Package ‘campsismod’

October 12, 2022

Type Package

Title Generic Implementation of a PK/PD Model

Version 0.9.0

Description A generic, easy-to-use and expandable implementation of a pharmacokinetic (PK) / pharmacodynamic (PD) model based on the S4 class system. This package allows the user to read/write a pharmacometric model from/to files and adapt it further on the fly in the R environment. For this purpose, this package provides an intuitive API to add, modify or delete equations, ordinary differential equations (ODE’s), model parameters or compartment properties (like infusion duration or rate, bioavailability and initial values). Finally, this package also provides a useful export of the model for use with simulation packages 'RxODE' and 'mrgsolve'. This package is designed and intended to be used with package 'campsis', a PK/PD simulation platform built on top of 'RxODE' and 'mrgsolve'.

License GPL (>= 3)

URL https://github.com/Calvagone/campsismod,
    https://calvagone.github.io/

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

Depends R (>= 4.0.0)

Imports assertthat, dplyr, magrittr, methods, plyr, purrr, readr, tibble, utils

Suggests devtools, knitr, pkgdown, rmarkdown, roxygen2, testthat, xfun

VignetteBuilder knitr

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.1.2
Collate 'global.R' 'data.R' 'utilities.R' 'special_operators.R'
 'check.R' 'generic.R' 'generic_element_list.R'
 'generic_element_position.R' 'generic_list.R' 'pattern.R'
 'model_statement.R' 'model_statements.R'
 'model_unknown_statement.R' 'model_comment.R'
 'model_line_break.R' 'model_equation.R' 'model_ode.R'
 'model_if_statement.R' 'compartment.R' 'compartment_property.R'
 'compartment_properties.R' 'compartment_bioavailability.R'
 'compartment_lag_time.R' 'compartment_infusion_duration.R'
 'compartment_infusion_rate.R' 'compartment_initial_condition.R'
 'compartments.R' 'parameter.R' 'parameters.R' 'code_record.R'
 'code_records.R' 'model_parser.R' 'campsis_model.R'
 'rxode_model.R' 'rxode_conversion.R' 'mrgsolve_model.R'
 'mrgsolve_conversion.R'

NeedsCompilation no

Author Nicolas Luyckx [aut, cre]

Maintainer Nicolas Luyckx <nicolas.luyckx@calvagone.com>

Repository CRAN

Date/Publication 2022-06-17 12:50:02 UTC

R topics documented:

add ................................................................. 5
as.data.frame .................................................. 7
assertSingleCharacterString .................................. 7
autoDetectNONMEM .............................................. 8
Bioavailability .................................................. 8
CampsisModel .................................................. 9
campsis_model-class ............................................. 9
CodeRecords .................................................... 10
code_record-class .............................................. 10
Comment ........................................................ 10
comment-class ................................................... 11
Compartments ................................................... 12
compartment-class .............................................. 11
Compartments ................................................... 12
compartments-class .......................................... 12
compartment_bioavailability-class ........................... 12
compartment_infusion_duration-class ......................... 12
compartment_infusion_rate-class ............................ 13
compartment_initial_condition-class ......................... 13
compartment_lag_time-class ................................... 13
compartment_properties-class ............................... 13
compartment_property-class .................................. 14
contains ......................................................... 14
default .......................................................... 15
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>delete</td>
<td>15</td>
</tr>
<tr>
<td>disable</td>
<td>16</td>
</tr>
<tr>
<td>double_array_parameter-class</td>
<td>17</td>
</tr>
<tr>
<td>duration_record-class</td>
<td>17</td>
</tr>
<tr>
<td>Equation</td>
<td>17</td>
</tr>
<tr>
<td>equation-class</td>
<td>18</td>
</tr>
<tr>
<td>ErrorRecord</td>
<td>18</td>
</tr>
<tr>
<td>error_record-class</td>
<td>18</td>
</tr>
<tr>
<td>export</td>
<td>19</td>
</tr>
<tr>
<td>export_type-class</td>
<td>19</td>
</tr>
<tr>
<td>extractLhs</td>
<td>20</td>
</tr>
<tr>
<td>extractRhs</td>
<td>20</td>
</tr>
<tr>
<td>extractTextBetweenBrackets</td>
<td>21</td>
</tr>
<tr>
<td>find</td>
<td>21</td>
</tr>
<tr>
<td>fixOmega</td>
<td>22</td>
</tr>
<tr>
<td>f_record-class</td>
<td>23</td>
</tr>
<tr>
<td>getByIndex</td>
<td>23</td>
</tr>
<tr>
<td>getNameInModel</td>
<td>27</td>
</tr>
<tr>
<td>getNames</td>
<td>27</td>
</tr>
<tr>
<td>getNONMEMName</td>
<td>28</td>
</tr>
<tr>
<td>getPrefix</td>
<td>28</td>
</tr>
<tr>
<td>getRecordDelimiter</td>
<td>29</td>
</tr>
<tr>
<td>getRecordName</td>
<td>30</td>
</tr>
<tr>
<td>getUncertainty</td>
<td>30</td>
</tr>
<tr>
<td>getVarCov</td>
<td>31</td>
</tr>
<tr>
<td>hasComment</td>
<td>32</td>
</tr>
<tr>
<td>IfStatement</td>
<td>32</td>
</tr>
<tr>
<td>indexOf</td>
<td>33</td>
</tr>
<tr>
<td>InfusionDuration</td>
<td>33</td>
</tr>
<tr>
<td>InfusionRate</td>
<td>34</td>
</tr>
<tr>
<td>InitialCondition</td>
<td>34</td>
</tr>
<tr>
<td>init_record-class</td>
<td>35</td>
</tr>
<tr>
<td>isComment</td>
<td>35</td>
</tr>
<tr>
<td>isDiag</td>
<td>36</td>
</tr>
<tr>
<td>isEmptyLine</td>
<td>36</td>
</tr>
<tr>
<td>isEquation</td>
<td>37</td>
</tr>
<tr>
<td>isIfStatement</td>
<td>37</td>
</tr>
<tr>
<td>isODE</td>
<td>38</td>
</tr>
<tr>
<td>isRecordDelimiter</td>
<td>38</td>
</tr>
<tr>
<td>isStrictRecordDelimiter</td>
<td>39</td>
</tr>
<tr>
<td>LagTime</td>
<td>39</td>
</tr>
<tr>
<td>lag_record-class</td>
<td>40</td>
</tr>
<tr>
<td>length_pmx_list-method</td>
<td>40</td>
</tr>
<tr>
<td>LineBreak</td>
<td>40</td>
</tr>
</tbody>
</table>
line_break-class ................................................. 41
MainRecord ......................................................... 41
main_record-class .............................................. 41
maxIndex .......................................................... 42
minIndex .......................................................... 42
ModelStatements .................................................. 43
model_library ..................................................... 43
model_statement-class ........................................... 44
model_statements-class ......................................... 44
model_suite ....................................................... 45
mrgsolveBlock .................................................... 45
mrgsolveCapture .................................................. 46
mrgsolveCompartment ........................................... 46
mrgsolveMain ...................................................... 47
mrgsolveMatrix .................................................... 47
mrgsolveOde ......................................................... 48
mrgsolveParam ..................................................... 48
mrgsolveTable ...................................................... 49
mrgsolve_type-class ............................................. 49
Ode ................................................................. 49
ode-class .......................................................... 50
OdeRecord .......................................................... 50
ode_record-class .................................................. 50
Omega ............................................................... 51
omega-class ........................................................ 51
parameter-class .................................................... 52
Parameters .......................................................... 52
parameters-class .................................................... 52
parseIfStatement ................................................... 53
parseStatements .................................................... 53
Pattern .............................................................. 54
pattern-class ........................................................ 54
pmx_element-class ................................................ 54
pmx_list-class ...................................................... 54
pmx_position-class ................................................. 55
pmx_position_by_element-class ................................. 55
pmx_position_by_index-class .................................... 55
Position .............................................................. 55
processExtraArg .................................................... 56
properties_record-class ......................................... 56
rate_record-class ................................................... 56
read ................................................................. 57
read.allparameters ............................................... 57
read.campsis ....................................................... 58
read.model .......................................................... 58
read.parameters .................................................... 59
read.varcov ........................................................ 59
replace .............................................................. 60
add

Add element to list.

Description
Add element to list.

Usage
add(object, x, ...)

## S4 method for signature 'pmx_list,pmx_element'
add(object, x, pos = NULL)

## S4 method for signature 'pmx_list,pmx_list'
add(object, x)

## S4 method for signature 'pmx_list,list'
add(object, x)

## S4 method for signature 'compartments,compartment_property'
add(object, x)
add(object, x)
## S4 method for signature 'compartments,compartments'
add(object, x)

## S4 method for signature 'parameters,single_array_parameter'
add(object, x)

## S4 method for signature 'parameters,double_array_parameter'
add(object, x)

## S4 method for signature 'parameters,parameters'
add(object, x)

## S4 method for signature 'code_record,model_statement'
add(object, x, pos = NULL)

## S4 method for signature 'code_record,code_record'
add(object, x)

## S4 method for signature 'code_records,code_records'
add(object, x)

## S4 method for signature 'code_records,model_statement'
add(object, x, pos = NULL)

## S4 method for signature 'campsis_model,compartment_property'
add(object, x)

## S4 method for signature 'campsis_model,parameter'
add(object, x)

## S4 method for signature 'campsis_model,code_record'
add(object, x)

## S4 method for signature 'campsis_model,model_statement'
add(object, x, pos = NULL)

## S4 method for signature 'campsis_model,campsis_model'
add(object, x)

### Arguments

- **object**: list object
- **x**: element to add
- **...**: extra arguments, unused by this generic list
- **pos**: position where x needs to be added in list
as.data.frame

Value
modified list object

Description
As data frame method.

Usage
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S4 method for signature 'theta,character,logical'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S4 method for signature 'omega,character,logical'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S4 method for signature 'sigma,character,logical'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

Arguments
x generic object
row.names row names
optional optional
... extra arguments

Value
data frame

assertSingleCharacterString

assertSingleCharacterString(x)

Description
Assert the given character vector is a single character string.

Usage
assertSingleCharacterString(x)
Bioavailability

Arguments

- x single character string

Value

- no return value

---

**autoDetectNONMEM**

Auto-detect special variables from NONMEM as compartment properties. Bioavailabilities, infusion durations/rates and lag times will be automatically detected.

---

Description

Auto-detect special variables from NONMEM as compartment properties. Bioavailabilities, infusion durations/rates and lag times will be automatically detected.

Usage

autoDetectNONMEM(object, ...)

## S4 method for signature 'campsis_model'

classifyModel(autoDetectNONMEM(object, ...))

Arguments

- object object that has NONMEM special variables to be identified
- ... extra arguments, unused

Value

- updated object

---

**Bioavailability**

Create a bioavailability for the specified compartment.

Description

Create a bioavailability for the specified compartment.

Usage

Bioavailability(compartment, rhs = "")
Arguments

- compartment: compartment index
- rhs: right-hand side part of the equation

Value

a bioavailability property

Create a new CAMPSIS model.

Description

Create a new CAMPSIS model.

Usage

CampsisModel()

Value

a CAMPSIS model, empty

campsis_model-class  CAMPSIS model class.

Description

CAMPSIS model class.

Slots

- model: a list of code records
- parameters: model parameters
- compartments: model compartments
CodeRecords

Create a list of code records.

Description
Create a list of code records.

Usage
CodeRecords()

Value
an empty list of code records

code_record-class

Code record class. See this code record as an abstract class. 2 implementations are possible: - properties record (lag, duration, rate & bioavailability properties) - statements record (main, ode & error records)

Description
Code record class. See this code record as an abstract class. 2 implementations are possible: - properties record (lag, duration, rate & bioavailability properties) - statements record (main, ode & error records)

Slots
comment a comment, single character value
statements model statements

Comment
Create a new comment.

Description
Create a new comment.

Usage
Comment(x)
**comment-class**

**Arguments**
- x: comment, single character string

**Value**
- a comment

---

**Description**
Comment class. A statement starting with #.

---

**Compartment**
Create a compartment.

---

**Description**
Create a compartment.

**Usage**
Compartment(index, name = NA)

**Arguments**
- index: compartment index
- name: compartment name (without prefix)

**Value**
- an empty list of compartments

---

**compartment-class**
Compartment class.

---

**Description**
Compartment class.

**Slots**
- name: compartment name (without prefix)
- index: compartment index
Compartments

Create a list of compartments

Description

Create a list of compartments

Usage

Compartments()

Value

an empty list of compartments

compartments-class

Compartments class.

Description

Compartments class.

Slots

properties  compartment properties of the compartments defined in this class

compartment_bioavailability-class

Compartment bioavailability class.

Description

Compartment bioavailability class.

compartment_infusion_duration-class

Compartment infusion duration class.

Description

Compartment infusion duration class.
compartment_infusion_rate-class

Compartment infusion rate class.

Description

Compartment infusion rate class.

compartment_initial_condition-class

Compartment initial condition class.

Description

Compartment initial condition class.

compartment_lag_time-class

Compartment lag time class.

Description

Compartment lag time class.

compartment_properties-class

Compartment properties class.

Description

Compartment properties class.
contains

compartment_property-class

Compartment property class.

Description

Compartment property class.

Slots

compartment  related compartment index
rhs  right-hand side formula
comment  comment if any, single character string

contains

Check if an element exists in list.

Description

Check if an element exists in list.

Usage

contains(object, x)

## S4 method for signature 'pmx_list,pmx_element'
contains(object, x)

## S4 method for signature 'statements_record,model_statement'
contains(object, x)

## S4 method for signature 'campsis_model,pmx_element'
contains(object, x)

Arguments

object  list object
x  element to check if exists

Value

logical value, TRUE or FALSE
default

Get default element from list.

Description
Get default element from list.

Usage
default(object, ...)

Arguments
object list object
... additional arguments

Value
the default element from list

delete
Delete an element from this list.

Description
Delete an element from this list.

Usage
delete(object, x)

## S4 method for signature 'pmx_list,pmx_element'
delete(object, x)

## S4 method for signature 'pmx_list,integer'
delete(object, x)

## S4 method for signature 'compartments,compartment_property'
delete(object, x)

## S4 method for signature 'parameters,single_array_parameter'
delete(object, x)

## S4 method for signature 'parameters,double_array_parameter'
delete(object, x)
disable

## S4 method for signature 'statements_record,model_statement'
delete(object, x)

## S4 method for signature 'statements_record,integer'
delete(object, x)

## S4 method for signature 'code_records,model_statement'
delete(object, x)

## S4 method for signature 'campsis_model,compartment_property'
delete(object, x)

## S4 method for signature 'campsis_model,parameter'
delete(object, x)

## S4 method for signature 'campsis_model,code_record'
delete(object, x)

## S4 method for signature 'campsis_model,model_statement'
delete(object, x)

### Arguments

- **object**: list object
- **x**: element to delete or element index

### Value

- the updated list

---

disable  
*Disable.*

### Description

Disable.

### Usage

disable(object, x, ...)

## S4 method for signature 'parameters,character'
disable(object, x, ...)

## S4 method for signature 'campsis_model,character'
disable(object, x, ...)
**double_array_parameter-class**

**Arguments**

- **object**: generic object
- **x**: what needs to be disabled
- **...**: extra arguments needed for disabling

**Value**

- object with some disabled features

---

**double_array_parameter-class**

Double-array parameter class. This parameter has 2 indexes. It can thus be used to define correlations between parameters.

---

**duration_record-class**

(Infusion)-duration record class.

---

**Description**

(Infusion)-duration record class.

---

**Equation**

Create a new equation.

---

**Description**

Create a new equation.

**Usage**

Equation(lhs, rhs = "", comment = as.character(NA))

**Arguments**

- **lhs**: left-hand side variable corresponding to the assigned variable name
- **rhs**: right-hand side expression corresponding to a formula
- **comment**: comment if any, single character string

**Value**

- an equation
equation-class

*Equation class. Any statement in the form $A = B$.*

**Description**

Equation class. Any statement in the form $A = B$.

**Slots**

- **lhs** left-hand side expression
- **rhs** right-hand side expression

---

ErrorRecord

*Create ERROR code record.*

**Description**

Create ERROR code record.

**Usage**

`ErrorRecord(code = character())`

**Arguments**

- **code** code record

**Value**

an ERROR code record

---

error_record-class

*Error record class.*

**Description**

Error record class.
Export function.

Usage

```r
export(object, dest, ...)  
## S4 method for signature 'campsis_model,character'
export(object, dest, outvars = NULL)

## S4 method for signature 'campsis_model,rxode_type'
export(object, dest, ...)

## S4 method for signature 'campsis_model,mrgsolve_type'
export(object, dest, outvars = NULL)
```

Arguments

- `object`: generic object
- `dest`: destination
- `...`: optional arguments
- `outvars`: additional variables to capture

Value

- specific object depending on given destination

Export type-class

Description

Export type class.
**extractLhs**

*Extract left-hand-side expression.*

**Description**

Extract left-hand-side expression.

**Usage**

\[\text{extractLhs}(x, \text{split} = "\=")\]

**Arguments**

- **x**: character value
- **split**: character where to split

**Value**

left-hand-side expression, not trimmed

---

**extractRhs**

*Extract right-hand-side expression.*

**Description**

Extract right-hand-side expression.

**Usage**

\[\text{extractRhs}(x, \text{split} = "\=")\]

**Arguments**

- **x**: character value
- **split**: character where to split

**Value**

right-hand side expression
extractTextBetweenBrackets

Extract text between brackets.

Description
Extract text between brackets.

Usage
extractTextBetweenBrackets(x)

Arguments
x character value

Value
text between brackets (trimmed)

find
Find an element in list.

Description
Find an element in list.

Usage
find(object, x)

## S4 method for signature 'pmx_list,pmx_element'
find(object, x)

## S4 method for signature 'compartments,compartment_property'
find(object, x)

## S4 method for signature 'statements_record,model_statement'
find(object, x)

## S4 method for signature 'code_records,model_statement'
find(object, x)

## S4 method for signature 'campsis_model,compartment'
find(object, x)
Fix omega matrix for SAME OMEGA parameters that have NA values due to imperfections in Pharmpy import.

## S4 method for signature 'campsis_model,compartment_property'
find(object, x)

## S4 method for signature 'campsis_model,parameter'
find(object, x)

## S4 method for signature 'campsis_model,code_record'
find(object, x)

## S4 method for signature 'campsis_model,model_statement'
find(object, x)

### Arguments

- **object**: list object
- **x**: element to find, only key slots need to be filled in

### Value

the element from the list that has same name as x, or NULL if no element was found

---

fixOmega

*Fix omega matrix for SAME OMEGA parameters that have NA values due to imperfections in Pharmpy import.*

### Description

Fix omega matrix for SAME OMEGA parameters that have NA values due to imperfections in Pharmpy import.

### Usage

```r
fixOmega(object)
```

### Arguments

- **object**: generic object

### Value

the parameter that matches
Bioavailability record class.

getByIndex

Get element by index.

Description

Get element by index.

Usage

gByIndex(object, x)

## S4 method for signature 'pmx_list, integer'
gByIndex(object, x)

## S4 method for signature 'pmx_list, numeric'
gByIndex(object, x)

## S4 method for signature 'compartment_properties, compartment_property'
gByIndex(object, x)

## S4 method for signature 'parameters, parameter'
gByIndex(object, x)

Arguments

object: list object

x: element index

Value

element from the list whose index matches with provided index
**getDescription**

Get an element from a list by name. Never return more than 1 element.

**Usage**

getDescription(object, name)

## S4 method for signature 'pmx_list,character'

description(object, name)

**Arguments**

<table>
<thead>
<tr>
<th>object</th>
<th>list object</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>element name to search for</td>
</tr>
</tbody>
</table>

**Value**

the element that was found or NULL if no element was found with the same name

---

**getCompartmentIndex**

Get the compartment index for the specified compartment name.

**Usage**

getCompartmentIndex(object, name)

## S4 method for signature 'compartments,character'

generateCompartmentIndex(object, name)

## S4 method for signature 'campsis_model,character'

generateCompartmentIndex(object, name)

**Arguments**

<table>
<thead>
<tr>
<th>object</th>
<th>generic object that contains compartments information</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>compartment name</td>
</tr>
</tbody>
</table>
**getValue**

the corresponding compartment index

---

**Description**

Get element name.

**Usage**

```r
getName(x)
```

## S4 method for signature 'unknown_statement'
```r
getName(x)
```

## S4 method for signature 'comment'
```r
getName(x)
```

## S4 method for signature 'line_break'
```r
getName(x)
```

## S4 method for signature 'equation'
```r
getName(x)
```

## S4 method for signature 'ode'
```r
getName(x)
```

## S4 method for signature 'if_statement'
```r
getName(x)
```

## S4 method for signature 'compartment'
```r
getName(x)
```

## S4 method for signature 'compartment_bioavailability'
```r
getName(x)
```

## S4 method for signature 'compartment_lag_time'
```r
getName(x)
```

## S4 method for signature 'compartment_infusion_duration'
```r
getName(x)
```

## S4 method for signature 'compartment_infusion_rate'
```r
getName(x)
```
## S4 method for signature 'compartment_initial_condition'
getName(x)

## S4 method for signature 'theta'
getName(x)

## S4 method for signature 'omega'
getName(x)

## S4 method for signature 'sigma'
getName(x)

## S4 method for signature 'main_record'
getName(x)

## S4 method for signature 'ode_record'
getName(x)

## S4 method for signature 'f_record'
getName(x)

## S4 method for signature 'lag_record'
getName(x)

## S4 method for signature 'duration_record'
getName(x)

## S4 method for signature 'rate_record'
getName(x)

## S4 method for signature 'init_record'
getName(x)

## S4 method for signature 'error_record'
getName(x)

### Arguments

- **x**: element to know the name

### Value

the name of this element
get\text{NameInModel} \quad \text{Get the name of the given parameter in the CAMPSIS model.}

\textbf{Description}

Get the name of the given parameter in the CAMPSIS model.

\textbf{Usage}

\begin{verbatim}
get\text{NameInModel}(x)

## S4 method for signature 'theta'
get\text{NameInModel}(x)

## S4 method for signature 'omega'
get\text{NameInModel}(x)

## S4 method for signature 'sigma'
get\text{NameInModel}(x)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
\item \texttt{x} \quad \text{element to know the name}
\end{itemize}

\textbf{Value}

the name of this parameter

\textbf{get\text{Names} \quad Get element names from list.}

\textbf{Description}

Get element names from list.

\textbf{Usage}

\begin{verbatim}
get\text{Names}(\text{object})

## S4 method for signature 'pmx\_list'
get\text{Names}(\text{object})
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
\item \texttt{object} \quad \text{list object}
\end{itemize}
Value

class character vector with all the element names of this list

getNONMEMName

Get NONMEM name.

Description

Get NONMEM name.

Usage

getNONMEMName(object)

## S4 method for signature 'theta'
getNONMEMName(object)

## S4 method for signature 'omega'
getNONMEMName(object)

## S4 method for signature 'sigma'
getNONMEMName(object)

Arguments

object generic object

Value

the NONMEM name associated with this object

getPrefix

Get prefix.

Description

Get prefix.
getRecordDelimiter

Usage

getPrefix(object, ...)

## S4 method for signature 'compartment_bioavailability'
getPrefix(object, ...)

## S4 method for signature 'compartment_lag_time'
getPrefix(object, ...)

## S4 method for signature 'compartment_infusion_duration'
getPrefix(object, ...)

## S4 method for signature 'compartment_infusion_rate'
getPrefix(object, ...)

## S4 method for signature 'compartment_initial_condition'
getPrefix(object, ...)

Arguments

object  generic object
...
  e.g. dest='mrgsolve'

Value

the prefix of this object

getRecordDelimiter

Get record delimiter.

Description

Get record delimiter.

Usage

getRecordDelimiter(line)

Arguments

line  any line, single character value

Value

the record delimiter between brackets
getRecordName  Get record name.

Description
Get record name.

Usage
getRecordName(object)

## S4 method for signature 'compartment_bioavailability'
getRecordName(object)

## S4 method for signature 'compartment_lag_time'
getRecordName(object)

## S4 method for signature 'compartment_infusion_duration'
getRecordName(object)

## S4 method for signature 'compartment_infusion_rate'
getRecordName(object)

## S4 method for signature 'compartment_initial_condition'
getRecordName(object)

Arguments

object    generic object

Value
the name of the record

generate uncertainty  Get uncertainty on the parameters.

Description
Get uncertainty on the parameters.
getVarCov

Usage

getUncertainty(object, ...)

## S4 method for signature 'parameter'
getUncertainty(object, varcov, ...)

## S4 method for signature 'parameters'
getUncertainty(object, ...)

## S4 method for signature 'campsis_model'
getUncertainty(object, ...)

Arguments

- object: generic object
- ...: extra arguments
- varcov: variance covariance matrix

Value

data frame with standard error (se) and relative standard error (rse)

ggetVarCov

Get variance-covariance matrix.

Description

Get variance-covariance matrix.

Usage

ggetVarCov(object)

## S4 method for signature 'parameters'
ggetVarCov(object)

## S4 method for signature 'campsis_model'
ggetVarCov(object)

Arguments

- object: generic object

Value

- a variance-covariance matrix (data frame) or NULL if no matrix present
hasComment  

Check if string contains CAMPSIS-style comments.

Description
Check if string contains CAMPSIS-style comments.

Usage
hasComment(x)

Arguments
x  
character vector

Value
logical value

IfStatement  

Create a new IF-statement.

Description
Create a new IF-statement.

Usage
IfStatement(condition, equation, comment = as.character(NA))

Arguments

ccondition  
condition, single character string
equation  
equation if condition is met
ccomment  
comment if any, single character string

Value
an IF-statement
**if_statement-class**

*If-statement class. Any statement in the form if (condition) A = B.*

**Description**

If-statement class. Any statement in the form if (condition) A = B.

**Slots**

- `condition`: IF statement condition
- `equation`: any equation or ODE

**indexOf**

*Get the index of an element in list.*

**Description**

Get the index of an element in list.

**Usage**

```r
indexOf(object, x)
```

## S4 method for signature 'pmx_list,pmx_element'

```r
indexOf(object, x)
```

**Arguments**

- `object`: list object
- `x`: element to know the index

**Value**

index of this element
InfusionDuration Create an infusion duration.

Description
Create an infusion duration.

Usage
InfusionDuration(compartment, rhs = "")

Arguments
- compartment: compartment index
- rhs: right-hand side part of the equation

Value
an infusion duration property

InfusionRate Create an infusion rate.

Description
Create an infusion rate.

Usage
InfusionRate(compartment, rhs = "")

Arguments
- compartment: compartment index
- rhs: right-hand side part of the equation

Value
an infusion rate property
**InitialCondition**

*Create an initial condition.*

**Description**

Create an initial condition.

**Usage**

```r
InitialCondition(compartment, rhs = "")
```

**Arguments**

- **compartment**: compartment index
- **rhs**: right-hand side part of the equation

**Value**

an initial condition property

---

**init_record-class**

*Init record class.*

**Description**

Init record class.

---

**isComment**

*Check if string is a CAMPSIS comment (i.e. not an equation).*

**Description**

Check if string is a CAMPSIS comment (i.e. not an equation).

**Usage**

```r
isComment(x)
```

**Arguments**

- **x**: character vector

**Value**

logical value
isDiag

Is diagonal.

Description
Is diagonal.

Usage

isDiag(object)

## S4 method for signature 'double_array_parameter'
isDiag(object)

Arguments
object generic object

Value
logical value

isEmptyLine

Check if string is an empty line.

Description
Check if string is an empty line.

Usage

isEmptyLine(x)

Arguments
x character vector

Value
logical value
**isEquation**

*Say if line in record is an equation not.*

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Say if line in record is an equation not.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>isEquation(x)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>logical value</td>
</tr>
</tbody>
</table>

**isIfStatement**

*Say if line in record is an IF-statement.*

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Say if line in record is an IF-statement.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>isIfStatement(x)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>logical value</td>
</tr>
</tbody>
</table>
isODE

Say if line(s) in record is/are ODE or not.

Description
Say if line(s) in record is/are ODE or not.

Usage
isODE(x)

Arguments
x character vector

Value
logical vector

isRecordDelimiter

Is record delimiter. A record delimiter is any line starting with [...].

Description
Is record delimiter. A record delimiter is any line starting with [...].

Usage
isRecordDelimiter(line)

Arguments
line any line, single character value

Value
a logical value
isStrictRecordDelimiter

*isStrictRecordDelimiter*  
*Is strict record delimiter.* A strict record delimiter is any line starting with [...] and followed by nothing but spaces or a possible comment.

**Description**

Is strict record delimiter. A strict record delimiter is any line starting with [...] and followed by nothing but spaces or a possible comment.

**Usage**

isStrictRecordDelimiter(line)

**Arguments**

- line: any line, single character value

**Value**

a logical value

---

LagTime

Create a lag time for the specified compartment.

**Description**

Create a lag time for the specified compartment.

**Usage**

LagTime(compartment, rhs = "")

**Arguments**

- compartment: compartment index
- rhs: right-hand side part of the equation

**Value**

a lag time property
lag_record-class

Lag record class.

Description

Lag record class.

length,pmx_list-method

Return the length of this list.

Description

Return the length of this list.

Usage

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

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

Arguments

- `x` : list object

Value

the length of this list, integer value

LineBreak

Create a new line break.

Description

Create a new line break.

Usage

```
LineBreak()
```

Value

a line break
Description

Line-break class. A linebreak in the model.

MainRecord

Create MAIN code record.

Description

Create MAIN code record.

Usage

MainRecord(code = character())

Arguments

code code record

Description

Main record class.
maxIndex

**Max index.**

**Description**
Max index.

**Usage**
```r
maxIndex(object)
```

## S4 method for signature 'parameters'
maxIndex(object)

**Arguments**
- `object` generic object

**Value**
max index

minIndex

**Min index.**

**Description**
Min index.

**Usage**
```r
minIndex(object)
```

## S4 method for signature 'parameters'
minIndex(object)

**Arguments**
- `object` generic object

**Value**
min index
ModelStatements

Create an empty list of model statements.

Description
Create an empty list of model statements.

Usage
ModelStatements()

Value
a model statements object

model_library
CAMPSIS model library.

Description
A list containing pharmacokinetic (PK) and pharmacodynamic (PD) model templates.

Usage
model_library

Format
A list with all the models:

- **advan1_trans1**: 1-compartment PK model (K,V)
- **advan1_trans2**: 1-compartment PK model (CL,V)
- **advan2_trans1**: 1-compartment PK model with absorption compartment (KA,K,V)
- **advan2_trans2**: 1-compartment PK model with absorption compartment (KA,CL,V)
- **advan3_trans1**: 2-compartment PK model (K,V,K12,K21)
- **advan3_trans2**: 2-compartment PK model (CL,V,Q,VSS)
- **advan3_trans3**: 2-compartment PK model (CL,V1,V2,Q)
- **advan3_trans4**: 2-compartment PK model with absorption compartment (KA,K,V,K12,K21)
- **advan4_trans3**: 2-compartment PK model with absorption compartment (KA,CL,V,Q,VSS)
- **advan4_trans4**: 2-compartment PK model with absorption compartment (KA,CL,V1,V2,Q)
- **advan11_trans4**: 3-compartment PK model (CL,V1,V2,V3,Q2,Q3)
- **advan12_trans4**: 3-compartment PK model with absorption compartment (KA,CL,V1,V2,V3,Q2,Q3)
direct_effect_model  direct effect PD model (EMAX,EC50,GAMMA,E0)
effect_cmt_model  effect compartment PD model (KE0)
irm_kin_inhibition  indirect response PD model - inhibition of KIN (IC50,KIN,KOUT)
irm_kin_stimulation  indirect response PD model - stimulation of KIN (EMAX,EC50,KIN,KOUT)
irm_kout_inhibition  indirect response PD model - inhibition of KOUT (IC50,KIN,KOUT)
irm_kout_stimulation  indirect response PD model - stimulation of KOUT (EMAX,EC50,KIN,KOUT)
transit_cmt_model  transit compartment PD model (BASE,POWER,MTT,SLOPE,KTR)
filgrastim_pkpd_krzyzanski  PK/PD model of filgrastim (Krzyzanski et al., see URL below)
my_model1  Example of 2-compartment PK model with variance-covariance matrix

Source

https://www.iconplc.com/innovation/nonmem/
http://repository.ddmore.eu/model/DDMODEL00000077/
https://calvagone.github.io/campsis.doc/

---

**model_statement-class**  *Model statement class. Any statement in a code record.*

---

**Description**

Model statement class. Any statement in a code record.

---

**Slots**

comment  a comment associated to this model statement

---

**model_statements-class**  *Model statements class. A list of statements.*

---

**Description**

Model statements class. A list of statements.
**model_suite**

**Description**

A library of models of all kinds, ready to be simulated in Campsis. These model templates are sorted into the following categories: pharmacokinetic (PK), pharmacodynamic (PD), target-mediated drug disposition (TMDD), NONMEM, literature and other (custom models).

**Usage**

```r
model_suite
```

**Format**

A list with all the models:

- **pk**  extensive list of pharmacokinetic (PK) model templates
- **pd**  list of pharmacodynamic (PD) model templates, to be plugged into any pharmacokinetic (PK) model
- **tmdd**  extensive list of target-mediated drug disposition (TMDD) model templates
- **nonmem**  list of model templates translated from standard NONMEM control streams
- **literature**  a couple of models coming from the literature
- **other**  a couple of custom models

**Source**

- [https://calvagone.github.io/campsis.doc/](https://calvagone.github.io/campsis.doc/)
- [https://www.iconplc.com/innovation/nonmem/](https://www.iconplc.com/innovation/nonmem/)
- [http://repository.ddmore.eu/model/DDMODEL00000077/](http://repository.ddmore.eu/model/DDMODEL00000077/)

---

**mrgsolveBlock**

**Convert code record for mrgsolve.**

**Description**

Convert code record for mrgsolve.

**Usage**

```r
mrgsolveBlock(record, init = NULL, capture = FALSE)
```
**Arguments**

- `record` code record
- `init` name of mrgsolve block
- `capture` 'capture' instead of 'double'

**Value**

translated record for mrgsolve

---

**mrgsolveCapture** *Get the CAPTURE block for mrgsolve.*

**Description**

Get the CAPTURE block for mrgsolve.

**Usage**

`mrgsolveCapture(outvars, model)`

**Arguments**

- `outvars` outvars in method simulate
- `model` CAMPSIS model

**Value**

CAPTURE block or character(0) if no variable in outvars

---

**mrgsolveCompartment** *Get the compartment block for mrgsolve.*

**Description**

Get the compartment block for mrgsolve.

**Usage**

`mrgsolveCompartment(model)`

**Arguments**

- `model` CAMPSIS model

**Value**

character vector, each value is a line
### mrgsolveMain

Get the MAIN block for mrgsolve.

**Description**

Get the MAIN block for mrgsolve.

**Usage**

```r
mrgsolveMain(model)
```

**Arguments**

- `model`: CAMPSIS model

**Value**

MAIN block

### mrgsolveMatrix

Get the OMEGA/SIGMA matrix for mrgsolve.

**Description**

Get the OMEGA/SIGMA matrix for mrgsolve.

**Usage**

```r
mrgsolveMatrix(model, type = "omega")
```

**Arguments**

- `model`: CAMPSIS model
- `type`: either omega or sigma

**Value**

- named matrix or character(0) if matrix is empty
**mrgsolveOde**  
*Get the ODE block for mrgsolve.*

**Description**
Get the ODE block for mrgsolve.

**Usage**
mrgsolveOde(model)

**Arguments**

- **model**  
  CAMPSIS model

**Value**
ODE block

---

**mrgsolveParam**  
*Get the parameters block for mrgsolve.*

**Description**
Get the parameters block for mrgsolve.

**Usage**
mrgsolveParam(model)

**Arguments**

- **model**  
  CAMPSIS model

**Value**
character vector, 1 parameter per line. First one is header [PARAM].
mrgsolveTable

Get the TABLE block for mrgsolve.

Description

Get the TABLE block for mrgsolve.

Usage

mrgsolveTable(model)

Arguments

model CAMPISIS model

Value

TABLE block if at least one line in error record, character(0) otherwise

mrgsolve_type-class

Mrgsolve export type class.

Description

Mrgsolve export type class.

Ode

Create a new ordinary differential equation (ODE).

Description

Create a new ordinary differential equation (ODE).

Usage

Ode(lhs, rhs = "", comment = as.character(NA))

Arguments

lhs left-hand side variable corresponding to derivative name, must start with 'A_'
rhs right-hand side expression corresponding to derivative value
comment comment if any, single character string

Value

an ODE
ode-class

ODE class. Any statement in the form d/dt(A_CMT) = B.

Description

ODE class. Any statement in the form d/dt(A_CMT) = B.

OdeRecord

Create ODE code record.

Description

Create ODE code record.

Usage

OdeRecord(code = character())

Arguments

code code record

Value

an ODE code record

ode_record-class

ODE record class.

Description

ODE record class.
Create an OMEGA parameter.

**Usage**

```r
Omega(
    name = NA,
    index = NA,
    index2 = NA,
    value = NA,
    fix = FALSE,
    type = NULL,
    same = NA
)
```

**Arguments**

- **name**: parameter name, e.g. CL (prefix OMEGA will be added automatically)
- **index**: parameter index
- **index2**: second parameter index
- **value**: parameter value
- **fix**: parameter was fixed in estimation, logical value
- **type**: variance type: 'var', 'sd', 'covar', 'cor', 'cv' or 'cv%'
- **same**: NA by default, FALSE for first OMEGA followed by 'SAME' OMEGA's, TRUE for 'SAME' OMEGA's

**Value**

an OMEGA parameter

**omega-class**

Omega parameter class.

**Description**

Omega parameter class.

**Slots**

- **same**: logical value, tell if this omega is the same as the previous one
Parameter class. Any parameter in a pharmacometric model.

Slots

- name: parameter name, optional (although recommended)
- index: parameter index, integer
- value: parameter value (e.g. the estimated value from a modelling tool)
- fix: logical value, say if parameter was fixed in the modelling phase

Parameters

Create a list of parameters.

Description

Create a list of parameters.

Usage

Parameters()

Value

an empty list of parameters

Parameters class

Description

Parameters class.

Slots

- varcov: associated variance-covariance matrix
parseIfStatement

Parse IF-statement. Assumption: isIfStatement method already called and returned TRUE.

Description

Parse IF-statement. Assumption: isIfStatement method already called and returned TRUE.

Usage

parseIfStatement(line, comment = as.character(NA))

Arguments

- **line**: IF-statement as single character string value, comment omitted
- **comment**: any comment, NA by default

Value

an IF statement object

parseStatements

Parse statements code and return CAMPSIS statements.

Description

Parse statements code and return CAMPSIS statements.

Usage

parseStatements(code)

Arguments

- **code**: character vector containing all statements (text form)

Value

a list of CAMPSIS statements
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pmx_list-class</td>
<td>PMX list class.</td>
</tr>
<tr>
<td>pmx_element-class</td>
<td>PMX element class.</td>
</tr>
<tr>
<td>pattern-class</td>
<td>Pattern class.</td>
</tr>
</tbody>
</table>

**Pattern**

Create a pattern.

**Usage**

```
Pattern(x)
```

**Arguments**

- `x` regular expression

**Value**

a pattern

**Slots**

- `list` effective list which will contain the elements
- `type` type of the elements this list may contain
pmx_position-class

Description
PMX position class.

pmx_position_by_element-class

Description
PMX position by element class.

pmx_position_by_index-class

Description
PMX position by index class.

Position
Element position in list.

Description
Element position in list.

Usage
Position(x, after = TRUE)

Arguments

- x: either an integer position (useful to add an element in a code record at a specified position) or an model element (element can be a model statement or a code record)
- after: element to be added will be added after x (if after is TRUE) or before x (if after is FALSE)

Value
a position object
**processExtraArg**  
*Process extra arguments.*

**Description**  
Process extra arguments.

**Usage**  
```r  
processExtraArg(args, name, default = NULL, mandatory = FALSE)  
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>args</td>
<td>arguments list</td>
</tr>
<tr>
<td>name</td>
<td>argument name to retrieve</td>
</tr>
<tr>
<td>default</td>
<td>default value if argument is not present</td>
</tr>
<tr>
<td>mandatory</td>
<td>mandatory argument, logical value</td>
</tr>
</tbody>
</table>

**Value**

requested argument value

---

**properties_record-class**  
*Properties record class.*

**Description**  
Properties record class.

---

**rate_record-class**  
*(Infusion)-rate record class.*

**Description**  
*(Infusion)-rate record class.*
read

**Generic read method to read data from a file or a folder.**

**Description**

Generic read method to read data from a file or a folder.

**Usage**

`read(file, ...)`

**Arguments**

- **file**
  - path to the file or folder to be read
- **...**
  - extra arguments

**Value**

the object representation of the data contained in the file

---

**read.allparameters**

**Read all parameters files at once.**

**Description**

Read all parameters files at once.

**Usage**

`read.allparameters(folder)`

**Arguments**

- **folder**
  - path to folder or path to zipped project

**Value**

parameters object
read.campsis

Read a CAMPSIS model.

Description

Read a CAMPSIS model.

Usage

read.campsis(file)

Arguments

file path to folder

Value

a CAMPSIS model

read.model

Read model file.

Description

Read model file.

Usage

read.model(file)

Arguments

file path to records

Value

records object
**read.parameters**

Read parameters file.

**Description**

Read parameters file.

**Usage**

read.parameters(file, type)

**Arguments**

- **file**: path to CSV file
- **type**: parameter type: 'theta', 'omega' or 'sigma'

**Value**

parameters sub list

---

**read.varcov**

Read variance-covariance file.

**Description**

Read variance-covariance file.

**Usage**

read.varcov(file)

**Arguments**

- **file**: path to CSV file

**Value**

variance-covariance matrix
replace

**Replace element by another in list.**

Description

Replace element by another in list.

Usage

```r
replace(object, x)

## S4 method for signature 'pmx_list,pmx_element'
replace(object, x)

## S4 method for signature 'pmx_list,pmx_list'
replace(object, x)

## S4 method for signature 'pmx_list,list'
replace(object, x)

## S4 method for signature 'compartments,compartment_property'
replace(object, x)

## S4 method for signature 'parameters,single_array_parameter'
replace(object, x)

## S4 method for signature 'parameters,double_array_parameter'
replace(object, x)

## S4 method for signature 'statements_record,model_statement'
replace(object, x)

## S4 method for signature 'code_records,model_statement'
replace(object, x)

## S4 method for signature 'campsis_model,compartment'
replace(object, x)

## S4 method for signature 'campsis_model,compartment_property'
replace(object, x)

## S4 method for signature 'campsis_model,parameter'
replace(object, x)

## S4 method for signature 'campsis_model,code_record'
replace(object, x)
```
replaceAll

## S4 method for signature 'campsis_model,model_statement'
replaceAll(object, x)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>list object</td>
</tr>
<tr>
<td>x</td>
<td>element to replace</td>
</tr>
</tbody>
</table>

**Value**

list object or an error if the element does not exist in the list

---

**replaceAll**

*Replace all occurrences in object.*

**Description**

Replace all occurrences in object.

**Usage**

replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'character,variable_pattern,character'
replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'character,pattern,character'
replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'model_statement,pattern,character'
replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'equation,pattern,character'
replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'if_statement,pattern,character'
replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'code_record,pattern,character'
replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'code_records,pattern,character'
replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'campsis_model,pattern,character'
replaceAll(object, pattern, replacement, ...)

## S4 method for signature 'campsis_model,character,character'
replaceAll(object, pattern, replacement, ...)
Arguments

<table>
<thead>
<tr>
<th>object</th>
<th>generic object (e.g. model, code_record(s), etc.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pattern</td>
<td>pattern to be replaced</td>
</tr>
<tr>
<td>replacement</td>
<td>replacement string</td>
</tr>
<tr>
<td>...</td>
<td>extra arguments</td>
</tr>
</tbody>
</table>

Value

the same object with all occurrences replaced

---

```r	rxodeCode(model)
```

Description

Get code for RxODE.

Usage

```r
rxodeCode(model)
```

Arguments

| model     | CAMPSIS model                                    |

Value

corresponding model code for RxODE

---

```r
rxodeMatrix(model, type = "omega")
```

Description

Get the OMEGA/SIGMA matrix for RxODE.

Usage

```r
rxodeMatrix(model, type = "omega")
```

Arguments

| model     | CAMPSIS model                                    |
| type      | either omega or sigma                             |

Value

omega/omega named matrix
**rxodeParams**

Get the parameters vector for RxODE.

**Description**

Get the parameters vector for RxODE.

**Usage**

rxodeParams(model)

**Arguments**

- **model**: CAMPSIS model

**Value**

named vector with THETA values

---

**rxode_type-class**

RxODE export type class.

**Description**

RxODE export type class.

---

**select**

Get a subset of an object.

**Description**

Get a subset of an object.

**Usage**

select(object, ...)

## S4 method for signature 'data.frame'
select(object, ...)

## S4 method for signature 'compartment_properties'
select(object, ...)

## S4 method for signature 'parameters'
select(object, ...)
Arguments

- object: generic object
- ...: arguments to select

Value

subset of an object

---

Sigma 
Create a SIGMA parameter.

---

Description

Create a SIGMA parameter.

Usage

Sigma(name = NA, index = NA, index2 = NA, value = NA, fix = FALSE, type = NULL)

Arguments

- name: parameter name, e.g. CL (prefix SIGMA will be added automatically)
- index: parameter index
- index2: second parameter index
- value: parameter value
- fix: parameter was fixed in estimation, logical value
- type: variance type: 'var', 'sd', 'covar', 'cv' or 'cv%'

Value

a SIGMA parameter

---

sigma-class 
Sigma parameter class.

---

Description

Sigma parameter class.
Single-array parameter class. This parameter has a single index value.

**Description**

Single-array parameter class. This parameter has a single index value.

**sort**

*Sort the specified list.*

**Description**

Sort the specified list.

**Usage**

```
sort(x, decreasing = FALSE, ...)
```

```
## S4 method for signature 'compartment_properties'
sort(x, decreasing = FALSE, ...)
```

```
## S4 method for signature 'compartments'
sort(x, decreasing = FALSE, ...)
```

```
## S4 method for signature 'parameters'
sort(x, decreasing = FALSE, ...)
```

```
## S4 method for signature 'code_records'
sort(x, decreasing = FALSE, ...)
```

```
## S4 method for signature 'campsis_model'
sort(x, decreasing = FALSE, ...)
```

**Arguments**

- `x` list object
- `decreasing` increasing or decreasing order
- `...` extra arguments

**Value**

same list but ordered
**standardise**  
*Standardise.*

**Description**

Standardise.

**Usage**

```r
standardise(object, ...)  
## S4 method for signature 'theta'
standardise(object, ...)  
## S4 method for signature 'double_array_parameter'
standardise(object, parameters = NULL, ...)  
## S4 method for signature 'parameters'
standardise(object, ...)  
```

**Arguments**

- **object**  
generic object
- **...**  
extra arguments needed for standardisation
- **parameters**  
the list of parameters, to be provided only if parameter type is 'cor'

**Value**

standardised object

---

**statements_record-class**  
*Statements record class.*

**Description**

Statements record class.
**Theta**

Create a THETA parameter.

**Description**

Create a THETA parameter.

**Usage**

Theta(name = NA, index = NA, value = NA, fix = FALSE)

**Arguments**

- **name**: parameter name, e.g. CL (prefix THETA will be added automatically)
- **index**: parameter index
- **value**: parameter value
- **fix**: parameter was fixed in estimation, logical value

**Value**

a THETA parameter

---

**theta-class**

Theta parameter class.

**Description**

Theta parameter class.

**toString**

ToString generic method.

**Description**

ToString generic method.
Usage

toString(object, ...)

## S4 method for signature 'unknown_statement'
toString(object, ...)

## S4 method for signature 'comment'
toString(object, ...)

## S4 method for signature 'line_break'
toString(object, ...)

## S4 method for signature 'equation'
toString(object, ...)

## S4 method for signature 'ode'
toString(object, ...)

## S4 method for signature 'if_statement'
toString(object, ...)

## S4 method for signature 'compartment'
toString(object, ...)

## S4 method for signature 'compartment_property'
toString(object, ...)

## S4 method for signature 'compartment_initial_condition'
toString(object, ...)

## S4 method for signature 'mrgsolve_model'
toString(object, ...)

Arguments

object              generic object
...                  extra arguments needed for toString conversion

Value

character value/vector

trim                  Trim character vector. Remove all leading and trailing spaces.
**UnknownStatement**

**Description**

Trim character vector. Remove all leading and trailing spaces.

**Usage**

\[
\text{trim}(x)
\]

**Arguments**

- \(x\) character vector

**Value**

character vector without leading and trailing spaces

---

**UnknownStatement** *Create a new ordinary differential equation (ODE).*

**Description**

Create a new ordinary differential equation (ODE).

**Usage**

\[
\text{UnknownStatement}(\text{line}, \text{comment} = \text{as.character(NA)})
\]

**Arguments**

- \(\text{line}\) line which was not recognised
- \(\text{comment}\) comment if any, single character string

**Value**

an unknown statement

---

**unknown_statement-class**

*Unknown statement class. Any statement not recognized by campsismod.*

**Description**

Unknown statement class. Any statement not recognized by campsismod.
updateCompartments  

Update compartments list from the persisted records. Exported especially for package pmxtran. However, this method should not be called.

Description
Update compartments list from the persisted records. Exported especially for package pmxtran. However, this method should not be called.

Usage
updateCompartments(model)

Arguments
model  
CAMPSIS model

Value
an updated CAMPSIS model, with an updated compartments list

VariablePattern  

Create a variable pattern.

Description
Create a variable pattern.

Usage
VariablePattern(x)

Arguments
x  
variable name

Value
a variable pattern
Variable pattern class.

**Description**

Variable pattern class.

**write**

Write generic object to files.

**Description**

Write generic object to files.

**Usage**

```r
write(object, file, ...)

## S4 method for signature 'parameters,character'
write(object, file, ...)

## S4 method for signature 'code_records,character'
write(object, file, ...)

## S4 method for signature 'campsis_model,character'
write(object, file, ...)
```

**Arguments**

- `object`  
  generic object
- `file`  
  path of the output file or directory
- `...`  
  extra arguments

**Value**

logical value, TRUE for success, FALSE for failure
writeParameters

Write subset of parameters (theta, omega or sigma).

Description
Write subset of parameters (theta, omega or sigma).

Usage
writeParameters(object, file, ...)

Arguments
- object: subset of parameters
- file: filename
- ...: extra arguments, like defaultDf for empty parameters list

Value
TRUE if success

writeVarcov

Write variance-covariance matrix.

Description
Write variance-covariance matrix.

Usage
writeVarcov(object, file)

Arguments
- object: matrix
- file: filename

Value
TRUE if success
Index

* datasets
    model_library, 43
    model_suite, 45

add, 5
add,campsis_model,campsis_model-method (add), 5
add,campsis_model,code_record-method (add), 5
add,campsis_model,compartment_property-method (add), 5
add,campsis_model,model_statement-method (add), 5
add,campsis_model,parameter-method (add), 5
add,code_record,code_record-method (add), 5
add,code_record,model_statement-method (add), 5
add,code_records,code_records-method (add), 5
add,code_records,model_statement-method (add), 5
add,compartments,compartment_property-method (add), 5
add,compartments,compartments-method (add), 5
add,parameters,double_array_parameter-method (add), 5
add,parameters,parameters-method (add), 5
add,parameters,single_array_parameter-method (add), 5
add,pmx_list,list-method (add), 5
add,pmx_list,pmx_element-method (add), 5
add,pmx_list,pmx_list-method (add), 5
as.data.frame, 7
as.data.frame,omega,character,logical-method (as.data.frame), 7
as.data.frame,sigma,character,logical-method (as.data.frame), 7
as.data.frame,theta,character,logical-method (as.data.frame), 7
assertSingleCharacterString, 7
autoDetectNONMEM, 8
autoDetectNONMEM,campsis_model-method (autoDetectNONMEM), 8
Bioavailability, 8
campsis_model-class, 9
CampsisModel, 9
code_record-class, 10
CodeRecords, 10
Comment, 10
comment-class, 11
Compartment, 11
compartments-class, 11
compartments-class, 12
compartment_bioavailability-class, 12
compartment_infusion_duration-class, 12
compartment_infusion_rate-class, 13
compartment_initial_condition-class, 13
compartment_lag_time-class, 13
compartment_properties-class, 13
compartment_property-class, 14
Compartments, 12
compartments-class, 12
contains, 14
contains,campsis_model,pmx_element-method (contains), 14
contains,pmx_list,pmx_element-method (contains), 14
contains,statements_record,model_statement-method (contains), 14
default, 15
delete, 15
getName, compartment_initial_condition-method (getName), 25
getName, compartment_lag_time-method (getName), 25
getName, duration_record-method (getName), 25
generate error_record-method (getName), 25
getName, f_record-method (getName), 25
getName, if_statement-method (getName), 25
getName, init_record-method (getName), 25
generate lag_record-method (getName), 25
generate line_break-method (getName), 25
generate main_record-method (getName), 25
generate ode-method (getName), 25
generate ode_record-method (getName), 25
generate omega-method (getName), 25
generate rate_record-method (getName), 25
genome, sigma-method (getName), 25
genome, theta-method (getName), 25
genome, unknown_statement-method (getName), 25
getNameInModel, 27
getNameInModel, omega-method (getNameInModel), 27
getNameInModel, sigma-method (getNameInModel), 27
getNameInModel, theta-method (getNameInModel), 27
getName, 27
getNames, pmx_list-method (getNames), 27
getNONMEMName, 28
genome, omega-method (getNONMEMName), 28
getNONMEMName, sigma-method (getNONMEMName), 28
getNONMEMName, theta-method (getNONMEMName), 28
genome, 28
genome, compartment_bioavailability-method (getPrefix), 28
genome, compartment_lag_time-method (getPrefix), 28
genome, compartment_bioavailability-method (getRecordName), 30
genome, compartment infraflow_duration-method (getRecordName), 30
genome, compartment infusion_duration-method (getRecordName), 30
genome, compartment infusion_rate-method (getRecordName), 30
genome, compartment_initial_condition-method (getRecordName), 30
genome, compartment lag_time-method (getRecordName), 30
getUncertainty, 30
genome, compartment bioavailability-method (getUncertainty), 30
getUncertainty, parameter-method (getUncertainty), 30
getUncertainty, parameters-method (getUncertainty), 30
getVarCov, 31
getVarCov, campsis_model-method (getVarCov), 31
getVarCov, parameters-method (getVarCov), 31
hasComment, 32
if_statement-class, 33
IfStatement, 32
indexOf, 33
indexOf, pmx_list, pmx_element-method (indexOf), 33
InfusionDuration, 34
InfusionRate, 34
init_record-class, 35
InitialCondition, 35
isComment, 35
isDiag, 36
isDiag, double_array_parameter-method (isDiag), 36
isEmptyLine, 36
isIfStatement, 37
isODE, 38
isRecordDelimiter, 39
isStrictRecordDelimiter, 39
lag_record-class, 40
LagTime, 39
length, pmx_list-method, 40
length, statements_record-method
    (length, pmx_list-method), 40
line_break-class, 41
LineBreak, 40
main_record-class, 41
MainRecord, 41
maxIndex, 42
maxIndex, parameters-method (maxIndex), 42
minIndex, 42
minIndex, parameters-method (minIndex), 42
model_library, 43
model_statement-class, 44
model_statements-class, 44
model_suite, 45
ModelStatements, 43
mrgsolveType-class, 49
mrgsolveBlock, 45
mrgsolveCapture, 46
mrgsolveCompartment, 46
mrgsolveMain, 47
mrgsolveMatrix, 47
mrgsolveOde, 48
mrgsolveParam, 48
mrgsolveTable, 49
Ode, 49
ode-class, 50
ode_record-class, 50
OdeRecord, 50
Omega, 51
omega-class, 51
parameter-class, 52
Parameters, 52
parameters-class, 52
parseIfStatement, 53
parseStatements, 53
Pattern, 54
pattern-class, 54
pmx_element-class, 54
pmx_list-class, 54
pmx_position-class, 55
pmx_position_by_element-class, 55
pmx_position_by_index-class, 55
Position, 55
processExtraArg, 56
properties_record-class, 56
rate_record-class, 56
read, 57
read.allparameters, 57
read.campsis, 58
read.model, 58
read.parameters, 59
read.varcov, 59
replace, 60
replace, campsis_model, code_record-method
    (replace), 60
replace, campsis_model, compartment-method
    (replace), 60
replace, campsis_model, compartment_property-method
    (replace), 60
replace, campsis_model, model_statement-method
    (replace), 60
replace, campsis_model, parameter-method
    (replace), 60
replace, code_records, model_statement-method
    (replace), 60
replace, compartments, compartment_property-method
    (replace), 60
replace, parameters, double_array_parameter-method
    (replace), 60
replace, parameters, single_array_parameter-method
    (replace), 60
replace, pmx_list, list-method (replace), 60
replace, pmx_list, pmx_element-method
    (replace), 60
replace, statements_record, model_statement-method
    (replace), 60
replaceAll, 61
replaceAll, campsis_model, character, character-method
    (replaceAll), 61
replaceAll, campsis_model, pattern, character-method
    (replaceAll), 61
replaceAll, character, pattern, character-method
    (replaceAll), 61
replaceAll, character, variable_pattern, character-method
    (replaceAll), 61
replaceAll, code_record, pattern, character-method (replaceAll), 61
replaceAll, code_records, pattern, character-method (replaceAll), 61
replaceAll, equation, pattern, character-method (replaceAll), 61
replaceAll, if_statement, pattern, character-method (replaceAll), 61
replaceAll, model_statement, pattern, character-method (replaceAll), 61
rxode_type-class, 63
rxodeCode, 62
rxodeMatrix, 62
rxodeParams, 63
select, 63
select, compartment_properties-method (select), 63
select, data.frame-method (select), 63
select, parameters-method (select), 63
Sigma, 64
sigma-class, 64
single_array_parameter-class, 65
sort, 65
sort, campsis_model-method (sort), 65
sort, code_records-method (sort), 65
sort, compartment_properties-method (sort), 65
sort, compartments-method (sort), 65
sort, parameters-method (sort), 65
standardise, 66
standardise, double_array_parameter-method (standardise), 66
standardise, parameters-method (standardise), 66
standardise, theta-method (standardise), 66
statements_record-class, 66
Theta, 67
theta-class, 67
toString, 67
toString, comment-method (toString), 67
toString, compartment-method (toString), 67
toString, compartment_initial_condition-method (toString), 67
toString, compartment_property-method (toString), 67
toString, equation-method (toString), 67
toString, if_statement-method (toString), 67
toString, line_break-method (toString), 67
toString, mrgsolve_model-method (toString), 67
toString, ode-method (toString), 67
toString, unknown_statement-method (toString), 67
trim, 68
unknown_statement-class, 69
UnknownStatement, 69
updateCompartments, 70
variable_pattern-class, 71
VariablePattern, 70
write, 71
write, campsis_model, character-method (write), 71
write, code_records, character-method (write), 71
write, parameters, character-method (write), 71
writeParameters, 72
writeVarcov, 72