable across individuals within each replicate), before the confidence interval around each percentile is computed across replicates.

Usage

```
## S3 method for class 'std_campsis_tbl'
vpc_plot(
  x,
  variable = "CONC",
  strata = "auto",
  pi_level = 0.9,
  ci_level = 0.9,
  alpha = 0.15,
  facet = TRUE,
  ...
)
```

Arguments

x	a std_campsis_tbl object with a replicate column
variable	name of the column to summarise on the y-axis. Defaults to "CONC". An informative error is raised when the column is absent.
strata	stratification for the VPC. One of "auto", NULL, or a named character vector of length 1. See Details.
pi_level	prediction interval level, i.e. the percentiles of the data computed within each replicate. Default is 0.90 (90% PI, i.e. the 5th, 50th and 95th percentiles).
ci_level	confidence interval level for the ribbons around each percentile across replicates. Default is 0.90 (90% CI).
alpha	transparency of the ribbons. Default is 0.15.
facet	how the stratification variable is displayed when present. TRUE (default) draws one panel per stratum; FALSE overlays the strata in a single panel and maps the stratum to the ribbon fill colour. Forwarded to vpcPlot.
...	additional arguments (currently unused, reserved for future use)

Details

Stratification is resolved as follows:

- `"auto"` (default): `ARM` is used when more than one distinct arm is detected; otherwise `SCENARIO` is used when more than one distinct scenario is detected; otherwise `NULL` (no stratification). Only one stratification variable is applied because `vpcPlot` supports at most one.
- `NULL`: no stratification.
- named character vector: passed directly to `vpcPlot` (e.g. `c(ARM = "all")` or `c(SCENARIO = "all")`).

Value

a ggplot object

See Also

`vpcPlot`, `vpc_plot.pi_campsis_tbl`

---

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, 49

apply_action, 6
apply_action,campsis_model,replace_action-method *(apply_action)*, 6
apply_action,dataset,replace_action-method *(apply_action)*, 6
apply_compartment_characteristics, 7
apply_outfun, 7
apply_outfun,outfun-method *(apply_outfun)*, 7
apply_outfun,outfuns-method *(apply_outfun)*, 7
Arm, 8
arm-class, 9
arms-class, 9
as.numeric,time_sequence-method, 9
as.numeric,time_vector-method, 10

binomial_distribution-class, 11
BinomialDistribution, 10
Bolus, 11
bolus-class, 12
bolus_wrapper-class, 12
Bootstrap, 12
bootstrap-class, 13
bootstrap_distribution-class, 14
BootstrapDistribution, 13

campsis_handler, 14
campsis_metadata-class, 14
campsis_tbl *(campsis_tbl-class)*, 15
campsis_tbl-class, 15
compute_pi, 15
compute_stats, 15
constant_distribution-class, 16
ConstantDistribution, 16
convert_time, 17
Covariate, 17
covariate-class, 18
covariates-class, 18
cyclic_schedule-class, 19
CyclicSchedule, 18

Dataset, 19
dataset-class, 19
dataset_config-class, 20
DatasetConfig, 20
days, 21
Declare, 21
declare_settings-class, 22
default_outfun-class, 23
default_settings-class, 23
DefaultOutfun, 22, *91*
DefaultSettings, 22
discrete_distribution-class, 24
DiscreteDistribution, 24
distribution-class, 24
dose_adaptation-class, 25
dose_adaptations-class, 25
DoseAdaptation, 25
dosing_only, 26
dosing_schedule-class, 26
DosingSchedule, 26

EtaDistribution, 27
Event, 27
event-class, 28
event_covariate-class, 29
EventCovariate, 28
Events, 29
events-class, 29

fixed_covariate-class, 30
fixed_distribution-class, 30
FixedDistribution, 30
function_distribution-class, 31
FunctionDistribution, 31

generate_iiv, 32

generate_iiv_, 32
get_available_time_units, 33
get_campsis_option, 33
get_covariates, 34
get_covariates,arm-method
(get_covariates), 34
get_covariates,arms-method
(get_covariates), 34
get_covariates,covariates-method
(get_covariates), 34
get_covariates,dataset-method
(get_covariates), 34
get_event_covariates, 34
get_event_covariates,arm-method
(get_event_covariates), 34
get_event_covariates,arms-method
(get_event_covariates), 34
get_event_covariates,covariates-method
(get_event_covariates), 34
get_event_covariates,dataset-method
(get_event_covariates), 34
get_fixed_covariates, 35
get_fixed_covariates,arm-method
(get_fixed_covariates), 35
get_fixed_covariates,arms-method
(get_fixed_covariates), 35
get_fixed_covariates,covariates-method
(get_fixed_covariates), 35
get_fixed_covariates,dataset-method
(get_fixed_covariates), 35
get_iovs, 36
get_iovs,arm-method (get_iovs), 36
get_iovs,arms-method (get_iovs), 36
get_iovs,dataset-method (get_iovs), 36
get_occasions, 36
get_occasions,arm-method
(get_occasions), 36
get_occasions,arms-method
(get_occasions), 36
get_occasions,dataset-method
(get_occasions), 36
get_splitting_configuration, 37
get_time_varying_covariates, 38
get_time_varying_covariates,arm-method
(get_time_varying_covariates),
38
get_time_varying_covariates,arms-method
(get_time_varying_covariates),
38
get_time_varying_covariates,covariates-method
(get_time_varying_covariates),
38
get_time_varying_covariates,dataset-method
(get_time_varying_covariates),
38
get_times, 37
get_times,arm-method (get_times), 37
get_times,arms-method (get_times), 37
get_times,dataset-method (get_times), 37
get_times,events-method (get_times), 37
get_times,observations-method
(get_times), 37
get_times,observations_set-method
(get_times), 37
get_times,treatment-method (get_times),
37

Hardware, 39
hardware_settings-class, 40
hours, 41

Infusion, 41
infusion-class, 42
infusion_wrapper-class, 42
internal_settings-class, 43
IOV, 43
iov-class, 44

length,arm-method, 44
length,cyclic_schedule-method, 45
length,dataset-method, 45
length,repeat_at_schedule-method, 46
lognormal_distribution-class, 47
LogNormalDistribution, 46

minutes, 47
months, 47
mrgsolve_engine-class, 48

nca_table_outfun-class, 48
NCATableOutfun, 48
nhanes, 49
NOCB, 49
nocb_settings-class, 50
normal_distribution-class, 50
NormalDistribution, 50

obs_only, 52

Observations, 51
observations-class, 51
observations_set-class, 51
Occasion, 52
occasion-class, 53
occasions-class, 53
Outfun, 53
outfun-class, 54
Outfuns, 54
outfuns-class, 55

ParameterDistribution, 55
pi_campsis_tbl *(pi_campsis_tbl-class)*, 56
pi_campsis_tbl-class, 56
pi_outfun-class, 56
PIOutfun, 55, *90, 91*
Progress, 57
progress_settings-class, 57
protocol-class, 57

repeat_at_schedule-class, 58
repeat_schedule, 59
repeat_schedule,numeric,cyclic_schedule-method *(repeat_schedule)*, 59
repeat_schedule,numeric,repeat_at_schedule-method *(repeat_schedule)*, 59
repeat_schedule,numeric,undefined_schedule-method *(repeat_schedule)*, 59
RepeatAtSchedule, 58
repeated_schedule-class, 58
replace_action-class, 60
ReplaceAction, 59
retrieve_parameter_value, 60
rxode_engine-class, 60

sample, 61
sample,bolus,integer-method *(sample)*, 61
sample,bootstrap,integer-method *(sample)*, 61
sample,bootstrap_distribution,integer-method *(sample)*, 61
sample,constant_distribution,integer-method *(sample)*, 61
sample,covariate,integer-method *(sample)*, 61
sample,fixed_distribution,integer-method *(sample)*, 61
sample,function_distribution,integer-method *(sample)*, 61
sample,infusion,integer-method *(sample)*, 61
sample,observations,integer-method *(sample)*, 61
scatter_plot, 62
scatter_plot.std_campsis_tbl, 62
scatterPlot, *62, 63*
Scenario, 63
scenario-class, 64
scenario_action-class, 65
scenario_actions-class, 65
Scenarios, 64
scenarios-class, 64
seconds, 65
set_label, 67
set_label,arm,character-method *(set_label)*, 67
set_label,dataset,character-method *(set_label)*, 67
set_subjects, 68
set_subjects,arm,integer-method *(set_subjects)*, 68
set_subjects,dataset,integer-method *(set_subjects)*, 68
Settings, 66
setup_plan_default, 66
setup_plan_sequential, 67
shaded_plot, 68
shaded_plot.std_campsis_tbl, 69
shadedPlot, *69, 70*
simulate, 70
simulate,campsis_model,data.frame,character,events,scenari *(simulate)*, 70
simulate,campsis_model,dataset,character,events,scenarios, *(simulate)*, 70
simulate,campsis_model,tbl_df,character,events,scenarios,f *(simulate)*, 70
simulate,campsis_model,tbl_df,mrgsolve_engine,events,scena *(simulate)*, 70
simulate,campsis_model,tbl_df,rxode_engine,events,scenario *(simulate)*, 70
simulate,replicated_campsis_model,dataset,character,events *(simulate)*, 70
simulation_engine-class, 75
simulation_progress-class, 75
simulation_settings-class, 76

SimulationProgress, 74
Solver, 76
solver_settings-class, 77
spaghetti_plot, 77
spaghetti_plot.std_campsis_tbl, 78
spaghettiPlot, *78*
standardise_time, 79
stats_campsis_tbl
    *(*stats_campsis_tbl-class*)*, 80
stats_campsis_tbl-class, 80
stats_outfun-class, 80
StatsOutfun, 79
std_campsis_tbl
    *(*std_campsis_tbl-class*)*, 80
std_campsis_tbl-class, 80

time_sequence-class, 82
time_varying_covariate-class, 82
time_vector-class, 82
TimeSequence, 81
TimeVaryingCovariate, 81
TimeVector, 82
treatment-class, 83
treatment_iovs-class, 83

undefined_distribution-class, 83
undefined_schedule-class, 83
uniform_distribution-class, 84
UniformDistribution, 84
unwrap_treatment, 84
unwrap_treatment,arm-method
    *(*unwrap_treatment*)*, 84
unwrap_treatment,arms-method
    *(*unwrap_treatment*)*, 84
unwrap_treatment,bolus-method
    *(*unwrap_treatment*)*, 84
unwrap_treatment,bolus_wrapper-method
    *(*unwrap_treatment*)*, 84
unwrap_treatment,dataset-method
    *(*unwrap_treatment*)*, 84
unwrap_treatment,infusion-method
    *(*unwrap_treatment*)*, 84
unwrap_treatment,infusion_wrapper-method
    *(*unwrap_treatment*)*, 84
unwrap_treatment,treatment-method
    *(*unwrap_treatment*)*, 84
update_addl, 85
update_addl,arm,integer,character-method
    *(*update_addl*)*, 85
update_addl,arms,integer,character-method
    *(*update_addl*)*, 85
update_addl,bolus,integer,character-method
    *(*update_addl*)*, 85
update_addl,bolus_wrapper,integer,character-method
    *(*update_addl*)*, 85
update_addl,dataset,integer,character-method
    *(*update_addl*)*, 85
update_addl,infusion,integer,character-method
    *(*update_addl*)*, 85
update_addl,infusion_wrapper,integer,character-method
    *(*update_addl*)*, 85
update_addl,treatment,integer,character-method
    *(*update_addl*)*, 85
update_amount, 86
update_amount,arm,numeric,character-method
    *(*update_amount*)*, 86
update_amount,arms,numeric,character-method
    *(*update_amount*)*, 86
update_amount,bolus,numeric,character-method
    *(*update_amount*)*, 86
update_amount,bolus_wrapper,numeric,character-method
    *(*update_amount*)*, 86
update_amount,dataset,numeric,character-method
    *(*update_amount*)*, 86
update_amount,infusion,numeric,character-method
    *(*update_amount*)*, 86
update_amount,infusion_wrapper,numeric,character-method
    *(*update_amount*)*, 86
update_amount,treatment,numeric,character-method
    *(*update_amount*)*, 86
update_ii, 87
update_ii,arm,numeric,character-method
    *(*update_ii*)*, 87
update_ii,arms,numeric,character-method
    *(*update_ii*)*, 87
update_ii,bolus,numeric,character-method
    *(*update_ii*)*, 87
update_ii,bolus_wrapper,numeric,character-method
    *(*update_ii*)*, 87
update_ii,dataset,numeric,character-method
    *(*update_ii*)*, 87
update_ii,infusion,numeric,character-method
    *(*update_ii*)*, 87
update_ii,infusion_wrapper,numeric,character-method
    *(*update_ii*)*, 87
update_ii,treatment,numeric,character-method
    *(*update_ii*)*, 87

update_repeat, 88
update_repeat,arm,repeated_schedule,character-method
(update_repeat), 88
update_repeat,arms,repeated_schedule,character-method
(update_repeat), 88
update_repeat,bolus,repeated_schedule,character-method
(update_repeat), 88
update_repeat,bolus_wrapper,repeated_schedule,character-method
(update_repeat), 88
update_repeat,dataset,repeated_schedule,character-method
(update_repeat), 88
update_repeat,infusion,repeated_schedule,character-method
(update_repeat), 88
update_repeat,infusion_wrapper,repeated_schedule,character-method
(update_repeat), 88
update_repeat,treatment,repeated_schedule,character-method
(update_repeat), 88

vpc_plot, 89
vpc_plot.pi_campsis_tbl, 90, *92*
vpc_plot.std_campsis_tbl, 91
vpcPlot, *90–92*

weeks, 92

years, 93