

Package ‘crmPack’

June 19, 2015

Maintainer Daniel Sabanes Bove <sabanesd@roche.com>

License GPL (≥ 2)

Title Object-oriented implementation of CRM designs

LazyLoad yes

Author Daniel Sabanes Bove <sabanesd@roche.com>,
Giuseppe Palermo <giuseppe.palermo@roche.com>

Description Object-oriented implementation of CRM designs

Version 0.1.2

Date 2015-06-19

Depends R ($\geq 3.0.0$),
ggplot2,
graphics

Imports methods,
grid,
gridExtra,
GenSA,
mvtnorm,
parallel,
BayesLogit,
rjags,
utils,
tools,
httr

Suggests ggmcmc,
R2WinBUGS,
Rcpp,
RcppArmadillo

Collate 'helpers.R'
'Data-class.R'
'Data-methods.R'
'Rules-class.R'
'Model-class.R'
'Design-class.R'
'writeModel.R'
'McmcOptions-methods.R'
'McmcOptions-class.R'
'Samples-class.R'

'mcmc.R'
 'Simulations-class.R'
 'Model-methods.R'
 'Rules-methods.R'
 'Design-methods.R'
 'fromQuantiles.R'
 'Samples-methods.R'
 'Simulations-methods.R'
 'crmPack-package.R'
 'crmUpgrade.R'

R topics documented:

crmPack-package	5
approximate	5
as.list,GeneralData-method	6
biomLevel	7
CohortSize-class	7
CohortSizeConst	8
CohortSizeConst-class	8
CohortSizeDLT	8
CohortSizeDLT-class	9
CohortSizeMax	9
CohortSizeMax-class	9
CohortSizeMin	10
CohortSizeMin-class	10
CohortSizeParts	10
CohortSizeParts-class	11
CohortSizeRange	11
CohortSizeRange-class	11
crmPackExample	12
crmPackHelp	12
crmPackUpgrade	13
Data	13
Data-class	14
DataDual	14
DataDual-class	15
DataParts	15
DataParts-class	16
Design	16
Design-class	16
dose	17
DualDesign	17
DualDesign-class	18
DualEndpoint	18
DualEndpoint-class	19
DualEndpointBeta	20
DualEndpointBeta-class	20
DualEndpointEmax	21
DualEndpointEmax-class	21
DualEndpointRW	22
DualEndpointRW-class	22

DualSimulations	23
DualSimulations-class	23
DualSimulationsSummary-class	24
examine	24
fit	25
GeneralData-class	26
GeneralModel-class	26
GeneralSimulations	27
GeneralSimulations-class	27
GeneralSimulationsSummary-class	28
get,Samples,character-method	28
getMinInfBeta	29
Increments-class	29
IncrementsRelative	30
IncrementsRelative-class	30
IncrementsRelativeDLT	31
IncrementsRelativeDLT-class	31
IncrementsRelativeParts	32
IncrementsRelativeParts-class	32
initialize,DualEndpointOld-method	33
LogisticKadane	33
LogisticKadane-class	34
LogisticLogNormal	34
LogisticLogNormal-class	35
LogisticLogNormalSub	35
LogisticLogNormalSub-class	36
LogisticNormal	37
LogisticNormal-class	37
LogisticNormalFixedMixture	38
LogisticNormalFixedMixture-class	38
LogisticNormalMixture	39
LogisticNormalMixture-class	40
logit	40
maxDose	41
maxSize	42
mcmc	42
McmcOptions	43
McmcOptions-class	44
MinimalInformative	44
minSize	45
Model-class	45
nextBest	46
NextBest-class	48
NextBestDualEndpoint	48
NextBestDualEndpoint-class	48
NextBestMTD	49
NextBestMTD-class	49
NextBestNCRM	50
NextBestNCRM-class	50
NextBestThreePlusThree	51
NextBestThreePlusThree-class	51
or-Stopping-Stopping	51

or-Stopping-StoppingAny	52
or-StoppingAny-Stopping	52
plot,Data,missing-method	53
plot,DataDual,missing-method	53
plot,DualSimulations,missing-method	54
plot,DualSimulationsSummary,missing-method	54
plot,GeneralSimulations,missing-method	55
plot,GeneralSimulationsSummary,missing-method	56
plot,Samples,DualEndpoint-method	57
plot,Samples,Model-method	57
plot,SimulationsSummary,missing-method	58
plot.arrange	59
prob	59
Quantiles2LogisticNormal	60
Report	61
RuleDesign	61
RuleDesign-class	61
Samples	62
Samples-class	62
sampleSize	62
setSeed	63
show,DualSimulationsSummary-method	63
show,GeneralSimulationsSummary-method	64
show,SimulationsSummary-method	64
simulate,Design-method	65
simulate,DualDesign-method	66
simulate,RuleDesign-method	67
Simulations	67
Simulations-class	68
SimulationsSummary-class	68
size	69
Stopping-class	70
StoppingAll	70
StoppingAll-class	70
StoppingAny	71
StoppingAny-class	71
StoppingCohortsNearDose	72
StoppingCohortsNearDose-class	72
StoppingList	72
StoppingList-class	73
StoppingMinCohorts	73
StoppingMinCohorts-class	74
StoppingMinPatients	74
StoppingMinPatients-class	74
StoppingMTDdistribution	75
StoppingMTDdistribution-class	75
StoppingPatientsNearDose	76
StoppingPatientsNearDose-class	76
StoppingTargetBiomarker	76
StoppingTargetBiomarker-class	77
StoppingTargetProb	77
StoppingTargetProb-class	78

stopTrial	78
summary,DualSimulations-method	80
summary,GeneralSimulations-method	80
summary,Simulations-method	81
ThreePlusThreeDesign	82
update,Data-method	82
update,DataDual-method	83
update,DataParts-method	83
Validate	84
writeModel	84
&,Stopping,Stopping-method	85
&,Stopping,StoppingAll-method	85
&,StoppingAll,Stopping-method	86

Index	87
--------------	-----------

crmPack-package	<i>Object-oriented implementation of CRM designs</i>
-----------------	--

Description

Object-oriented implementation of CRM designs

Author(s)

Daniel Sabanes Bove <sabanesd@roche.com>

approximate	<i>Approximate posterior with (log) normal distribution</i>
-------------	---

Description

It is recommended to use `set.seed` before, in order to be able to reproduce the resulting approximating model exactly.

Usage

```
approximate(object, model, data, ...)

## S4 method for signature 'Samples'
approximate(object, model, data, points = seq(from =
  min(data@doseGrid), to = max(data@doseGrid), length = 5L),
  refDose = median(points), logNormal = FALSE, verbose = TRUE, ...)
```

Arguments

object	the Samples object
model	the Model object
data	the Data object
points	optional parameter, which gives the dose values at which the approximation should rely on (default: 5 values equally spaced from minimum to maximum of the dose grid)
refDose	the reference dose to be used (default: median of points)
logNormal	use the log-normal prior? (not default) otherwise, the normal prior for the logistic regression coefficients is used
verbose	be verbose (progress statements and plot)? (default)
...	additional arguments (see methods)

Value

the approximation model

Methods (by class)

- Samples: Here the ... argument can transport additional arguments for [Quantiles2LogisticNormal](#), e.g. in order to control the approximation quality, etc.

as.list, GeneralData-method

as.list method for the "GeneralData" class

Description

as.list method for the "GeneralData" class

Usage

```
## S4 method for signature 'GeneralData'
as.list(x, ...)
```

Arguments

x	the GeneralData object we want to convert
...	unused

Value

a list of all slots in x

biomLevel	<i>Compute the biomarker level for a given dose, given model and samples</i>
-----------	--

Description

Compute the biomarker level for a given dose, given model and samples

Usage

```
biomLevel(dose, model, samples, ...)

## S4 method for signature 'numeric,DualEndpoint,Samples'
biomLevel(dose, model, samples, xLevel,
  ...)
```

Arguments

dose	the dose
model	the DualEndpoint object
samples	the Samples object
xLevel	the grid index of dose
...	unused

Methods (by class)

- dose = numeric,model = DualEndpoint,samples = Samples: Here it is very easy, we just return the corresponding column (index xLevel) of the biomarker samples matrix, since we save that in the samples

CohortSize-class	<i>The virtual class for cohort sizes</i>
------------------	---

Description

The virtual class for cohort sizes

See Also

[CohortSizeMax](#), [CohortSizeMin](#), [CohortSizeRange](#), [CohortSizeDLT](#), [CohortSizeConst](#), [CohortSizeParts](#)

CohortSizeConst	<i>Initialization function for "CohortSizeConst"</i>
-----------------	--

Description

Initialization function for "CohortSizeConst"

Usage

CohortSizeConst(size)

Arguments

size see [CohortSizeConst](#)

Value

the [CohortSizeConst](#) object

CohortSizeConst-class	<i>Constant cohort size</i>
-----------------------	-----------------------------

Description

This class is used when the cohort size should be kept constant.

Slots

size the constant integer size

CohortSizeDLT	<i>Initialization function for "CohortSizeDLT"</i>
---------------	--

Description

Initialization function for "CohortSizeDLT"

Usage

CohortSizeDLT(DLTintervals, cohortSize)

Arguments

DLTintervals see [CohortSizeDLT](#)
cohortSize see [CohortSizeDLT](#)

Value

the [CohortSizeDLT](#) object

CohortSizeDLT-class	<i>Cohort size based on number of DLTs</i>
---------------------	--

Description

Cohort size based on number of DLTs

Slots

DLTintervals an integer vector with the left bounds of the relevant DLT intervals
cohortSize an integer vector of the same length with the cohort sizes in the DLTintervals

CohortSizeMax	<i>Initialization function for "CohortSizeMax"</i>
---------------	--

Description

Initialization function for "CohortSizeMax"

Usage

CohortSizeMax(cohortSizeList)

Arguments

cohortSizeList see [CohortSizeMax](#)

Value

the [CohortSizeMax](#) object

CohortSizeMax-class	<i>Size based on maximum of multiple cohort size rules</i>
---------------------	--

Description

This class can be used to combine multiple cohort size rules with the MAX operation.

Details

cohortSizeList contains all cohort size rules, which are again objects of class [CohortSize](#). The maximum of these individual cohort sizes is taken to give the final cohort size.

Slots

cohortSizeList list of cohort size rules

CohortSizeMin	<i>Initialization function for "CohortSizeMin"</i>
---------------	--

Description

Initialization function for "CohortSizeMin"

Usage

CohortSizeMin(cohortSizeList)

Arguments

cohortSizeList see [CohortSizeMin](#)

Value

the [CohortSizeMin](#) object

CohortSizeMin-class	<i>Size based on minimum of multiple cohort size rules</i>
---------------------	--

Description

This class can be used to combine multiple cohort size rules with the MIN operation.

Details

cohortSizeList contains all cohort size rules, which are again objects of class [CohortSize](#). The minimum of these individual cohort sizes is taken to give the final cohort size.

Slots

cohortSizeList list of cohort size rules

CohortSizeParts	<i>Initialization function for "CohortSizeParts"</i>
-----------------	--

Description

Initialization function for "CohortSizeParts"

Usage

CohortSizeParts(sizes)

Arguments

sizes see [CohortSizeParts](#)

Value

the CohortSizeParts object

CohortSizeParts-class	Cohort size based on the parts
-----------------------	--------------------------------

Description

This class is used when the cohort size should change for the second part of the dose escalation. Only works in conjunction with DataParts objects.

Slots

sizes the two sizes for part 1 and part 2

CohortSizeRange	Initialization function for "CohortSizeRange"
-----------------	---

Description

Initialization function for "CohortSizeRange"

Usage

CohortSizeRange(intervals, cohortSize)

Arguments

intervals see CohortSizeRange
cohortSize see CohortSizeRange

Value

the CohortSizeRange object

CohortSizeRange-class	Cohort size based on dose range
-----------------------	---------------------------------

Description

Cohort size based on dose range

Slots

intervals a vector with the left bounds of the relevant dose intervals
cohortSize an integer vector of the same length with the cohort sizes in the intervals

crmPackExample	<i>Open the example pdf for crmPack</i>
----------------	---

Description

Calling this helper function should open the example.pdf document, residing in the doc subfolder of the package installation directory.

Usage

```
crmPackExample()
```

Value

nothing

Author(s)

Daniel Sabanes Bove <sabanesd@roche.com>

crmPackHelp	<i>Open the browser with help pages for crmPack</i>
-------------	---

Description

This convenience function opens your browser with the help pages for crmPack.

Usage

```
crmPackHelp()
```

Value

nothing

Author(s)

Daniel Sabanes Bove <sabanesd@roche.com>

 crmPackUpgrade

Upgrade your crmPack installation with the latest version

Description

Executing this function upgrades your crmPack installation with the newest version available on the server. You will need connection to the Roche network (i.e. RANGE connection when you are working off-site) in order to successfully run it.

Usage

```
crmPackUpgrade(lib = NULL, devel = FALSE, force = FALSE,
  repos = structure(c(CRAN = "http://stat.ethz.ch/CRAN/")))
```

Arguments

lib	library where to install the new version (and other required packages) into. Default: same location as the last crmPack version.
devel	Should the development version from Stash be installed? (default FALSE) Otherwise and by default the latest version on the internal package repository GRAN is installed.
force	Should the installation be forced, i.e., even if you have already the latest version, should the package be re-installed?
repos	which repository to use for installing required packages (default: ETHZ in Switzerland)

Details

After installation, the new features in this version will be shown by printing the relevant parts of the NEWS file.

Value

nothing

Author(s)

Daniel Sabanes Bove <sabanesd@roche.com>

 Data

Initialization function for the "Data" class

Description

This is the function for initializing a "Data" class object.

Usage

```
Data(x = numeric(), y = integer(), ID = integer(), cohort = integer(),
  doseGrid = numeric(), ...)
```

Arguments

<code>x</code>	the doses for the patients
<code>y</code>	the vector of toxicity events (0 or 1 integers). You can also normal numeric vectors, but these will then be converted to integers.
<code>ID</code>	unique patient IDs (integer vector)
<code>cohort</code>	the cohort indices (sorted values from 0, 1, 2, ...)
<code>doseGrid</code>	the vector of all possible doses
<code>...</code>	not used

Details

Note that `ID` and `cohort` can be missing, then a warning will be issued and the variables will be filled with default IDs and best guesses, respectively.

Value

the initialized [Data](#) object

Data-class	<i>Class for the data input</i>
------------	---------------------------------

Description

This class inherits from [GeneralData](#).

Slots

<code>x</code>	the doses for the patients
<code>y</code>	the vector of toxicity events (0 or 1 integers)
<code>doseGrid</code>	the vector of all possible doses (sorted), i.e. the dose grid
<code>nGrid</code>	number of gridpoints
<code>xLevel</code>	the levels for the doses the patients have been given

DataDual	<i>Initialization function for the "DataDual" class</i>
----------	---

Description

This is the function for initializing a "DataDual" class object.

Usage

```
DataDual(w = numeric(), ...)
```

Arguments

w the continuous vector of biomarker values
 ... additional parameters from [Data](#)

Value

the initialized [DataDual](#) object

DataDual-class	<i>Class for the dual endpoint data input</i>
----------------	---

Description

This is a subclass of [Data](#), so contains all slots from [Data](#), and in addition biomarker values.

Slots

w the continuous vector of biomarker values

DataParts	<i>Initialization function for the "DataParts" class</i>
-----------	--

Description

This is the function for initializing a [DataParts](#) object.

Usage

```
DataParts(part = integer(), nextPart = 1L, part1Ladder = numeric(), ...)
```

Arguments

part which part does each of the patients belong to?
 nextPart what is the part for the next cohort? (1 or 2)
 part1Ladder what is the escalation ladder for part 1?
 ... additional parameters from [Data](#)

Value

the initialized [DataParts](#) object

DataParts-class	<i>Class for the data with two study parts</i>
-----------------	--

Description

This is a subclass of [Data](#), so contains all slots from [Data](#), and in addition information on the two study parts.

Slots

part integer vector; which part does each of the patients belong to?

nextPart integer; what is the part for the next cohort?

part1Ladder sorted numeric vector; what is the escalation ladder for part 1? This shall be a subset of the doseGrid.

Design	<i>Initialization function for "Design"</i>
--------	---

Description

Initialization function for "Design"

Usage

```
Design(model, stopping, increments, ...)
```

Arguments

model	see Design
stopping	see Design
increments	see Design
...	additional arguments for RuleDesign

Value

the [Design](#) object

Design-class	<i>Class for the CRM design</i>
--------------	---------------------------------

Description

In addition to the slots in the more simple [RuleDesign](#), objects of this class contain:

Slots

model the model to be used, an object of class [Model](#)

stopping stopping rule(s) for the trial, an object of class [Stopping](#)

increments how to control increments between dose levels, an object of class [Increments](#)

dose	<i>Compute the doses for a given probability, given model and samples</i>
------	---

Description

Compute the doses for a given probability, given model and samples

Usage

```
dose(prob, model, samples, ...)
```

```
## S4 method for signature 'numeric,Model,Samples'
dose(prob, model, samples, ...)
```

Arguments

prob	the probability
model	the Model
samples	the Samples
...	unused

Methods (by class)

- prob = numeric,model = [Model](#),samples = [Samples](#):

DualDesign	<i>Initialization function for "DualDesign"</i>
------------	---

Description

Initialization function for "DualDesign"

Usage

```
DualDesign(model, data, ...)
```

Arguments

model	see DualDesign
data	see DualDesign
...	additional arguments for Design

Value

the [DualDesign](#) object

DualDesign-class	<i>Class for the dual-endpoint CRM design</i>
------------------	---

Description

This class has special requirements for the model and data slots in comparison to the parent class [Design](#):

Slots

model the model to be used, an object of class [DualEndpoint](#)

data what is the dose grid, any previous data, etc., contained in an object of class [DataDual](#)

Note that the NextBest slot can be of any class, this allows for easy comparison with recommendation methods that don't use the biomarker information.

DualEndpoint	<i>Initialization function for the "DualEndpoint" class</i>
--------------	---

Description

Initialization function for the "DualEndpoint" class

Usage

```
DualEndpoint(mu, Sigma, sigma2W, rho)
```

Arguments

mu see [DualEndpoint](#)

Sigma see [DualEndpoint](#)

sigma2W see [DualEndpoint](#)

rho see [DualEndpoint](#)

Value

the [DualEndpoint](#) object

DualEndpoint-class	<i>General class for the dual endpoint model</i>
--------------------	--

Description

The idea of the dual-endpoint models is to model not only the dose-toxicity relationship, but also to model at the same time the relationship of a PD biomarker with the dose. The subclasses of this class detail how the dose-biomarker relationship is parametrized and are those to be used. This class here shall contain all the common features to reduce duplicate code. (However, this class must not be virtual, because we need to create objects of it during the construction of subclass objects.)

Details

Currently a probit regression model

$$\Phi^{-1}[p(x)] = \beta_{Z1} + \beta_{Z2} \cdot x$$

is used, where $p(x)$ is the probability of observing a DLT for a given dose x , and Φ is the standard normal cdf. This could later be generalized to have a reference dose or a log transformation for the dose. The prior is

$$\beta_Z \sim \text{Normal}(\mu, \Sigma)$$

.

For the biomarker response w at a dose x , we assume

$$w(x) \sim \text{Normal}(f(x), \sigma_W^2)$$

and $f(x)$ is a function of the dose x , which is further specified in the subclasses. The biomarker variance σ_W^2 can be fixed or assigned an inverse gamma prior distribution; see the details below under slot `sigma2W`.

Finally, the two endpoints y (the binary DLT variable) and w (the biomarker) can be correlated, by assuming a correlation ρ between the underlying continuous latent toxicity variable z and the biomarker w . Again, this correlation can be fixed or assigned a prior distribution from the scaled beta family; see the details below under slot `rho`.

Please see the Hive page for more details on the model and the example vignette by typing `crmPackExample()` for a full example.

Slots

`mu` For the probit toxicity model, `mu` contains the prior mean vector

`Sigma` For the probit toxicity model, contains the prior covariance matrix

`sigma2W` Either a fixed value for the biomarker variance, or a vector with elements `a` and `b` for the inverse-gamma prior parameters.

`rho` Either a fixed value for the correlation (between -1 and 1), or a vector with elements `a` and `b` for the Beta prior on the transformation $\kappa = (\rho + 1) / 2$, which is in $(0, 1)$. For example, `a=1, b=1` leads to a uniform prior on `rho`.

`useFixed` a list with logical value for each of the parameters indicating whether a fixed value is used or not; this slot is needed for internal purposes and not to be touched by the user.

See Also

Current subclasses: [DualEndpointRW](#), [DualEndpointBeta](#)

DualEndpointBeta	<i>Initialization function for the "DualEndpointBeta" class</i>
------------------	---

Description

Initialization function for the "DualEndpointBeta" class

Usage

```
DualEndpointBeta(E0, Emax, delta1, mode, refDose, ...)
```

Arguments

E0	see DualEndpointBeta
Emax	see DualEndpointBeta
delta1	see DualEndpointBeta
mode	see DualEndpointBeta
refDose	see DualEndpointBeta
...	additional parameters, see DualEndpoint

Value

the [DualEndpointBeta](#) object

DualEndpointBeta-class	<i>Dual endpoint model with beta function for dose-biomarker relationship</i>
------------------------	---

Description

This class extends the [DualEndpoint](#) class. Here the dose-biomarker relationship $f(x)$ is modelled by a parametric, rescaled beta density function:

Details

$$f(x) = E_0 + (E_{max} - E_0) * Beta(\delta_1, \delta_2) * (x/x^*)^{\delta_1} * (1 - x/x^*)^{\delta_2}$$

where x^* is the maximum dose (end of the dose range to be considered), δ_1 and δ_2 are the two beta parameters, and E_0 and E_{max} are the minimum and maximum levels, respectively. For ease of interpretation, we parametrize with δ_1 and the mode of the curve instead, where

$$mode = \delta_1 / (\delta_1 + \delta_2),$$

and multiplying this with x^* gives the mode on the dose grid.

All parameters can currently be assigned uniform distributions or be fixed in advance.

Slots

`E0` either a fixed number or the two uniform distribution parameters

`Emax` either a fixed number or the two uniform distribution parameters

`delta1` either a fixed number or the two uniform distribution parameters

`mode` either a fixed number or the two uniform distribution parameters

`refDose` the reference dose x^*

DualEndpointEmax	Initialization function for the "DualEndpointEmax" class
------------------	--

Description

Initialization function for the "DualEndpointEmax" class

Usage

```
DualEndpointEmax(E0, Emax, ED50, refDose, ...)
```

Arguments

<code>E0</code>	see DualEndpointEmax
<code>Emax</code>	see DualEndpointEmax
<code>ED50</code>	see DualEndpointEmax
<code>refDose</code>	see DualEndpointEmax
<code>...</code>	additional parameters, see DualEndpoint

Value

the [DualEndpointEmax](#) object

DualEndpointEmax-class	<i>Dual endpoint model with emax function for dose-biomarker relationship</i>
------------------------	---

Description

This class extends the [DualEndpoint](#) class. Here the dose-biomarker relationship $f(x)$ is modelled by a parametric EMAX function:

Details

$$f(x) = E_0 + \frac{(E_{max} - E_0) * (x/x^*)}{ED_{50} + (x/x^*)}$$

where x^* is a reference dose, E_0 and E_{max} are the minimum and maximum levels for the biomarker and ED_{50} is the dose achieving half of the maximum effect $0.5 * E_{max}$.

All parameters can currently be assigned uniform distributions or be fixed in advance.

Slots

E0 either a fixed number or the two uniform distribution parameters
 Emax either a fixed number or the two uniform distribution parameters
 ED50 either a fixed number or the two uniform distribution parameters
 refDose the reference dose x^*

DualEndpointRW	<i>Initialization function for the "DualEndpointRW" class</i>
----------------	---

Description

Initialization function for the "DualEndpointRW" class

Usage

```
DualEndpointRW(sigma2betaW, smooth = c("RW1", "RW2"), ...)
```

Arguments

sigma2betaW	see DualEndpointRW
smooth	either "RW1" (default) or "RW2", for specifying the random walk prior on the biomarker level.
...	additional parameters, see DualEndpoint

Value

the [DualEndpointRW](#) object

DualEndpointRW-class	<i>Dual endpoint model with RW prior for biomarker</i>
----------------------	--

Description

This class extends the [DualEndpoint](#) class. Here the dose-biomarker relationship $f(x)$ is modelled by a non-parametric random-walk of first (RW1) or second order (RW2) (todo: warning: at the moment only the first order random walk produces useful results).

Details

That means, for the RW1 we assume

$$\beta_{W,i} - \beta_{W,i-1} \sim \text{Normal}(0, \sigma_{\beta_W}^2),$$

where $\beta_{W,i} = f(x_i)$ is the biomarker mean at the i -th dose gridpoint x_i . For the RW2, the second-order differences instead of the first-order differences of the biomarker means follow the normal distribution.

The variance parameter $\sigma_{\beta_W}^2$ is important because it steers the smoothness of the function $f(x)$: if it is large, then $f(x)$ will be very wiggly; if it is small, then $f(x)$ will be smooth. This parameter can either be fixed or assigned an inverse gamma prior distribution.

Usually this modelling will only make sense if a regular dose grid is used, with equidistant grid points ensuring that the distance $x_i - x_{i-1}$ is the same for all grid positions i .

Slots

- `sigma2betaW` Contains the prior variance factor of the random walk prior for the biomarker model. If it is not a single number, it can also contain a vector with elements `a` and `b` for the inverse-gamma prior on `sigma2betaW`.
- `useRW1` for specifying the random walk prior on the biomarker level: if TRUE, RW1 is used, otherwise RW2.

DualSimulations	<i>Initialization function for "DualSimulations"</i>
-----------------	--

Description

Initialization function for "DualSimulations"

Usage

```
DualSimulations(rhoEst, sigma2West, fitBiomarker,...)
```

Arguments

- | | |
|---------------------------|--|
| <code>rhoEst</code> | see DualSimulations |
| <code>sigma2West</code> | see DualSimulations |
| <code>fitBiomarker</code> | see DualSimulations |
| <code>...</code> | additional parameters from Simulations |

Value

the [DualSimulations](#) object

DualSimulations-class	<i>Class for the simulations output from dual-endpoint model based designs</i>
-----------------------	--

Description

This class captures the trial simulations from dual-endpoint model based designs. In comparison to the parent class [Simulations](#), it contains additional slots to capture the dose-biomarker fits, and the `sigma2W` and `rho` estimates.

Slots

- `rhoEst` the vector of final posterior median `rho` estimates
- `sigma2West` the vector of final posterior median `sigma2W` estimates
- `fitBiomarker` list with the final dose-biomarker curve fits

DualSimulationsSummary-class

Class for the summary of dual-endpoint simulations output

Description

In addition to the slots in the parent class [SimulationsSummary](#), it contains two slots for the biomarker model fit information.

Details

Note that objects should not be created by users, therefore no initialization function is provided for this class.

Slots

biomarkerFitAtDoseMostSelected fitted biomarker level at dose most often selected

meanBiomarkerFit list with the average, lower (2.5 quantiles of the mean fitted biomarker level at each dose level

examine

Obtain hypothetical trial course table for a design

Description

This generic function takes a design and generates a dataframe showing the beginning of several hypothetical trial courses under the design. This means, from the generated dataframe one can read off: - how many cohorts are required in the optimal case (no DLTs observed) in order to reach the highest dose of the specified dose grid - assuming no DLTs are observed until a certain dose level, what the next recommended dose is for all possible number of DLTs observed - the actual relative increments that will be used in these cases - whether the trial would stop at a certain cohort Examining the "single trial" behavior of a dose escalation design is the first important step in evaluating a design, and cannot be replaced by studying solely the operating characteristics in "many trials". The cohort sizes are also taken from the design, assuming no DLTs occur until the dose listed.

Usage

```
examine(object, ...)
```

```
## S4 method for signature 'Design'
```

```
examine(object, mcmcOptions = McmcOptions(), ...)
```

```
## S4 method for signature 'RuleDesign'
```

```
examine(object, ...)
```


Arguments

object	the design (Design or RuleDesign object) we want to examine
mcmcOptions	object of class McmcOptions , giving the MCMC options for each evaluation in the trial. By default, the standard options are used
...	additional arguments (see methods)

Value

The data frame

Methods (by class)

- Design: Examine a model-based CRM
- RuleDesign: Examine a rule-based design

fit	<i>Fit method for the Samples class</i>
-----	---

Description

Note this new generic function is necessary because the [fitted](#) function only allows the first argument object to appear in the signature. But we need also other arguments in the signature.

Usage

```
fit(object, model, data, ...)

## S4 method for signature 'Samples,Model,Data'
fit(object, model, data,
     points = data@doseGrid, quantiles = c(0.025, 0.975), middle = mean, ...)

## S4 method for signature 'Samples,DualEndpoint,DataDual'
fit(object, model, data,
     quantiles = c(0.025, 0.975), middle = mean, ...)
```

Arguments

object	the Samples object
model	the Model object
data	the Data object
points	at which dose levels is the fit requested? default is the dose grid
quantiles	the quantiles to be calculated (default: 0.025 and 0.975)
middle	the function for computing the middle point. Default: mean
...	unused

Value

the data frame with required information (see method details)

Methods (by class)

- `object = Samples, model = Model, data = Data`: This method returns a data frame with dose, middle, lower and upper quantiles for the dose-toxicity curve
- `object = Samples, model = DualEndpoint, data = DataDual`: This method returns a data frame with dose, and middle, lower and upper quantiles, for both the dose-tox and dose-biomarker (suffix "Biomarker") curves, for all grid points (Note that currently only the grid points can be used, because the DualEndpointRW models only allow that)

GeneralData-class	<i>Class for general data input</i>
-------------------	-------------------------------------

Description

Class for general data input

Slots

ID unique patient IDs (integer vector)
 cohort the cohort indices (sorted values from 0, 1, 2, ...)
 nObs number of observations

GeneralModel-class	<i>General class for model input</i>
--------------------	--------------------------------------

Description

This is the general model class, from which all other specific models inherit.

Details

The `datamodel` must obey the convention that the data input is called exactly as in the corresponding data class. All prior distributions for parameters should be contained in the model function `priormodel`. The background is that this can be used to simulate from the prior distribution, before obtaining any data.

Slots

`datamodel` a function representing the BUGS data model specification (see the details above)
`priormodel` a function representing the BUGS prior specification (see the details above)
`datanames` The names of all data slots that are used in the `datamodel` and/or `priormodel` definition. Note that you cannot specify more variables than those that are really used in the model!
`modelspecs` a function computing the list of the data model and prior model specifications that are required for fully specifying them (e.g. prior parameters, reference dose, etc.), based on the data slots that are then required as arguments of this function. This will then be passed to BUGS for the computations.
`init` a function computing the list of starting values for parameters required to be initialized in the MCMC sampler, based on the data slots that are then required as arguments of this function
`sample` names of all parameters from which you would like to save the MCMC samples.

See Also[Model](#)

GeneralSimulations	<i>Initialization function for "GeneralSimulations"</i>
--------------------	---

Description

Initialization function for "GeneralSimulations"

Usage

```
GeneralSimulations(data, doses, seed)
```

Arguments

data	see GeneralSimulations
doses	see GeneralSimulations
seed	see GeneralSimulations

Value

the [GeneralSimulations](#) object

GeneralSimulations-class	<i>General class for the simulations output</i>
--------------------------	---

Description

This class captures trial simulations.

Details

Here also the random generator state before starting the simulation is saved, in order to be able to reproduce the outcome. For this just use [set.seed](#) with the seed as argument before running [simulate,Design-method](#).

Slots

data	list of produced Data objects
doses	the vector of final dose recommendations
seed	random generator state before starting the simulation

GeneralSimulationsSummary-class

Class for the summary of general simulations output

Description

Note that objects should not be created by users, therefore no initialization function is provided for this class.

Slots

target target toxicity interval
 targetDoseInterval corresponding target dose interval
 nsim number of simulations
 propDLTs proportions of DLTs in the trials
 meanToxRisk mean toxicity risks for the patients
 doseSelected doses selected as MTD
 toxAtDosesSelected true toxicity at doses selected
 propAtTarget Proportion of trials selecting target MTD
 doseMostSelected dose most often selected as MTD
 obsToxRateAtDoseMostSelected observed toxicity rate at dose most often selected
 nObs number of patients overall
 nAboveTarget number of patients treated above target tox interval
 doseGrid the dose grid that has been used

get,Samples,character-method

Get specific parameter samples and produce a data.frame

Description

Here you have to specify with pos which parameter you would like to extract from the [Samples](#) object

Usage

```
## S4 method for signature 'Samples,character'
get(x, pos = -1L, envir = NULL, mode = NULL,
    inherits = NULL)
```

Arguments

x	the Samples object
pos	the name of the parameter
envir	for vectorial parameters, you can give the indices of the elements you would like to extract. If NULL, the whole vector samples will be returned
mode	not used
inherits	not used

Value

the data frame suitable for use with [ggmcmc](#)

getMinInfBeta	<i>Get the minimal informative unimodal beta distribution</i>
---------------	---

Description

As defined in Neuenschwander et al (2008), this function computes the parameters of the minimal informative unimodal beta distribution, given the request that the p-quantile should be q, i.e. $X \sim \text{Be}(a, b)$ with $\Pr(X \leq q) = p$.

Usage

```
getMinInfBeta(p, q)
```

Arguments

p	the probability (> 0 and < 1)
q	the quantile (> 0 and < 1)

Value

the two resulting beta parameters a and b in a list

Increments-class	<i>The virtual class for controlling increments</i>
------------------	---

Description

The virtual class for controlling increments

See Also

[IncrementsRelative](#), [IncrementsRelativeDLT](#), [IncrementsRelativeParts](#)

IncrementsRelative	<i>Initialization function for "IncrementsRelative"</i>
--------------------	---

Description

Initialization function for "IncrementsRelative"

Usage

```
IncrementsRelative(intervals, increments)
```

Arguments

intervals	see IncrementsRelative
increments	see IncrementsRelative

Value

the [IncrementsRelative](#) object

IncrementsRelative-class	<i>Increments control based on relative differences in intervals</i>
--------------------------	--

Description

Note that `intervals` is to be read as follows. If for example, we want to specify three intervals: First 0 to less than 50, second at least 50 up to less than 100 mg, and third at least 100 mg, then we specify `intervals` to be `c(0, 50, 100)`. That means, the right bound of the intervals are exclusive to the interval, and the last interval goes from the last value until infinity.

Slots

`intervals` a vector with the left bounds of the relevant intervals

`increments` a vector of the same length with the maximum allowable relative increments in the `intervals`

IncrementsRelativeDLT *Initialization function for "IncrementsRelativeDLT"*

Description

Initialization function for "IncrementsRelativeDLT"

Usage

IncrementsRelativeDLT(DLTintervals, increments)

Arguments

DLTintervals see [IncrementsRelativeDLT](#)
 increments see [IncrementsRelativeDLT](#)

Value

the [IncrementsRelativeDLT](#) object

IncrementsRelativeDLT-class
Increments control based on relative differences in terms of DLTs

Description

Note that DLTintervals is to be read as follows. If for example, we want to specify three intervals: First 0 DLTs, second 1 or 2 DLTs, and third at least 3 DLTs, then we specify DLTintervals to be $c(0, 1, 3)$. That means, the right bound of the intervals are exclusive to the interval – the vector only gives the left bounds of the intervals. The last interval goes from 3 to infinity.

Slots

DLTintervals an integer vector with the left bounds of the relevant DLT intervals
 increments a vector of the same length with the maximum allowable relative increments in the DLTintervals

IncrementsRelativeParts

Initialization function for "IncrementsRelativeParts"

Description

Initialization function for "IncrementsRelativeParts"

Usage

```
IncrementsRelativeParts(dltStart, cleanStart, ...)
```

Arguments

dltStart	see IncrementsRelativeParts
cleanStart	see IncrementsRelativeParts
...	additional slots from IncrementsRelative

Value

the [IncrementsRelativeParts](#) object

IncrementsRelativeParts-class

Increments control based on relative differences in intervals, with special rules for part 1 and beginning of part 2

Description

Note that this only works in conjunction with [DataParts](#) objects. If the part 2 will just be started in the next cohort, then the next maximum dose will be either dltStart (e.g. -1) shift of the last part 1 dose in case of a DLT in part 1, or cleanStart shift (e.g. 0) in case of no DLTs in part 1. If part 1 will still be on in the next cohort, then the next dose level will be the next higher dose level in the part1Ladder of the data object. If part 2 has been started before, the usual relative increment rules apply, see [IncrementsRelative](#).

Slots

dltStart integer giving the dose level increment for starting part 2 in case of a DLT in part 1

cleanStart integer giving the dose level increment for starting part 2 in case of a DLT in part 1. If this is less or equal to 0, then the part 1 ladder will be used to find the maximum next dose. If this is larger than 0, then the relative increment rules will be applied to find the next maximum dose level.

initialize,DualEndpointOld-method

Initialization method for the "DualEndpointOld" class

Description

Initialization method for the "DualEndpointOld" class

Usage

```
## S4 method for signature 'DualEndpointOld'
initialize(.Object, mu, Sigma, sigma2betaW, sigma2W,
  rho, smooth = c("RW1", "RW2"), ...)
```

Arguments

.Object	the DualEndpointOld we want to initialize
mu	see DualEndpointOld
Sigma	see DualEndpointOld
sigma2betaW	see DualEndpointOld
sigma2W	see DualEndpointOld
rho	see DualEndpointOld
smooth	either "RW1" (default) or "RW2", for specifying the random walk prior on the biomarker level.
...	not used

LogisticKadane

Initialization function for the "LogisticKadane" class

Description

Initialization function for the "LogisticKadane" class

Usage

```
LogisticKadane(theta, xmin, xmax)
```

Arguments

theta	the target toxicity probability
xmin	the minimum of the dose range
xmax	the maximum of the dose range

Value

the [LogisticKadane](#)

LogisticKadane-class	<i>Reparametrized logistic model</i>
----------------------	--------------------------------------

Description

This is the logistic model in the parametrization of Kadane et al. (1980).

Details

Let $\rho_0 = p(x_{min})$ be the probability of a DLT and the minimum dose x_{min} , and let γ be the dose with target toxicity probability θ , i.e. $p(\gamma) = \theta$. Then it can easily be shown that the logistic regression model has intercept

$$\frac{\gamma \text{logit}(\rho_0) - x_{min} \text{logit}(\theta)}{\gamma - x_{min}}$$

and slope

$$\frac{\text{logit}(\theta) - \text{logit}(\rho_0)}{\gamma - x_{min}}$$

The prior is a uniform distribution for γ between x_{min} and x_{max} , and for ρ_0 as well a uniform distribution between 0 and θ .

The slots of this class, required for creating the model, are the target toxicity, as well as the minimum and maximum of the dose range. Note that these can be different from the minimum and maximum of the dose grid in the data later on.

Slots

theta the target toxicity probability θ
 xmin the minimum of the dose range x_{min}
 xmax the maximum of the dose range x_{max}

LogisticLogNormal	<i>Initialization function for the "LogisticLogNormal" class</i>
-------------------	--

Description

Initialization function for the "LogisticLogNormal" class

Usage

```
LogisticLogNormal(mean, cov, refDose)
```

Arguments

mean	the prior mean vector
cov	the prior covariance matrix
refDose	the reference dose

Value

the [LogisticLogNormal](#) object

LogisticLogNormal-class

Standard logistic model with bivariate (log) normal prior

Description

This is the usual logistic regression model with a bivariate normal prior on the intercept and log slope.

Details

The covariate is the natural logarithm of the dose x divided by the reference dose x^* :

$$\text{logit}[p(x)] = \alpha + \beta \cdot \log(x/x^*)$$

where $p(x)$ is the probability of observing a DLT for a given dose x .

The prior is

$$(\alpha, \log(\beta)) \sim \text{Normal}(\mu, \Sigma)$$

The slots of this class contain the mean vector and the covariance matrix of the bivariate normal distribution, as well as the reference dose.

Note that the parametrization inside the class uses `alpha0` and `alpha1`. `alpha0` is identical to the intercept α above and is the log-odds for a DLT at the reference dose x^* . Therefore, the prior mean for `alpha0` is the expected log-odds at the reference dose x^* before observing any data. Note that the expected odds is not just the exp of the prior mean of `alpha0`, because the non-linearity of the exp transformation. The log-normal distribution on Wikipedia gives the formula for computing the prior mean of $\exp(\alpha_0)$. `alpha0` is the $\log(\alpha)$ in the Neuenschwander et al. (2008) paper. `alpha1` is identical to $\log(\beta)$ above and equals the beta in the Neuenschwander et al paper. $\exp(\alpha_1)$ gives the odds-ratio for DLT between two doses that differ by the factor $\exp(1) \sim 2.7$. `alpha1` has a log-normal distribution in the LogisticLogNormal model in order to ensure positivity of `alpha1` and thus $\exp(\alpha_1) > 1$.

Slots

`mean` the prior mean vector μ
`cov` the prior covariance matrix Σ
`refDose` the reference dose x^*

LogisticLogNormalSub *Initialization function for the "LogisticLogNormalSub" class*

Description

Initialization function for the "LogisticLogNormalSub" class

Usage

```
LogisticLogNormalSub(mean, cov, refDose)
```

Arguments

mean	the prior mean vector
cov	the prior covariance matrix
refDose	the reference dose

Value

the `LogisticLogNormalSub` object

LogisticLogNormalSub-class

Standard logistic model with bivariate (log) normal prior with subtractive dose standardization

Description

This is the usual logistic regression model with a bivariate normal prior on the intercept and log slope.

Details

The covariate is the dose x minus the reference dose x^* :

$$\text{logit}[p(x)] = \alpha + \beta \cdot (x - x^*)$$

where $p(x)$ is the probability of observing a DLT for a given dose x .

The prior is

$$(\alpha, \log(\beta)) \sim \text{Normal}(\mu, \Sigma)$$

The slots of this class contain the mean vector and the covariance matrix of the bivariate normal distribution, as well as the reference dose.

Slots

mean	the prior mean vector μ
cov	the prior covariance matrix Σ
refDose	the reference dose x^*

LogisticNormal	<i>Initialization function for the "LogisticNormal" class</i>
----------------	---

Description

Initialization function for the "LogisticNormal" class

Usage

```
LogisticNormal(mean, cov, refDose)
```

Arguments

mean	the prior mean vector
cov	the prior covariance matrix
refDose	the reference dose

Value

the [LogisticNormal](#) object

LogisticNormal-class	<i>Standard logistic model with bivariate normal prior</i>
----------------------	--

Description

This is the usual logistic regression model with a bivariate normal prior on the intercept and slope.

Details

The covariate is the natural logarithm of the dose x divided by the reference dose x^* :

$$\text{logit}[p(x)] = \alpha + \beta \cdot \log(x/x^*)$$

where $p(x)$ is the probability of observing a DLT for a given dose x .

The prior is

$$(\alpha, \beta) \sim \text{Normal}(\mu, \Sigma)$$

The slots of this class contain the mean vector, the covariance and precision matrices of the bivariate normal distribution, as well as the reference dose.

Slots

mean	the prior mean vector μ
cov	the prior covariance matrix Σ
prec	the prior precision matrix Σ^{-1}
refDose	the reference dose x^*

LogisticNormalFixedMixture

Initialization function for the "LogisticNormalFixedMixture" class

Description

Initialization function for the "LogisticNormalFixedMixture" class

Usage

```
LogisticNormalFixedMixture(components, weights, refDose, logNormal = FALSE)
```

Arguments

components	the specifications of the mixture components: a list with one list of mean and cov for each bivariate (log) normal prior
weights	the weights of the components, these must be positive and will be normalized to sum to 1
refDose	the reference dose
logNormal	should a log normal prior be specified, such that the mean vectors and covariance matrices are valid for the intercept and log slope? (not default)

Value

the [LogisticNormalFixedMixture](#) object

LogisticNormalFixedMixture-class

Standard logistic model with fixed mixture of multiple bivariate (log) normal priors

Description

This is standard logistic regression model with a mixture of multiple bivariate (log) normal priors on the intercept and slope parameters. The weights of the normal priors are fixed, hence no additional model parameters are introduced. This type of prior is often used to better approximate a given posterior distribution, or when the information is given in terms of a mixture.

Details

The covariate is the natural logarithm of the dose x divided by the reference dose x^* :

$$\text{logit}[p(x)] = \alpha + \beta \cdot \log(x/x^*)$$

where $p(x)$ is the probability of observing a DLT for a given dose x .

The prior is

$$(\alpha, \beta) \sim \sum_{j=1}^K w_j \text{Normal}(\mu_j, \Sigma_j)$$

if a normal prior is used and

$$(\alpha, \log(\beta)) \sim \sum_{j=1}^K w_j \text{Normal}(\mu_j, \Sigma_j)$$

if a log normal prior is used.

The weight w_j of the components are fixed and sum to 1.

The (additional) slots of this class comprise two lists, containing the mean vector, the covariance and precision matrices of the two bivariate normal distributions each, the parameters of the beta prior for the first component weight, as well as the reference dose. Moreover, a slot specifies whether a log normal prior is used.

Slots

components a list with one entry per component of the mixture. Each entry is a list with mean, cov and prec for the bivariate normal prior

weights the weights of the components, these must be positive and sum to 1

refDose the reference dose x^*

logNormal is a log normal prior specified for each of the components?

LogisticNormalMixture *Initialization function for the "LogisticNormalMixture" class*

Description

Initialization function for the "LogisticNormalMixture" class

Usage

```
LogisticNormalMixture(comp1, comp2, weightpar, refDose)
```

Arguments

comp1	the specifications of the first component: a list with mean and cov for the first bivariate normal prior
comp2	the specifications of the second component
weightpar	the beta parameters for the weight of the first component
refDose	the reference dose

Value

the [LogisticNormalMixture](#) object

LogisticNormalMixture-class

Standard logistic model with flexible mixture of two bivariate normal priors

Description

This is standard logistic regression model with a mixture of two bivariate normal priors on the intercept and slope parameters. The weight of the two normal priors is a model parameter, hence it is a flexible mixture. This type of prior is often used with a mixture of a minimal informative and an informative component, in order to make the CRM more robust to data deviations from the informative component.

Details

The covariate is the natural logarithm of the dose x divided by the reference dose x^* :

$$\text{logit}[p(x)] = \alpha + \beta \cdot \log(x/x^*)$$

where $p(x)$ is the probability of observing a DLT for a given dose x .

The prior is

$$(\alpha, \beta) \sim w * \text{Normal}(\mu_1, \Sigma_1) + (1 - w) * \text{Normal}(\mu_2, \Sigma_2)$$

The weight w for the first component is assigned a beta prior $B(a, b)$.

The slots of this class comprise two lists, containing the mean vector, the covariance and precision matrices of the two bivariate normal distributions each, the parameters of the beta prior for the first component weight, as well as the reference dose.

Slots

comp1 the specifications of the first component: a list with mean, cov and prec for the first bivariate normal prior
 comp2 the specifications of the second component
 weightpar the beta parameters for the weight of the first component
 refDose the reference dose x^*

logit

Shorthand for logit function

Description

Shorthand for logit function

Usage

logit(x)

Arguments

x the function argument

Value

the logit(x)

maxDose	<i>Determine the maximum possible next dose</i>
---------	---

Description

Determine the upper limit of the next dose based on the increments rule.

Usage

```
maxDose(increments, data, ...)

## S4 method for signature 'IncrementsRelative,Data'
maxDose(increments, data, ...)

## S4 method for signature 'IncrementsRelativeParts,DataParts'
maxDose(increments, data, ...)

## S4 method for signature 'IncrementsRelativeDLT,Data'
maxDose(increments, data, ...)
```

Arguments

increments The rule, an object of class [Increments](#)
 data The data input, an object of class [Data](#)
 ... further arguments

Details

This function outputs the maximum possible next dose, based on the corresponding rule increments and the data.

Value

the maximum possible next dose

Methods (by class)

- increments = IncrementsRelative, data = Data: Determine the maximum possible next dose based on relative increments
- increments = IncrementsRelativeParts, data = DataParts: Determine the maximum possible next dose based on relative increments and part 1 and 2
- increments = IncrementsRelativeDLT, data = Data: Determine the maximum possible next dose based on relative increments determined by DLTs so far

maxSize	<i>"MAX" combination of cohort size rules</i>
---------	---

Description

This function combines cohort size rules by taking the maximum of all sizes.

Usage

```
maxSize(...)

## S4 method for signature 'CohortSize'
maxSize(...)
```

Arguments

... Objects of class [CohortSize](#)

Value

the combination as an object of class [CohortSizeMax](#)

Methods (by class)

- CohortSize: The method combining cohort size rules by taking maximum

See Also

[minSize](#)

mcmc	<i>Obtain posterior samples for all model parameters</i>
------	--

Description

This is the function to actually run the MCMC machinery to produce posterior samples from all model parameters and required derived values. It is a generic function, so that customized versions may be conveniently defined for specific subclasses of `GeneralData`, `GeneralModel`, and `McmcOptions` input.

Usage

```
mcmc(data, model, options, ...)

## S4 method for signature 'GeneralData,GeneralModel,McmcOptions'
mcmc(data, model, options,
      program = c("JAGS", "OpenBUGS", "WinBUGS"), verbose = FALSE, ...)

## S4 method for signature 'Data,LogisticNormal,McmcOptions'
mcmc(data, model, options,
      verbose = FALSE, ...)
```

Arguments

data	The data input, an object of class GeneralData
model	The model input, an object of class GeneralModel
options	MCMC options, an object of class McmcOptions
program	the program which shall be used: either “JAGS” (default), “OpenBUGS” or “WinBUGS”
verbose	shall progress bar and messages be printed? (not default)
...	unused

Details

Reproducible samples can be obtained by setting the seed via [set.seed](#) before in the user code as usual. However, note that because the RNG sampler used is external to R, running this MCMC function will not change the seed position – that is, the repeated call to this function will then result in exactly the same output.

Value

The posterior samples, an object of class [Samples](#).

Methods (by class)

- `data = GeneralData, model = GeneralModel, options = McmcOptions`: Standard method which uses JAGS/BUGS
- `data = Data, model = LogisticNormal, options = McmcOptions`: The fast method for the `LogisticNormal` class

McmcOptions	<i>Initialization function for the "McmcOptions" class</i>
-------------	--

Description

Initialization function for the "McmcOptions" class

Usage

```
McmcOptions(burnin = 10000L, step = 2L, samples = 10000L)
```

Arguments

burnin	number of burn-in iterations which are not saved (default: 10,000)
step	only every step-th iteration is saved after the burn-in (default: 2)
samples	number of resulting samples (by default 10,000 will result)

Value

the [McmcOptions](#) object

McmcOptions-class	<i>Class for the three canonical MCMC options</i>
-------------------	---

Description

Class for the three canonical MCMC options

Slots

iterations number of MCMC iterations
 burnin number of burn-in iterations which are not saved
 step only every step-th iteration is saved after the burn-in

MinimalInformative	<i>Construct a minimally informative prior</i>
--------------------	--

Description

This function constructs a minimally informative prior, which is captured in a [LogisticNormal](#) (or [LogisticLogNormal](#)) object.

Usage

```
MinimalInformative(dosegrid, threshmin = 0.2, threshmax = 0.3, ...)
```

Arguments

dosegrid	the dose grid
threshmin	Any toxicity probability above this threshold would be very unlikely (5%) at the minimum dose (default: 0.2)
threshmax	Any toxicity probability below this threshold would be very unlikely (5%) at the maximum dose (default: 0.3)
...	additional arguments for computations, see Quantiles2LogisticNormal , e.g. refDose and logNormal=TRUE to obtain a minimal informative log normal prior.

Details

Based on the proposal by Neuenschwander et al (2008, Statistics in Medicine), a minimally informative prior distribution is constructed. The required key input is the minimum (d_1 in the notation of the Appendix A.1 of that paper) and the maximum value (d_J) of the dose grid supplied to this function. Then threshmin is the probability threshold q_1 , such that any probability of DLT larger than q_1 has only 5% probability. Therefore q_1 is the 95% quantile of the beta distribution and hence $p_1 = 0.95$. Likewise, threshmax is the probability threshold q_J , such that any probability of DLT smaller than q_J has only 5% probability ($p_J = 0.05$). Subsequently, for all doses supplied in the dosegrid argument, beta distributions are set up from the assumption that the prior medians are linear in log-dose on the logit scale, and [Quantiles2LogisticNormal](#) is used to transform the resulting quantiles into an approximating [LogisticNormal](#) (or [LogisticLogNormal](#)) model. Note that the reference dose is not required for these computations.

Value

see [Quantiles2LogisticNormal](#)

minSize	<i>"MIN" combination of cohort size rules</i>
---------	---

Description

This function combines cohort size rules by taking the minimum of all sizes.

Usage

```
minSize(...)

## S4 method for signature 'CohortSize'
minSize(...)
```

Arguments

... Objects of class [CohortSize](#)

Value

the combination as an object of class [CohortSizeMin](#)

Methods (by class)

- CohortSize: The method combining cohort size rules by taking minimum

See Also

[maxSize](#)

Model-class	<i>Class for the model input</i>
-------------	----------------------------------

Description

This is the model class for single agent dose escalation, from which all other specific models inherit. It inherits all slots from [GeneralModel](#).

Details

The `datamodel` must obey the convention that the data input is called exactly as in the [Data](#) class. All prior distributions for parameters should be contained in the model function `priormodel`. The background is that this can be used to simulate from the prior distribution, before obtaining any data.

The dose function has as first argument `prob`, a scalar toxicity probability which is targeted. Additional arguments are model parameters. Then it computes, using model parameter(s) (samples), the resulting dose. Note that the model parameters are called exactly as in the `model` and must be included in the `sample` vector. The vectors of all samples for these parameters will then be supplied to the function. So your function must be able to process vectors of the model parameters, i.e. it must vectorize over them.

The `prob` function has as first argument `dose`, which is a scalar dose. Additional arguments are model parameters. Then it computes, using model parameter(s) (samples), the resulting probability of toxicity at that dose. Again here, the function must vectorize over the model parameters.

If you work with multivariate parameters, then please assume that your the two functions receive either one parameter value as a row vector, or a samples matrix where the rows correspond to the sampling index, i.e. the layout is then `nSamples x dimParameter`.

Note that `dose` and `prob` are the inverse functions of each other.

Slots

`dose` a function computing the dose reaching a specific target probability, based on the model parameters and additional prior settings (see the details above)

`prob` a function computing the probability of toxicity for a specific dose, based on the model parameters and additional prior settings (see the details above)

See Also

[LogisticNormal](#), [LogisticLogNormal](#), [LogisticLogNormalSub](#), [LogisticKadane](#), [DualEndpoint](#)

nextBest

Find the next best dose

Description

Compute the recommended next best dose.

Usage

```
nextBest(nextBest, doselimit, samples, model, data, ...)
```

```
## S4 method for signature 'NextBestMTD,numeric,Samples,Model,Data'
nextBest(nextBest, doselimit,
  samples, model, data, ...)
```

```
## S4 method for signature 'NextBestNCRM,numeric,Samples,Model,Data'
nextBest(nextBest,
  doselimit, samples, model, data, ...)
```

```
## S4 method for signature 'NextBestNCRM,numeric,Samples,Model,DataParts'
nextBest(nextBest,
  doselimit, samples, model, data, ...)

## S4 method for signature
## 'NextBestThreePlusThree,missing,missing,missing,Data'
nextBest(nextBest,
  doselimit, samples, model, data, ...)

## S4 method for signature
## 'NextBestDualEndpoint,numeric,Samples,DualEndpoint,Data'
nextBest(nextBest,
  doselimit, samples, model, data, ...)
```

Arguments

nextBest	The rule, an object of class NextBest
doselimit	The maximum allowed next dose. If this is an empty (length 0) vector, then no dose limit will be applied in the course of dose recommendation calculation, and a corresponding warning is given.
samples	the Samples object
model	The model input, an object of class Model
data	The data input, an object of class Data
...	possible additional arguments without method dispatch

Details

This function outputs the next best dose recommendation based on the corresponding rule nextBest, the posterior samples from the model and the underlying data.

Value

a list with the next best dose (element value) on the grid defined in data, and a plot depicting this recommendation (element plot)

Methods (by class)

- nextBest = NextBestMTD, doselimit = numeric, samples = Samples, model = Model, data = Data:
Find the next best dose based on the MTD rule
- nextBest = NextBestNCRM, doselimit = numeric, samples = Samples, model = Model, data = Data:
Find the next best dose based on the NCRM method
- nextBest = NextBestNCRM, doselimit = numeric, samples = Samples, model = Model, data = DataParts:
Find the next best dose based on the NCRM method when two parts trial is used
- nextBest = NextBestThreePlusThree, doselimit = missing, samples = missing, model = missing, data = Data:
Find the next best dose based on the 3+3 method
- nextBest = NextBestDualEndpoint, doselimit = numeric, samples = Samples, model = DualEndpoint, data = Data:
Find the next best dose based on the dual endpoint model

NextBest-class	<i>The virtual class for finding next best dose</i>
----------------	---

Description

The virtual class for finding next best dose

See Also

[NextBestMTD](#), [NextBestNCRM](#), [NextBestDualEndpoint](#), [NextBestThreePlusThree](#)

NextBestDualEndpoint	<i>Initialization function for "NextBestDualEndpoint"</i>
----------------------	---

Description

Initialization function for "NextBestDualEndpoint"

Usage

NextBestDualEndpoint(target, overdose, maxOverdoseProb)

Arguments

- target see [NextBestDualEndpoint](#)
- overdose see [NextBestDualEndpoint](#)
- maxOverdoseProb see [NextBestDualEndpoint](#)

Value

the [NextBestDualEndpoint](#) object

NextBestDualEndpoint-class	<i>The class with the input for finding the next dose based on the dual endpoint model</i>
----------------------------	--

Description

This rule first excludes all doses that exceed the probability maxOverdoseProb of having an overdose toxicity, as specified by the overdose interval overdose. Then, it picks under the remaining admissible doses the one that maximizes the probability to be in the target biomarker range, relative to the maximum biomarker level across the dose grid or relative to the Emax parameter in case a parametric model was selected (e.g. [DualEndpointBeta](#), [DualEndpointEmax](#)))

Slots

`target` the biomarker target range, relative to the maximum, that needs to be reached. For example, (0.8, 1.0) means we target a dose with at least 80 (0.5, 0.8) would mean that we target a dose between 50 the maximum biomarker level.

`overdose` the overdose toxicity interval (lower limit excluded, upper limit included)

`maxOverdoseProb` maximum overdose probability that is allowed

NextBestMTD	<i>Initialization function for class "NextBestMTD"</i>
-------------	--

Description

Initialization function for class "NextBestMTD"

Usage

```
NextBestMTD(target, derive)
```

Arguments

`target` see [NextBestMTD](#)

`derive` see [NextBestMTD](#)

Value

the [NextBestMTD](#) object

NextBestMTD-class	<i>The class with the input for finding the next best MTD estimate</i>
-------------------	--

Description

The class with the input for finding the next best MTD estimate

Slots

`target` the target toxicity probability

`derive` the function which derives from the input, a vector of posterior MTD samples called `mtdSamples`, the final next best MTD estimate.

NextBestNCRM	<i>Initialization function for "NextBestNCRM"</i>
--------------	---

Description

Initialization function for "NextBestNCRM"

Usage

NextBestNCRM(target, overdose, maxOverdoseProb)

Arguments

- target see [NextBestNCRM](#)
- overdose see [NextBestNCRM](#)
- maxOverdoseProb see [NextBestNCRM](#)

Value

the [NextBestNCRM](#) object

NextBestNCRM-class	<i>The class with the input for finding the next dose in target interval</i>
--------------------	--

Description

Note that to avoid numerical problems, the dose selection algorithm has been implemented as follows: First admissible doses are found, which are those with probability to fall in overdose category being below maxOverdoseProb. Next, within the admissible doses, the maximum probability to fall in the target category is calculated. If that is above 5% (i.e., it is not just numerical error), then the corresponding dose is the next recommended dose. Otherwise, the highest admissible dose is the next recommended dose.

Slots

- target the target toxicity interval (limits included)
- overdose the overdose toxicity interval (lower limit excluded, upper limit included)
- maxOverdoseProb maximum overdose probability that is allowed

NextBestThreePlusThree

Initialization function for "NextBestThreePlusThree"

Description

Initialization function for "NextBestThreePlusThree"

Usage

NextBestThreePlusThree()

Value

the [NextBestThreePlusThree](#) object

NextBestThreePlusThree-class

The class with the input for finding the next dose in target interval

Description

Implements the classical 3+3 dose recommendation. No input is required, hence this class has no slots.

or-Stopping-Stopping *The method combining two atomic stopping rules*

Description

The method combining two atomic stopping rules

Usage

```
## S4 method for signature 'Stopping,Stopping'
e1 | e2
```

Arguments

e1	First Stopping object
e2	Second Stopping object

Value

The [StoppingAny](#) object

or-Stopping-StoppingAny

The method combining a stopping list and an atomic

Description

The method combining a stopping list and an atomic

Usage

```
## S4 method for signature 'StoppingAny,Stopping'
e1 | e2
```

Arguments

e1 [StoppingAny](#) object
e2 [Stopping](#) object

Value

The modified [StoppingAny](#) object

or-StoppingAny-Stopping

The method combining an atomic and a stopping list

Description

The method combining an atomic and a stopping list

Usage

```
## S4 method for signature 'Stopping,StoppingAny'
e1 | e2
```

Arguments

e1 [Stopping](#) object
e2 [StoppingAny](#) object

Value

The modified [StoppingAny](#) object

`plot,Data,missing-method`*Plot method for the "Data" class*

Description

Plot method for the "Data" class

Usage

```
## S4 method for signature 'Data,missing'  
plot(x, y, ...)
```

Arguments

x	the Data object we want to plot
y	missing
...	not used

Value

the [ggplot](#) object

`plot,DataDual,missing-method`*Plot method for the "DataDual" class*

Description

Plot method for the "DataDual" class

Usage

```
## S4 method for signature 'DataDual,missing'  
plot(x, y, ...)
```

Arguments

x	the DataDual object we want to plot
y	missing
...	not used

Value

the [ggplot](#) object

```
plot, DualSimulations, missing-method
```

Plot dual-endpoint simulations

Description

This plot method can be applied to [DualSimulations](#) objects in order to summarize them graphically. In addition to the standard plot types, there is

sigma2W Plot a boxplot of the final biomarker variance estimates in the simulated trials

rho Plot a boxplot of the final correlation estimates in the simulated trials

Usage

```
## S4 method for signature 'DualSimulations,missing'
plot(x, y, type = c("trajectory",
  "dosesTried", "sigma2W", "rho"), ...)
```

Arguments

x	the DualSimulations object we want to plot from
y	missing
type	the type of plots you want to obtain.
...	not used

Value

A single [ggplot2](#) object if a single plot is asked for, otherwise a [gridExtra](#){gTree} object.

```
plot, DualSimulationsSummary, missing-method
```

Plot summaries of the dual-endpoint design simulations

Description

This plot method can be applied to [DualSimulationsSummary](#) objects in order to summarize them graphically. Possible type of plots at the moment are those listed in [plot, SimulationsSummary, missing-method](#) plus:

meanBiomarkerFit Plot showing the average fitted dose-biomarker curve across the trials, together with 95% credible intervals, and comparison with the assumed truth (as specified by the trueBiomarker argument to [summary, DualSimulations-method](#))

You can specify any subset of these in the type argument.

Usage

```
## S4 method for signature 'DualSimulationsSummary,missing'
plot(x, y, type = c("nObs",
  "doseSelected", "propDLTs", "nAboveTarget", "meanFit", "meanBiomarkerFit"),
  ...)
```

Arguments

x	the DualSimulationsSummary object we want to plot from
y	missing
type	the types of plots you want to obtain.
...	not used

Value

A single [ggplot2](#) object if a single plot is asked for, otherwise a [gridExtra](#){gTree} object.

plot, GeneralSimulations, missing-method
Plot simulations

Description

Summarize the simulations with plots

Usage

```
## S4 method for signature 'GeneralSimulations,missing'
plot(x, y, type = c("trajectory",
  "dosesTried"), ...)
```

Arguments

x	the GeneralSimulations object we want to plot from
y	missing
type	the type of plots you want to obtain.
...	not used

Details

This plot method can be applied to [GeneralSimulations](#) objects in order to summarize them graphically. Possible types of plots at the moment are:

trajectory Summary of the trajectory of the simulated trials

dosesTried Average proportions of the doses tested in patients

You can specify one or both of these in the type argument.

Value

A single [ggplot2](#) object if a single plot is asked for, otherwise a [gridExtra](#){gTree} object.

```
plot, GeneralSimulationsSummary, missing-method
```

Plot summaries of the general simulations

Description

Graphical display of the general simulation summary

Usage

```
## S4 method for signature 'GeneralSimulationsSummary,missing'
plot(x, y, type = c("nObs",
  "doseSelected", "propDLTs", "nAboveTarget"), ...)
```

Arguments

x	the GeneralSimulationsSummary object we want to plot from
y	missing
type	the types of plots you want to obtain.
...	not used

Details

This plot method can be applied to [GeneralSimulationsSummary](#) objects in order to summarize them graphically. Possible types of plots at the moment are:

nObs Distribution of the number of patients in the simulated trials

doseSelected Distribution of the final selected doses in the trials. Note that this can include zero entries, meaning that the trial was stopped because all doses in the dose grid appeared too toxic.

propDLTs Distribution of the proportion of patients with DLTs in the trials

nAboveTarget Distribution of the number of patients treated at doses which are above the target toxicity interval (as specified by the truth and target arguments to [summary, GeneralSimulations-method](#))

You can specify any subset of these in the type argument.

Value

A single [ggplot2](#) object if a single plot is asked for, otherwise a [gridExtra](#){gTree} object.

plot,Samples,DualEndpoint-method

Plotting dose-toxicity and dose-biomarker model fits

Description

When we have the dual endpoint model, also the dose-biomarker fit is shown in the plot

Usage

```
## S4 method for signature 'Samples,DualEndpoint'
plot(x, y, data, extrapolate = TRUE,
     showLegend = FALSE, ...)
```

Arguments

x	the Samples object
y	the DualEndpoint object
data	the DataDual object
extrapolate	should the biomarker fit be extrapolated to the whole dose grid? (default)
showLegend	should the legend be shown? (not default)
...	additional arguments for the parent method plot,Samples,Model-method

Value

This returns the [ggplot](#) object with the dose-toxicity and dose-biomarker model fits

plot,Samples,Model-method

Plotting dose-toxicity model fits

Description

Plotting dose-toxicity model fits

Usage

```
## S4 method for signature 'Samples,Model'
plot(x, y, data, ..., xlab = "Dose level",
     ylab = "Probability of DLT [%]", showLegend = TRUE)
```

Arguments

x	the Samples object
y	the Model object
data	the Data object
xlab	the x axis label
ylab	the y axis label
showLegend	should the legend be shown? (default)
...	not used

Value

This returns the [ggplot](#) object for the dose-toxicity model fit

```
plot, SimulationsSummary, missing-method
```

Plot summaries of the model-based design simulations

Description

Graphical display of the simulation summary

Usage

```
## S4 method for signature 'SimulationsSummary,missing'
plot(x, y, type = c("nObs",
  "doseSelected", "propDLTs", "nAboveTarget", "meanFit"), ...)
```

Arguments

x	the SimulationsSummary object we want to plot from
y	missing
type	the types of plots you want to obtain.
...	not used

Details

This plot method can be applied to [SimulationsSummary](#) objects in order to summarize them graphically. Possible type of plots at the moment are those listed in [plot, GeneralSimulationsSummary, missing-method](#) plus:

meanFit Plot showing the average fitted dose-toxicity curve across the trials, together with 95% credible intervals, and comparison with the assumed truth (as specified by the truth argument to [summary, Simulations-method](#))

You can specify any subset of these in the type argument.

Value

A single [ggplot2](#) object if a single plot is asked for, otherwise a [gridExtra](#){gTree} object.

plot.arrange	<i>Plots arrange objects</i>
--------------	------------------------------

Description

Plots arrange objects

Usage

```
## S3 method for class 'arrange'
plot(x, ...)
```

Arguments

x	the arrange object
...	additional parameters for grid.draw

prob	<i>Compute the probability for a given dose, given model and samples</i>
------	--

Description

Compute the probability for a given dose, given model and samples

Usage

```
prob(dose, model, samples, ...)

## S4 method for signature 'numeric,Model,Samples'
prob(dose, model, samples, ...)
```

Arguments

dose	the dose
model	the Model object
samples	the Samples
...	unused

Value

the vector (for [Model](#) objects) of probability samples.

Methods (by class)

- dose = numeric,model = [Model](#),samples = [Samples](#):

Quantiles2LogisticNormal

Convert prior quantiles (lower, median, upper) to logistic (log) normal model

Description

This function uses generalised simulated annealing to optimise a [LogisticNormal](#) model to be as close as possible to the given prior quantiles.

Usage

```
Quantiles2LogisticNormal(dosegrid, refDose, lower, median, upper,
  level = 0.95, logNormal = FALSE, parstart = NULL, parlower = c(-10,
  -10, 0, 0, -0.95), parupper = c(10, 10, 10, 10, 0.95), verbose = TRUE,
  control = list(threshold.stop = 0.01, maxit = 50000, temperature = 50000,
  max.time = 600))
```

Arguments

dosegrid	the dose grid
refDose	the reference dose
lower	the lower quantiles
median	the medians
upper	the upper quantiles
level	the credible level of the (lower, upper) intervals (default: 0.95)
logNormal	use the log-normal prior? (not default) otherwise, the normal prior for the logistic regression coefficients is used
parstart	starting values for the parameters. By default, these are determined from the medians supplied.
parlower	lower bounds on the parameters (intercept alpha and the slope beta, the corresponding standard deviations and the correlation.)
parupper	upper bounds on the parameters
verbose	be verbose? (default)
control	additional options for the optimisation routine, see GenSA for more details

Value

a list with the best approximating model ([LogisticNormal](#) or [LogisticLogNormal](#)), the resulting quantiles, the required quantiles and the distance to the required quantiles, as well as the final parameters (which could be used for running the algorithm a second time)

Report	<i>A Reference Class to represent sequentially updated reporting objects.</i>
--------	---

Description

A Reference Class to represent sequentially updated reporting objects.

Fields

object The object from which to report
df the data frame to which columns are sequentially added
dfNames the names to which strings are sequentially added

RuleDesign	<i>Initialization function for "RuleDesign"</i>
------------	---

Description

Initialization function for "RuleDesign"

Usage

```
RuleDesign(nextBest, cohortSize, data, startingDose)
```

Arguments

nextBest	see RuleDesign
cohortSize	see RuleDesign
data	see RuleDesign
startingDose	see RuleDesign

Value

the [RuleDesign](#) object

RuleDesign-class	<i>Class for rule-based designs</i>
------------------	-------------------------------------

Description

The difference to [Design](#) class is that model, stopping and increments slots are missing.

Slots

nextBest how to find the next best dose, an object of class [NextBest](#)
cohortSize rules for the cohort sizes, an object of class [CohortSize](#)
data what is the dose grid, any previous data, etc., contained in an object of class [Data](#)
startingDose what is the starting dose? Must lie on the grid in data

Samples	<i>Initialization function for "Samples"</i>
---------	--

Description

Initialization function for "Samples"

Usage

```
Samples(data, options)
```

Arguments

data see [Samples](#)
options see [Samples](#)

Value

the [Samples](#) object

Samples-class	<i>Class for the MCMC output</i>
---------------	----------------------------------

Description

Class for the MCMC output

Slots

data a list where each entry contains the samples of a (vector-valued) parameter in a vector/matrix in the format (number of samples) x (dimension of the parameter).
options the [McmcOptions](#) which have been used

sampleSize	<i>Compute the number of samples for a given MCMC options triple</i>
------------	--

Description

Compute the number of samples for a given MCMC options triple

Usage

```
sampleSize(mcmcOptions)
```

Arguments

mcmcOptions the [McmcOptions](#) object

Value

the resulting sample size

setSeed

Helper function to set and save the RNG seed

Description

This is basically copied from simulate.lm

Usage

```
setSeed(seed = NULL)
```

Arguments

seed	an object specifying if and how the random number generator should be initialized (“seeded”). Either NULL (default) or an integer that will be used in a call to set.seed before simulating the response vectors. If set, the value is saved as the seed slot of the returned object. The default, NULL will not change the random generator state.
------	---

Value

The RNGstate will be returned, in order to call this function with this input to reproduce the obtained simulation results

Author(s)

Daniel Sabanes Bove <sabanesd@roche.com>

show, DualSimulationsSummary-method

Show the summary of the dual-endpoint simulations

Description

Show the summary of the dual-endpoint simulations

Usage

```
## S4 method for signature 'DualSimulationsSummary'
show(object)
```

Arguments

object	the DualSimulationsSummary object we want to print
--------	--

Value

invisibly returns a data frame of the results with one row and appropriate column names

show, GeneralSimulationsSummary-method
Show the summary of the simulations

Description

Show the summary of the simulations

Usage

```
## S4 method for signature 'GeneralSimulationsSummary'  
show(object)
```

Arguments

object the [GeneralSimulationsSummary](#) object we want to print

Value

invisibly returns a data frame of the results with one row and appropriate column names

show, SimulationsSummary-method
Show the summary of the simulations

Description

Show the summary of the simulations

Usage

```
## S4 method for signature 'SimulationsSummary'  
show(object)
```

Arguments

object the [SimulationsSummary](#) object we want to print

Value

invisibly returns a data frame of the results with one row and appropriate column names

simulate, Design-method

Simulate outcomes from a CRM design

Description

Simulate outcomes from a CRM design

Usage

```
## S4 method for signature 'Design'
simulate(object, nsim = 1L, seed = NULL, truth,
  args = NULL, firstSeparate = FALSE, mcmcOptions = McmcOptions(),
  parallel = FALSE, ...)
```

Arguments

object	the Design object we want to simulate data from
nsim	the number of simulations (default: 1)
seed	see setSeed
truth	a function which takes as input a dose (vector) and returns the true probability (vector) for toxicity. Additional arguments can be supplied in args.
args	data frame with arguments for the truth function. The column names correspond to the argument names, the rows to the values of the arguments. The rows are appropriately recycled in the nsim simulations. In order to produce outcomes from the posterior predictive distribution, e.g, pass an object that contains the data observed so far, truth contains the prob function from the model in object, and args contains posterior samples from the model.
firstSeparate	enroll the first patient separately from the rest of the cohort? (not default) If yes, the cohort will be closed if a DLT occurs in this patient.
mcmcOptions	object of class McmcOptions , giving the MCMC options for each evaluation in the trial. By default, the standard options are used
parallel	should the simulation runs be parallelized across the clusters of the computer? (not default)
...	not used

Value

an object of class [Simulations](#)

simulate, DualDesign-method

Simulate outcomes from a dual-endpoint design

Description

Simulate outcomes from a dual-endpoint design

Usage

```
## S4 method for signature 'DualDesign'
simulate(object, nsim = 1L, seed = NULL, trueTox,
  trueBiomarker, args = NULL, sigma2W, rho = 0, firstSeparate = FALSE,
  mcmcOptions = McmcOptions(), parallel = FALSE, ...)
```

Arguments

object	the DualDesign object we want to simulate data from
nsim	the number of simulations (default: 1)
seed	see setSeed
trueTox	a function which takes as input a dose (vector) and returns the true probability (vector) for toxicity. Additional arguments can be supplied in args.
trueBiomarker	a function which takes as input a dose (vector) and returns the true biomarker level (vector). Additional arguments can be supplied in args.
args	data frame with arguments for the trueTox and trueBiomarker function. The column names correspond to the argument names, the rows to the values of the arguments. The rows are appropriately recycled in the nsim simulations.
sigma2W	variance for the biomarker measurements
rho	correlation between toxicity and biomarker measurements (default: 0)
firstSeparate	enroll the first patient separately from the rest of the cohort? (not default) If yes, the cohort will be closed if a DLT occurs in this patient.
mcmcOptions	object of class McmcOptions , giving the MCMC options for each evaluation in the trial. By default, the standard options are used
parallel	should the simulation runs be parallelized across the clusters of the computer? (not default)
...	not used

Value

an object of class [DualSimulations](#)

 simulate, RuleDesign-method

Simulate outcomes from a rule-based design

Description

Simulate outcomes from a rule-based design

Usage

```
## S4 method for signature 'RuleDesign'
simulate(object, nsim = 1L, seed = NULL, truth,
         args = NULL, parallel = FALSE, ...)
```

Arguments

object	the RuleDesign object we want to simulate data from
nsim	the number of simulations (default: 1)
seed	see setSeed
truth	a function which takes as input a dose (vector) and returns the true probability (vector) for toxicity. Additional arguments can be supplied in args.
args	data frame with arguments for the truth function. The column names correspond to the argument names, the rows to the values of the arguments. The rows are appropriately recycled in the nsim simulations.
parallel	should the simulation runs be parallelized across the clusters of the computer? (not default)
...	not used

Value

an object of class [GeneralSimulations](#)

 Simulations

Initialization function for the "Simulations" class

Description

Initialization function for the "Simulations" class

Usage

```
Simulations(fit, stopReasons, ...)
```

Arguments

fit	see Simulations
stopReasons	see Simulations
...	additional parameters from GeneralSimulations

Value

the [Simulations](#) object

Simulations-class	<i>Class for the simulations output from model based designs</i>
-------------------	--

Description

This class captures the trial simulations from model based designs. Additional slots fit and stopReasons compared to the general class [GeneralSimulations](#).

Slots

- fit list with the final fits
- stopReasons list of stopping reasons for each simulation run

SimulationsSummary-class	<i>Class for the summary of model-based simulations output</i>
--------------------------	--

Description

In addition to the slots in the parent class [GeneralSimulationsSummary](#), it contains two slots with model fit information.

Details

Note that objects should not be created by users, therefore no initialization function is provided for this class.

Slots

- fitAtDoseMostSelected fitted toxicity rate at dose most often selected
- meanFit list with the average, lower (2.5 quantiles of the mean fitted toxicity at each dose level

size	<i>Determine the size of the next cohort</i>
------	--

Description

This function determines the size of the next cohort.

Usage

```
size(cohortSize, dose, data, ...)

## S4 method for signature 'CohortSizeRange,ANY,Data'
size(cohortSize, dose, data, ...)

## S4 method for signature 'CohortSizeDLT,ANY,Data'
size(cohortSize, dose, data, ...)

## S4 method for signature 'CohortSizeMax,ANY,Data'
size(cohortSize, dose, data, ...)

## S4 method for signature 'CohortSizeMin,ANY,Data'
size(cohortSize, dose, data, ...)

## S4 method for signature 'CohortSizeConst,ANY,Data'
size(cohortSize, dose, data, ...)

## S4 method for signature 'CohortSizeParts,ANY,DataParts'
size(cohortSize, dose, data, ...)
```

Arguments

cohortSize	The rule, an object of class CohortSize
dose	the next dose
data	The data input, an object of class Data
...	additional arguments

Value

the size as integer value

Methods (by class)

- cohortSize = CohortSizeRange, dose = ANY, data = Data: Determine the cohort size based on the range into which the next dose falls into
- cohortSize = CohortSizeDLT, dose = ANY, data = Data: Determine the cohort size based on the number of DLTs so far
- cohortSize = CohortSizeMax, dose = ANY, data = Data: Size based on maximum of multiple cohort size rules
- cohortSize = CohortSizeMin, dose = ANY, data = Data: Size based on minimum of multiple cohort size rules

- cohortSize = CohortSizeConst,dose = ANY,data = Data: Constant cohort size
- cohortSize = CohortSizeParts,dose = ANY,data = DataParts: Cohort size based on the parts

Stopping-class	<i>The virtual class for stopping rules</i>
----------------	---

Description

The virtual class for stopping rules

See Also

[StoppingList](#), [StoppingCohortsNearDose](#), [StoppingPatientsNearDose](#), [StoppingMinCohorts](#), [StoppingMinPatients](#), [StoppingTargetProb](#) [StoppingMTDdistribution](#), [StoppingTargetBiomarker](#)

StoppingAll	<i>Initialization function for "StoppingAll"</i>
-------------	--

Description

Initialization function for "StoppingAll"

Usage

StoppingAll(stopList)

Arguments

stopList see [StoppingAll](#)

Value

the [StoppingAll](#) object

StoppingAll-class	<i>Stop based on fulfillment of all multiple stopping rules</i>
-------------------	---

Description

This class can be used to combine multiple stopping rules with an AND operator.

Details

stopList contains all stopping rules, which are again objects of class [Stopping](#). All stopping rules must be fulfilled in order that the result of this rule is to stop.

Slots

stopList list of stopping rules

StoppingAny	<i>Initialization function for "StoppingAny"</i>
-------------	--

Description

Initialization function for "StoppingAny"

Usage

StoppingAny(stopList)

Arguments

stopList see [StoppingAny](#)

Value

the [StoppingAny](#) object

StoppingAny-class	<i>Stop based on fulfillment of any stopping rule</i>
-------------------	---

Description

This class can be used to combine multiple stopping rules with an OR operator.

Details

stopList contains all stopping rules, which are again objects of class [Stopping](#). Any of these rules must be fulfilled in order that the result of this rule is to stop.

Slots

stopList list of stopping rules

StoppingCohortsNearDose

Initialization function for "StoppingCohortsNearDose"

Description

Initialization function for "StoppingCohortsNearDose"

Usage

StoppingCohortsNearDose(nCohorts, percentage)

Arguments

nCohorts see [StoppingCohortsNearDose](#)

percentage see [StoppingCohortsNearDose](#)

Value

the [StoppingCohortsNearDose](#) object

StoppingCohortsNearDose-class

Stop based on number of cohorts near to next best dose

Description

Stop based on number of cohorts near to next best dose

Slots

nCohorts number of required cohorts

percentage percentage (between 0 and 100) within the next best dose the cohorts must lie

StoppingList

Initialization function for "StoppingList"

Description

Initialization function for "StoppingList"

Usage

StoppingList(stopList, summary)

Arguments

stopList see [StoppingList](#)
summary see [StoppingList](#)

Value

the [StoppingList](#) object

StoppingList-class *Stop based on multiple stopping rules*

Description

This class can be used to combine multiple stopping rules.

Details

stopList contains all stopping rules, which are again objects of class [Stopping](#), and the summary is a function taking a logical vector of the size of stopList and returning a single logical value. For example, if the function all is given as summary function, then this means that all stopping rules must be fulfilled in order that the result of this rule is to stop.

Slots

stopList list of stopping rules
summary the summary function to combine the results of the stopping rules into a single result

StoppingMinCohorts *Initialization function for "StoppingMinCohorts"*

Description

Initialization function for "StoppingMinCohorts"

Usage

StoppingMinCohorts(nCohorts)

Arguments

nCohorts see [StoppingMinCohorts](#)

Value

the [StoppingMinCohorts](#) object

StoppingMinCohorts-class

Stop based on minimum number of cohorts

Description

Stop based on minimum number of cohorts

Slots

nCohorts minimum required number of cohorts

StoppingMinPatients

Initialization function for "StoppingMinPatients"

Description

Initialization function for "StoppingMinPatients"

Usage

StoppingMinPatients(nPatients)

Arguments

nPatients see [StoppingMinPatients](#)

Value

the [StoppingMinPatients](#) object

StoppingMinPatients-class

Stop based on minimum number of patients

Description

Stop based on minimum number of patients

Slots

nPatients minimum allowed number of patients

StoppingMTDdistribution

Initialization function for "StoppingMTDdistribution"

Description

Initialization function for "StoppingMTDdistribution"

Usage

StoppingMTDdistribution(target, thresh, prob)

Arguments

target see [StoppingMTDdistribution](#)

thresh see [StoppingMTDdistribution](#)

prob see [StoppingMTDdistribution](#)

Value

the [StoppingMTDdistribution](#) object

StoppingMTDdistribution-class

Stop based on MTD distribution

Description

Has 90% probability above a threshold of 50% of the current MTD been reached? This class is used for this question.

Slots

target the target toxicity probability (e.g. 0.33) defining the MTD

thresh the threshold relative to the MTD (e.g. 0.5)

prob required probability (e.g. 0.9)

StoppingPatientsNearDose

Initialization function for "StoppingPatientsNearDose"

Description

Initialization function for "StoppingPatientsNearDose"

Usage

StoppingPatientsNearDose(nPatients, percentage)

Arguments

nPatients see [StoppingPatientsNearDose](#)
percentage see [StoppingPatientsNearDose](#)

Value

the [StoppingPatientsNearDose](#) object

StoppingPatientsNearDose-class

Stop based on number of patients near to next best dose

Description

Stop based on number of patients near to next best dose

Slots

nPatients number of required patients
percentage percentage (between 0 and 100) within the next best dose the patients must lie

StoppingTargetBiomarker

Initialization function for "StoppingTargetBiomarker"

Description

Initialization function for "StoppingTargetBiomarker"

Usage

StoppingTargetBiomarker(target, prob)

Arguments

target see [StoppingTargetBiomarker](#)
prob see [StoppingTargetBiomarker](#)

Value

the [StoppingTargetBiomarker](#) object

StoppingTargetBiomarker-class
Stop based on probability of target biomarker

Description

Stop based on probability of target biomarker

Slots

target the biomarker target range, relative to the maximum, that needs to be reached. So this must be a probability range (1 is allowed here)
prob required target probability for reaching sufficient precision

StoppingTargetProb *Initialization function for "StoppingTargetProb"*

Description

Initialization function for "StoppingTargetProb"

Usage

StoppingTargetProb(target, prob)

Arguments

target see [StoppingTargetProb](#)
prob see [StoppingTargetProb](#)

Value

the [StoppingTargetProb](#) object

StoppingTargetProb-class

Stop based on probability of target tox interval

Description

Stop based on probability of target tox interval

Slots

target the target toxicity interval, e.g. `c(0.2, 0.35)`

prob required target toxicity probability (e.g. 0.4) for reaching sufficient precision

stopTrial

Stop the trial?

Description

This function returns whether to stop the trial.

Usage

```
stopTrial(stopping, dose, samples, model, data, ...)
```

```
## S4 method for signature 'StoppingList,ANY,ANY,ANY,ANY'
stopTrial(stopping, dose, samples,
  model, data, ...)
```

```
## S4 method for signature 'StoppingAll,ANY,ANY,ANY,ANY'
stopTrial(stopping, dose, samples,
  model, data, ...)
```

```
## S4 method for signature 'StoppingAny,ANY,ANY,ANY,ANY'
stopTrial(stopping, dose, samples,
  model, data, ...)
```

```
## S4 method for signature 'StoppingCohortsNearDose,numeric,ANY,ANY,Data'
stopTrial(stopping,
  dose, samples, model, data, ...)
```

```
## S4 method for signature 'StoppingPatientsNearDose,numeric,ANY,ANY,Data'
stopTrial(stopping,
  dose, samples, model, data, ...)
```

```
## S4 method for signature 'StoppingMinCohorts,ANY,ANY,ANY,Data'
stopTrial(stopping, dose,
  samples, model, data, ...)
```

```
## S4 method for signature 'StoppingMinPatients,ANY,ANY,ANY,Data'
```

```

stopTrial(stopping, dose,
          samples, model, data, ...)

## S4 method for signature 'StoppingTargetProb,numeric,Samples,Model,ANY'
stopTrial(stopping,
          dose, samples, model, data, ...)

## S4 method for signature 'StoppingMTDdistribution,numeric,Samples,Model,ANY'
stopTrial(stopping,
          dose, samples, model, data, ...)

## S4 method for signature
## 'StoppingTargetBiomarker,numeric,Samples,DualEndpoint,ANY'
stopTrial(stopping,
          dose, samples, model, data, ...)

```

Arguments

stopping	The rule, an object of class Stopping
dose	the recommended next best dose
samples	the Samples object
model	The model input, an object of class Model
data	The data input, an object of class Data
...	additional arguments

Value

logical value: TRUE if the trial can be stopped, FALSE otherwise. It should have an attribute message which gives the reason for the decision.

Methods (by class)

- stopping = StoppingList, dose = ANY, samples = ANY, model = ANY, data = ANY: Stop based on multiple stopping rules
- stopping = StoppingAll, dose = ANY, samples = ANY, model = ANY, data = ANY: Stop based on fulfillment of all multiple stopping rules
- stopping = StoppingAny, dose = ANY, samples = ANY, model = ANY, data = ANY: Stop based on fulfillment of any stopping rule
- stopping = StoppingCohortsNearDose, dose = numeric, samples = ANY, model = ANY, data = Data: Stop based on number of cohorts near to next best dose
- stopping = StoppingPatientsNearDose, dose = numeric, samples = ANY, model = ANY, data = Data: Stop based on number of patients near to next best dose
- stopping = StoppingMinCohorts, dose = ANY, samples = ANY, model = ANY, data = Data: Stop based on minimum number of cohorts
- stopping = StoppingMinPatients, dose = ANY, samples = ANY, model = ANY, data = Data: Stop based on minimum number of patients
- stopping = StoppingTargetProb, dose = numeric, samples = Samples, model = Model, data = ANY: Stop based on probability of target tox interval

- stopping = StoppingMTDdistribution, dose = numeric, samples = Samples, model = Model, data = ANY:
Stop based on MTD distribution
- stopping = StoppingTargetBiomarker, dose = numeric, samples = Samples, model = DualEndpoint, data = ANY:
Stop based on probability of targeting biomarker

summary, DualSimulations-method

Summarize the dual-endpoint design simulations, relative to given true dose-toxicity and dose-biomarker curves

Description

Summarize the dual-endpoint design simulations, relative to given true dose-toxicity and dose-biomarker curves

Usage

```
## S4 method for signature 'DualSimulations'
summary(object, trueTox, trueBiomarker,
        target = c(0.2, 0.35), ...)
```

Arguments

object	the DualSimulations object we want to summarize
trueTox	a function which takes as input a dose (vector) and returns the true probability (vector) for toxicity.
trueBiomarker	a function which takes as input a dose (vector) and returns the true biomarker level (vector).
target	the target toxicity interval (default: 20-35%) used for the computations
...	Additional arguments can be supplied here for trueTox and trueBiomarker

Value

an object of class [DualSimulationsSummary](#)

summary, GeneralSimulations-method

Summarize the simulations, relative to a given truth

Description

Summarize the simulations, relative to a given truth

Usage

```
## S4 method for signature 'GeneralSimulations'
summary(object, truth, target = c(0.2, 0.35),
        ...)
```


Arguments

object	the GeneralSimulations object we want to summarize
truth	a function which takes as input a dose (vector) and returns the true probability (vector) for toxicity
target	the target toxicity interval (default: 20-35%) used for the computations
...	Additional arguments can be supplied here for truth

Value

an object of class [GeneralSimulationsSummary](#)

summary, Simulations-method

Summarize the model-based design simulations, relative to a given truth

Description

Summarize the model-based design simulations, relative to a given truth

Usage

```
## S4 method for signature 'Simulations'
summary(object, truth, target = c(0.2, 0.35), ...)
```

Arguments

object	the Simulations object we want to summarize
truth	a function which takes as input a dose (vector) and returns the true probability (vector) for toxicity
target	the target toxicity interval (default: 20-35%) used for the computations
...	Additional arguments can be supplied here for truth

Value

an object of class [SimulationsSummary](#)

ThreePlusThreeDesign *Creates a new 3+3 design object from a dose grid*

Description

Creates a new 3+3 design object from a dose grid

Usage

```
ThreePlusThreeDesign(doseGrid)
```

Arguments

doseGrid the dose grid to be used

Value

the object of class [RuleDesign](#) with the 3+3 design

Author(s)

Daniel Sabanes Bove <sabanesd@roche.com>

update,Data-method *Update method for the "Data" class*

Description

Add new data to the [Data](#) object

Usage

```
## S4 method for signature 'Data'
update(object, x, y, ID = (if (length(object@ID))
  max(object@ID) else 0L) + seq_along(y), ...)
```

Arguments

object	the old Data object
x	the dose level (one level only!)
y	the DLT vector (0/1 vector), for all patients in this cohort
ID	the patient IDs
...	not used

Value

the new [Data](#) object

 update,DataDual-method

Update method for the "DataDual" class

Description

Add new data to the [DataDual](#) object

Usage

```
## S4 method for signature 'DataDual'
update(object, x, y, w, ID = (if (length(object@ID))
  max(object@ID) else 0L) + seq_along(y), ...)
```

Arguments

object	the old DataDual object
x	the dose level (one level only!)
y	the DLT vector (0/1 vector), for all patients in this cohort
w	the biomarker vector, for all patients in this cohort
ID	the patient IDs
...	not used

Value

the new [DataDual](#) object

 update,DataParts-method

Update method for the "DataParts" class

Description

Add new data to the [DataParts](#) object

Usage

```
## S4 method for signature 'DataParts'
update(object, x, y, ID = (if (length(object@ID))
  max(object@ID) else 0L) + seq_along(y), ...)
```

Arguments

object	the old DataParts object
x	the dose level (one level only!)
y	the DLT vector (0/1 vector), for all patients in this cohort
ID	the patient IDs
...	not used

Value

the new [DataParts](#) object

Validate

A Reference Class to help programming validation for new S4 classes

Description

Starting from an empty msg vector, with each check that is returning FALSE the vector gets a new element - the string explaining the failure of the validation

Fields

msg the message character vector

writeModel

Creating a WinBUGS model file

Description

Convert R function to a **WinBUGS** model file. BUGS models follow closely S syntax. It is therefore possible to write most BUGS models as R functions. As a difference, BUGS syntax allows truncation specification like this: `dnorm(...)` `I(...)` but this is illegal in R. To overcome this incompatibility, use dummy operator `%_%` before `I(...)`: `dnorm(...)` `%_% I(...)`. The dummy operator `%_%` will be removed before the BUGS code is saved. In S-PLUS, a warning is generated when the model function is defined if the last statement in the model is an assignment. To avoid this warning, add the line `invisible()` to the end of the model definition. This line will be removed before the BUGS code is saved.

Usage

```
writeModel(model, con = "model.bug", digits = 5)
```

Arguments

model	R function containing the BUGS model in the BUGS model language, for minor differences see Section Details.
con	passed to writelines which actually writes the model file
digits	number of significant digits used for WinBUGS input, see formatC

Value

Nothing, but as a side effect, the model file is written

Author(s)

original idea by Jouni Kerman, modified by Uwe Ligges, DSB removed S Plus part

&,Stopping,Stopping-method

The method combining two atomic stopping rules

Description

The method combining two atomic stopping rules

Usage

```
## S4 method for signature 'Stopping,Stopping'
e1 & e2
```

Arguments

e1	First Stopping object
e2	Second Stopping object

Value

The [StoppingAll](#) object

&,Stopping,StoppingAll-method

The method combining an atomic and a stopping list

Description

The method combining an atomic and a stopping list

Usage

```
## S4 method for signature 'Stopping,StoppingAll'
e1 & e2
```

Arguments

e1	Stopping object
e2	StoppingAll object

Value

The modified [StoppingAll](#) object

&,StoppingAll,Stopping-method

The method combining a stopping list and an atomic

Description

The method combining a stopping list and an atomic

Usage

```
## S4 method for signature 'StoppingAll,Stopping'  
e1 & e2
```

Arguments

e1 [StoppingAll](#) object
e2 [Stopping](#) object

Value

The modified [StoppingAll](#) object

Index

*Topic **classes**

CohortSize-class, [7](#)
CohortSizeConst-class, [8](#)
CohortSizeDLT-class, [9](#)
CohortSizeMax-class, [9](#)
CohortSizeMin-class, [10](#)
CohortSizeParts-class, [11](#)
CohortSizeRange-class, [11](#)
Data-class, [14](#)
DataDual-class, [15](#)
DataParts-class, [16](#)
Design-class, [16](#)
DualDesign-class, [18](#)
DualEndpoint-class, [19](#)
DualEndpointBeta-class, [20](#)
DualEndpointEmax-class, [21](#)
DualEndpointRW-class, [22](#)
DualSimulations-class, [23](#)
DualSimulationsSummary-class, [24](#)
GeneralData-class, [26](#)
GeneralModel-class, [26](#)
GeneralSimulations-class, [27](#)
GeneralSimulationsSummary-class, [28](#)
Increments-class, [29](#)
IncrementsRelative-class, [30](#)
IncrementsRelativeDLT-class, [31](#)
IncrementsRelativeParts-class, [32](#)
LogisticKadane-class, [34](#)
LogisticLogNormal-class, [35](#)
LogisticLogNormalSub-class, [36](#)
LogisticNormal-class, [37](#)
LogisticNormalFixedMixture-class, [38](#)
LogisticNormalMixture-class, [40](#)
McmcOptions-class, [44](#)
Model-class, [45](#)
NextBest-class, [48](#)
NextBestDualEndpoint-class, [48](#)
NextBestMTD-class, [49](#)
NextBestNCRM-class, [50](#)
NextBestThreePlusThree-class, [51](#)
RuleDesign-class, [61](#)

Samples-class, [62](#)
Simulations-class, [68](#)
SimulationsSummary-class, [68](#)
Stopping-class, [70](#)
StoppingAll-class, [70](#)
StoppingAny-class, [71](#)
StoppingCohortsNearDose-class, [72](#)
StoppingList-class, [73](#)
StoppingMinCohorts-class, [74](#)
StoppingMinPatients-class, [74](#)
StoppingMTDdistribution-class, [75](#)
StoppingPatientsNearDose-class, [76](#)
StoppingTargetBiomarker-class, [77](#)
StoppingTargetProb-class, [78](#)

*Topic **documentation**

crmPackExample, [12](#)
crmPackHelp, [12](#)

*Topic **methods**

&, Stopping, Stopping-method, [85](#)
&, Stopping, StoppingAll-method, [85](#)
&, StoppingAll, Stopping-method, [86](#)
approximate, [5](#)
as.list, GeneralData-method, [6](#)
biomLevel, [7](#)
CohortSizeConst, [8](#)
CohortSizeDLT, [8](#)
CohortSizeMax, [9](#)
CohortSizeMin, [10](#)
CohortSizeParts, [10](#)
CohortSizeRange, [11](#)
Design, [16](#)
dose, [17](#)
DualDesign, [17](#)
DualEndpoint, [18](#)
DualEndpointBeta, [20](#)
DualEndpointEmax, [21](#)
DualEndpointRW, [22](#)
DualSimulations, [23](#)
examine, [24](#)
fit, [25](#)
GeneralSimulations, [27](#)
get, Samples, character-method, [28](#)
IncrementsRelative, [30](#)

- IncrementsRelativeDLT, [31](#)
- IncrementsRelativeParts, [32](#)
- initialize, DualEndpointOld-method, [33](#)
- LogisticKadane, [33](#)
- LogisticLogNormal, [34](#)
- LogisticLogNormalSub, [35](#)
- LogisticNormal, [37](#)
- LogisticNormalFixedMixture, [38](#)
- LogisticNormalMixture, [39](#)
- maxDose, [41](#)
- maxSize, [42](#)
- mcmc, [42](#)
- McmcOptions, [43](#)
- minSize, [45](#)
- nextBest, [46](#)
- NextBestDualEndpoint, [48](#)
- NextBestMTD, [49](#)
- NextBestNCRM, [50](#)
- NextBestThreePlusThree, [51](#)
- or-Stopping-Stopping, [51](#)
- or-Stopping-StoppingAny, [52](#)
- or-StoppingAny-Stopping, [52](#)
- plot, Data, missing-method, [53](#)
- plot, DataDual, missing-method, [53](#)
- plot, DualSimulations, missing-method, [54](#)
- plot, DualSimulationsSummary, missing-method, [54](#)
- plot, GeneralSimulations, missing-method, [55](#)
- plot, GeneralSimulationsSummary, missing-method, [56](#)
- plot, SimulationsSummary, missing-method, [58](#)
- prob, [59](#)
- RuleDesign, [61](#)
- Samples, [62](#)
- show, DualSimulationsSummary-method, [63](#)
- show, GeneralSimulationsSummary-method, [64](#)
- show, SimulationsSummary-method, [64](#)
- simulate, Design-method, [65](#)
- simulate, DualDesign-method, [66](#)
- simulate, RuleDesign-method, [67](#)
- Simulations, [67](#)
- size, [69](#)
- StoppingAll, [70](#)
- StoppingAny, [71](#)
- StoppingCohortsNearDose, [72](#)
- StoppingList, [72](#)
- StoppingMinCohorts, [73](#)
- StoppingMinPatients, [74](#)
- StoppingMTDdistribution, [75](#)
- StoppingPatientsNearDose, [76](#)
- StoppingTargetBiomarker, [76](#)
- StoppingTargetProb, [77](#)
- stopTrial, [78](#)
- summary, DualSimulations-method, [80](#)
- summary, GeneralSimulations-method, [80](#)
- summary, Simulations-method, [81](#)
- update, Data-method, [82](#)
- update, DataDual-method, [83](#)
- update, DataParts-method, [83](#)
- *Topic package**
 - crmPack-package, [5](#)
- *Topic programming**
 - crmPackUpgrade, [13](#)
 - Data, [13](#)
 - DataDual, [14](#)
 - DataParts, [15](#)
 - getMinInfBeta, [29](#)
 - logit, [40](#)
 - MinimalInformative, [44](#)
 - Quantiles2LogisticNormal, [60](#)
 - sampleSize, [62](#)
 - setSeed, [63](#)
 - ThreePlusThreeDesign, [82](#)
- *Topic regression**
 - examine, [24](#)
 - mcmc, [42](#)
- tools**
 - CohortSizeConst (CohortSizeConst-class), [8](#)
 - .CohortSizeDLT (CohortSizeDLT-class), [9](#)
 - .CohortSizeMax (CohortSizeMax-class), [9](#)
 - .CohortSizeMin (CohortSizeMin-class), [10](#)
 - .CohortSizeParts (CohortSizeParts-class), [11](#)
 - .CohortSizeRange (CohortSizeRange-class), [11](#)
 - .Data (Data-class), [14](#)
 - .DataDual (DataDual-class), [15](#)
 - .DataParts (DataParts-class), [16](#)
 - .Design (Design-class), [16](#)
 - .DualDesign (DualDesign-class), [18](#)
 - .DualEndpoint (DualEndpoint-class), [19](#)
 - .DualEndpointBeta (DualEndpointBeta-class), [20](#)
 - .DualEndpointEmax (DualEndpointEmax-class), [21](#)
 - .DualEndpointRW (DualEndpointRW-class), [22](#)

- .DualSimulations
 - (DualSimulations-class), [23](#)
- .DualSimulationsSummary
 - (DualSimulationsSummary-class), [24](#)
- .GeneralData (GeneralData-class), [26](#)
- .GeneralModel (GeneralModel-class), [26](#)
- .GeneralSimulations
 - (GeneralSimulations-class), [27](#)
- .GeneralSimulationsSummary
 - (GeneralSimulationsSummary-class), [28](#)
- .IncrementsRelative
 - (IncrementsRelative-class), [30](#)
- .IncrementsRelativeDLT
 - (IncrementsRelativeDLT-class), [31](#)
- .IncrementsRelativeParts
 - (IncrementsRelativeParts-class), [32](#)
- .LogisticKadane (LogisticKadane-class), [34](#)
- .LogisticLogNormal
 - (LogisticLogNormal-class), [35](#)
- .LogisticLogNormalSub
 - (LogisticLogNormalSub-class), [36](#)
- .LogisticNormal (LogisticNormal-class), [37](#)
- .LogisticNormalFixedMixture
 - (LogisticNormalFixedMixture-class), [38](#)
- .LogisticNormalMixture
 - (LogisticNormalMixture-class), [40](#)
- .McmcOptions (McmcOptions-class), [44](#)
- .Model (Model-class), [45](#)
- .NextBestDualEndpoint
 - (NextBestDualEndpoint-class), [48](#)
- .NextBestMTD (NextBestMTD-class), [49](#)
- .NextBestNCRM (NextBestNCRM-class), [50](#)
- .NextBestThreePlusThree
 - (NextBestThreePlusThree-class), [51](#)
- .RuleDesign (RuleDesign-class), [61](#)
- .Samples (Samples-class), [62](#)
- .Simulations (Simulations-class), [68](#)
- .SimulationsSummary
 - (SimulationsSummary-class), [68](#)
- .StoppingAll (StoppingAll-class), [70](#)
- .StoppingAny (StoppingAny-class), [71](#)
- .StoppingCohortsNearDose
 - (StoppingCohortsNearDose-class), [72](#)
- .StoppingList (StoppingList-class), [73](#)
- .StoppingMTDdistribution
 - (StoppingMTDdistribution-class), [75](#)
- .StoppingMinCohorts
 - (StoppingMinCohorts-class), [74](#)
- .StoppingMinPatients
 - (StoppingMinPatients-class), [74](#)
- .StoppingPatientsNearDose
 - (StoppingPatientsNearDose-class), [76](#)
- .StoppingTargetBiomarker
 - (StoppingTargetBiomarker-class), [77](#)
- .StoppingTargetProb
 - (StoppingTargetProb-class), [78](#)
- &, Stopping, Stopping-method, [85](#)
- &, Stopping, StoppingAll-method, [85](#)
- &, StoppingAll, Stopping-method, [86](#)
- approximate, [5](#)
- approximate, Samples-method (approximate), [5](#)
- as.list, GeneralData-method, [6](#)
- biomLevel, [7](#)
- biomLevel, numeric, DualEndpoint, Samples-method (biomLevel), [7](#)
- CohortSize, [9](#), [10](#), [42](#), [45](#), [61](#), [69](#)
- CohortSize-class, [7](#)
- CohortSizeConst, [7](#), [8](#), [8](#)
- CohortSizeConst-class, [8](#)
- CohortSizeDLT, [7](#), [8](#), [8](#)
- CohortSizeDLT-class, [9](#)
- CohortSizeMax, [7](#), [9](#), [9](#), [42](#)
- CohortSizeMax-class, [9](#)
- CohortSizeMin, [7](#), [10](#), [10](#), [45](#)
- CohortSizeMin-class, [10](#)
- CohortSizeParts, [7](#), [10](#), [10](#), [11](#)
- CohortSizeParts-class, [11](#)
- CohortSizeRange, [7](#), [11](#), [11](#)
- CohortSizeRange-class, [11](#)
- crmPack (crmPack-package), [5](#)
- crmPack-package, [5](#)
- crmPackExample, [12](#)
- crmPackHelp, [12](#)
- crmPackUpgrade, [13](#)
- Data, [6](#), [13](#), [14–16](#), [25](#), [27](#), [41](#), [46](#), [47](#), [53](#), [57](#), [61](#), [69](#), [79](#), [82](#)

- Data-class, 14
- DataDual, 14, 15, 18, 53, 57, 83
- DataDual-class, 15
- DataParts, 11, 15, 15, 32, 83, 84
- DataParts-class, 16
- Design, 16, 16, 17, 18, 25, 61, 65
- Design-class, 16
- dose, 17
- dose, numeric, Model, Samples-method (dose), 17
- DualDesign, 17, 17, 66
- DualDesign-class, 18
- DualEndpoint, 7, 18, 18, 20–22, 46, 57
- DualEndpoint-class, 19
- DualEndpointBeta, 19, 20, 20, 48
- DualEndpointBeta-class, 20
- DualEndpointEmax, 21, 21, 48
- DualEndpointEmax-class, 21
- DualEndpointOld, 33
- DualEndpointRW, 19, 22, 22
- DualEndpointRW-class, 22
- DualSimulations, 23, 23, 54, 66, 80
- DualSimulations-class, 23
- DualSimulationsSummary, 54, 55, 63, 80
- DualSimulationsSummary-class, 24
- examine, 24
- examine, Design-method (examine), 24
- examine, RuleDesign-method (examine), 24
- fit, 25
- fit, Samples, DualEndpoint, DataDual-method (fit), 25
- fit, Samples, Model, Data-method (fit), 25
- fitted, 25
- formatC, 84
- GeneralData, 6, 14, 43
- GeneralData-class, 26
- GeneralModel, 43, 45
- GeneralModel-class, 26
- GeneralSimulations, 27, 27, 55, 67, 68, 81
- GeneralSimulations-class, 27
- GeneralSimulationsSummary, 56, 64, 68, 81
- GeneralSimulationsSummary-class, 28
- GenSA, 60
- get, Samples, character-method, 28
- getMinInfBeta, 29
- ggmcmc, 29
- ggplot, 53, 57, 58
- ggplot2, 54–56, 58
- grid.draw, 59
- gridExtra, 54–56, 58
- Increments, 16, 41
- Increments-class, 29
- IncrementsRelative, 29, 30, 30, 32
- IncrementsRelative-class, 30
- IncrementsRelativeDLT, 29, 31, 31
- IncrementsRelativeDLT-class, 31
- IncrementsRelativeParts, 29, 32, 32
- IncrementsRelativeParts-class, 32
- initialize, DualEndpointOld-method, 33
- LogisticKadane, 33, 33, 46
- LogisticKadane-class, 34
- LogisticLogNormal, 34, 34, 44, 46, 60
- LogisticLogNormal-class, 35
- LogisticLogNormalSub, 35, 36, 46
- LogisticLogNormalSub-class, 36
- LogisticNormal, 37, 37, 44, 46, 60
- LogisticNormal-class, 37
- LogisticNormalFixedMixture, 38, 38
- LogisticNormalFixedMixture-class, 38
- LogisticNormalMixture, 39, 39
- LogisticNormalMixture-class, 40
- logit, 40
- maxDose, 41
- maxDose, IncrementsRelative, Data-method (maxDose), 41
- maxDose, IncrementsRelativeDLT, Data-method (maxDose), 41
- maxDose, IncrementsRelativeParts, DataParts-method (maxDose), 41
- maxSize, 42, 45
- maxSize, CohortSize-method (maxSize), 42
- mcmc, 42
- mcmc, Data, LogisticNormal, McmcOptions-method (mcmc), 42
- mcmc, GeneralData, GeneralModel, McmcOptions-method (mcmc), 42
- McmcOptions, 25, 43, 43, 62, 65, 66
- McmcOptions-class, 44
- mean, 25
- MinimalInformative, 44
- minSize, 42, 45
- minSize, CohortSize-method (minSize), 45
- Model, 6, 16, 17, 25, 27, 47, 57, 59, 79
- Model-class, 45
- NextBest, 47, 61
- nextBest, 46
- nextBest, NextBestDualEndpoint, numeric, Samples, DualEndpoint-method (nextBest), 46
- nextBest, NextBestMTD, numeric, Samples, Model, Data-method (nextBest), 46

- nextBest, NextBestNCRM, numeric, Samples, Model, Data, missing-method, 64
(nextBest), 46
- nextBest, NextBestNCRM, numeric, Samples, Model, DataParts, Design-method, 66
(nextBest), 46
- nextBest, NextBestThreePlusThree, missing, missing-method, 67, 68, 81
(nextBest), 46
- NextBest-class, 48
- NextBestDualEndpoint, 48, 48
- NextBestDualEndpoint-class, 48
- NextBestMTD, 48, 49, 49
- NextBestMTD-class, 49
- NextBestNCRM, 48, 50, 50
- NextBestNCRM-class, 50
- NextBestThreePlusThree, 48, 51, 51
- NextBestThreePlusThree-class, 51
- or-Stopping-Stopping, 51
- or-Stopping-StoppingAny, 52
- or-StoppingAny-Stopping, 52
- plot, Data, missing-method, 53
- plot, DataDual, missing-method, 53
- plot, DualSimulations, missing-method, 54
- plot, DualSimulationsSummary, missing-method, 54
- plot, GeneralSimulations, missing-method, 55
- plot, GeneralSimulationsSummary, missing-method, 56
- plot, Samples, DualEndpoint-method, 57
- plot, Samples, Model-method, 57
- plot, SimulationsSummary, missing-method, 58
- plot.arrange, 59
- prob, 59
- prob, numeric, Model, Samples-method
(prob), 59
- Quantiles2LogisticNormal, 6, 44, 45, 60
- Report, 61
- RuleDesign, 16, 25, 61, 61, 67, 82
- RuleDesign-class, 61
- Samples, 6, 7, 17, 25, 28, 43, 47, 57, 59, 62, 62, 79
- Samples-class, 62
- sampleSize, 62
- set.seed, 5, 27, 43, 63
- setSeed, 63, 65–67
- show, DualSimulationsSummary-method, 63
- show, GeneralSimulationsSummary-method, 64
- show, SimulationsSummary-method, 64
- simulate, Design-method, 65
- simulate, RuleDesign-method, 67
- simulate, Data, missing-method, 67, 68, 81
- Simulations-class, 68
- SimulationsSummary, 24, 58, 64, 81
- SimulationsSummary-class, 68
- size, 69
- size, CohortSizeConst, ANY, Data-method
(size), 69
- size, CohortSizeDLT, ANY, Data-method
(size), 69
- size, CohortSizeMax, ANY, Data-method
(size), 69
- size, CohortSizeMin, ANY, Data-method
(size), 69
- size, CohortSizeParts, ANY, DataParts-method
(size), 69
- size, CohortSizeRange, ANY, Data-method
(size), 69
- Stopping, 16, 51, 52, 70, 71, 73, 79, 85, 86
- Stopping-class, 70
- StoppingAll, 70, 70, 85, 86
- StoppingAll-class, 70
- StoppingAny, 51, 52, 71, 71
- StoppingAny-class, 71
- StoppingCohortsNearDose, 70, 72, 72
- StoppingCohortsNearDose-class, 72
- StoppingList, 70, 72, 73
- StoppingList-class, 73
- StoppingMinCohorts, 70, 73, 73
- StoppingMinCohorts-class, 74
- StoppingMinPatients, 70, 74, 74
- StoppingMinPatients-class, 74
- StoppingMTDdistribution, 70, 75, 75
- StoppingMTDdistribution-class, 75
- StoppingPatientsNearDose, 70, 76, 76
- StoppingPatientsNearDose-class, 76
- StoppingTargetBiomarker, 70, 76, 77
- StoppingTargetBiomarker-class, 77
- StoppingTargetProb, 70, 77, 77
- StoppingTargetProb-class, 78
- stopTrial, 78
- stopTrial, StoppingAll, ANY, ANY, ANY, ANY-method
(stopTrial), 78
- stopTrial, StoppingAny, ANY, ANY, ANY, ANY-method
(stopTrial), 78
- stopTrial, StoppingCohortsNearDose, numeric, ANY, ANY, Data-
(stopTrial), 78
- stopTrial, StoppingList, ANY, ANY, ANY, ANY-method
(stopTrial), 78

stopTrial, StoppingMinCohorts, ANY, ANY, ANY, Data-method
(stopTrial), [78](#)

stopTrial, StoppingMinPatients, ANY, ANY, ANY, Data-method
(stopTrial), [78](#)

stopTrial, StoppingMTDdistribution, numeric, Samples, Model, ANY-method
(stopTrial), [78](#)

stopTrial, StoppingPatientsNearDose, numeric, ANY, ANY, Data-method
(stopTrial), [78](#)

stopTrial, StoppingTargetBiomarker, numeric, Samples, DualEndpoint, ANY-method
(stopTrial), [78](#)

stopTrial, StoppingTargetProb, numeric, Samples, Model, ANY-method
(stopTrial), [78](#)

summary, DualSimulations-method, [80](#)

summary, GeneralSimulations-method, [80](#)

summary, Simulations-method, [81](#)

ThreePlusThreeDesign, [82](#)

update, Data-method, [82](#)

update, DataDual-method, [83](#)

update, DataParts-method, [83](#)

Validate, [84](#)

writeLines, [84](#)

writeModel, [84](#)