

Package 'trialr'

August 9, 2018

Type Package

Title Clinical Trial Designs in 'RStan'

Version 0.0.3

Date 2018-08-09

Maintainer Kristian Brock <kristian.brock@gmail.com>

Description A showcase of Bayesian clinical trial designs, implemented in 'RStan' and R. Some designs are implemented in R for the first time (e.g. 'EffTox' by Thall and Cook, 2004, <https://biostatistics.mdanderson.org/softwaredownload/SingleSoftware.aspx?Software_Id=2>). Given the emphasis on flexibility in Bayesian analysis, the implementations in a common language and style may serve as a cookbook to encourage the use of Bayesian methods in trials. Compiled 'RStan' models are provided in addition to helper classes and functions to perform simulations and inference on observed trial outcomes. There is a preponderance of early phase trial designs because this is where Bayesian methods are used most. If there is a published Bayesian design you want implemented in 'Stan', get in touch.

License GPL-3

Depends R (>= 3.0.2), Rcpp (>= 0.12.8), methods (>= 3.3.2)

Imports rstan (>= 2.12.1), rstantools (>= 1.1.0), loo (>= 1.1.0), gtools, stringr, dplyr, tidyr, magrittr, ggplot2

LinkingTo StanHeaders (>= 2.12.0.1), rstan (>= 2.12.1), BH (>= 1.62.0.1), Rcpp (>= 0.12.8), RcppEigen (>= 0.3.2.9.0)

Suggests knitr, rmarkdown, ggridges, testthat

URL <https://github.com/brockk/trialr>

BugReports <https://github.com/brockk/trialr/issues>

RoxygenNote 6.0.1

VignetteBuilder knitr

NeedsCompilation yes

Author Kristian Brock [aut, cre],
Trustees of Columbia University [cph]

Repository CRAN

Date/Publication 2018-08-09 10:20:03 UTC

R topics documented:

trialr-package	3
as.data.frame.crm_fit	7
as.data.frame.efftox_fit	7
crm_fit-class	8
crm_params-class	9
crm_process	10
df_parse_outcomes	11
efftox_analysis	12
efftox_analysis_to_df	13
efftox_contour_plot	13
efftox_dtps	15
efftox_fit-class	16
efftox_get_tox	17
efftox_parameters_demo	18
efftox_params-class	19
efftox_parse_outcomes	20
efftox_process	21
efftox_simulate	22
efftox_solve_p	23
efftox_superiority	24
efftox_utility	25
efftox_utility_density_plot	26
gather_samples.crm_fit	27
gather_samples.efftox_fit	27
model_BebopInPeps2	28
model_EffTox	28
model_ThallHierarchicalBinary	29
peps2_get_data	30
peps2_params	32
peps2_process	33
peps2_run_sims	35
plot.crm_fit	36
plot.efftox_fit	37
print.crm_fit	37
print.efftox_fit	38
ranBin2	38
Rcpp_model_BebopInPeps2-class	39
Rcpp_model_EffTox-class	39
Rcpp_model_ThallHierarchicalBinary-class	39
stanmodels	40
stan_crm	40
stan_efftox	42

stan_efftox_demo 45
stan_files 46
summary.crm_fit 46
summary.efftox_fit 47
thallhierarchicalbinary_parameters_demo 47
thallhierarchicalbinary_params 48

Index 49

trialr-package *Clinical Trial Designs in 'RStan'*

Description

A showcase of Bayesian clinical trial designs, implemented in 'RStan' and R. Some designs are implemented in R for the first time (e.g. 'EffTox' by Thall and Cook, 2004, <<https://biostatistics.mdanderson.org/softwaredownload>>). Given the emphasis on flexibility in Bayesian analysis, the implementations in a common language and style may serve as a cookbook to encourage the use of Bayesian methods in trials. Compiled 'RStan' models are provided in addition to helper classes and functions to perform simulations and inference on observed trial outcomes. There is a preponderance of early phase trial designs because this is where Bayesian methods are used most. If there is a published Bayesian design you want implemented in 'Stan', get in touch.

Details

trialr is a collection of Bayesian clinical trial designs implemented in rstan and R. Bayesian designs are being used more commonly in clinical trials however software implementation remains a frequent challenge. rstan is the R implementation of Stan, a probabilistic programming language that performs full Bayesian statistical inference using MCMC sampling. It allows the statistician to focus on specifying the model (the good bit) without having to worry about the prior-to-posterior analysis (the nuisance bit). trialr seeks to be a showcase of published examples of Bayesian clinical trial designs, implemented in a common language and style, and brought together in a single R package.

Clinical trial designs currently implemented:

- EffTox, a design for seamless phase I-II dose-finding, published by Thall & Cook 2004
- Hierarchical model for a phase II trial of a treatment in a disease with multiple sub-types using binary response outcomes, published by Thall et al 2003
- BEBOP, a stratified medicine design for studying efficacy and toxicity in phase II that incorporates predictive baseline information, as developed for the PePS2 trial and submitted for publication by Brock, et al. 2017

Index of help topics:

```
Rcpp_model_BebopInPeps2-class
      Compiled Stan model for BEBOP implementation in
      PePS2 clinical trial
Rcpp_model_EffTox-class
```

	Compiled Stan model for EffTox dose-finding design
Rcpp_model_ThallHierarchicalBinary-class	Compiled Stan model for Thall et al.'s hierarchical Bayesian model for binary data
as.data.frame.crm_fit	Convert <code>crm_fit</code> object to 'data.frame'.
as.data.frame.efftox_fit	Convert <code>efftox_fit</code> object to 'data.frame'.
crm_fit-class	Class of model fit by 'trialr' using the CRM dose-finding design.
crm_params-class	Container class for parameters to fit the CRM models in trialr.
crm_process	Process RStan samples from a CRM model
df_parse_outcomes	Parse a string of dose-finding trial outcomes to binary vector notation.
efftox_analysis	Processed results of an EffTox dose-update analysis
efftox_analysis_to_df	EffTox analysis to data.frame
efftox_contour_plot	Plot EffTox utility contours
efftox_dtps	Calculate dose-transition pathways for an EffTox study
efftox_fit-class	Class of model fit by 'trialr' using the EffTox dose-finding design.
efftox_get_tox	Get the Prob(Tox) for Prob(Eff) and utility pairs
efftox_parameters_demo	Get parameters to run the EffTox demo
efftox_params-class	Container class for parameters to fit the EffTox model in trialr.
efftox_parse_outcomes	Parse a string of EffTox outcomes to binary vector notation.
efftox_process	Process RStan samples from an EffTox model
efftox_simulate	Run EffTox simulations
efftox_solve_p	Calculate the p-index for EffTox utility contours
efftox_superiority	Get dose-superiority matrix in EffTox
efftox_utility	Get the utility of efficacy & toxicity probability pairs
efftox_utility_density_plot	Plot densities of EffTox dose utilities
gather_samples.crm_fit	Extract tall data.frame of posterior prob_tox samples.
gather_samples.efftox_fit	Extract tall data.frame of posterior variable samples.
model_BebopInPeps2	Stan model for BEBOP implementation in PePS2 clinical trial

model_EffTox	Stan model for EffTox dose-finding design
model_ThallHierarchicalBinary	Stan model for Thall et al.'s hierarchical Bayesian model for binary data
peps2_get_data	Get data to run the PePS2 trial example
peps2_params	Parameters to be passed to 'BebopInPeps2' model in Stan
peps2_process	Process RStan samples from a BEBOP model fit to PePS2 data
peps2_run_sims	Run simulations of BEBOP in PePS2
plot.crm_fit	Plot an crm_fit
plot.efftox_fit	Plot an efftox_fit
print.crm_fit	Print crm_fit object.
print.efftox_fit	Print efftox_fit object.
ranBin2	Sample pairs of correlated binary events
stan_crm	Fit a CRM model
stan_efftox	Fit an EffTox model
stan_efftox_demo	Fit the EffTox model presented in Thall et al. (2014)
stan_files	Stan file locations
stanmodels	Stan models used by trialr
summary.crm_fit	Obtain summary of an crm_fit
summary.efftox_fit	Obtain summary of an efftox_fit
thallhierarchicalbinary_parameters_demo	Get parameters to run the demo of Thall Hierarchical Binary model
thallhierarchicalbinary_params	Parameters to be passed to the ThallHierarchicalBinary model in Stan
trialr-package	Clinical Trial Designs in 'RStan'

Further information is available in the following vignettes:

BEBOP	BEBOP: Bayesian Evaluation of Bivariate Binary Outcomes and Predictive Information
CRM-visualisation	Vignette Title (source, pdf)
CRM	Vignette Title (source, pdf)
EffTox	EffTox in trialr (source, pdf)
HierarchicalBayesianResponse	Hierarchical Bayesian Model for Binary Responses (source, pdf)
trialr-overview	Vignette Title (source, pdf)

Author(s)

NA

Maintainer: Kristian Brock <kristian.brock@gmail.com>

References

- Thall, PF, and JD Cook. 2004. "Dose-Finding Based on Efficacy-Toxicity Trade-Offs". *Biometrics* 60 (3): 684-93.
- Thall, Herrick, Nguyen, Venier, and Norris. 2014. "Effective sample size for computing prior hyperparameters in Bayesian phase I-II dose-finding". *Clinical Trials* 11 (6): 657-66. doi:10.1177/1740774514547397.
- Thall, Wathen, Bekele, Champlin, Baker, & Benjamin. 2003. "Hierarchical Bayesian approaches to phase II trials in diseases with multiple subtypes". *Statistics in Medicine*, 22(5), 763-780. <https://doi.org/10.1002/sim.1399>

See Also

[model_EffTox](#)
[model_ThallHierarchicalBinary](#)
[model_BebopInPeps2](#)

Examples

```
library(trialr)

# EffTox usage
dat <- efftox_parameters_demo()
# Add some outcomes
dat$num_patients <- 3
dat$eff <- c(0, 1, 1)
dat$tox <- c(0, 0, 1)
dat$doses <- c(1, 2, 3)
# Use rstan to obtain samples from the posterior distributions
samp <- rstan::sampling(stanmodels$EffTox, data = dat)
# Perform inference on posterior samples. E.g. posterior mean drug efficacy
colMeans(rstan::extract(samp, 'prob_eff')[[1]])

## Not run
## Hierarchical model for responses in a disease with multiple subtypes
# dat <- thallhierarchicalbinary_parameters_demo()
# samp <- rstan::sampling(stanmodels$ThallHierarchicalBinary, data = dat)
# rstan::plot(samp, pars = 'p') # Plot the modelled response rates in the ten subtypes

## BEBOP in PePS2 usage
# set.seed(123)
# dat <- peps2_get_data(num_patients = 60,
#                       prob_eff = c(0.167, 0.192, 0.5, 0.091, 0.156, 0.439),
#                       prob_tox = rep(0.1, 6),
#                       eff_tox_or = rep(1, 6))
# samp <- rstan::sampling(stanmodels$BebopInPeps2, data = dat)
# decision <- peps2_process(dat, samp)
# decision$Accept # Accept in cohort 2, 3, 5, 6 but not 1 or 4
# decision$ProbEff # The probability of efficacy is driving that decision
```

as.data.frame.crm_fit *Convert crm_fit object to data.frame.*

Description

Convert crm_fit object to data.frame.

Usage

```
## S3 method for class 'crm_fit'  
as.data.frame(x, ...)
```

Arguments

x [crm_fit](#) object to convert.
... Extra parameters, passed onwards.

Value

A data.frame

as.data.frame.efftox_fit
Convert efftox_fit object to data.frame.

Description

Convert efftox_fit object to data.frame.

Usage

```
## S3 method for class 'efftox_fit'  
as.data.frame(x, ...)
```

Arguments

x [efftox_fit](#) object to convert.
... Extra parameters, passed onwards.

Value

A data.frame

 crm_fit-class

*Class of model fit by **trialr** using the CRM dose-finding design.*

Description

Class of model fit by **trialr** using the CRM dose-finding design.

Usage

```
crm_fit(dose_indices, recommended_dose, prob_tox, median_prob_tox,
        modal_mtd_candidate, prob_mtd, dat, fit)
```

Arguments

dose_indices	A vector of integers representing the dose-levels under consideration.
recommended_dose	An integer representing the dose-level recommended for the next patient or cohort; or NA if stopping is recommended. The recommended dose typically has associated probability of DLT closest to the target toxicity rate. Contrast to modal_mtd_candidate.
prob_tox	The posterior mean probabilities of toxicity at doses 1:n; a vector of numbers between 0 and 1.
median_prob_tox	The posterior median probabilities of toxicity at doses 1:n; a vector of numbers between 0 and 1.
modal_mtd_candidate	An integer representing the dose-level most likely to be the MTD, i.e. the dose-level that maximises prob_mtd.
prob_mtd	The posterior probability that each dose is the MTD, by the chosen model; a vector of numbers between 0 and 1.
dat	Object crm_params containing data passed to sampling .
fit	An object of class stanfit , containing the posterior samples.

Details

See `methods(class = "crm_fit")` for an overview of available methods.

See Also

[stan_crm](#) [crm_process](#)

crm_params-class	<i>Container class for parameters to fit the CRM models in trialr.</i>
------------------	--

Description

Container class for parameters to fit the CRM models in trialr.

Usage

```
crm_params(skeleton, target, a0 = NULL, alpha_mean = NULL,
  alpha_sd = NULL, beta_mean = NULL, beta_sd = NULL, beta_shape = NULL,
  beta_inverse_scale = NULL)
```

Arguments

skeleton	a vector of the prior guesses of toxicity at doses. This should be a monotonically-increasing vector of numbers between 0 and 1.
target	the target toxicity probability, a number between 0 and 1. This value would normally be one of the values in skeleton, but that is not a requirement.
a0	Value of fixed intercept parameter. Only required for certain models. See Details.
alpha_mean	Prior mean of intercept variable for normal prior. Only required for certain models. See Details.
alpha_sd	Prior standard deviation of intercept variable for normal prior. Only required for certain models. See Details.
beta_mean	Prior mean of gradient variable for normal prior. Only required for certain models. See Details.
beta_sd	Prior standard deviation of slope variable for normal prior. Only required for certain models. See Details.
beta_shape	Prior shape parameter of slope variable for gamma prior. Only required for certain models. See Details.
beta_inverse_scale	Prior inverse scale parameter of slope variable for gamma prior. Only required for certain models. See Details.

Details

Different model parameterisations require that difference parameter values are specified.

Requirements of empiric model

* beta_sd

Requirements of logistic model

* a0 * beta_mean * beta_sd

Requirements of logistic_gamma model

* a0 * beta_shape * beta_inverse_scale

Requirements of logistics model

* a0 * alpha_mean * alpha_sd * beta_mean * beta_sd

See Also

[stan_crm](#) [crm_process](#)

crm_process

Process RStan samples from a CRM model

Description

Process RStan samples from a CRM model to make inferences about dose-toxicity and which dose should be recommended next. Typically, this function is not required to be called explicitly by the user because [stan_crm](#) will call it implicitly.

Usage

```
crm_process(dat, fit)
```

Arguments

dat	An instance of crm_params , a list of CRM parameters.
fit	An instance of <code>rstan::stanmodel</code> , derived by fitting one of the trialr CRM models.

Value

An instance of [crm_fit](#).

See Also

[stan_crm](#) [crm_params](#)

Examples

```
## Not run:  
dat <- list(  
  num_doses = 5,  
  skeleton = c(0.05, 0.12, 0.25, 0.40, 0.55),  
  target = 0.25,  
  beta_sd = sqrt(1.34),  
  num_patients = 3,  
  doses = c(1, 2, 3),
```

```
  tox = c(0, 0, 1)
)
samp <- rstan::sampling(stanmodels$CrmEmpiricNormalPrior,
  data = dat, seed = 123)
decision <- crm_process(dat, samp)

## End(Not run)
```

df_parse_outcomes *Parse a string of dose-finding trial outcomes to binary vector notation.*

Description

Parse a string of dose-finding trial outcomes to the binary vector notation required by Stan for model invocation. The outcome string describes the doses given and outcomes observed. The format of the string is the pure phase I analogue to that described in Brock et al. (2017). The letters T and N are used to represent patients that experienced (T)oxicity and (N)o toxicity. These letters are concatenated after numerical dose-levels to convey the outcomes of cohorts of patients. For instance, 2NNT represents a cohort of three patients that were treated at dose-level 2, one of whom experienced toxicity, and two that did not. The results of cohorts are separated by spaces. Thus, 2NNT 1NN extends our previous example, where the next cohort of two were treated at dose-level 1 and neither experienced toxicity. See examples.

Usage

```
df_parse_outcomes(outcome_string, as.list = TRUE)
```

Arguments

`outcome_string` character string, conveying doses given and outcomes observed.
`as.list` TRUE (be default) to return a list; FALSE to return a data.frame

Value

If `as.list == TRUE`, a list with elements `tox`, `doses` and `num_patients`. These elements are congruent with those of the same name in `crm_params`, for example. If `as.list == FALSE`, a data.frame with columns `tox` and `doses`.

References

Brock, K., Billingham, L., Copland, M., Siddique, S., Sirovica, M., & Yap, C. (2017). Implementing the EffTox dose-finding design in the Matchpoint trial. *BMC Medical Research Methodology*, 17(1), 112. <https://doi.org/10.1186/s12874-017-0381-x>

Examples

```
x = efftox_parse_outcomes('1NNE 2EEN 3TBB')
x$num_patients == 9
x$eff == c(0, 0, 1, 1, 1, 0, 0, 1, 1)
sum(x$tox) == 3
```

 efftox_analysis

Processed results of an EffTox dose-update analysis

Description

Processed results of an EffTox dose-update analysis, yielded by `efftox_process`. The model is invoked on data and RStan performs sampling from the posterior distribution. `efftox_process` then conducts the required processing to ultimately arrive at a dose recommendation.

Details

The object is essentially a list with slots:

- `dose_indices`, a vector of integers representing the dose-levels under consideration.
- `recommended_dose`, an integer representing the dose recommended for the next patient or cohort. TODO: what happens when stop recommended?
- `prob_eff`, the posterior mean probabilities of efficacy at doses 1:n; a vector of numbers between 0 and 1.
- `prob_tox`, the posterior mean probabilities of toxicity at doses 1:n; a vector of numbers between 0 and 1.
- `prob_acc_eff`, the posterior mean probabilities that efficacy at the doses is acceptable, i.e. that it exceeds the minimum acceptable efficacy threshold; a vector of numbers between 0 and 1.
- `prob_acc_tox`, the posterior mean probabilities that toxicity at the doses is acceptable, i.e. that it is less than the maximum toxicity threshold; a vector of numbers between 0 and 1.
- `utility`, the utilities of doses 1:n, calculated by plugging the posterior mean probabilities of efficacy and toxicity into the utility formula, as advocated by Thall & Cook. Contrast to `post_utility`; a vector of numbers.
- `post_utility`, the posterior mean utilities of doses 1:n, calculated from the posterior distributions of the utilities. This is in contrast to `utility`, which uses plug-in posterior means of efficacy and toxicity, as advocated by Thall & Cook; a vector of numbers.
- `acceptable`, a vector of logical values to indicate whether doses 1:n are acceptable, according to the rules for acceptable efficacy & toxicity, and rules on not skipping untested doses.
- `fit`, An object of class `stanfit`, containing the posterior samples.

Value

A list with elements identified above.

See Also[efftox_process](#)

 efftox_analysis_to_df *EffTox analysis to data.frame*

Description

Convenient function to turn an [efftox_analysis](#) into a `data.frame`.

Usage

```
efftox_analysis_to_df(x)
```

Arguments

`x` An [efftox_analysis](#)

Value

a `data.frame`

Examples

```
dat <- efftox_parameters_demo()
dat$num_patients <- 3
dat$eff <- c(0, 1, 1)
dat$tox <- c(0, 0, 1)
dat$doses <- c(1, 2, 3)
fit <- rstan::sampling(stanmodels$EffTox, data = dat)
decision <- efftox_process(dat, fit)
df <- efftox_analysis_to_df(decision)
round(df$Utility, 2) == c(-0.64, 0.04, 0.24, -0.05, -0.19)
```

 efftox_contour_plot *Plot EffTox utility contours*

Description

Plot EffTox utility contours. The probability of efficacy is on the x-axis and toxicity on the y-axis. The zero-utility curve is plotted bolder. The three "hinge points" are plotted as blue triangles. Optional Prob(Efficacy) vs Prob(Toxicity) points can be added; these are shown as red numerals, enumerated in the order provided.

Usage

```
efftox_contour_plot(dat, use_ggplot = FALSE, prob_eff = NULL,
  prob_tox = NULL, num_points = 1000, util_vals = seq(-3, 3, by = 0.2))
```

Arguments

<code>dat</code>	An instance of <code>efftox_params</code> , a list of EffTox parameters. An example is yielded by <code>efftox_parameters_demo</code> .
<code>use_ggplot</code>	logical, TRUE to use ggplot2. Defaults to FALSE to use standard R graphics.
<code>prob_eff</code>	an optional vector of numbers between 0 and 1, containing the efficacy probabilities of extra points to add to the plot as points, e.g. the posterior mean efficacy probabilities of the doses under investigation. Paired with <code>prob_tox</code> , thus they should be the same length.
<code>prob_tox</code>	an optional vector of numbers between 0 and 1, containing the toxicity probabilities of extra points to add to the plot as points, e.g. the posterior mean toxicity probabilities of the doses under investigation. Paired with <code>prob_eff</code> , thus they should be the same length.
<code>num_points</code>	integer for number of points to calculate on each curve. The default is 1000 and this should be plenty.
<code>util_vals</code>	A contour is plotted for each of these utility values. The default is contours spaced by 0.2 between from -3 and 3, i.e. <code>seq(-3, 3, by = 0.2)</code> .

Value

if `use_ggplot = TRUE`, an instance of `ggplot`; else no object is returned. Omit assignment in either case to just view the plot.

See Also

[efftox_params](#)
[efftox_parameters_demo](#)

Examples

```
dat <- efftox_parameters_demo()
efftox_contour_plot(dat)
# Add posterior beliefs
dat$num_patients <- 3
dat$eff <- c(0, 1, 1)
dat$tox <- c(0, 0, 1)
dat$doses <- c(1, 2, 3)
fit <- rstan::sampling(stanmodels$EffTox, data = dat)
decision <- efftox_process(dat, fit)
efftox_contour_plot(dat, prob_eff = decision$prob_eff,
  prob_tox = decision$prob_tox)
title('EffTox utility contours')
# The same with ggplot2
efftox_contour_plot(dat, prob_eff = decision$prob_eff,
```

```

prob_tox = decision$prob_tox,
use_ggplot = TRUE) +
ggplot2::ggtitle('EffTox utility contours')

```

efftox_dtps

Calculate dose-transition pathways for an EffTox study

Description

Calculate dose-transition pathways for an EffTox study.

Usage

```
efftox_dtps(dat, cohort_sizes, next_dose, ...)
```

Arguments

dat	An instance of efftox_params , a list of EffTox parameters. An example is yielded by efftox_parameters_demo .
cohort_sizes	vector of future cohort sizes, i.e. positive integers. E.g. To calculate paths for the the next cohort of two followed by the next cohort of three, use <code>c(2, 3)</code> .
next_dose	the dose-level to be given to the immediately next cohort.
...	extra params passed to <code>rstan::sampling</code> .

Value

dose pathways in a `data.frame`.

References

Brock et al. (submitted 2017), Implementing the EffTox Dose-Finding Design in the Matchpoint Trial.

See Also

[efftox_params](#)
[efftox_parameters_demo](#)

Examples

```

# Calculate the paths for the first cohort of 3 in Thall et al 2014 example
dat <- efftox_parameters_demo()
## Not run:
dtps1 <- efftox_dtps(dat = dat, cohort_sizes = c(3), next_dose = 1)

## End(Not run)
# To calculate future paths in a partially-observed trial

```

```

dat <- efftox_parameters_demo()
dat$doses = array(c(1,1,1))
dat$eff = array(c(0,0,0))
dat$tox = array(c(1,1,1))
dat$num_patients = 3
## Not run:
dtps2 <- efftox_dtps(dat = dat, cohort_sizes = c(3), next_dose = 1)

## End(Not run)

```

efftox_fit-class *Class of model fit by **trialr** using the EffTox dose-finding design.*

Description

Class of model fit by **trialr** using the EffTox dose-finding design.

Usage

```

efftox_fit(dose_indices, recommended_dose, prob_eff, prob_tox, prob_acc_eff,
  prob_acc_tox, utility, post_utility, acceptable, dat, fit)

```

Arguments

dose_indices	A vector of integers representing the dose-levels under consideration.
recommended_dose	An integer representing the dose-level recommended for the next patient or cohort; or NA if stopping is recommended.
prob_eff	The posterior mean probabilities of efficacy at doses 1:n; a vector of numbers between 0 and 1.
prob_tox	The posterior mean probabilities of toxicity at doses 1:n; a vector of numbers between 0 and 1.
prob_acc_eff	The posterior mean probabilities that efficacy at the doses is acceptable, i.e. that it exceeds the minimum acceptable efficacy threshold; a vector of numbers between 0 and 1.
prob_acc_tox	The posterior mean probabilities that toxicity at the doses is acceptable, i.e. that it is less than the maximum toxicity threshold; a vector of numbers between 0 and 1.
utility	The utilities of doses 1:n, calculated by plugging the posterior mean probabilities of efficacy and toxicity into the utility formula, as advocated by Thall & Cook. Contrast to post_utility; a vector of numbers.
post_utility	The posterior mean utilities of doses 1:n, calculated from the posterior distributions of the utilities. This is in contrast to utility, which uses plug-in posterior means of efficacy and toxicity, as advocated by Thall & Cook; a vector of numbers.

acceptable	A vector of logical values to indicate whether doses 1:n are acceptable, according to the rules for acceptable efficacy & toxicity, and rules on not skipping untested doses.
dat	Object <code>efftox_params</code> containing data passed to <code>sampling</code> .
fit	An object of class <code>stanfit</code> , containing the posterior samples.

Details

See `methods(class = "efftox_fit")` for an overview of available methods.

See Also

[stan_efftox](#) [stan_efftox_demo](#) [efftox_process](#)

efftox_get_tox	<i>Get the Prob(Tox) for Prob(Eff) and utility pairs</i>
----------------	--

Description

Get the probability of toxicity for probability-of-efficacy and utility pairs

Usage

```
efftox_get_tox(eff, util, p, eff0, tox1)
```

Arguments

eff	Probability of efficacy; number between 0 and 1
util	Utility score; number
p	p-index of EffTox utility contours. Use <code>efftox_solve_p</code>
eff0	Efficacy probability required when toxicity is impossible; a number between 0 and 1
tox1	Toxicity probability permitted when efficacy is guaranteed; a number between 0 and 1

Value

Probability(s) of toxicity

Note

Various ways of vectorising the function are demonstrated in the examples

See Also

[efftox_solve_p](#)

Examples

```
p <- efftox_solve_p(0.5, 0.65, 0.7, 0.25)

prob_tox <- efftox_get_tox(0.7, 0, p, eff0 = 0.5, tox1 = 0.65)
round(prob_tox, 2) == 0.25

prob_tox <- efftox_get_tox(0.7, seq(-0.5, 0.25, by = 0.25), p, eff0 = 0.5,
                          tox1 = 0.65)
round(prob_tox, 2) == c(0.57, 0.41, 0.25, 0.09)

prob_tox <- efftox_get_tox(c(0.5, 0.7, 0.8), 0.25, p, eff0 = 0.5, tox1 = 0.65)
round(prob_tox, 2) == c(NaN, 0.09, 0.22)

prob_tox <- efftox_get_tox(c(0.5, 0.7, 0.8), c(-1, 0, 1), p, eff0 = 0.5,
                          tox1 = 0.65)
round(prob_tox, 2) == c(0.63, 0.25, NaN)
```

efftox_parameters_demo

Get parameters to run the EffTox demo

Description

Get parameters to run the EffTox demo. These match those used to demonstrate EffTox in Thall et al. 2014.

Usage

```
efftox_parameters_demo()
```

Value

a list of parameters, described in `efftox_params`

References

Thall, Herrick, Nguyen, Venier & Norris. 2014, Effective sample size for computing prior hyperparameters in Bayesian phase I-II dose-finding

See Also

[efftox_params](#)

Examples

```
dat <- efftox_parameters_demo()
names(dat)
dat$real_doses == c(1, 2, 4, 6.6, 10)
```

efftox_params-class *Container class for parameters to fit the EffTox model in trialr.*

Description

Container class for parameters to fit the EffTox model in trialr.

Usage

```
efftox_params(real_doses, efficacy_hurdle, toxicity_hurdle, p_e, p_t, eff0,
  tox1, eff_star, tox_star, alpha_mean, alpha_sd, beta_mean, beta_sd,
  gamma_mean, gamma_sd, zeta_mean, zeta_sd, eta_mean, eta_sd, psi_mean, psi_sd)
```

Arguments

real_doses	a vector of numbers. The doses under investigation. They should be ordered from lowest to highest and be in consistent units. E.g., to conduct a dose-finding trial of doses 10mg, 20mg and 50mg, use c(10, 20, 50).
efficacy_hurdle	Minimum acceptable efficacy probability. A number between 0 and 1.
toxicity_hurdle	Maximum acceptable toxicity probability. A number between 0 and 1.
p_e	Certainty required to infer a dose is acceptable with regards to being probably efficacious; a number between 0 and 1.
p_t	Certainty required to infer a dose is acceptable with regards to being probably tolerable; a number between 0 and 1.
eff0	Efficacy probability required when toxicity is impossible; a number between 0 and 1 (see Details).
tox1	Toxicity probability permitted when efficacy is guaranteed; a number between 0 and 1 (see Details).
eff_star	Efficacy probability of an equi-utility third point (see Details).
tox_star	Toxicity probability of an equi-utility third point (see Details).
alpha_mean	The prior normal mean of the intercept term in the toxicity logit model. A number.
alpha_sd	The prior normal standard deviation of the intercept term in the toxicity logit model. A number.
beta_mean	The prior normal mean of the slope term in the toxicity logit model. A number.
beta_sd	The prior normal standard deviation of the slope term in the toxicity logit model. A number.
gamma_mean	The prior normal mean of the intercept term in the efficacy logit model. A number.
gamma_sd	The prior normal standard deviation of the intercept term in the efficacy logit model. A number.

zeta_mean	The prior normal mean of the slope term in the efficacy logit model. A number.
zeta_sd	The prior normal standard deviation of the slope term in the efficacy logit model. A number.
eta_mean	The prior normal mean of the squared term coefficient in the efficacy logit model. A number.
eta_sd	The prior normal standard deviation of the squared term coefficient in the efficacy logit model. A number.
psi_mean	The prior normal mean of the association term in the combined efficacy-toxicity model. A number.
psi_sd	The prior normal standard deviation of the association term in the combined efficacy-toxicity model. A number.

See Also

[stan_efftox](#) [stan_efftox_demo](#) [efftox_process](#)

efftox_parse_outcomes *Parse a string of EffTox outcomes to binary vector notation.*

Description

Parse a string of EffTox outcomes to the binary vector notation required by Stan for model invocation. The outcome string describes the doses given and outcomes observed. The format of the string is described in Brock et al. (2017). The letters E, T, N and B are used to represent patients that experienced (E)fficacy only, (T)oxicity only, (B)oth efficacy and toxicity, and (N)either. These letters are concatenated after numerical dose-levels to convey the outcomes of cohorts of patients. For instance, 2ETB represents a cohort of three patients that were treated at dose-level 2, and experienced efficacy, toxicity and both events, respectively. The results of cohorts are separated by spaces. Thus, 2ETB 1NN extends our previous example, where the next cohort of two were treated at dose-level 1 and both patients experienced neither efficacy nor toxicity. See examples.

We present the notation in the EffTox setting but it is applicable in general seamless phase I/II dose-finding scenarios.

Usage

```
efftox_parse_outcomes(outcome_string, as.list = TRUE)
```

Arguments

outcome_string character string, conveying doses given and outcomes observed.
as.list TRUE (be default) to return a list; FALSE to return a data.frame

Value

If `as.list == TRUE`, a list with elements `eff`, `tox`, `doses` and `num_patients`. These elements are congruent with those of the same name in `efftox_params`. If `as.list == FALSE`, a `data.frame` with columns `eff`, `tox`, and `doses`.

References

Brock, K., Billingham, L., Copland, M., Siddique, S., Sirovica, M., & Yap, C. (2017). Implementing the EffTox dose-finding design in the Matchpoint trial. *BMC Medical Research Methodology*, 17(1), 112. <https://doi.org/10.1186/s12874-017-0381-x>

Examples

```
x = efftox_parse_outcomes('1NNE 2EEN 3TBB')
x$num_patients == 9
x$eff == c(0, 0, 1, 1, 1, 0, 0, 1, 1)
sum(x$tox) == 3
```

efftox_process

Process RStan samples from an EffTox model

Description

Process RStan samples from an EffTox model to make inferences about dose-acceptability, dose-utility and which dose should be recommended next.

Usage

```
efftox_process(dat, fit)
```

Arguments

dat	An instance of efftox_params , a list of EffTox parameters. An example is yielded by efftox_parameters_demo .
fit	An instance of <code>rstan::stanmodel</code> , derived by fitting the trialr EffTox model.

Value

An instance of [efftox_fit](#).

See Also

[efftox_params](#)

[efftox_parameters_demo](#)

Examples

```

dat <- efftox_parameters_demo()
dat$num_patients <- 3
dat$eff <- c(0, 1, 1)
dat$tox <- c(0, 0, 1)
dat$doses <- c(1, 2, 3)
fit <- rstan::sampling(stanmodels$EffTox, data = dat)
decision <- efftox_process(dat, fit)
decision$recommended_dose == 3

```

efftox_simulate *Run EffTox simulations*

Description

Run EffTox simulations for assumed true efficacy and toxicity curves.

Usage

```

efftox_simulate(dat, num_sims, first_dose, true_eff, true_tox, cohort_sizes,
  ...)

```

Arguments

dat	An instance of <code>efftox_params</code> , a list of EffTox parameters. An example is yielded by <code>efftox_parameters_demo</code> .
num_sims	integer, number of simulated iterations
first_dose	integer, the dose-level to give to patient 1, e.g. 1 for the lowest dose.
true_eff	the true probabilities of efficacy at the doses under investigation; a vector of numbers between 0 and 1.
true_tox	the true probabilities of toxicity at the doses under investigation; a vector of numbers between 0 and 1.
cohort_sizes	a vector of integer cohort sizes. A dose decision is made when each cohort is completed and the next cohort is treated at the recommended dose. To conduct a trial using at most 20 patients, where dose is re-evaluated after every second patient, use <code>rep(2, 10)</code> . To conduct a trial of 8 patients where dose is re-evaluated after each single patient, use <code>rep(1, 8)</code> . Cohort size need not be uniform. E.g. <code>c(rep(1, 5), rep(3, 10))</code> represents a trial where the dose is re-evaluated after each patient for the first 5 patients, and then after every third patient for a further 30 patients.
...	Extra parameters provided via the ellipsis are passed to <code>stan::sampling</code>

Value

A list with named elements `recommended_dose`, `efficacies`, `toxicities`, and `doses_given`.

Examples

```

dat <- efftox_parameters_demo()
set.seed(123)
# Let's say we want to use only 2 chains. Extra args are passed to stan
## Not run:
sims <- efftox_simulate(dat, num_sims = 10, first_dose = 1,
                        true_eff = c(0.20, 0.40, 0.60, 0.80, 0.90),
                        true_tox = c(0.05, 0.10, 0.15, 0.20, 0.40),
                        cohort_sizes = rep(3, 13),
                        chains = 2)
table(sims$recommended_dose) / length(sims$recommended_dose)
table(unlist(sims$doses_given)) / length(unlist(sims$doses_given))
table(unlist(sims$doses_given)) / length(sims$recommended_dose)

## End(Not run)
# In real life, we would run thousands of iterations, not 10.
# This is an example.

```

efftox_solve_p

Calculate the p-index for EffTox utility contours

Description

Calculate the p-index for EffTox utility contours so that the neutral utility contour intersects the following points in the Prob(Efficacy) - Prob>Toxicity) plane: (eff0, 0), (1, tox1) and (eff_star, tox_star)

Usage

```
efftox_solve_p(eff0, tox1, eff_star, tox_star)
```

Arguments

eff0	Efficacy probability required when toxicity is impossible; a number between 0 and 1
tox1	Toxicity probability permitted when efficacy is guaranteed; a number between 0 and 1
eff_star	Efficacy probability of an equi-utility third point
tox_star	Toxicity probability of an equi-utility third point

Value

The p-index

References

Thall, Herrick, Nguyen, Venier & Norris. 2014, Effective sample size for computing prior hyper-parameters in Bayesian phase I-II dose-finding

Examples

```
efftox_solve_p(0.5, 0.65, 0.7, 0.25)
```

efftox_superiority *Get dose-superiority matrix in EffTox*

Description

Get a dose-superiority matrix from an EffTox dose analysis. EffTox seeks to choose the dose with the highest utility, thus superiority is inferred by posterior utility. The item in row i , col j is the posterior probability that the utility of dose j exceeds that of dose i .

Usage

```
efftox_superiority(fit)
```

Arguments

fit An instance of `rstan::stanmodel`, derived by sampling an EffTox model. Use `stan::sampling(stanmodels$EffTox, data = dat)`.

Value

n by n matrix, where n is number of doses under investigation. The item in row i , col j is the posterior probability that the utility of dose j exceeds that of dose i .

Examples

```
dat <- efftox_parameters_demo()
dat$num_patients <- 3
dat$eff <- c(0, 1, 1)
dat$tox <- c(0, 0, 1)
dat$doses <- c(1, 2, 3)
fit <- rstan::sampling(stanmodels$EffTox, data = dat)
sup_mat <- efftox_superiority(fit)
```

efftox_utility *Get the utility of efficacy & toxicity probability pairs*

Description

Get the utility of efficacy & toxicity probability pairs

Usage

```
efftox_utility(p, eff0, tox1, prob_eff, prob_tox)
```

Arguments

p	p-index of EffTox utility contours. Use <code>efftox_solve_p</code>
eff0	Efficacy probability required when toxicity is impossible; a number between 0 and 1
tox1	Toxicity probability permitted when efficacy is guaranteed; a number between 0 and 1
prob_eff	Probability of efficacy; number between 0 and 1
prob_tox	Probability of toxicity; number between 0 and 1

Value

Utility value(s)

See Also

[efftox_solve_p](#)

Examples

```
p <- efftox_solve_p(0.5, 0.65, 0.7, 0.25)

u <- efftox_utility(p, 0.5, 0.65, prob_eff = 0.7, prob_tox = 0.25)
round(u, 4) == 0

u <- efftox_utility(p, 0.5, 0.65, prob_eff = c(0.6, 0.7, 0.8),
                   prob_tox = c(0.1, 0.2, 0.3))
round(u, 2) == c(0.04, 0.08, 0.12)
```

`efftox_utility_density_plot`*Plot densities of EffTox dose utilities*

Description

Plot densities of EffTox dose utilities. Optionally plot only a subset of the doses by specifying the `doses` parameter. This function requires `ggplot2` be installed.

Usage

```
efftox_utility_density_plot(fit, doses = NULL)
```

Arguments

<code>fit</code>	An instance of <code>rstan::stanmodel</code> , derived by sampling an EffTox model. Use <code>stan::sampling(stanmodels\$EffTox, data = dat)</code> .
<code>doses</code>	optional, vector of integer dose-levels to plot. E.g. to plot only dose-levels 1, 2 & 3 (and suppress the plotting of any other doses), use <code>doses = 1:3</code>

Value

an instance of `ggplot`. Omit assignment to just view the plot.

Note

This function requires that `ggplot2` be installed.

Examples

```
dat <- efftox_parameters_demo()
dat$num_patients <- 3
dat$eff <- c(0, 1, 1)
dat$tox <- c(0, 0, 1)
dat$doses <- c(1, 2, 3)
fit <- rstan::sampling(stanmodels$EffTox, data = dat)
efftox_utility_density_plot(fit) + ggplot2::ggtitle('My doses') # Too busy?
# Specify subset of doses to make plot less cluttered
efftox_utility_density_plot(fit, doses = 1:3) + ggplot2::ggtitle('My doses')
```

`gather_samples.crm_fit`*Extract tall data.frame of posterior prob_tox samples.*

Description

Extract tall data.frame of posterior prob_tox samples.

Usage

```
gather_samples.crm_fit(x)
```

Arguments

x `crm_fit` object.

Value

data.frame

`gather_samples.efftox_fit`*Extract tall data.frame of posterior variable samples.*

Description

Extract tall data.frame of posterior variable samples.

Usage

```
gather_samples.efftox_fit(x, pars = "utility")
```

Arguments

x `efftox_fit` object.

pars One of 'utility', 'prob_tox', 'prob_eff', 'prob_acc_eff', or 'prob_acc_tox'

Value

data.frame

model_BebopInPeps2 *Stan model for BEBOP implementation in PePS2 clinical trial*

Description

This is the Stan implementation of the BEBOP design for the PePS2 trial, that incorporates baseline predictive information to study co-primary efficacy and toxicity outcomes.

The model is compiled when `trialr` is installed and is available at run-time under `stanmodels$BebopInPeps2`.

The design has been submitted for publication by Brock et al. in 2017.

See the BEBOP vignette for a detailed description of the probability model and a demonstration of the pertinent methods implemented in `trial`.

See Also

[peps2_params](#)
[peps2_get_data](#)
[peps2_process](#)

Examples

```
# Get model parameters as used in the PePS2 trial.
# This call randomly samples patient outcomes so set a seed
set.seed(123)
dat <- peps2_get_data(num_patients = 60,
                     prob_eff = c(0.167, 0.192, 0.5, 0.091, 0.156, 0.439),
                     prob_tox = rep(0.1, 6),
                     eff_tox_or = rep(1, 6))
# Fit the observed data to the model using rstan
samp <- rstan::sampling(stanmodels$BebopInPeps2, data = dat)
# The fit object is quite crude. Post-process to perform useful inference:
decision <- peps2_process(dat, samp)
decision$Accept    # Accept in cohort 2, 3, 5, 6 but not 1 or 4
decision$ProbEff   # The probability of efficacy is driving that decision
```

model_EffTox *Stan model for EffTox dose-finding design*

Description

This is the Stan implementation of the EffTox design for dose-finding in scenarios where dose decisions are guided by efficacy and toxicity outcomes.

The model is compiled when `trialr` is installed and is available at run-time under `stanmodels$EffTox`.

The design was first published by Thall & Cook in 2004. Advice on its use was materially updated in 2014.

See the EffTox vignette for a detailed description of the probability model and a demonstration of the pertinent methods implemented in `trial`.

References

- Thall & Cook. 2004, Dose-Finding Based on Efficacy-Toxicity Trade-Offs
- Thall, Herrick, Nguyen, Venier & Norris. 2014, Effective sample size for computing prior hyper-parameters in Bayesian phase I-II dose-finding

See Also

[efftox_params](#)

[efftox_parameters_demo](#)

[efftox_process](#)

Examples

```
# Get example parameters from Thall et al. 2014
dat <- efftox_parameters_demo()
# Add outcomes for three patients
dat$num_patients <- 3
dat$eff <- c(0, 1, 1)
dat$tox <- c(0, 0, 1)
dat$doses <- c(1, 2, 3)
# Fit the observed data to the model using rstan
fit <- rstan::sampling(stanmodels$EffTox, data = dat)
# The fit object is quite crude. Post-process to perform useful inference:
decision <- efftox_process(dat, fit)
decision$recommended_dose == 3
# Dose-level 3 is recommended for the next cohort of patients.
```

model_ThallHierarchicalBinary

Stan model for Thall et al.'s hierarchical Bayesian model for binary data

Description

This is the Stan implementation of Thall et al.'s hierarchical model for analysing the response rate of a common drug in multiple sub-types of a single disease.

The model is compiled when `trialr` is installed and is available at run-time under `stanmodels$ThallHierarchicalBinary`.

The design was first published by Thall et al. in 2003.

See the Hierarchical Bayesian Response vignette for a detailed description of the probability model and a demonstration of the implementation in `trial`.

References

- Thall, Wathen, Bekele, Champlin, Baker, & Benjamin. 2003. "Hierarchical Bayesian approaches to phase II trials in diseases with multiple subtypes". *Statistics in Medicine*, 22(5), 763-780. <https://doi.org/10.1002/sim.1399>

See Also[thallhierarchicalbinary_params](#)[thallhierarchicalbinary_parameters_demo](#)**Examples**

```
# Hierarchical model for responses in a disease with multiple subtypes
dat <- thallhierarchicalbinary_parameters_demo()
samp <- rstan::sampling(stanmodels$ThallHierarchicalBinary, data = dat)
rstan::plot(samp, pars = 'p') # Plot the modelled response rates in the subtypes
```

`peps2_get_data`*Get data to run the PePS2 trial example*

Description

Get data to run the BEBOP model in the PePS2 trial. The trial investigates pembrolizumab in non-small-cell lung cancer. Patients may be previously treated (PT) or treatment naive (TN). Pembro response rates in lung cancer have been shown to increase with PD-L1 tumour proportion score. PD-L1 score is measured at baseline. Each patient belongs to one of the Low, Medium or High categories. These two baseline variables stratify the patient population and are used as predictive variables to stratify the analysis. The BEBOP model studies co-primary efficacy and toxicity outcomes in the presence of predictive data. Thus, PePS2 studies efficacy and toxicity in 6 distinct cohorts: TN Low, TN Medium, TN High, PT Low, PT Medium, PT High. The design admits all-comers and does not target specific sample sizes in the individual cohorts. Hyperprior parameters have defaults to match those used in PePS2, but all may be overridden. The returned object includes randomly-sampled outcomes, as well as parameters to run the model. These are all combined in the same list object for passing to RStan, as is the convention. See the accompanying vignette for a full description.

Usage

```
peps2_get_data(num_patients, cohort_probs = NULL, prob_eff, prob_tox,
  eff_tox_or, cohort_rho = c(15.7, 21.8, 12.4, 20.7, 18, 11.4),
  alpha_mean = -2.2, alpha_sd = 2, beta_mean = -0.5, beta_sd = 2,
  gamma_mean = -0.5, gamma_sd = 2, zeta_mean = -0.5, zeta_sd = 2,
  lambda_mean = -2.2, lambda_sd = 2, psi_mean = 0, psi_sd = 1)
```

Arguments

<code>num_patients</code>	Total number of patients to use, positive integer.
<code>cohort_probs</code>	Probabilities that a patient belongs to each of the 6 cohorts, in the order given above; a vector of numbers between 0 and 1 that add up to 1. <code>cohort_probs</code> or <code>cohort_rho</code> must be specified.
<code>prob_eff</code>	Probabilities of efficacy in each of the 6 cohorts, in the order given above; a vector of numbers between 0 and 1

prob_tox	Probabilities of toxicity in each of the 6 cohorts, in the order given above; a vector of numbers between 0 and 1
eff_tox_or	Measure of strength of association between efficacy and toxicity, in each of the 6 cohorts, in the order given above; a vector of numbers. Use 1 for no association; numbers increasingly greater than 1 for stronger positive associations, and numbers less than 1 for stronger negative associations
cohort_rho	Concentration parameters for cohort membership, in the order given above, using a Dirichlet distribution. This leads to randomly- sampled cohort sizes distributed $\text{Dir}(\text{cohort_rho})$. <code>cohort_probs</code> or <code>cohort_rho</code> must be specified.
alpha_mean	The prior mean of alpha. Alpha is the efficacy model intercept.
alpha_sd	The prior standard deviation of alpha. Alpha is the efficacy model intercept.
beta_mean	The prior mean of beta. Beta is the efficacy model term for being previously treated.
beta_sd	The prior standard deviation of beta. Beta is the efficacy model term for being previously treated.
gamma_mean	The prior mean of gamma. Gamma is the efficacy model term for being PD-L1 score = Low.
gamma_sd	The prior standard deviation of gamma. Gamma is the efficacy model term for being PD-L1 score = Low.
zeta_mean	The prior mean of zeta. Zeta is the efficacy model term for being PD-L1 score = Medium.
zeta_sd	The prior standard deviation of zeta. Zeta is the efficacy model term for being PD-L1 score = Medium.
lambda_mean	The prior mean of lambda. Lambda is the toxicity model intercept.
lambda_sd	The prior standard deviation of lambda. Lambda is the toxicity model intercept.
psi_mean	The prior mean of psi. Psi is the joint model association parameter.
psi_sd	The prior standard deviation of psi. Psi is the joint model association parameter.

Value

a list of parameters, described in `peps2_params`

See Also

[peps2_params](#)

Examples

```
set.seed(123)
dat <- peps2_get_data(num_patients = 60,
  prob_eff = c(0.167, 0.192, 0.5, 0.091, 0.156, 0.439),
  prob_tox = rep(0.1, 6),
  eff_tox_or = rep(1, 6))
samp = rstan::sampling(stanmodels$BebopInPeps2, data = dat)
```

peps2_params

*Parameters to be passed to BebopInPeps2 model in Stan***Description**

The parameters required by the BebopInPeps2 Stan model are bundled up in a list, for convenience. This list contains elements that we might call "data" (e.g. trial outcomes), and "parameters" (e.g. modelling choices).

Details

The list object has the following elements:

- J, Total number of patients used, positive integer.
- eff, a vector of logical values to represent the presence (=1) or absence (=0) of efficacy events in the patients.
- tox, a vector of logical values to represent the presence (=1) or absence (=0) of toxicity events in the patients.
- x1, a vector of logical values to represent whether the patients are pre-treated (=1) or treatment-naive (=0).
- x2, a vector of logical values to represent whether the patients are Low PD-L1 (=1) or not (=0).
- x3, a vector of logical values to represent whether the patients are Medium PD-L1 (=1) or not (=0). Logically, if x2 and x3 are both zero, then the patient is High PD-L1.
- cohort_n, vector of cohort sizes, provided for convenience.
- cohort_eff, vector of counts of efficacy events by cohort, provided for convenience.
- cohort_tox, vector of counts of toxicity events by cohort, provided for convenience.
- alpha_mean, The prior mean of alpha. Alpha is the efficacy model intercept.
- alpha_sd, The prior standard deviation of alpha. Alpha is the efficacy model intercept.
- beta_mean, The prior mean of beta. Beta is the efficacy model term for being previously treated.
- beta_sd, The prior standard deviation of beta. Beta is the efficacy model term for being previously treated.
- gamma_mean, The prior mean of gamma. Gamma is the efficacy model term for being PD-L1 score = Low.
- gamma_sd, The prior standard deviation of gamma. Gamma is the efficacy model term for being PD-L1 score = Low.
- zeta_mean, The prior mean of zeta. Zeta is the efficacy model term for being PD-L1 score = Medium.
- zeta_sd, The prior standard deviation of zeta. Zeta is the efficacy model term for being PD-L1 score = Medium.
- lambda_mean, The prior mean of lambda. Lambda is the toxicity model intercept.

- `lambda_sd`, The prior standard deviation of `lambda`. `lambda` is the toxicity model intercept.
- `psi_mean`, The prior mean of `psi`. `psi` is the joint model association parameter.
- `psi_sd`, The prior standard deviation of `psi`. `psi` is the joint model association parameter.

An example of the `peps2_params` list is returned by [peps2_get_data](#).

Value

A list with elements identified above.

See Also

[peps2_get_data](#)

`peps2_process`

Process RStan samples from a BEBOP model fit to PePS2 data

Description

Process RStan samples from a BEBOP model fit to PePS2 data. This step lets us make inferences about whether the modelled efficacy and toxicity probabilities suggest the treatment is acceptable in each of the cohorts under study. The parameters have default values to match those used in the PePS2 trial. See the accompanying vignette for a full description.

Usage

```
peps2_process(dat, fit, min_eff = 0.1, max_tox = 0.3, eff_cert = 0.7,
             tox_cert = 0.9)
```

Arguments

<code>dat</code>	An instance of peps2_params , a list of PePS2 data and parameters. An example is yielded by peps2_get_data .
<code>fit</code>	An instance of <code>rstan::stanmodel</code> , derived by fitting data to the BEBOP in PePS2 model. Use <code>stan::sampling(stanmodels\$BebopInPeps2, data = dat)</code> .
<code>min_eff</code>	The lower efficacy probability threshold; a number between 0 and 1.
<code>max_tox</code>	The upper toxicity probability threshold; a number between 0 and 1.
<code>eff_cert</code>	Certainty required to infer the treatment is acceptable with regards to being probably efficacious; a number between 0 and 1.
<code>tox_cert</code>	Certainty required to infer the treatment is acceptable with regards to being probably tolerable; a number between 0 and 1.

Value

a list with the following items:

- ProbEff, the posterior mean probability of efficacy in the 6 cohorts.
- ProbAccEff, the posterior mean probability that the probability of efficacy exceeds `min_eff`, in the 6 cohorts.
- ProbTox, the posterior mean probability of toxicity in the 6 cohorts.
- ProbAccTox, the posterior mean probability that the probability of toxicity is less than `max_tox`, in the 6 cohorts.
- Accept, a vector of logical values to show whether treatment should be accepted in the 6 cohorts. Treatment is acceptable when it is probably efficacious and probably not toxic, with respect to the described rules.
- alpha, the posterior mean estimate of alpha.
- beta, the posterior mean estimate of beta.
- gamma, the posterior mean estimate of gamma.
- zeta, the posterior mean estimate of zeta.
- lambda, the posterior mean estimate of lambda.
- psi, the posterior mean estimate of psi.

See Also

[peps2_params](#)

[peps2_get_data](#)

Examples

```
set.seed(123)
dat <- peps2_get_data(num_patients = 60,
                     prob_eff = c(0.167, 0.192, 0.5,
                                   0.091, 0.156, 0.439),
                     prob_tox = rep(0.1, 6),
                     eff_tox_or = rep(1, 6))
samp = rstan::sampling(stanmodels$BebopInPeps2, data = dat)
decision <- peps2_process(dat, samp)
decision$Accept # Accept in cohort 2, 3, 5, 6 but not 1, 4
decision$ProbEff # The probability of efficacy is driving that decision
```

peps2_run_sims	<i>Run simulations of BEBOP in PePS2</i>
----------------	--

Description

Run simulations of the BEBOP model in a PePS2 scenario. The logic of creating the data to be passed to the Stan model is delegated to a function, thus the user may tailor rates of efficacy and toxicity, priors, cohort membership probabilities, by specialising this function. Similarly, the post-processing to perform on the RStan sample is delegated to another function. This pattern gives a flexible interface.

Usage

```
peps2_run_sims(num_sims, sample_data_func, summarise_func, ...)
```

Arguments

<code>num_sims</code>	integer, number of simulated iterations
<code>sample_data_func</code>	delegate function that takes no parameters and returns a list like <code>peps2_params</code> . This function is called during each simulated iteration so it should contain random outcomes.
<code>summarise_func</code>	delegate function that takes args <code>dat</code> and <code>fit</code> , where <code>dat</code> is an instance of <code>peps2_params</code> returned by <code>sample_data_func()</code> , and <code>fit</code> is an instance of <code>rstan::stanmodel</code> , derived by sampling the <code>BebopInPeps2</code> model. The simulation routine collects this using <code>stan::sampling(stanmodels\$BebopInPeps2, data = dat)</code> . The objects returned by this delegate are ultimately returned to the user. Thus, this function should perform the post-processing to get the RStan sample into some useful format. An example if peps2_process .
<code>...</code>	extra parameters passed to <code>rstan::sampling</code> .

Value

a list of length `num_sims`, with element `i` being the result of the `i`th call to `summarise_func(dat, fit)`.

See Also

[peps2_params](#)
[peps2_get_data](#)
[peps2_process](#)

Examples

```

prob_eff <- c(0.167, 0.192, 0.5, 0.091, 0.156, 0.439)
prob_tox = c(0.1, 0.1, 0.1, 0.1, 0.1, 0.1)
peps2_scenario_data <- function() peps2_get_data(num_patients = 60,
                                                prob_eff = prob_eff,
                                                prob_tox = prob_tox,
                                                eff_tox_or = rep(1, 6))

set.seed(123)
## Not run:
sims <- peps2_run_sims(num_sims = 10, sample_data_func = peps2_scenario_data,
                     summarise_func = peps2_process)
apply(sapply(sims, function(x) x$Accept), 1, mean) # 0.5 0.7 1.0 0.2 0.5 1.0

## End(Not run)
# Simulation suggests we are likely to approve in cohorts 3 and 6.
# In real life, we would run thousands of iterations, not 10.
# This is an example.

```

plot.crm_fit

Plot an crm_fit

Description

Plot an crm_fit

Usage

```

## S3 method for class 'crm_fit'
plot(x, pars = "prob_tox", ...)

```

Arguments

x [crm_fit](#) object to plot.

pars Parameters to plot. Plots utility scores by default.

... Extra parameters, passed onwards.

Value

A plot

plot.efftox_fit	<i>Plot an efftox_fit</i>
-----------------	---------------------------

Description

Plot an efftox_fit

Usage

```
## S3 method for class 'efftox_fit'  
plot(x, pars = "utility", ...)
```

Arguments

x	efftox_fit object to plot.
pars	Parameters to plot. Plots utility scores by default.
...	Extra parameters, passed onwards.

Value

A plot

print.crm_fit	<i>Print crm_fit object.</i>
---------------	------------------------------

Description

Print crm_fit object.

Usage

```
## S3 method for class 'crm_fit'  
print(x, ...)
```

Arguments

x	crm_fit object to convert.
...	Extra parameters, passed onwards.

```
print.efftox_fit      Print efftox_fit object.
```

Description

Print efftox_fit object.

Usage

```
## S3 method for class 'efftox_fit'
print(x, ...)
```

Arguments

`x` `efftox_fit` object to convert.
`...` Extra parameters, passed onwards.

```
ranBin2              Sample pairs of correlated binary events
```

Description

This function is reproduced from the `binarySimCLF` package on CRAN. The original package appears no longer to be maintained. View the original source at: <https://github.com/cran/binarySimCLF/blob/master/R/ranBin2>.

Usage

```
ranBin2(nRep, u, psi)
```

Arguments

`nRep` Number of simulated event pairs, positive integer.
`u` Mean event probabilities, expressed as a vector of length 2. E.g. to simulate associated bivariate events with probabilities $u = c(0.8, 0.3)$.
`psi` Odds ratio, number. This parameter controls the strength of association. Use `psi = 1` for no association. Values greater than 1 correspond to increasingly positive association between the two events, and vice-versa.

Value

Matrix of events represented as 0s and 1s, with `nRep` rows and 2 columns. The first column is the incidence of event 1.

Examples

```
probs <- c(0.8, 0.3)
s <- ranBin2(1000, probs, psi=0.2) # 1000 pairs of outcomes
cor(s) # Negatively correlated because psi < 1
colMeans(s) # Event rates as expected
```

Rcpp_model_BebopInPeps2-class

Compiled Stan model for BEBOP implementation in PePS2 clinical trial

Description

Compiled Stan model for BEBOP implementation in PePS2 clinical trial.

See Also

[model_BebopInPeps2](#)

Rcpp_model_EffTox-class

Compiled Stan model for EffTox dose-finding design

Description

Compiled Stan model for EffTox dose-finding design

See Also

[model_EffTox](#)

Rcpp_model_ThallHierarchicalBinary-class

Compiled Stan model for Thall et al.'s hierarchical Bayesian model for binary data

Description

Compiled Stan model for Thall et al.'s hierarchical Bayesian model for binary data

stanmodels	<i>Stan models used by trialr</i>
------------	-----------------------------------

Description

A list of the Stan models installed by trialr.

stan_crm	<i>Fit a CRM model</i>
----------	------------------------

Description

Fit a CRM model using Stan for full Bayesian inference.

Usage

```
stan_crm(outcome_str = NULL, skeleton, target, model = c("empiric",
  "logistic", "logistic_gamma", "logistic2"), a0 = NULL, alpha_mean = NULL,
  alpha_sd = NULL, beta_mean = NULL, beta_sd = NULL, beta_shape = NULL,
  beta_inverse_scale = NULL, doses_given = NULL, tox = NULL, ...)
```

Arguments

outcome_str	A string representing the outcomes observed hitherto. See df_parse_outcomes for a description of syntax and examples. Alternatively, you may provide doses_given and tox parameters. See Details.
skeleton	a vector of the prior guesses of toxicity at doses. This should be a monotonically-increasing vector of numbers between 0 and 1.
target	the target toxicity probability, a number between 0 and 1. This value would normally be one of the values in skeleton, but that is not a requirement.
model	Character string to denote desired model. One of empiric, logistic, logistic_gamma, or logistic2. The choice of model determines which parameters are required. See Details.
a0	Value of fixed intercept parameter. Only required for certain models. See Details.
alpha_mean	Prior mean of intercept variable for normal prior. Only required for certain models. See Details.
alpha_sd	Prior standard deviation of intercept variable for normal prior. Only required for certain models. See Details.
beta_mean	Prior mean of gradient variable for normal prior. Only required for certain models. See Details.
beta_sd	Prior standard deviation of slope variable for normal prior. Only required for certain models. See Details.

beta_shape	Prior shape parameter of slope variable for gamma prior. Only required for certain models. See Details.
beta_inverse_scale	Prior inverse scale parameter of slope variable for gamma prior. Only required for certain models. See Details.
doses_given	A optional vector of dose-levels given to patients 1:num_patients, where 1=lowest dose, 2=second dose, etc. Only required when outcome_str is not provided.
tox	An optional vector of toxicity outcomes for patients 1:num_patients, where 1=toxicity and 0=no toxicity. Only required when outcome_str is not provided.
...	Extra parameters are passed to rstan::sampling. Commonly used options are iter, chains, warmup, cores, control. sampling .

Details

The quickest and easiest way to fit a CRM model to some observed outcomes is to describe the outcomes using **trialr**'s syntax for dose-finding outcomes. See [df_parse_outcomes](#) for full details and examples.

Different model parameterisations require that difference parameter values are specified.

Value

An object of class [crm_fit](#)

Requirements of empiric model

* beta_sd

Requirements of logistic model

* a0 * beta_mean * beta_sd

Requirements of logistic_gamma model

* a0 * beta_shape * beta_inverse_scale

Requirements of logistics model

* a0 * alpha_mean * alpha_sd * beta_mean * beta_sd

Author(s)

Kristian Brock <kristian.brock@gmail.com>

References

O'Quigley, J., Pepe, M., & Fisher, L. (1990). Continual reassessment method: a practical design for phase 1 clinical trials in cancer. *Biometrics*, 46(1), 33-48. <https://doi.org/10.2307/2531628>

See Also

[crm_fit](#) [crm_process](#)

Examples

```
## Not run:
# This model is presented in Thall et al. (2014)
mod1 <- stan_crm('1N 2N 3T', skeleton = c(0.1, 0.2, 0.35, 0.6),
               target = 0.2, model = 'empiric', beta_sd = sqrt(1.34),
               seed = 123)

# Shorthand for the above is:
mod2 <- stan_efftox_demo('1N 2E 3B', seed = 123)

# the seed is passed to the Stan sampler. The usual Stan sampler params like
# cores, iter, chains etc are passed on too via the ellipsis operator.

## End(Not run)
```

stan_efftox

Fit an EffTox model

Description

Fit an EffTox model using Stan for full Bayesian inference.

Usage

```
stan_efftox(outcome_str = NULL, real_doses, efficacy_hurdle, toxicity_hurdle,
            p_e, p_t, eff0, tox1, eff_star, tox_star, alpha_mean, alpha_sd, beta_mean,
            beta_sd, gamma_mean, gamma_sd, zeta_mean, zeta_sd, eta_mean, eta_sd, psi_mean,
            psi_sd, doses_given = NULL, eff = NULL, tox = NULL, ...)
```

Arguments

outcome_str	A string representing the outcomes observed hitherto. See efftox_parse_outcomes for a description of syntax and examples. Alternatively, you may provide doses_given, eff and tox parameters. See Details.
real_doses	A vector of numbers. The doses under investigation. They should be ordered from lowest to highest and be in consistent units. E.g., # to conduct a dose-finding trial of doses 10mg, 20mg and 50mg, use c(10, 20, 50).
efficacy_hurdle	Minimum acceptable efficacy probability. A number between 0 and 1.
toxicity_hurdle	Maximum acceptable toxicity probability. A number between 0 and 1.
p_e	Certainty required to infer a dose is acceptable with regards to being probably efficacious; a number between 0 and 1.

p_t	Certainty required to infer a dose is acceptable with regards to being probably tolerable; a number between 0 and 1.
eff0	Efficacy probability required when toxicity is impossible; a number between 0 and 1 (see Details).
tox1	Toxicity probability permitted when efficacy is guaranteed; a number between 0 and 1 (see Details).
eff_star	Efficacy probability of an equi-utility third point (see Details).
tox_star	Toxicity probability of an equi-utility third point (see Details).
alpha_mean	The prior normal mean of the intercept term in the toxicity logit model. A number.
alpha_sd	The prior normal standard deviation of the intercept term in the toxicity logit model. A number.
beta_mean	The prior normal mean of the slope term in the toxicity logit model. A number.
beta_sd	The prior normal standard deviation of the slope term in the toxicity logit model. A number.
gamma_mean	The prior normal mean of the intercept term in the efficacy logit model. A number.
gamma_sd	The prior normal standard deviation of the intercept term in the efficacy logit model. A number.
zeta_mean	The prior normal mean of the slope term in the efficacy logit model. A number.
zeta_sd	The prior normal standard deviation of the slope term in the efficacy logit model. A number.
eta_mean	The prior normal mean of the squared term coefficient in the efficacy logit model. A number.
eta_sd	The prior normal standard deviation of the squared term coefficient in the efficacy logit model. A number.
psi_mean	The prior normal mean of the association term in the combined efficacy-toxicity model. A number.
psi_sd	The prior normal standard deviation of the association term in the combined efficacy-toxicity model. A number.
doses_given	A optional vector of dose-levels given to patients 1:num_patients, where 1=lowest dose, 2=second dose, etc. Only required when outcome_str is not provided.
eff	An optional vector of efficacy outcomes for patients 1:num_patients, where 1=efficacy and 0=no efficacy. Only required when outcome_str is not provided.
tox	An optional vector of toxicity outcomes for patients 1:num_patients, where 1=toxicity and 0=no toxicity. Only required when outcome_str is not provided.
...	Extra parameters are passed to <code>rstan::sampling</code> . Commonly used options are <code>iter</code> , <code>chains</code> , <code>warmup</code> , <code>cores</code> , <code>control</code> . sampling .

Details

The quickest and easiest way to fit an EffTox model to some observed outcomes is to describe the outcomes using **trialr**'s syntax for efficacy-toxicity dose-finding outcomes. See [efftox_parse_outcomes](#) for full details and examples.

Utility or attractiveness scores are calculated in EffTox using L^p norms. Imagine the first quadrant of a scatter plot with `prob_eff` along the x-axis and `prob_tox` along the y-axis. The point (1, 0) (i.e. guaranteed efficacy & no toxicity) is the holy grail. The neutral contour intersects the points (`eff0`, 0), (1, `tox1`) and (`eff_star`, `tox_star`). A unique curve intersects these three points and identifies a value for `p`, the exponent in the L^p norm. On this neutral- utility contour, scores are equal to zero. A family of curves with different utility scores is defined that are "parallel" to this neutral curve. Points with probabilities of efficacy and toxicity that are nearer to (1, 0) will yield greater scores, and vice-versa.

Value

An object of class `efftox_fit`

Author(s)

Kristian Brock <kristian.brock@gmail.com>

References

Thall, P., & Cook, J. (2004). Dose-Finding Based on Efficacy-Toxicity Trade-Offs. *Biometrics*, 60(3), 684-693.

Thall, P., Herrick, R., Nguyen, H., Venier, J., & Norris, J. (2014). Effective sample size for computing prior hyperparameters in Bayesian phase I-II dose-finding. *Clinical Trials*, 11(6), 657-666. <https://doi.org/10.1177/1740774514547397>

Brock, K., Billingham, L., Copland, M., Siddique, S., Sirovica, M., & Yap, C. (2017). Implementing the EffTox dose-finding design in the Matchpoint trial. *BMC Medical Research Methodology*, 17(1), 112. <https://doi.org/10.1186/s12874-017-0381-x>

See Also

[efftox_fit](#) [stan_efftox_demo](#) [efftox_process](#)

Examples

```
## Not run:
# This model is presented in Thall et al. (2014)
mod1 <- stan_efftox('1N 2E 3B',
  real_doses = c(1.0, 2.0, 4.0, 6.6, 10.0),
  efficacy_hurdle = 0.5, toxicity_hurdle = 0.3,
  p_e = 0.1, p_t = 0.1,
  eff0 = 0.5, tox1 = 0.65,
  eff_star = 0.7, tox_star = 0.25,
  alpha_mean = -7.9593, alpha_sd = 3.5487,
  beta_mean = 1.5482, beta_sd = 3.5018,
  gamma_mean = 0.7367, gamma_sd = 2.5423,
```

```
zeta_mean = 3.4181, zeta_sd = 2.4406,  
eta_mean = 0, eta_sd = 0.2,  
psi_mean = 0, psi_sd = 1, seed = 123)  
  
# Shorthand for the above is:  
mod2 <- stan_efftox_demo('1N 2E 3B', seed = 123)  
  
# the seed is passed to the Stan sampler. The usual Stan sampler params like  
# cores, iter, chains etc are passed on too via the ellipsis operator.  
  
## End(Not run)
```

stan_efftox_demo *Fit the EffTox model presented in Thall et al. (2014)*

Description

Fit the EffTox model presented in Thall et al. (2014) using Stan for full Bayesian inference.

Usage

```
stan_efftox_demo(outcome_str, ...)
```

Arguments

outcome_str	A string representing the outcomes observed hitherto. See efftox_parse_outcomes for a description of syntax and examples. Alternatively, you may provide doses_given, eff and tox parameters. See Details.
...	Extra parameters are passed to <code>rstan::sampling</code> . Commonly used options are iter, chains, warmup, cores, control. sampling .

Value

An object of class [efftox_fit](#)

Author(s)

Kristian Brock <kristian.brock@gmail.com>

References

- Thall, P., & Cook, J. (2004). Dose-Finding Based on Efficacy-Toxicity Trade-Offs. *Biometrics*, 60(3), 684-693.
- Thall, P., Herrick, R., Nguyen, H., Venier, J., & Norris, J. (2014). Effective sample size for computing prior hyperparameters in Bayesian phase I-II dose-finding. *Clinical Trials*, 11(6), 657-666. <https://doi.org/10.1177/1740774514547397>
- Brock, K., Billingham, L., Copland, M., Siddique, S., Sirovica, M., & Yap, C. (2017). Implementing the EffTox dose-finding design in the Matchpoint trial. *BMC Medical Research Methodology*, 17(1), 112. <https://doi.org/10.1186/s12874-017-0381-x>

See Also

[efftox_fit](#) [stan_efftox](#) [efftox_process](#)

Examples

```
## Not run:
# This model is presented in Thall et al. (2014)
mod2 <- stan_efftox_demo('1N 2E 3B', seed = 123)

# The seed is passed to the Stan sampler. The usual Stan sampler params like
# cores, iter, chains etc are passed on too via the ellipsis operator.

## End(Not run)
```

stan_files	<i>Stan file locations</i>
------------	----------------------------

Description

A character vector of the locations of Stan models installed by trialr.

summary.crm_fit	<i>Obtain summary of an crm_fit</i>
-----------------	-------------------------------------

Description

Obtain summary of an crm_fit

Usage

```
## S3 method for class 'crm_fit'
summary(object, ...)
```

Arguments

object	crm_fit object to summarise.
...	Extra parameters, passed onwards.

Value

A summary object.

summary.efftox_fit *Obtain summary of an efftox_fit*

Description

Obtain summary of an efftox_fit

Usage

```
## S3 method for class 'efftox_fit'  
summary(object, ...)
```

Arguments

object [efftox_fit](#) object to summarise.
... Extra parameters, passed onwards.

Value

A summary object.

thallhierarchicalbinary_parameters_demo
Get parameters to run the demo of Thall Hierarchical Binary model

Description

Get parameters to run the demo of Thall Hierarchical Binary model. These match those used to demonstrate EffTox in Thall et al. 2003.

Usage

```
thallhierarchicalbinary_parameters_demo()
```

Value

a list of parameters

References

Thall et al. 2014, Effective sample size for computing prior hyperparameters in Bayesian phase I-II dose-finding

See Also

[efftox_params](#)

Examples

```
dat <- thallhierarchicalbinary_parameters_demo()
names(dat)
dat$target_resp == 0.3
```

thallhierarchicalbinary_params

Parameters to be passed to the `ThallHierarchicalBinary` model in Stan

Description

The parameters required by the `trialr` implementation of Thall et al's hierarchical Bayesian model for binary outcomes are bundled up in a list, for convenience. The model analyses the binary response rates to a common drug of multiple subtypes of a single disease. See the included vignette for a full description and demonstration.

Details

The list object has the following elements:

- `m`, the number of disease sub-types under investigation, an integer.
- `x`, the counts of responses in the `m` cohorts, a vector of non-negative integers.
- `n`, the counts of patients in the `m` cohorts, a vector of non-negative integers.
- `target_resp` the threshold response rate sought in the cohorts, a number between 0 and 1.
- `mu_mean`, mean of the normal prior distribution on the treatment effects.
- `mu_sd`, standard deviation of the normal prior distribution on the treatment effects.
- `tau_alpha`, alpha parameter of the gamma prior distribution on the precision, τ .
- `tau_beta`, beta parameter of the gamma prior distribution on the precision, τ .

An example of the `thallhierarchicalbinary_params` list is returned by [thallhierarchicalbinary_parameters_demo](#).

Value

A list with elements identified above.

See Also

[thallhierarchicalbinary_parameters_demo](#)

Index

- *Topic **Bayesian**
 - trialr-package, 3
- *Topic **clinical trial**
 - trialr-package, 3
- *Topic **package**
 - trialr-package, 3
- *Topic **rstan**
 - trialr-package, 3
- *Topic **stan**
 - trialr-package, 3

- as.data.frame.crm_fit, 7
- as.data.frame.efftox_fit, 7

- BebopInPeps2 (model_BebopInPeps2), 28

- crm_fit, 7, 10, 27, 36, 37, 41, 42, 46
- crm_fit (crm_fit-class), 8
- crm_fit-class, 8
- crm_params, 8, 10
- crm_params (crm_params-class), 9
- crm_params-class, 9
- crm_process, 8, 10, 10, 42

- df_parse_outcomes, 11, 40, 41

- EffTox (model_EffTox), 28
- efftox_analysis, 12, 13
- efftox_analysis_to_df, 13
- efftox_contour_plot, 13
- efftox_dtps, 15
- efftox_fit, 7, 21, 27, 37, 38, 44–47
- efftox_fit (efftox_fit-class), 16
- efftox_fit-class, 16
- efftox_get_tox, 17
- efftox_parameters_demo, 14, 15, 18, 21, 22, 29
- efftox_params, 14, 15, 17, 18, 21, 22, 29, 47
- efftox_params (efftox_params-class), 19
- efftox_params-class, 19
- efftox_parse_outcomes, 20, 42, 44, 45

- efftox_process, 12, 13, 17, 20, 21, 29, 44, 46
- efftox_simulate, 22
- efftox_solve_p, 17, 23, 25
- efftox_superiority, 24
- efftox_utility, 25
- efftox_utility_density_plot, 26

- gather_samples.crm_fit, 27
- gather_samples.efftox_fit, 27

- model_BebopInPeps2, 6, 28, 39
- model_EffTox, 6, 28, 39
- model_ThallHierarchicalBinary, 6, 29

- peps2_get_data, 28, 30, 33–35
- peps2_params, 28, 31, 32, 33–35
- peps2_process, 28, 33, 35
- peps2_run_sims, 35
- plot.crm_fit, 36
- plot.efftox_fit, 37
- print.crm_fit, 37
- print.efftox_fit, 38

- ranBin2, 38
- Rcpp_model_BebopInPeps2
 - (Rcpp_model_BebopInPeps2-class), 39
- Rcpp_model_BebopInPeps2-class, 39
- Rcpp_model_EffTox
 - (Rcpp_model_EffTox-class), 39
- Rcpp_model_EffTox-class, 39
- Rcpp_model_ThallHierarchicalBinary
 - (Rcpp_model_ThallHierarchicalBinary-class), 39
- Rcpp_model_ThallHierarchicalBinary-class, 39

- sampling, 8, 17, 41, 43, 45
- stan_crm, 8, 10, 40
- stan_efftox, 17, 20, 42, 46
- stan_efftox_demo, 17, 20, 44, 45

stan_files, [46](#)
stanfit, [8](#), [12](#), [17](#)
stanmodels, [40](#)
summary.crm_fit, [46](#)
summary.efftox_fit, [47](#)

ThallHierarchicalBinary
 (model_ThallHierarchicalBinary),
 [29](#)

thallhierarchicalbinary_parameters_demo,
 [30](#), [47](#), [48](#)

thallhierarchicalbinary_params, [30](#), [48](#)

trialr (trialr-package), [3](#)
trialr-package, [3](#)