

# Package ‘crmPack’

May 11, 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>

**Description** Object-oriented implementation of CRM designs

**Version** 0.0.24

**Date** 2015-05-11

**Depends** R (>= 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
ComboLogistic	12
ComboLogistic-class	12
crmPackExample	13
crmPackHelp	14
crmPackUpgrade	14
Data	15
Data-class	16
DataCombo	16
DataCombo-class	17
DataDual	17
DataDual-class	18
DataParts	18
DataParts-class	18
Design	19
Design-class	19
dose	20
DualDesign	20
DualDesign-class	21
DualEndpoint	21
DualEndpoint-class	22
DualEndpointBeta	23
DualEndpointBeta-class	23
DualEndpointEmax	24

DualEndpointEmax-class . . . . .	24
DualEndpointRW . . . . .	25
DualEndpointRW-class . . . . .	25
DualSimulations . . . . .	26
DualSimulations-class . . . . .	26
DualSimulationsSummary-class . . . . .	27
examine . . . . .	27
fit . . . . .	28
GeneralData-class . . . . .	29
GeneralModel-class . . . . .	30
GeneralSimulations . . . . .	30
GeneralSimulations-class . . . . .	31
GeneralSimulationsSummary-class . . . . .	31
get,Samples,character-method . . . . .	32
Increments-class . . . . .	32
IncrementsRelative . . . . .	33
IncrementsRelative-class . . . . .	33
IncrementsRelativeDLT . . . . .	34
IncrementsRelativeDLT-class . . . . .	34
IncrementsRelativeParts . . . . .	35
IncrementsRelativeParts-class . . . . .	35
initialize,DualEndpointOld-method . . . . .	36
LogisticKadane . . . . .	36
LogisticKadane-class . . . . .	37
LogisticLogNormal . . . . .	37
LogisticLogNormal-class . . . . .	38
LogisticLogNormalSub . . . . .	38
LogisticLogNormalSub-class . . . . .	39
LogisticNormal . . . . .	40
LogisticNormal-class . . . . .	40
LogisticNormalFixedMixture . . . . .	41
LogisticNormalFixedMixture-class . . . . .	41
LogisticNormalMixture . . . . .	42
LogisticNormalMixture-class . . . . .	43
logit . . . . .	43
maxDose . . . . .	44
maxSize . . . . .	45
mcmc . . . . .	45
McmcOptions . . . . .	46
McmcOptions-class . . . . .	47
MinimalInformative . . . . .	47
minSize . . . . .	48
Model-class . . . . .	48
nextBest . . . . .	49
NextBest-class . . . . .	51
NextBestDualEndpoint . . . . .	51
NextBestDualEndpoint-class . . . . .	51
NextBestMTD . . . . .	52
NextBestMTD-class . . . . .	52
NextBestNCRM . . . . .	53
NextBestNCRM-class . . . . .	53
NextBestThreePlusThree . . . . .	54

NextBestThreePlusThree-class . . . . .	54
or-Stopping-Stopping . . . . .	54
or-Stopping-StoppingAny . . . . .	55
or-StoppingAny-Stopping . . . . .	55
plot,Data,missing-method . . . . .	56
plot,DataCombo,missing-method . . . . .	56
plot,DataDual,missing-method . . . . .	57
plot,DualSimulations,missing-method . . . . .	57
plot,DualSimulationsSummary,missing-method . . . . .	58
plot,GeneralSimulations,missing-method . . . . .	58
plot,GeneralSimulationsSummary,missing-method . . . . .	59
plot,Samples,ComboLogistic-method . . . . .	60
plot,Samples,DualEndpoint-method . . . . .	61
plot,Samples,Model-method . . . . .	61
plot,SimulationsSummary,missing-method . . . . .	62
plot.arrange . . . . .	63
prob . . . . .	63
Quantiles2LogisticNormal . . . . .	64
Report . . . . .	65
RuleDesign . . . . .	65
RuleDesign-class . . . . .	65
Samples . . . . .	66
Samples-class . . . . .	66
sampleSize . . . . .	66
setSeed . . . . .	67
show,DualSimulationsSummary-method . . . . .	67
show,GeneralSimulationsSummary-method . . . . .	68
show,SimulationsSummary-method . . . . .	68
simulate,Design-method . . . . .	69
simulate,DualDesign-method . . . . .	70
simulate,RuleDesign-method . . . . .	71
Simulations . . . . .	71
Simulations-class . . . . .	72
SimulationsSummary-class . . . . .	72
size . . . . .	73
Stopping-class . . . . .	74
StoppingAll . . . . .	74
StoppingAll-class . . . . .	74
StoppingAny . . . . .	75
StoppingAny-class . . . . .	75
StoppingCohortsNearDose . . . . .	76
StoppingCohortsNearDose-class . . . . .	76
StoppingList . . . . .	76
StoppingList-class . . . . .	77
StoppingMinCohorts . . . . .	77
StoppingMinCohorts-class . . . . .	78
StoppingMinPatients . . . . .	78
StoppingMinPatients-class . . . . .	78
StoppingMTDdistribution . . . . .	79
StoppingMTDdistribution-class . . . . .	79
StoppingPatientsNearDose . . . . .	80
StoppingPatientsNearDose-class . . . . .	80

StoppingTargetBiomarker . . . . .	80
StoppingTargetBiomarker-class . . . . .	81
StoppingTargetProb . . . . .	81
StoppingTargetProb-class . . . . .	82
stopTrial . . . . .	82
summary,DualSimulations-method . . . . .	84
summary,GeneralSimulations-method . . . . .	84
summary,Simulations-method . . . . .	85
ThreePlusThreeDesign . . . . .	86
update,Data-method . . . . .	86
update,DataCombo-method . . . . .	87
update,DataDual-method . . . . .	87
update,DataParts-method . . . . .	88
Validate . . . . .	88
writeModel . . . . .	89
&,Stopping,Stopping-method . . . . .	89
&,Stopping,StoppingAll-method . . . . .	90
&,StoppingAll,Stopping-method . . . . .	90
<b>Index</b>	<b>91</b>

---

 crmPack-package

*Object-oriented implementation of CRM designs*


---

## Description

Object-oriented implementation of CRM designs

## Author(s)

Daniel Sabanes Bove <sabanesd@roche.com>

---

 approximate

*Approximate posterior with (log) normal distribution*


---

## 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 <a href="#">Samples</a> object
model	the <a href="#">Model</a> object
data	the <a href="#">Data</a> 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 <a href="#">GeneralData</a> 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 <a href="#">DualEndpoint</a> object
samples	the <a href="#">Samples</a> 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	<i>Cohort size based on the parts</i>
-----------------------	---------------------------------------

---

**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	<i>Initialization function for "CohortSizeRange"</i>
-----------------	--

---

**Description**

Initialization function for "CohortSizeRange"

**Usage**

CohortSizeRange(intervals, cohortSize)

**Arguments**

intervals see CohortSizeRange  
cohortSize see CohortSizeRange

**Value**

the CohortSizeRange object

---

CohortSizeRange-class	<i>Cohort size based on dose range</i>
-----------------------	--

---

**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

---

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

---

**Description**

Initialization function for the "ComboLogistic" class

**Usage**

```
ComboLogistic(singlePriors, gamma, tau)
```

**Arguments**

singlePriors	a named list where each element is a <a href="#">LogisticLogNormal</a> object, specifying the prior for this drug. The list names are the drug names.
gamma	see <a href="#">ComboLogistic</a>
tau	see <a href="#">ComboLogistic</a>

**Value**

the [ComboLogistic](#) object

---

ComboLogistic-class	<i>Combo model with logistic regression</i>
---------------------	---

---

**Description**

Currently, this model is for double combination dose escalation trials.

**Details**

todo: Later, it will be extended to higher-order combinations. The model and code is building on the work by Simon Wandel et al (Novartis).

The regression model is defined as follows. Let  $odds(p) = p/(1-p)$  be the odds transformation of the probability  $p$ , such that  $logit(p) = \log(odds(p))$ . Let  $x_i$  be the dose of drug  $i=1,2$ , and  $p(x_1, x_2)$  be the probability of DLT with doses  $x_1$  and  $x_2$ . The reference doses for the two compounds are again denoted by stars. Then the model assumes:

$$odds(p(x_1, x_2)) = odds(p_0(x_1, x_2)) \cdot \exp(\eta x_1/x_1^* x_2/x_2^*),$$

where  $\eta$  is the interaction coefficient (positive values correspond to synergistic toxicity, zero corresponds to additive effect without interaction, and negative values correspond to antagonistic toxicity). A normal prior

$$\eta \sim Normal(\gamma, \tau^{-1})$$

is used. Under no interaction with  $\eta = 0$ , this reduces the probability  $\eta odds(p(x_1, x_2))$  to

$$p_0(x_1, x_2) = p(x_1) + p(x_2) - p(x_1)p(x_2) = 1 - (1 - p(x_1))(1 - p(x_2)).$$

Now for the single-agent DLT probabilities  $p(x_1)$  and  $p(x_2)$  we assume the logistic log-normal models (compare [LogisticLogNormal](#)):

$$\text{logit}[p(x_i)] = \alpha_i + \beta_i \cdot \log(x_i/x_i^*),$$

with prior

$$(\alpha_i, \log(\beta_i)) \sim \text{Normal}(\mu_i, \Sigma_i)$$

for  $i = 1, 2$ . todo: Note that in principle any model could be used for the individual agents. For now we stick to this simple, fixed implementation.

The prob function has as first argument dose, which is either a single dose combination (a vector with names specifying the drug names) or multiple dose combinations in rows of a matrix (with the column names specifying the drug names). Additional arguments are the model parameters alpha0 (intercepts, nSamples x nDrugs matrix), alpha1 (slopes, nSamples x nDrugs matrix) and eta (vector of length nSamples), containing the nSamples MCMC samples. Then prob computes the resulting samples of the probability of toxicity at that dose combination(s) (nSamples x nCombinations).

### Slots

**singlePriors** a list with one [LogisticLogNormal](#) model per drug specifying the bivariate log normal prior described above. The names of this list are the drug names for this model - they have to be specified correctly in any interactions with the model object (e.g. when calling the prob function).

**gamma** the mean for the interaction parameter

**tau** the precision for the interaction parameter

**prob** function calculating the probability of toxicity for a specific dose combination, based on the model parameters (see the details above)

---

 crmPackExample

---

*Open the example pdf for crmPack*


---

### 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	<i>Upgrade your crmPack installation with the latest version</i>
----------------	--

---

**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	<i>Initialization function for the "Data" class</i>
------	---

---

**Description**

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

**Usage**

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

**Arguments**

x	the doses for the patients
y	the vector of toxicity events (0 or 1 integers). You can also normal numeric vectors, but these will then be converted to integers.
ID	unique patient IDs (integer vector)
cohort	the cohort indices (sorted values from 0, 1, 2, ...)
doseGrid	the vector of all possible doses
...	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

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

---

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

---

### Description

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

### Usage

```
DataCombo(x, y, ID, cohort, doseGrid)
```

### Arguments

<code>x</code>	the matrix with the doses for the patients. Recommendation: Use <code>cbind(drugA=..., drugB=...)</code> to create this matrix.
<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 list with vectors of all possible doses for each of the drugs

### 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 [DataCombo](#) object



---

DataCombo-class	<i>Class for the data input in combo trials</i>
-----------------	---

---

### Description

This class inherits from [GeneralData](#).

### Slots

`x` a matrix with the doses of all `nDrugs` drugs (columns) for the `nObs` patients (rows). The column names are the `drugNames`.

`y` the vector of toxicity events (0 or 1 integers)

`doseGrid` a list containing a vector of all possible doses (sorted) for each of the `nDrugs` drugs (named with `drugNames`).

`nDrugs` number of drugs

`drugNames` character vector with the drug names

`nGrid` vector with the number of gridpoints for each of the drugs

`xLevel` an integer matrix with the levels for the doses the patients have been given, same dimensions as `x`.

---

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

<code>w</code>	the continuous vector of biomarker values
<code>...</code>	additional parameters from <a href="#">Data</a>

### 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 <a href="#">Data</a>

**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 <a href="#">Design</a>
stopping	see <a href="#">Design</a>
increments	see <a href="#">Design</a>
...	additional arguments for <a href="#">RuleDesign</a>

**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 <a href="#">Model</a>
stopping	stopping rule(s) for the trial, an object of class <a href="#">Stopping</a>
increments	how to control increments between dose levels, an object of class <a href="#">Increments</a>

---

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 <a href="#">Model</a>
samples	the <a href="#">Samples</a>
...	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 <a href="#">DualDesign</a>
data	see <a href="#">DualDesign</a>
...	additional arguments for <a href="#">Design</a>

**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 <a href="#">DualEndpoint</a>
Sigma	see <a href="#">DualEndpoint</a>
sigma2W	see <a href="#">DualEndpoint</a>
rho	see <a href="#">DualEndpoint</a>

**Value**

the [DualEndpoint](#) object

---

DualEndpoint-class      *General class for the dual endpoint model*


---

## 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  $\Phi$  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  $\sigma_W^2$  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  $\rho$  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 <a href="#">DualEndpointBeta</a>
Emax	see <a href="#">DualEndpointBeta</a>
delta1	see <a href="#">DualEndpointBeta</a>
mode	see <a href="#">DualEndpointBeta</a>
refDose	see <a href="#">DualEndpointBeta</a>
...	additional parameters, see <a href="#">DualEndpoint</a>

### 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),  $\delta_1$  and  $\delta_2$  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  $\delta_1$  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	<i>Initialization function for the "DualEndpointEmax" class</i>
------------------	---

---

**Description**

Initialization function for the "DualEndpointEmax" class

**Usage**

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

**Arguments**

<code>E0</code>	see <a href="#">DualEndpointEmax</a>
<code>Emax</code>	see <a href="#">DualEndpointEmax</a>
<code>ED50</code>	see <a href="#">DualEndpointEmax</a>
<code>refDose</code>	see <a href="#">DualEndpointEmax</a>
<code>...</code>	additional parameters, see <a href="#">DualEndpoint</a>

**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 <a href="#">DualEndpointRW</a>
smooth	either "RW1" (default) or "RW2", for specifying the random walk prior on the biomarker level.
...	additional parameters, see <a href="#">DualEndpoint</a>

**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 <a href="#">DualSimulations</a>                    |
| <code>sigma2West</code>   | see <a href="#">DualSimulations</a>                    |
| <code>fitBiomarker</code> | see <a href="#">DualSimulations</a>                    |
| <code>...</code>          | additional parameters from <a href="#">Simulations</a> |

**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 ( <a href="#">Design</a> or <a href="#">RuleDesign</a> object) we want to examine
mcmcOptions	object of class <a href="#">McmcOptions</a> , 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, ...)

## S4 method for signature 'Samples,ComboLogistic,DataCombo'
fit(object, model, data,
     focus = head(names(model@singlePriors), 2L),
     points = as.matrix(expand.grid(data@doseGrid[focus])),
     quantiles = c(0.025, 0.975), middle = mean, ...)
```

**Arguments**

object	the <a href="#">Samples</a> object
model	the <a href="#">Model</a> object
data	the <a href="#">Data</a> 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: <a href="#">mean</a>

focus	two drug names of this combo for which the model fit should be produced. Defaults to the first two drugs.
...	unused
points	matrix with the dose combos at which the fit is requested. Default is the expanded dose grid of the focus drugs.
quantiles	the quantiles to be calculated (default: 0.025 and 0.975)
middle	the function for computing the middle point. Default: <a href="#">mean</a>

### 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)
- `object = Samples, model = Combologistic, data = DataCombo`: This method returns a data frame with doses for the two focus drugs, middle, lower and upper quantiles for the dose-toxicity surface

---

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](#), [ComboLogistic](#)

---

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

---

### Description

Initialization function for "GeneralSimulations"

### Usage

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

### Arguments

<code>data</code>	see <a href="#">GeneralSimulations</a>
<code>doses</code>	see <a href="#">GeneralSimulations</a>
<code>seed</code>	see <a href="#">GeneralSimulations</a>

**Value**

the `GeneralSimulations` object

---

`GeneralSimulations-class`

*General class for the simulations output*

---

**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

<code>x</code>	the <a href="#">Samples</a> object
<code>pos</code>	the name of the parameter
<code>envir</code>	for vectorial parameters, you can give the indices of the elements you would like to extract. If <code>NULL</code> , the whole vector samples will be returned
<code>mode</code>	not used
<code>inherits</code>	not used

### Value

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

---

`Increments-class`

*The virtual class for controlling increments*

---

### 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 <a href="#">IncrementsRelative</a>
increments	see <a href="#">IncrementsRelative</a>

## 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

<code>intervals</code>	a vector with the left bounds of the relevant intervals
<code>increments</code>	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 <a href="#">IncrementsRelativeParts</a>
cleanStart	see <a href="#">IncrementsRelativeParts</a>
...	additional slots from <a href="#">IncrementsRelative</a>

## 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 <a href="#">DualEndpointOld</a> we want to initialize
mu	see <a href="#">DualEndpointOld</a>
Sigma	see <a href="#">DualEndpointOld</a>
sigma2betaW	see <a href="#">DualEndpointOld</a>
sigma2W	see <a href="#">DualEndpointOld</a>
rho	see <a href="#">DualEndpointOld</a>
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     *Reparametrized logistic model*


---

### 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  $\gamma$  be the dose with target toxicity probability  $\theta$ , 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  $\gamma$  between  $x_{min}$  and  $x_{max}$ , and for  $\rho_0$  as well a uniform distribution between 0 and  $\theta$ .

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  $\theta$   
xmin the minimum of the dose range  $x_{min}$   
xmax the maximum of the dose range  $x_{max}$

---

LogisticLogNormal     *Initialization function for the "LogisticLogNormal" class*


---

### 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  $\alpha$  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  $\mu$   
`cov` the prior covariance matrix  $\Sigma$   
`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 $\mu$
cov	the prior covariance matrix $\Sigma$
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 $\mu$
cov	the prior covariance matrix $\Sigma$
prec	the prior precision matrix $\Sigma^{-1}$
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 <a href="#">GeneralData</a>
model	The model input, an object of class <a href="#">GeneralModel</a>
options	MCMC options, an object of class <a href="#">McmcOptions</a>
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, refDose, threshmin = 0.2, threshmax = 0.3, ...)

**Arguments**

dosegrid	the dose grid
refDose	the reference dose
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 <a href="#">Quantiles2LogisticNormal</a> , e.g. 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. Likewise, threshmax is the probability threshold  $q_J$ , such that any probability of DLT smaller than  $q_J$  has only 5% probability. Subsequently, for all doses supplied in the dosegrid argument, Beta distributions are set up, and [Quantiles2LogisticNormal](#) is used to transform the resulting quantiles into an approximating [LogisticNormal](#) (or [LogisticLogNormal](#)) model.

**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](#), [ComboLogistic](#)

---

nextBest	<i>Find the next best dose</i>
----------	--------------------------------

---

## 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 <a href="#">NextBest</a>
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 <a href="#">Samples</a> object
model	The model input, an object of class <a href="#">Model</a>
data	The data input, an object of class <a href="#">Data</a>
...	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 <a href="#">NextBestDualEndpoint</a>
overdose	see <a href="#">NextBestDualEndpoint</a>
maxOverdoseProb	see <a href="#">NextBestDualEndpoint</a>

**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. [NextBestDualEndpointBeta](#), [NextBestDualEndpointEmax](#)))

**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

**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 <a href="#">NextBestNCRM</a>
overdose	see <a href="#">NextBestNCRM</a>
maxOverdoseProb	see <a href="#">NextBestNCRM</a>

### 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
overdose	the overdose toxicity interval
maxOverdoseProb	maximum overdose probability that is allowed

---

NextBestThreePlusThree
<i>Initialization function for "NextBestThreePlusThree"</i>

---

**Description**

Initialization function for "NextBestThreePlusThree"

**Usage**

NextBestThreePlusThree()

**Value**

the [NextBestThreePlusThree](#) object

---

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

---

**Description**

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

---

or-Stopping-Stopping	<i>The method combining two atomic stopping rules</i>
----------------------	---

---

**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	<a href="#">StoppingAny</a> object
e2	<a href="#">Stopping</a> 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	<a href="#">Stopping</a> object
e2	<a href="#">StoppingAny</a> 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 <a href="#">Data</a> object we want to plot
y	missing
...	not used

### Value

the [ggplot](#) object

---

```
plot,DataCombo,missing-method
```

*Plot method for the "DataCombo" class*

---

### Description

Plot method for the "DataCombo" class

### Usage

```
## S4 method for signature 'DataCombo,missing'
plot(x, y, select = head(x@drugNames, 2L),
     shorten = 0.03, ...)
```

### Arguments

x	the <a href="#">DataDual</a> object we want to plot
y	missing
select	two drug names that we want to use for plotting. Defaults to the first two drug names.
shorten	relative amount of shortening the arrows. default: 0.05
...	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 <a href="#">DataDual</a> 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 <a href="#">DualSimulations</a> 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

<code>x</code>	the <a href="#">DualSimulationsSummary</a> object we want to plot from
<code>y</code>	missing
<code>type</code>	the types of plots you want to obtain.
<code>...</code>	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 <a href="#">GeneralSimulations</a> 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 <a href="#">GeneralSimulationsSummary</a> 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, ComboLogistic-method

*Plotting dose-toxicity combo model fits*

---

## Description

Plotting dose-toxicity combo model fits

## Usage

```
## S4 method for signature 'Samples,ComboLogistic'
plot(x, y, data, focus, resolution = 20,
     ...)
```

## Arguments

x	the <a href="#">Samples</a> object
y	the <a href="#">ComboLogistic</a> object
data	the <a href="#">DataCombo</a> object
focus	one or two drug names of this combo for which the model fit plot should be produced
resolution	number of points to be used in each dimension for 2D plots (default: 20) - higher number is higher resolution
...	passed to the single agent plotting method if focus specifies only one drug name

## Value

todo

---

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 <a href="#">Samples</a> object
y	the <a href="#">DualEndpoint</a> object
data	the <a href="#">DataDual</a> 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 <a href="#">plot,Samples,Model-method</a>

### 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 <a href="#">Samples</a> object
y	the <a href="#">Model</a> object
data	the <a href="#">Data</a> 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 <a href="#">SimulationsSummary</a> 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 <a href="#">grid.draw</a>

---

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, ...)

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

**Arguments**

dose	the dose
model	the <a href="#">Model</a> or <a href="#">ComboLogistic</a> object
samples	the <a href="#">Samples</a>
...	unused

**Value**

either the vector (for [Model](#) objects) or the matrix (for [ComboLogistic](#) objects) of probability samples.

**Methods (by class)**

- dose = numeric,model = [Model](#),samples = [Samples](#):
- dose = matrix,model = [ComboLogistic](#),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 <a href="#">GenSA</a> 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 <a href="#">RuleDesign</a>
cohortSize	see <a href="#">RuleDesign</a>
data	see <a href="#">RuleDesign</a>
startingDose	see <a href="#">RuleDesign</a>

**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 <code>set.seed</code> 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 <code>DualSimulationsSummary</code> 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 <a href="#">Design</a> object we want to simulate data from
nsim	the number of simulations (default: 1)
seed	see <a href="#">setSeed</a>
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 <a href="#">McmcOptions</a> , 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 <a href="#">DualDesign</a> object we want to simulate data from
nsim	the number of simulations (default: 1)
seed	see <a href="#">setSeed</a>
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 <a href="#">McmcOptions</a> , 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 <a href="#">RuleDesign</a> object we want to simulate data from
nsim	the number of simulations (default: 1)
seed	see <a href="#">setSeed</a>
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 <a href="#">Simulations</a>
stopReasons	see <a href="#">Simulations</a>
...	additional parameters from <a href="#">GeneralSimulations</a>

**Value**

the [Simulations](#) object

---

Simulations-class

*Class for the simulations output from model based designs*

---

**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

*Class for the summary of model-based simulations output*

---

**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 <a href="#">CohortSize</a>
dose	the next dose
data	The data input, an object of class <a href="#">Data</a>
...	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	<i>Initialization function for "StoppingCohortsNearDose"</i>
-------------------------	--

---

**Description**

Initialization function for "StoppingCohortsNearDose"

**Usage**

StoppingCohortsNearDose(nCohorts, percentage)

**Arguments**

nCohorts            see [StoppingCohortsNearDose](#)  
percentage        see [StoppingCohortsNearDose](#)

**Value**

the [StoppingCohortsNearDose](#) object

---

StoppingCohortsNearDose-class	<i>Stop based on number of cohorts near to next best dose</i>
-------------------------------	---

---

**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	<i>Initialization function for "StoppingList"</i>
--------------	---

---

**Description**

Initialization function for "StoppingList"

**Usage**

StoppingList(stopList, summary)

**Arguments**

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

**Value**

the [StoppingList](#) object

---

StoppingList-class	<i>Stop based on multiple stopping rules</i>
--------------------	--

---

**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	<i>Initialization function for "StoppingMinCohorts"</i>
--------------------	---

---

**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	
	<i>Stop based on probability of target biomarker</i>

---

**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	<i>Initialization function for "StoppingTargetProb"</i>
--------------------	---

---

**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 <a href="#">Stopping</a>
dose	the recommended next best dose
samples	the <a href="#">Samples</a> object
model	The model input, an object of class <a href="#">Model</a>
data	The data input, an object of class <a href="#">Data</a>
...	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 <a href="#">DualSimulations</a> 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 <a href="#">GeneralSimulations</a> 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 <a href="#">Simulations</a> 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 <a href="#">Data</a> 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,DataCombo-method

*Update method for the "DataCombo" class*


---

### Description

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

### Usage

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

### Arguments

object	the old <a href="#">DataCombo</a> object
x	the dose levels vector (one dose level combination only!)
y	the DLT vector (0/1 vector), for all patients in this cohort
ID	the patient IDs
...	not used

### Value

the new [DataCombo](#) 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 <a href="#">DataDual</a> 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 <code>DataParts</code> 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 <a href="#">writelines</a> which actually writes the model file
digits	number of significant digits used for <b>WinBUGS</b> input, see <a href="#">formatC</a>

**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 <a href="#">Stopping</a> object
e2	Second <a href="#">Stopping</a> 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](#)  
ComboLogistic-class, [12](#)  
Data-class, [16](#)  
DataCombo-class, [17](#)  
DataDual-class, [18](#)  
DataParts-class, [18](#)  
Design-class, [19](#)  
DualDesign-class, [21](#)  
DualEndpoint-class, [22](#)  
DualEndpointBeta-class, [23](#)  
DualEndpointEmax-class, [24](#)  
DualEndpointRW-class, [25](#)  
DualSimulations-class, [26](#)  
DualSimulationsSummary-class, [27](#)  
GeneralData-class, [29](#)  
GeneralModel-class, [30](#)  
GeneralSimulations-class, [31](#)  
GeneralSimulationsSummary-class, [31](#)  
Increments-class, [32](#)  
IncrementsRelative-class, [33](#)  
IncrementsRelativeDLT-class, [34](#)  
IncrementsRelativeParts-class, [35](#)  
LogisticKadane-class, [37](#)  
LogisticLogNormal-class, [38](#)  
LogisticLogNormalSub-class, [39](#)  
LogisticNormal-class, [40](#)  
LogisticNormalFixedMixture-class, [41](#)  
LogisticNormalMixture-class, [43](#)  
McmcOptions-class, [47](#)  
Model-class, [48](#)  
NextBest-class, [51](#)  
NextBestDualEndpoint-class, [51](#)  
NextBestMTD-class, [52](#)  
NextBestNCRM-class, [53](#)

NextBestThreePlusThree-class, [54](#)  
RuleDesign-class, [65](#)  
Samples-class, [66](#)  
Simulations-class, [72](#)  
SimulationsSummary-class, [72](#)  
Stopping-class, [74](#)  
StoppingAll-class, [74](#)  
StoppingAny-class, [75](#)  
StoppingCohortsNearDose-class, [76](#)  
StoppingList-class, [77](#)  
StoppingMinCohorts-class, [78](#)  
StoppingMinPatients-class, [78](#)  
StoppingMTDdistribution-class, [79](#)  
StoppingPatientsNearDose-class, [80](#)  
StoppingTargetBiomarker-class, [81](#)  
StoppingTargetProb-class, [82](#)

## \*Topic **documentation**

crmPackExample, [13](#)  
crmPackHelp, [14](#)

## \*Topic **methods**

&, Stopping, Stopping-method, [89](#)  
&, Stopping, StoppingAll-method, [90](#)  
&, StoppingAll, Stopping-method, [90](#)  
approximate, [5](#)  
as.list, GeneralData-method, [6](#)  
biomLevel, [7](#)  
CohortSizeConst, [8](#)  
CohortSizeDLT, [8](#)  
CohortSizeMax, [9](#)  
CohortSizeMin, [10](#)  
CohortSizeParts, [10](#)  
CohortSizeRange, [11](#)  
ComboLogistic, [12](#)  
Design, [19](#)  
dose, [20](#)  
DualDesign, [20](#)  
DualEndpoint, [21](#)  
DualEndpointBeta, [23](#)  
DualEndpointEmax, [24](#)  
DualEndpointRW, [25](#)  
DualSimulations, [26](#)  
examine, [27](#)  
fit, [28](#)

- GeneralSimulations, 30
- get, Samples, character-method, 32
- IncrementsRelative, 33
- IncrementsRelativeDLT, 34
- IncrementsRelativeParts, 35
- initialize, DualEndpointOld-method, 36
- LogisticKadane, 36
- LogisticLogNormal, 37
- LogisticLogNormalSub, 38
- LogisticNormal, 40
- LogisticNormalFixedMixture, 41
- LogisticNormalMixture, 42
- maxDose, 44
- maxSize, 45
- mcmc, 45
- McmcOptions, 46
- minSize, 48
- nextBest, 49
- NextBestDualEndpoint, 51
- NextBestMTD, 52
- NextBestNCRM, 53
- NextBestThreePlusThree, 54
- or-Stopping-Stopping, 54
- or-Stopping-StoppingAny, 55
- or-StoppingAny-Stopping, 55
- plot, Data, missing-method, 56
- plot, DataCombo, missing-method, 56
- plot, DataDual, missing-method, 57
- plot, DualSimulations, missing-method, 57
- plot, DualSimulationsSummary, missing-method, 58
- plot, GeneralSimulations, missing-method, 58
- plot, GeneralSimulationsSummary, missing-method, 59
- plot, SimulationsSummary, missing-method, 62
- prob, 63
- RuleDesign, 65
- Samples, 66
- show, DualSimulationsSummary-method, 67
- show, GeneralSimulationsSummary-method, 68
- show, SimulationsSummary-method, 68
- simulate, Design-method, 69
- simulate, DualDesign-method, 70
- simulate, RuleDesign-method, 71
- Simulations, 71
- size, 73
- StoppingAll, 74
- StoppingAny, 75
- StoppingCohortsNearDose, 76
- StoppingList, 76
- StoppingMinCohorts, 77
- StoppingMinPatients, 78
- StoppingMTDdistribution, 79
- StoppingPatientsNearDose, 80
- StoppingTargetBiomarker, 80
- StoppingTargetProb, 81
- stopTrial, 82
- summary, DualSimulations-method, 84
- summary, GeneralSimulations-method, 84
- summary, Simulations-method, 85
- update, Data-method, 86
- update, DataCombo-method, 87
- update, DataDual-method, 87
- update, DataParts-method, 88
- \*Topic **package**
  - crmPack-package, 5
- \*Topic **programming**
  - crmPackUpgrade, 14
  - Data, 15
  - DataCombo, 16
  - DataDual, 17
  - DataParts, 18
  - logit, 43
  - MinimalInformative, 47
  - Quantiles2LogisticNormal, 64
  - sampleSize, 66
  - setSeed, 67
  - ThreePlusThreeDesign, 86
- \*Topic **regression**
  - examine, 27
  - mcmc, 45
  - .CohortSizeConst
    - (CohortSizeConst-class), 8
  - .CohortSizeDLT (CohortSizeDLT-class), 9
  - .CohortSizeMax (CohortSizeMax-class), 9
  - .CohortSizeMin (CohortSizeMin-class), 10
  - .CohortSizeParts
    - (CohortSizeParts-class), 11
  - .CohortSizeRange
    - (CohortSizeRange-class), 11
  - .ComboLogistic (ComboLogistic-class), 12
  - .Data (Data-class), 16
  - .DataCombo (DataCombo-class), 17
  - .DataDual (DataDual-class), 18
  - .DataParts (DataParts-class), 18
  - .Design (Design-class), 19
  - .DualDesign (DualDesign-class), 21

- .DualEndpoint (DualEndpoint-class), 22
- .DualEndpointBeta
  - (DualEndpointBeta-class), 23
- .DualEndpointEmax
  - (DualEndpointEmax-class), 24
- .DualEndpointRW (DualEndpointRW-class), 25
- .DualSimulations
  - (DualSimulations-class), 26
- .DualSimulationsSummary
  - (DualSimulationsSummary-class), 27
- .GeneralData (GeneralData-class), 29
- .GeneralModel (GeneralModel-class), 30
- .GeneralSimulations
  - (GeneralSimulations-class), 31
- .GeneralSimulationsSummary
  - (GeneralSimulationsSummary-class), 31
- .IncrementsRelative
  - (IncrementsRelative-class), 33
- .IncrementsRelativeDLT
  - (IncrementsRelativeDLT-class), 34
- .IncrementsRelativeParts
  - (IncrementsRelativeParts-class), 35
- .LogisticKadane (LogisticKadane-class), 37
- .LogisticLogNormal
  - (LogisticLogNormal-class), 38
- .LogisticLogNormalSub
  - (LogisticLogNormalSub-class), 39
- .LogisticNormal (LogisticNormal-class), 40
- .LogisticNormalFixedMixture
  - (LogisticNormalFixedMixture-class), 41
- .LogisticNormalMixture
  - (LogisticNormalMixture-class), 43
- .McmcOptions (McmcOptions-class), 47
- .Model (Model-class), 48
- .NextBestDualEndpoint
  - (NextBestDualEndpoint-class), 51
- .NextBestMTD (NextBestMTD-class), 52
- .NextBestNCRM (NextBestNCRM-class), 53
- .NextBestThreePlusThree
  - (NextBestThreePlusThree-class), 54
- .RuleDesign (RuleDesign-class), 65
- .Samples (Samples-class), 66
- .Simulations (Simulations-class), 72
- .SimulationsSummary
  - (SimulationsSummary-class), 72
- .StoppingAll (StoppingAll-class), 74
- .StoppingAny (StoppingAny-class), 75
- .StoppingCohortsNearDose
  - (StoppingCohortsNearDose-class), 76
- .StoppingList (StoppingList-class), 77
- .StoppingMTDdistribution
  - (StoppingMTDdistribution-class), 79
- .StoppingMinCohorts
  - (StoppingMinCohorts-class), 78
- .StoppingMinPatients
  - (StoppingMinPatients-class), 78
- .StoppingPatientsNearDose
  - (StoppingPatientsNearDose-class), 80
- .StoppingTargetBiomarker
  - (StoppingTargetBiomarker-class), 81
- .StoppingTargetProb
  - (StoppingTargetProb-class), 82
- &, Stopping, Stopping-method, 89
- &, Stopping, StoppingAll-method, 90
- &, StoppingAll, Stopping-method, 90
- approximate, 5
- approximate, Samples-method
  - (approximate), 5
- as.list, GeneralData-method, 6
- biomLevel, 7
- biomLevel, numeric, DualEndpoint, Samples-method
  - (biomLevel), 7
- CohortSize, 9, 10, 45, 48, 65, 73
- CohortSize-class, 7
- CohortSizeConst, 7, 8, 8
- CohortSizeConst-class, 8
- CohortSizeDLT, 7, 8, 8
- CohortSizeDLT-class, 9
- CohortSizeMax, 7, 9, 9, 45
- CohortSizeMax-class, 9
- CohortSizeMin, 7, 10, 10, 48
- CohortSizeMin-class, 10
- CohortSizeParts, 7, 10, 10, 11
- CohortSizeParts-class, 11
- CohortSizeRange, 7, 11, 11
- CohortSizeRange-class, 11

- ComboLogistic, [12, 12, 30, 49, 60, 63](#)
- ComboLogistic-class, [12](#)
- crmPack (crmPack-package), [5](#)
- crmPack-package, [5](#)
- crmPackExample, [13](#)
- crmPackHelp, [14](#)
- crmPackUpgrade, [14](#)
  
- Data, [6, 15, 15, 17, 18, 28, 31, 44, 49, 50, 56, 61, 65, 73, 83, 86](#)
- Data-class, [16](#)
- DataCombo, [16, 16, 60, 87](#)
- DataCombo-class, [17](#)
- DataDual, [17, 17, 21, 56, 57, 61, 87, 88](#)
- DataDual-class, [18](#)
- DataParts, [11, 18, 18, 35, 88](#)
- DataParts-class, [18](#)
- Design, [19, 19, 20, 21, 28, 65, 69](#)
- Design-class, [19](#)
- dose, [20](#)
- dose, numeric, Model, Samples-method (dose), [20](#)
- DualDesign, [20, 20, 70](#)
- DualDesign-class, [21](#)
- DualEndpoint, [7, 21, 21, 23–25, 49, 61](#)
- DualEndpoint-class, [22](#)
- DualEndpointBeta, [22, 23, 23](#)
- DualEndpointBeta-class, [23](#)
- DualEndpointEmax, [24, 24](#)
- DualEndpointEmax-class, [24](#)
- DualEndpointOld, [36](#)
- DualEndpointRW, [22, 25, 25](#)
- DualEndpointRW-class, [25](#)
- DualSimulations, [26, 26, 57, 70, 84](#)
- DualSimulations-class, [26](#)
- DualSimulationsSummary, [58, 67, 84](#)
- DualSimulationsSummary-class, [27](#)
  
- examine, [27](#)
- examine, Design-method (examine), [27](#)
- examine, RuleDesign-method (examine), [27](#)
  
- fit, [28](#)
- fit, Samples, ComboLogistic, DataCombo-method (fit), [28](#)
- fit, Samples, DualEndpoint, DataDual-method (fit), [28](#)
- fit, Samples, Model, Data-method (fit), [28](#)
- fitted, [28](#)
- formatC, [89](#)
  
- GeneralData, [6, 16, 17, 46](#)
- GeneralData-class, [29](#)
  
- GeneralModel, [46, 48](#)
- GeneralModel-class, [30](#)
- GeneralSimulations, [30, 30, 31, 59, 71, 72, 85](#)
- GeneralSimulations-class, [31](#)
- GeneralSimulationsSummary, [59, 68, 72, 85](#)
- GeneralSimulationsSummary-class, [31](#)
- GenSA, [64](#)
- get, Samples, character-method, [32](#)
- ggmcmc, [32](#)
- ggplot, [56, 57, 61, 62](#)
- ggplot2, [57–60, 62](#)
- grid.draw, [63](#)
- gridExtra, [57–60, 62](#)
  
- Increments, [19, 44](#)
- Increments-class, [32](#)
- IncrementsRelative, [32, 33, 33, 35](#)
- IncrementsRelative-class, [33](#)
- IncrementsRelativeDLT, [32, 34, 34](#)
- IncrementsRelativeDLT-class, [34](#)
- IncrementsRelativeParts, [32, 35, 35](#)
- IncrementsRelativeParts-class, [35](#)
- initialize, DualEndpointOld-method, [36](#)
  
- LogisticKadane, [36, 36, 49](#)
- LogisticKadane-class, [37](#)
- LogisticLogNormal, [12, 13, 37, 37, 47, 49, 64](#)
- LogisticLogNormal-class, [38](#)
- LogisticLogNormalSub, [38, 39, 49](#)
- LogisticLogNormalSub-class, [39](#)
- LogisticNormal, [40, 40, 47, 49, 64](#)
- LogisticNormal-class, [40](#)
- LogisticNormalFixedMixture, [41, 41](#)
- LogisticNormalFixedMixture-class, [41](#)
- LogisticNormalMixture, [42, 42](#)
- LogisticNormalMixture-class, [43](#)
- logit, [43](#)
  
- maxDose, [44](#)
- maxDose, IncrementsRelative, Data-method (maxDose), [44](#)
- maxDose, IncrementsRelativeDLT, Data-method (maxDose), [44](#)
- maxDose, IncrementsRelativeParts, DataParts-method (maxDose), [44](#)
- maxSize, [45, 48](#)
- maxSize, CohortSize-method (maxSize), [45](#)
- mcmc, [45](#)
- mcmc, Data, LogisticNormal, McmcOptions-method (mcmc), [45](#)
- mcmc, GeneralData, GeneralModel, McmcOptions-method (mcmc), [45](#)

- McmcOptions, [28, 46, 46, 66, 69, 70](#)
- McmcOptions-class, [47](#)
- mean, [28, 29](#)
- MinimalInformative, [47](#)
- minSize, [45, 48](#)
- minSize, CohortSize-method (minSize), [48](#)
- Model, [6, 19, 20, 28, 30, 50, 61, 63, 83](#)
- Model-class, [48](#)
- NextBest, [50, 65](#)
- nextBest, [49](#)
- nextBest, NextBestDualEndpoint, numeric, Samples, DualEndpoint, Data-method (nextBest), [49](#)
- nextBest, NextBestMTD, numeric, Samples, Model, Data-method (nextBest), [49](#)
- nextBest, NextBestNCRM, numeric, Samples, Model, Data-method (nextBest), [49](#)
- nextBest, NextBestNCRM, numeric, Samples, Model, DataParts-method (nextBest), [49](#)
- nextBest, NextBestThreePlusThree, missing, missing, Data-method (nextBest), [49](#)
- NextBest-class, [51](#)
- NextBestDualEndpoint, [51, 51](#)
- NextBestDualEndpoint-class, [51](#)
- NextBestDualEndpointBeta, [51](#)
- NextBestDualEndpointEmax, [51](#)
- NextBestMTD, [51, 52, 52](#)
- NextBestMTD-class, [52](#)
- NextBestNCRM, [51, 53, 53](#)
- NextBestNCRM-class, [53](#)
- NextBestThreePlusThree, [51, 54, 54](#)
- NextBestThreePlusThree-class, [54](#)
- or-Stopping-Stopping, [54](#)
- or-Stopping-StoppingAny, [55](#)
- or-StoppingAny-Stopping, [55](#)
- plot, Data, missing-method, [56](#)
- plot, DataCombo, missing-method, [56](#)
- plot, DataDual, missing-method, [57](#)
- plot, DualSimulations, missing-method, [57](#)
- plot, DualSimulationsSummary, missing-method, [58](#)
- plot, GeneralSimulations, missing-method, [58](#)
- plot, GeneralSimulationsSummary, missing-method, [59](#)
- plot, Samples, ComboLogistic-method, [60](#)
- plot, Samples, DualEndpoint-method, [61](#)
- plot, Samples, Model-method, [61](#)
- plot, SimulationsSummary, missing-method, [62](#)
- plot.arrange, [63](#)
- prob, [63](#)
- prob, matrix, ComboLogistic, Samples-method (prob), [63](#)
- prob, numeric, Model, Samples-method (prob), [63](#)
- Quantiles2LogisticNormal, [6, 47, 48, 64](#)
- Report, [65](#)
- RuleDesign, [19, 28, 65, 65, 71, 86](#)
- RuleDesign-class, [65](#)
- RuleDesign, Samples, DualEndpoint, Data-method (RuleDesign), [65](#)
- Samples, [6, 7, 20, 28, 32, 46, 50, 60, 61, 63, 66, 66, 83](#)
- Samples-class, [66](#)
- sampleSize, [66](#)
- setSeed, [5, 31, 46, 67](#)
- setSeed, [67, 69–71](#)
- show, DualSimulationsSummary-method, [67](#)
- show, GeneralSimulationsSummary-method, [68](#)
- show, SimulationsSummary-method, [68](#)
- simulate, Design-method, [69](#)
- simulate, DualDesign-method, [70](#)
- simulate, RuleDesign-method, [71](#)
- Simulations, [26, 69, 71, 71, 72, 85](#)
- Simulations-class, [72](#)
- SimulationsSummary, [27, 62, 68, 85](#)
- SimulationsSummary-class, [72](#)
- size, [73](#)
- size, CohortSizeConst, ANY, Data-method (size), [73](#)
- size, CohortSizeDLT, ANY, Data-method (size), [73](#)
- size, CohortSizeMax, ANY, Data-method (size), [73](#)
- size, CohortSizeMin, ANY, Data-method (size), [73](#)
- size, CohortSizeParts, ANY, DataParts-method (size), [73](#)
- size, CohortSizeRange, ANY, Data-method (size), [73](#)
- Stopping, [19, 54, 55, 74, 75, 77, 83, 89, 90](#)
- Stopping-class, [74](#)
- StoppingAll, [74, 74, 90](#)
- StoppingAll-class, [74](#)
- StoppingAny, [54, 55, 75, 75](#)
- StoppingAny-class, [75](#)
- StoppingCohortsNearDose, [74, 76, 76](#)
- StoppingCohortsNearDose-class, [76](#)
- StoppingList, [74, 76, 77](#)
- StoppingList-class, [77](#)

StoppingMinCohorts, [74](#), [77](#), [77](#)  
StoppingMinCohorts-class, [78](#)  
StoppingMinPatients, [74](#), [78](#), [78](#)  
StoppingMinPatients-class, [78](#)  
StoppingMTDdistribution, [74](#), [79](#), [79](#)  
StoppingMTDdistribution-class, [79](#)  
StoppingPatientsNearDose, [74](#), [80](#), [80](#)  
StoppingPatientsNearDose-class, [80](#)  
StoppingTargetBiomarker, [74](#), [80](#), [81](#)  
StoppingTargetBiomarker-class, [81](#)  
StoppingTargetProb, [74](#), [81](#), [81](#)  
StoppingTargetProb-class, [82](#)  
stopTrial, [82](#)  
stopTrial, StoppingAll, ANY, ANY, ANY, ANY-method  
(stopTrial), [82](#)  
stopTrial, StoppingAny, ANY, ANY, ANY, ANY-method  
(stopTrial), [82](#)  
stopTrial, StoppingCohortsNearDose, numeric, ANY, ANY, Data-method  
(stopTrial), [82](#)  
stopTrial, StoppingList, ANY, ANY, ANY, ANY-method  
(stopTrial), [82](#)  
stopTrial, StoppingMinCohorts, ANY, ANY, ANY, Data-method  
(stopTrial), [82](#)  
stopTrial, StoppingMinPatients, ANY, ANY, ANY, Data-method  
(stopTrial), [82](#)  
stopTrial, StoppingMTDdistribution, numeric, Samples, Model, ANY-method  
(stopTrial), [82](#)  
stopTrial, StoppingPatientsNearDose, numeric, ANY, ANY, Data-method  
(stopTrial), [82](#)  
stopTrial, StoppingTargetBiomarker, numeric, Samples, DualEndpoint, ANY-method  
(stopTrial), [82](#)  
stopTrial, StoppingTargetProb, numeric, Samples, Model, ANY-method  
(stopTrial), [82](#)  
summary, DualSimulations-method, [84](#)  
summary, GeneralSimulations-method, [84](#)  
summary, Simulations-method, [85](#)  
  
ThreePlusThreeDesign, [86](#)  
  
update, Data-method, [86](#)  
update, DataCombo-method, [87](#)  
update, DataDual-method, [87](#)  
update, DataParts-method, [88](#)  
  
Validate, [88](#)  
  
writeLines, [89](#)  
writeModel, [89](#)