

Package ‘trialr’

October 16, 2020

Version 0.1.5

Date 2020-10-14

Title Clinical Trial Designs in 'rstan'

Description A collection of clinical trial designs and methods, implemented in 'rstan' and R, including: the Continual Reassessment Method by O'Quigley et al. (1990) <doi:10.2307/2531628>; EffTox by Thall & Cook (2004) <doi:10.1111/j.0006-341X.2004.00218.x>; the two-parameter logistic method of Neuenschwander, Branson & Sponer (2008) <doi:10.1002/sim.3230>; and the Augmented Binary method by Wason & Seaman (2013) <doi:10.1002/sim.5867>; and more. We provide functions to aid model-fitting and analysis.

The 'rstan' implementations may also serve as a cookbook to anyone looking to extend or embellish these models. We hope that this package encourages the use of Bayesian methods in clinical trials. There is a preponderance of early phase trial designs because this is where Bayesian methods are used most. If there is a method you would like implemented, please get in touch.

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

License GPL (>= 3)

Encoding UTF-8

LazyData true

ByteCompile true

Depends R (>= 3.5.0), methods, Rcpp (>= 1.0.1)

Imports rstan (>= 2.18.2), rstantools (>= 1.5.1), rlang (>= 0.4.5), dplyr, purrr, magrittr, stringr, ggplot2, gtools, coda, tidybayes (>= 2.0.3), tibble (>= 3.0.0), binom, MASS

LinkingTo StanHeaders (>= 2.18.1), rstan (>= 2.18.2), BH (>= 1.69.0-1), Rcpp (>= 1.0.1), RcppEigen (>= 0.3.3.5.0)

SystemRequirements GNU make

NeedsCompilation yes

RoxygenNote 7.1.1

VignetteBuilder knitr

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

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

Suggests testthat, knitr, rmarkdown, tidyr, ggridges, DiagrammeR

Author Kristian Brock [aut, cre] (<<https://orcid.org/0000-0002-3921-0166>>),
Trustees of Columbia University [cph]

Repository CRAN

Date/Publication 2020-10-15 23:00:02 UTC

R topics documented:

trialr-package	3
as.data.frame.crm_fit	4
as.data.frame.efftox_fit	4
as.mcmc.list.crm_fit	5
as.mcmc.list.efftox_fit	5
as_tibble.augbin_2t_1a_fit	6
as_tibble.dose_finding_paths	6
augbin_2t_1a_fit	7
augbin_fit	8
binary_prob_success	8
careful_escalation	10
closest_to_target	11
crm_codified_dose_logistic	11
crm_dtpts	12
crm_fit-class	15
crm_params-class	16
crm_path_analysis	18
crm_prior_beliefs	20
crm_process	22
df_parse_outcomes	23
dose_finding_fit-class	24
dose_finding_path_node-class	25
efftox_analysis_to_df	26
efftox_contour_plot	27
efftox_dtpts	28
efftox_dtpts_to_dataframe	31
efftox_fit-class	32
efftox_get_tox	34
efftox_parameters_demo	35
efftox_params-class	36
efftox_parse_outcomes	37
efftox_path_analysis	38
efftox_priors	39
efftox_process	41
efftox_simulate	41
efftox_solve_p	43
efftox_superiority	44

efftox_utility	44
efftox_utility_density_plot	45
eff_at_dose	46
get_efftox_priors	47
n_at_dose	48
parse_dose_finding_outcomes	49
parse_eff_tox_dose_finding_outcomes	50
peps2_get_data	51
peps2_process	53
plot.crm_fit	55
plot.efftox_fit	55
predict.augbin_2t_1a_fit	56
print.augbin_fit	57
print.crm_fit	57
print.efftox_fit	58
print.nbg_fit	58
prior_predictive_augbin_2t_1a	59
prob_success	60
prob_tox_exceeds	62
ranBin2	63
rlkcorr	63
spread_paths	64
stan_augbin	65
stan_augbin_demo	67
stan_crm	68
stan_efftox	71
stan_efftox_demo	75
stan_hierarchical_response_thall	76
stan_nbg	78
stan_peps2	80
summary.crm_fit	82
summary.efftox_fit	83
total_weight_at_dose	83
tox_at_dose	84
trialr_simulate	85
weights_at_dose	86

Index**88**

trialr-package

*The 'trialr' package.***Description**

trialr collects in one place Bayesian clinical trial designs and methods. Models are implemented in Stan and helper functions are provided in R.

References

Stan Development Team (2018). RStan: the R interface to Stan. R package version 2.18.2. <https://mc-stan.org/>

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

as.mcmc.list.crm_fit *Convert `crm_fit` to instance of `mcmc.list`*

Description

This function allows trialr to use tidybayes functions.

Usage

```
## S3 method for class 'crm_fit'  
as.mcmc.list(crm_fit, ...)
```

Arguments

`crm_fit` Object of class `crm_fit`
... Extra variables that are passed onwards.

Value

Object of class `mcmc.list`

as.mcmc.list.efftox_fit
Convert `efftox_fit` to instance of `mcmc.list`

Description

This function allows trialr to use tidybayes functions.

Usage

```
## S3 method for class 'efftox_fit'  
as.mcmc.list(efftox_fit, ...)
```

Arguments

`efftox_fit` Object of class `efftox_fit`
... Extra variables that are passed onwards.

Value

Object of class `mcmc.list`

```
as_tibble.augbin_2t_1a_fit
  Cast augbin_2t_1a_fit object to tibble.
```

Description

Cast `augbin_2t_1a_fit` object to [tibble](#).

Usage

```
## S3 method for class 'augbin_2t_1a_fit'
as_tibble(x, ...)
```

Arguments

`x` Object of class `augbin_2t_1a_fit`.
`...` Extra args passed onwards.

Value

Object of class [tibble](#)

```
as_tibble.dose_finding_paths
  Cast dose_finding_paths object to tibble.
```

Description

Cast `dose_finding_paths` object to [tibble](#).

Usage

```
## S3 method for class 'dose_finding_paths'
as_tibble(x, ...)
```

Arguments

`x` Object of class `dose_finding_paths`.
`...` Extra args passed onwards.

Value

Object of class [tibble](#)

augbin_2t_1a_fit	<i>Class used by trialr to fit Wason & Seaman's Augmented Binary method in single arm trials with two post-baseline tumour assessments.</i>
------------------	--

Description

Class used by **trialr** to fit Wason & Seaman's Augmented Binary method in single arm trials with two post-baseline tumour assessments.

Usage

```
augbin_2t_1a_fit(num_patients, tumour_size, non_shrinkage_failure, fit)
```

Arguments

num_patients	Integer, the number of patients analysed.
tumour_size	matrix-like object containing tumour size measures, with rows representing patients and columns representing chronological assessment points. Column one is baseline.
non_shrinkage_failure	matrix-like object containing logical indicators of non-shrinkage failure, with rows representing patients and columns representing chronological assessment points.
fit	An object of class stanfit , containing the posterior samples.

References

Wason JMS, Seaman SR. Using continuous data on tumour measurements to improve inference in phase II cancer studies. *Statistics in Medicine*. 2013;32(26):4639-4650. doi:10.1002/sim.5867

Eisenhauer EA, Therasse P, Bogaerts J, et al. New response evaluation criteria in solid tumours: Revised RECIST guideline (version 1.1). *European Journal of Cancer*. 2009;45(2):228-247. doi:10.1016/j.ejca.2008.10.026

See Also

[augbin_fit](#) [stan_augbin](#)

augbin_fit	<i>Class used by trialr to fit Wason & Seaman's Augmented Binary method.</i>
------------	---

Description

Class used by **trialr** to fit Wason & Seaman's Augmented Binary method.

Usage

```
augbin_fit(num_patients, tumour_size, non_shrinkage_failure, fit)
```

Arguments

num_patients	Integer, the number of patients analysed.
tumour_size	matrix-like object containing tumour size measures, with rows representing patients and columns representing chronological standardised assessment points. Column one is baseline.
non_shrinkage_failure	matrix-like object containing logical indicators of non-shrinkage failure, with rows representing patients and columns representing chronological standardised assessment points.
fit	An object of class stanfit , containing the posterior samples.

References

Wason JMS, Seaman SR. Using continuous data on tumour measurements to improve inference in phase II cancer studies. *Statistics in Medicine*. 2013;32(26):4639-4650. doi:10.1002/sim.5867

Eisenhauer EA, Therasse P, Bogaerts J, et al. New response evaluation criteria in solid tumours: Revised RECIST guideline (version 1.1). *European Journal of Cancer*. 2009;45(2):228-247. doi:10.1016/j.ejca.2008.10.026

See Also

[stan_augbin](#)

binary_prob_success	<i>Calculate the binary probability of success.</i>
---------------------	---

Description

Calculate the binary probability of success.

Calculate the binary probability of success from an `augbin_2t_1a_fit` object.

Usage

```
binary_prob_success(x, ...)  
  
## S3 method for class 'augbin_2t_1a_fit'  
binary_prob_success(  
  x,  
  y1_lower = -Inf,  
  y1_upper = Inf,  
  y2_lower = -Inf,  
  y2_upper = log(0.7),  
  conf.level = 0.95,  
  ...  
)
```

Arguments

x	an R object of class "augbin_fit"
...	arguments passed to other methods
y1_lower	numeric, minimum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 1 to baseline. Defaults to negative infinity.
y1_upper	numeric, maximum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 1 to baseline. Defaults to positive infinity.
y2_lower	numeric, minimum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 2 to baseline.
y2_upper	numeric, maximum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 2 to baseline. Defaults to log(0.7).
conf.level	confidence level for interval.

Value

a data.frame-like object

Examples

```
## Not run:  
fit <- stan_augbin_demo()  
binary_prob_success(fit, y2_upper = log(0.7))  
  
## End(Not run)
```

careful_escalation *Dose selection function that practices careful escalation.*

Description

Dose selection function that avoids dose-skipping in escalation and advocates stopping when there is sufficient evidence that the risk of toxicity at a reference dose exceeds some threshold.

Usage

```
careful_escalation(  
  dose_finding_fit,  
  tox_threshold,  
  certainty_threshold,  
  reference_dose = 1,  
  start_dose = 1  
)
```

Arguments

`dose_finding_fit` Instance of `dose_finding_fit`.

`tox_threshold` numeric, the toxicity threshold.

`certainty_threshold` numeric, the required confidence that the risk of toxicity exceeds ‘`tox_threshold`’ to advocate stopping.

`reference_dose` the integer index of the reference dose. 1 by default, i.e. the lowest dose-level.

`start_dose` the integer index of the desired starting dose. 1 by default. This is required for the function to give the desired answer when no patients have yet been treated.

Value

an integer dose-level

Examples

```
## Not run:  
# CRM example  
fit <- stan_crm('1N 2N 3T', skeleton = c(0.1, 0.2, 0.35, 0.6),  
              target = 0.2, model = 'empiric', beta_sd = 1,  
              seed = 123)  
  
## End(Not run)
```

closest_to_target *Get index of element in vector with value closest to a target*

Description

Get index of element in vector with value closest to a target

Usage

```
closest_to_target(vector, target)
```

Arguments

vector	Identify element in this numeric vector
target	numeric target

Value

an integer indexing vector

Examples

```
closest_to_target(c(0.1, 0.2, 0.3), 0.05) # 1
closest_to_target(c(0.1, 0.2, 0.3), 0.22) # 2
closest_to_target(c(0.1, 0.2, 0.3), -0.05) # 1
closest_to_target(c(0.1, 0.2, 0.3), 8) # 3
```

crm_codified_dose_logistic
Calculate codified CRM doses.

Description

Calculate the codified CRM doses that map to probability of toxicity prob_tox in a logistic model with expected values for intercept and gradient. I.e. find $x[i]$ such that $\text{logit}(p[i]) = \alpha + \beta x[i]$, where p is prob_tox.

Usage

```
crm_codified_dose_logistic(prob_tox, alpha_mean, beta_mean)
```

Arguments

prob_tox	Numeric vector, seek codified doses that yield these probabilities of toxicity.
alpha_mean	Numeric, expected value of intercept.
beta_mean	Numeric, expected value of gradient with respect to dose.

Value

Numeric vector of codified doses.

Examples

```
skeleton <- c(0.05, 0.1, 0.2, 0.5)
crm_codified_dose_logistic(skeleton, 1, 0)
crm_codified_dose_logistic(skeleton, 3, 0.5)
```

 crm_dtps

Calculate dose-transition pathways for a CRM study

Description

Calculate dose-transition pathways (DTPs, Yap et al, 2017) for a dose-finding trial using the continual reassessment method (CRM) design. DTPs are a glimpse into the future for an in-progress trial. They tell us what the model would advise for all feasible future outcomes. They can be used in the design stages to detect possible undesirable behaviour. They can be used during the trial to aid planning and understanding.

Usage

```
crm_dtps(
  skeleton,
  target,
  model,
  cohort_sizes,
  previous_outcomes = "",
  next_dose = NULL,
  user_dose_func = NULL,
  verbose = FALSE,
  i_am_patient = FALSE,
  ...
)
```

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.
model	Character string to denote desired model. One of <code>empiric</code> , <code>logistic</code> , <code>logistic_gamma</code> , or <code>logistic2</code> . The choice of model determines which extra parameters are required by ... See Details.
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 another cohort of three, use <code>cohort_sizes = c(2,3)</code> .

previous_outcomes	Outcomes observed hitherto in the syntax required by df_parse_outcomes .
next_dose	optional, integer (1-based) dose-level to be given to the next cohort. If omitted, the dose suggested by the model is used.
user_dose_func	optional delegate for deciding dose. A function that takes a crm_fit as the sole argument and returns the integer (1-based) dose-level to be given next, or NA to show that no dose should be chosen and the trial stopped. This function gives the user the opportunity to build in custom behaviour to tailor the dose selection decision in response to the insights garnered by the fit model, or recommend that a trial path be halted immediately. If omitted, the dose ordinarily chosen by the model is used. An example is given below.
verbose	logical, TRUE to get log messages.
i_am_patient	logical. The number of paths to analyse grows faster than linearly in the number of future cohorts to resolve. Fitting many models by MCMC can take a long time. This function will not proceed unless you signify your patience when the number of paths to resolve exceeds 100.
...	Extra parameters passed to stan_crm .

Details

Different model choices require that different parameters are provided. See below.

Value

A [list](#) of [dose_finding_path_node](#) objects.

Parameter requirements of `empiric` model

- `beta_sd`

Parameter requirements of `logistic` model

- `a0`
- `beta_mean`
- `beta_sd`

Parameter requirements of `logistic_gamma` model

- `a0`
- `beta_shape`
- `beta_inverse_scale`

Parameter requirements of `logistic2` model

- `alpha_mean`
- `alpha_sd`
- `beta_mean`
- `beta_sd`

Author(s)

Kristian Brock

References

Yap C, Billingham LJ, Cheung YK, Craddock C, O'Quigley J. Dose transition pathways: The missing link between complex dose-finding designs and simple decision-making. *Clinical Cancer Research*. 2017;23(24):7440-7447. doi:10.1158/1078-0432.CCR-17-0582

See Also

[df_parse_outcomes](#), [stan_crm](#), [crm_path_analysis](#), [dose_finding_path_node](#)

Examples

```
## Not run:
target <- 0.25
skeleton <- c(0.05, 0.15, 0.25, 0.4, 0.6)

# Run DTPs for the first two cohorts of two for new a trial:
paths <- crm_dtps(skeleton = skeleton, target = target, model = 'empiric',
                 cohort_sizes = c(2, 2), next_dose = 3, beta_sd = 1)
length(paths) # 13

library(tibble)
df <- as_tibble(paths)
df

# Run DTPs for the next cohort of three in a trial that has already treated
# six patients, seeing some toxicity at dose-level 3:
paths2 <- crm_dtps(skeleton = skeleton, target = target, model = 'empiric',
                  cohort_sizes = c(3), previous_outcomes = '2NNN 3TTN',
                  beta_sd = 1)
length(paths2) # 5
as_tibble(paths2)
# We see that de-escalation to dose-level 2 should occur now, and that any
# further toxicity will result in advice for further de-escalation to
# dose-level 1.

# An example with a custom dose selection function
paths3 <- crm_dtps(skeleton = skeleton, target = target, model = 'empiric',
                  cohort_sizes = c(3, 3), previous_outcomes = '2NN 3TN',
                  next_dose = 2, beta_sd = 1,
                  user_dose_func = function(x) {
                    careful_escalation(x, tox_threshold = target + 0.1,
                                       certainty_threshold = 0.7)
                  }, seed = 123, refresh = 0)
spread_paths(as_tibble(paths3) %>% select(-fit, -parent_fit, -dose_index))
# Stopping is recommended when the dose selection function returns NA.
```

```
## End(Not run)
```

```
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,
  num_patients,
  doses,
  tox,
  weights,
  prob_tox,
  median_prob_tox,
  prob_mtd,
  recommended_dose,
  dat,
  fit,
  samples = NULL
)
```

Arguments

dose_indices	A vector of integers representing the dose-levels under consideration.
num_patients	Integer, the number of patients analysed.
doses	vector of integers representing the dose given to the patients.
tox	vector of integers representing the toxicity status of the patients.
weights	Vector of numeric weights for the observations for patients 1:num_patients, thus facilitating the TITE-CRM design.
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.
prob_mtd	The posterior probability that each dose is the MTD, by the chosen model; a vector of numbers between 0 and 1. This probability reflects the uncertainty remaining in the parameter distributions, whereas prob_tox and median_prob_tox do not.
recommended_dose	An integer representing the dose-level that is recommended for the next patient or cohort. Contrast to modal_mtd_candidate.

dat	Object <code>crm_params</code> containing data passed to <code>sampling</code> .
fit	An object of class <code>stanfit</code> , containing the posterior samples.
samples	An optional <code>data.frame</code> like object of samples.

Details

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

See Also

[stan_crm](#)

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

Description

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

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 <code>skeleton</code> , 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.

Parameter requirements of empiric model

- beta_sd

Parameter requirements of logistic model

- a0
- beta_mean
- beta_sd

Parameter requirements of logistic_gamma model

- a0
- beta_shape
- beta_inverse_scale

Parameter requirements of logistic2 model

- alpha_mean
- alpha_sd
- beta_mean
- beta_sd

See Also

[stan_crm](#)

crm_path_analysis	<i>Fit a CRM model to the incrementally observed outcomes on a trial pathway.</i>
-------------------	---

Description

Fit a continuous reassessment method (CRM) model to the outcomes cumulatively observed at the end of each cohort in a trial pathway. E.g. if the trial pathway is 1NN 2NN 3NT, we have three cohorts of two patients. This function will fit the model to the following four states: before any patients have been evaluated; after 1NN; after 1NN 2NN; and finally after 1NN 2NN 3NT. This allows us to analyse how the trial model is evolving in its estimation as trial data is accumulated.

Usage

```
crm_path_analysis(outcome_str, skeleton, target, model, verbose = FALSE, ...)
```

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 extra parameters are required by See Details.
verbose	logical, TRUE to get log messages.
...	Extra parameters passed to stan_crm .

Details

Different model choices require that different parameters are provided. See below.

Value

A [list](#) of [dose_finding_path_node](#) objects.

Parameter requirements of empiric model

- beta_sd

Parameter requirements of logistic model

- a_0
- beta_mean
- beta_sd

Parameter requirements of logistic_gamma model

- a_0
- beta_shape
- beta_inverse_scale

Parameter requirements of logistic2 model

- alpha_mean
- alpha_sd
- beta_mean
- beta_sd

Author(s)

Kristian Brock

See Also

[df_parse_outcomes](#), [stan_crm](#), [dose_finding_path_node](#)

Examples

```
## Not run:
# CRM example
target <- 0.25
skeleton <- c(0.05, 0.15, 0.25, 0.4, 0.6)
paths <- crm_path_analysis(
  outcome_str = '1NNN 2NTN 2NNN',
  skeleton = skeleton, target = target, model = 'empiric',
  beta_sd = 1, seed = 123, refresh = 0)
length(paths) # 4
names(paths)[1] # ""
names(paths)[2] # "1NNN"
names(paths)[3] # "1NNN 2NTN"
names(paths)[4] # "1NNN 2NTN 2NNN"
# Each node is an analysis fit to the cumulative outcomes
# Converting to a tibble presents some nice tidyverse-related opportunities
library(tibble)
df <- as_tibble(paths)
df

## End(Not run)
```

crm_prior_beliefs *Get the prior beliefs for a CRM trial scenario.*

Description

Infer the prior beliefs consistent with the parameters and model form for a CRM dose-finding trial. This function could be interpreted as fitting the model to no data, thus examining the beliefs on dose-toxicity that are suggested by the parameter priors alone. This function provides the task analogous to [stan_crm](#) before any data has been collected.

Usage

```
crm_prior_beliefs(
  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,
  ...
)
```

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.
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.
...	extra parameters passed to stan_crm .

Details

Different model choices require that different parameters are provided. See below.

Value

An object of class [crm_fit](#)

Parameter requirements of empiric model

- beta_sd

Parameter requirements of logistic model

- a0
- beta_mean
- beta_sd

Parameter requirements of logistic_gamma model

- a0
- beta_shape
- beta_inverse_scale

Parameter requirements of logistic2 model

- alpha_mean
- alpha_sd
- beta_mean
- beta_sd

Author(s)

Kristian Brock

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://www.jstor.org/stable/2531628>

Cheung, Y.K. (2011). *Dose Finding by the Continual Reassessment Method*. CRC Press. ISBN 9781420091519

See Also

[stan_crm](#) [crm_fit](#)

Examples

```
skeleton <- c(0.05, 0.1, 0.15, 0.33, 0.5)
target <- 0.33

prior_fit1 <- crm_prior_beliefs(skeleton, target, model = 'empiric',
                               beta_sd = sqrt(1.34))
prior_fit2 <- crm_prior_beliefs(skeleton, target, model = 'logistic_gamma',
                               a0 = 3, beta_shape = 1,
                               beta_inverse_scale = 2)
```

crm_process

Process RStan samples from a CRM model.

Description

Internal function to 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](#).

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 (the 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 = df_parse_outcomes('1NNN 2NTN 3TTT')
x$num_patients # 9
x$doses        # c(1, 1, 1, 2, 2, 2, 3, 3, 3)
x$tox          # c(0, 0, 0, 0, 1, 0, 1, 1, 1)
sum(x$tox)     # 4
```

dose_finding_fit-class

*Class of dose-finding model fit by **trialr** using Stan.*

Description

Class of dose-finding model fit by **trialr** using Stan.

Usage

```
dose_finding_fit(
  dose_indices,
  num_patients,
  doses,
  tox,
  prob_tox,
  median_prob_tox,
  recommended_dose,
  dat,
  fit
)
```

Arguments

dose_indices	A vector of integers representing the dose-levels under consideration.
num_patients	Integer, the number of patients analysed.
doses	vector of integers representing the dose given to the patients.
tox	vector of integers representing the toxicity status of the patients.
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.
recommended_dose	An integer representing the dose-level that is recommended for the next patient or cohort.
dat	Object crm_params containing data passed to sampling .
fit	An object of class stanfit , containing the posterior samples.

See Also

[crm_fit](#), [efftox_fit](#)

 dose_finding_path_node-class

Class to hold the elements of a single dose-finding analysis residing in a pathway of analyses.

Description

A pathway in a dose-finding trial is a series of successive analyses. For instance, the model will likely be fit to all of the outcomes observed at the end of the first cohort, the second cohort, etc. This class holds the elements reflecting the analysis, and the place of this analysis in the pathway.

Usage

```
dose_finding_path_node(
  node_id,
  parent_node_id,
  depth,
  outcomes,
  next_dose,
  fit,
  parent_fit
)
```

Arguments

<code>node_id</code>	An integer representing the id of this node in a pathway.
<code>parent_node_id</code>	An integer representing the id of this node's parent in the pathway.
<code>depth</code>	An integer representing the depth of this node in the pathway, where the root has depth 0.
<code>outcomes</code>	A string representing the outcomes observed at the time of analysis. See df_parse_outcomes for a description of syntax and examples.
<code>next_dose</code>	An integer representing the dose recommended by the model for the next patient or cohort of patients.
<code>fit</code>	Object obtained from fitting the dose-finding model to outcomes.
<code>parent_fit</code>	Object obtained from fitting the dose-finding model to the outcomes of the parent node. Comparing to <code>fit</code> will often be valuable.

Value

Instance of class `dose_finding_path_node`

Examples

```
## Not run:
parent_outcomes <- '1NNN'
outcomes <- '1NNN 2NNT'
target <- 0.25
skeleton <- c(0.05, 0.15, 0.25, 0.4, 0.6)
parent_fit <- stan_crm(outcome_str = parent_outcomes, skeleton = skeleton,
                      target = target, model = 'empiric', beta_sd = 1)
fit <- stan_crm(outcome_str = outcomes, skeleton = skeleton,
               target = target, model = 'empiric', beta_sd = 1)
dose_finding_path_node(node_id = 2, parent_node_id = 1, depth = 1,
                       outcomes = outcomes, next_dose = fit$recommended_dose,
                       fit = fit, parent_fit = parent_fit)

## End(Not run)
```

efftox_analysis_to_df *EffTox analysis to data.frame*

Description

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

Usage

```
efftox_analysis_to_df(x)
```

Arguments

x An instance of [efftox_fit](#)

Value

a data.frame

See Also

[stan_efftox](#)

Examples

```
fit <- stan_efftox_demo(outcome_str = '1N 2E 3B')
df <- efftox_analysis_to_df(fit)
df
```

 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(
  fit,
  use_ggplot = FALSE,
  prob_eff = fit$prob_eff,
  prob_tox = fit$prob_tox,
  num_points = 1000,
  util_vals = seq(-3, 3, by = 0.2)
)
```

Arguments

<code>fit</code>	An instance of efftox_fit .
<code>use_ggplot</code>	logical, TRUE to use ggplot2. Defaults to FALSE to use standard R graphics.
<code>prob_eff</code>	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. Defaults to the values fitted by the model. Use NULL to suppress.
<code>prob_tox</code>	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. Defaults to the values fitted by the model. Use NULL to suppress.
<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

[stan_efftox](#)

Examples

```
fit <- stan_efftox_demo(outcome_str = '1N 2E 3B')
efftox_contour_plot(fit)
title('EffTox utility contours')
# The same with ggplot2
efftox_contour_plot(fit, 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. The function [efftox_dtps_to_dataframe](#) performs a similar function, but is much less-flexible.

Usage

```
efftox_dtps(
  cohort_sizes,
  previous_outcomes = "",
  next_dose = NULL,
  user_dose_func = NULL,
  verbose = FALSE,
  i_am_patient = FALSE,
  ...
)
```

Arguments

<code>cohort_sizes</code>	vector of future cohort sizes, i.e. positive integers. E.g. To calculate paths for the the next cohort of two followed by another cohort of three, use <code>cohort_sizes = c(2, 3)</code> .
<code>previous_outcomes</code>	Outcomes observed hitherto in the syntax required by efftox_parse_outcomes .
<code>next_dose</code>	the dose-level to be given to the immediately next cohort.
<code>user_dose_func</code>	optional delegate for deciding dose. A function that takes a efftox_fit as the sole argument and returns the integer (1-based) dose-level to be given next, or NA to show that no dose should be chosen and the trial stopped. This function gives the user the opportunity to build in custom behaviour to tailor the dose selection decision in response to the insights garnered by the fit model, or recommend that a trial path be halted immediately. If omitted, the dose ordinarily chosen by the model is used. An example is given below.
<code>verbose</code>	logical, TRUE to get progress messages.
<code>i_am_patient</code>	logical, TRUE to show your tolerance for waiting for over 100 models to fit. Set to FALSE by default.
<code>...</code>	extra params passed to <code>rstan::sampling</code> .

Value

dose pathways in a data.frame.

References

Yap C, Billingham LJ, Cheung YK, Craddock C, O'Quigley J. Dose transition pathways: The missing link between complex dose-finding designs and simple decision-making. *Clinical Cancer Research*. 2017;23(24):7440-7447. doi:10.1158/1078-0432.CCR-17-0582

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

See Also

[efftox_parse_outcomes](#), [stan_efftox](#), [efftox_path_analysis](#), [dose_finding_path_node](#)

Examples

```
## Not run:
# Calculate paths for the first cohort of 3 in Thall et al 2014 example
paths1 <- efftox_dtps(cohort_sizes = c(3), next_dose = 1,
  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)

# Calculate paths for the next two cohorts of 2, in an in-progress trial
# Warning: this create 100 paths. It will run for a minute or two.
paths2 <- efftox_dtps(cohort_sizes = c(2, 2),
  previous_outcomes = '1NN 2EE',
  next_dose = 1,
  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,
```

```

        i_am_patient = TRUE)

# Paths can be converted to a tibble
library(tibble)
library(dplyr)
df <- as_tibble(paths2)
df %>% print(n = 200)

# And shaped in a wide format
spread_paths(df %>% select(-fit, -parent_fit, -dose_index)) %>%
  print(n = 100)
# Incredibly, there are 100 ways these two cohorts of two can end up.

# An example with a custom dose selection function.
# Define a function to select the maximal utility dose, no matter what.
# Note: this diverges from the original authors' intentions; we provide this
# for illustration only!
max_utility_dose <- function(efftox_fit) {
  return(which.max(efftox_fit$utility))
}
# Fit the paths, providing the user_dose_func parameter
# Warning: this create 100 paths. It will run for a minute or two.
paths3 <- efftox_dtps(cohort_sizes = c(2, 2),
  previous_outcomes = '1NN 2EE',
  next_dose = 1,
  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,
  user_dose_func = max_utility_dose,
  seed = 123, i_am_patient = TRUE)

# We can see where the dose-selections differ at the second future cohort
# by joining these paths to those calculated in the previous example:
left_join(
  as_tibble(paths2)%>%
    select(.node, .parent, .depth, outcomes, model_dose = next_dose),
  as_tibble(paths3) %>%
    select(.node, user_dose = next_dose),
  by = '.node'
) %>% spread_paths() %>%
  filter(model_dose2 != user_dose2)
# They differ in many places. The user defined functions sometimes selects
# higher doses; sometimes lower.

```

```
## End(Not run)
```

```
efftox_dtgs_to_dataframe
```

```
Calculate dose-transition pathways for an EffTox study
```

Description

Calculate dose-transition pathways for an EffTox study. Note that TODO TODO TODO

Usage

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

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>cohort_sizes</code>	vector of future cohort sizes, i.e. positive integers. E.g. To calculate paths for the the next cohort of two followed by another cohort of three, use <code>cohort_sizes = c(2, 3)</code> .
<code>next_dose</code>	the dose-level to be given to the immediately next cohort.
<code>...</code>	extra params passed to <code>rstan::sampling</code> .

Value

dose pathways in a `data.frame`.

References

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

See Also

[efftox_dtgs](#), [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_to_dataframe(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_to_dataframe(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

Phase I/II dose-finding trials, i.e. those that search for a dose my efficacy and toxicity outcomes search for the optimal biological dose (OBD), rather than the maximum tolerated dose (MTD) that is typically sought be traditional toxicity-only dose-finding.

Usage

```

efftox_fit(
  dose_indices,
  num_patients,
  doses,
  tox,
  eff,
  prob_tox,
  prob_eff,
  median_prob_tox,
  median_prob_eff,
  prob_acc_tox,
  prob_acc_eff,
  utility,
  post_utility,
  prob_obd,
  acceptable,

```



```

    recommended_dose,
    dat,
    fit
  )

```

Arguments

dose_indices	A vector of integers representing the dose-levels under consideration.
num_patients	Integer, the number of patients analysed.
doses	vector of integers representing the dose given to the patients.
tox	vector of integers representing the toxicity status of the patients.
eff	vector of integers representing the efficacy status of the patients.
prob_tox	The posterior mean probabilities of toxicity at doses 1:n; a vector of numbers between 0 and 1.
prob_eff	The posterior mean probabilities of efficacy 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.
median_prob_eff	The posterior mean probabilities of efficacy at doses 1:n; 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.
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.
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 <code>post_utility</code> ; 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 <code>utility</code> , which uses plug-in posterior means of efficacy and toxicity, as advocated by Thall & Cook; a vector of numbers.
prob_obd	The posterior probability that each dose is the optimal biological dose (OBD); a vector of numbers between 0 and 1. This probability reflects the uncertainty remaining in the parameter distributions, whereas <code>prob_tox</code> and <code>prob_eff</code> (etc) do not.
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.
recommended_dose	An integer representing the dose-level recommended for the next patient or cohort; or NA if stopping is recommended.
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](#)

<code>efftox_get_tox</code>	<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

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

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).
priors	instance of class <code>efftox_priors</code> , the hyperparameters for normal priors on the six model parameters.

See Also

[efftox_priors](#) [get_efftox_priors](#) [stan_efftox](#) [stan_efftox_demo](#)

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_path_analysis *Fit an EffTox model to the incrementally observed outcomes on a trial pathway.*

Description

Fit a EffTox model to the outcomes cumulatively observed at the end of each cohort in a trial pathway. E.g. if the trial pathway is 1EN 2NN 3BT, we have three cohorts of two patients. This function will fit the model to the following four states: before any patients have been evaluated; after 1EN; after 1EN 2NN; and finally after 1EN 2NN 3BT. This allows us to analyse how the trial model is evolving in its estimation as trial data is accumulated.

Usage

```
efftox_path_analysis(outcome_str, verbose = FALSE, ...)
```

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 and tox parameters. See Details.
verbose	logical, TRUE to get log messages.
...	All other parameters are passed to stan_efftox .

Value

A list of [dose_finding_path_node](#) objects.

Author(s)

Kristian Brock

See Also

[efftox_parse_outcomes](#), [stan_efftox](#), [dose_finding_path_node](#)

Examples

```
## Not run:
# EffTox example
paths <- efftox_path_analysis(
  outcome_str = '1NNN 2NEN 3NEB',
  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, refresh = 0)

length(paths) # 4
names(paths)[1] # ""
names(paths)[2] # "1NNN"
names(paths)[3] # "1NNN 2NEN"
names(paths)[4] # "1NNN 2NEN 3NEB"
# Each node is an analysis fit to the cumulative outcomes
# Converting to a tibble presents some nice tidyverse-related opportunities
library(tibble)
df <- as_tibble(paths)
df

## End(Not run)

```

efftox_priors

Simple class to hold prior hyperparameters for the EffTox model.

Description

Simple class to hold prior hyperparameters for the EffTox model.

Usage

```

efftox_priors(
  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

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.

Value

list-like, instance of class `efftox_priors`.

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>

Examples

```
# The priors used in Thall et al. (2014)
p <- efftox_priors(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)
# The class exists simply to hold these twelve values.
```

efftox_process	<i>Process RStan samples from an EffTox model</i>
----------------	---

Description

Internal function to 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](#).

efftox_simulate	<i>Run EffTox simulations</i>
-----------------	-------------------------------

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

<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>num_sims</code>	integer, number of simulated iterations
<code>first_dose</code>	integer, the dose-level to give to patient 1, e.g. 1 for the lowest dose.
<code>true_eff</code>	the true probabilities of efficacy at the doses under investigation; a vector of numbers between 0 and 1.
<code>true_tox</code>	the true probabilities of toxicity at the doses under investigation; a vector of numbers between 0 and 1.
<code>cohort_sizes</code>	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.
<code>...</code>	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 `efftox_fit`.

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

```
fit <- stan_efftox_demo('1N 2E 3B')
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 `efftox_solve_p`

`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

fit	An instance of <code>efftox_fit</code> .
doses	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

```
fit <- stan_efftox_demo('1N 2E 3B')
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')
```

 eff_at_dose

Get the number of efficacy events seen at the doses under investigation.

Description

Get the number of efficacy events seen at the doses under investigation.

Usage

```
eff_at_dose(x, dose, ...)

## S3 method for class 'efftox_fit'
eff_at_dose(x, dose = NULL, ...)
```

Arguments

x	An R object of class "dose_finding_fit"
dose	Optional integer, at which dose-level? Omit to get data on all doses.
...	arguments passed to other methods

Value

integer vector

Examples

```
## Not run:
# EffTox example
x <- stan_efftox_demo(outcome_str = '1N 2E')
eff_at_dose(fit) # c(0, 1, 0, 0)
eff_at_dose(fit, dose = 2) # 1
eff_at_dose(fit, dose = 3) # 0

## End(Not run)
```

get_efftox_priors *Get normal prior hyperparameters for the EffTox model.*

Description

Get normal prior hyperparameters for the EffTox model using the algorithm presented in Thall et al. (2014) that targets a family of priors with a pre-specified effective sample size (ESS).

Usage

```
get_efftox_priors(
  doses = NULL,
  scaled_doses = NULL,
  pi_T,
  ess_T,
  pi_E,
  ess_E,
  num_samples = 10^4,
  seed = 123
)
```

Arguments

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). Specify doses or scaled_doses.
scaled_doses	Optional, vector of numbers, representing the scaled doses under investigation. Thall et al. advocate $\text{scaled_doses} = \log(\text{doses}) - \text{mean}(\log(\text{doses}))$, and that is what we use here. Specify doses or scaled_doses.
pi_T	Vector of prior expectations of probabilities of toxicity at the doses. Should be congruent to doses or scaled_doses.
ess_T	Numerical, sought total effective sample size for priors on parameters in the toxicity sub-model. Thall et al. (2014) advocate values in (0.3, 1.0) but stress that stress-testing with values outside this range may be necessary.
pi_E	Vector of prior expectations of probabilities of efficacy at the doses. Should be congruent to doses or scaled_doses.
ess_E	Numerical, sought total effective sample size for priors on parameters in the efficacy sub-model. Thall et al. (2014) advocate values in (0.3, 1.0) but stress that stress-testing with values outside this range may be necessary.
num_samples	Number of samples to draw from priors. The default 10^4 seems to be a nice compromise between accuracy and speed. Orders of magnitude larger take a long time to run.
seed	Optional seed. This process involves randomness so seeds are used for repeatable results.

Value

An instance of class `efftox_priors`.

References

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>

Examples

```
## Not run:
# Reproduce the priors calculated in Thall et al. (2014)
p <- get_efftox_priors(
  doses = c(1.0, 2.0, 4.0, 6.6, 10.0),
  pi_T = c(0.02, 0.04, 0.06, 0.08, 0.10), ess_T = 0.9,
  pi_E = c(0.2, 0.4, 0.6, 0.8, 0.9), ess_E = 0.9
)
p
# These are close to the published example. They do not match exactly because
# the process of deriving them is iterative.

## End(Not run)
```

n_at_dose

Get the number of patients treated at the doses under investigation.

Description

Get the number of patients treated at the doses under investigation.

Usage

```
n_at_dose(x, dose, ...)

## S3 method for class 'dose_finding_fit'
n_at_dose(x, dose = NULL, ...)
```

Arguments

x	An R object of class "dose_finding_fit"
dose	Optional integer, at which dose-level? Omit to get data on all doses.
...	arguments passed to other methods

Value

integer vector

Examples

```
## Not run:
# CRM example
target <- 0.2
fit <- stan_crm('1N 2N 3T', skeleton = c(0.1, 0.2, 0.35, 0.6),
               target = target, model = 'empiric', beta_sd = sqrt(1.34),
               seed = 123)
n_at_dose(fit)           # c(1, 1, 1, 0)
n_at_dose(fit, dose = 3) # 1

## End(Not run)
```

```
parse_dose_finding_outcomes
```

Parse a string of dose-finding trial outcomes.

Description

Parse a string of dose-finding trial outcomes

Parse a string of dose-finding trial outcomes to a list. The outcome string describes the doses given, outcomes observed and the timing of analyses that recommend a dose. 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 and it is assumed that a dose-finding decision takes place at the end of a cohort. Thus, 2NNT 1NN builds on our previous example, where the next cohort of two were treated at dose-level 1 and neither of these patients experienced toxicity. See examples.

Usage

```
parse_dose_finding_outcomes(outcome_string)
```

Arguments

`outcome_string` character representing doses given, outcomes observed, and timing of analyses. See Description.

Value

a list with a slot for each cohort. Each cohort slot is itself a list, containing elements: * `dose`, the integer dose delivered to the cohort; * `outcomes`, a character string representing the T or N outcomes for the patients in this cohort.

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 = parse_dose_finding_outcomes('1NNN 2NNT 3TT')
length(x)
x[[1]]$dose
x[[1]]$outcomes
x[[2]]$dose
x[[2]]$outcomes
x[[3]]$dose
x[[3]]$outcomes
```

parse_eff_tox_dose_finding_outcomes

Parse a string of phase I/II dose-finding trial outcomes.

Description

Parse a string of phase I/II dose-finding trial outcomes. Phase I/II trials conduct dose-finding by efficacy and toxicity outcomes.

Parse a string of phase I/II dose-finding outcomes to a list. The outcome string describes the doses given, efficacy and toxicity outcomes observed and the timing of analyses that recommend a dose. The format of the string is described in Brock *et al.* (2017). The letters E, T, N & B are used to represent patients that experienced (E)fficacy, (T)oxicity, (N)either and (B)oth. These letters are concatenated after numerical dose-levels to convey the outcomes of cohorts of patients. For instance, 2NET represents a cohort of three patients that were treated at dose-level 2, one of whom experienced toxicity only, one that experienced efficacy only, and one that had neither. The results of cohorts are separated by spaces and it is assumed that a dose-finding decision takes place at the end of a cohort. Thus, 2NET 1NN builds on our previous example, where the next cohort of two were treated at dose-level 1 and neither of these patients experienced either event. See examples.

Usage

```
parse_eff_tox_dose_finding_outcomes(outcome_string)
```

Arguments

`outcome_string` character representing doses given, outcomes observed, and timing of analyses. See Description.

Value

a list with a slot for each cohort. Each cohort slot is itself a list, containing elements: * dose, the integer dose delivered to the cohort; * outcomes, a character string representing the E, T N or B outcomes for the patients in this cohort.

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 = parse_eff_tox_dose_finding_outcomes('1NEN 2ENT 3TB')
length(x)
x[[1]]$dose
x[[1]]$outcomes
x[[2]]$dose
x[[2]]$outcomes
x[[3]]$dose
x[[3]]$outcomes
```

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

num_patients	Total number of patients to use, positive integer.
cohort_probs	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. cohort_probs or cohort_rho must be specified.
prob_eff	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 $\text{Dir}(\text{cohort_rho})$. cohort_probs or cohort_rho 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

Examples

```
## Not run:
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 <- stan_peps2(
  eff = dat$eff,
  tox = dat$tox,
  cohorts = dat$cohorts
)

## End(Not run)
```

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

Arguments

<code>fit</code>	An instance of <code>rstan::stanmodel</code> , derived by fitting data to the BEBOP in PePS2 model. Use <code>stan_peps2</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_get_data](#)

Examples

```
set.seed(123)
fit <- stan_peps2(
  eff = c(0, 1, 0, 1, 0, 0),
  tox = c(0, 0, 1, 1, 0, 0),
  cohorts = c(3, 1, 1, 4, 5, 6)
)
decision <- peps2_process(fit)
decision$Accept
decision$ProbEff
decision$ProbAccEff
```

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

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

```
predict.augbin_2t_1a_fit
```

Predict probability of success for given tumour size measurements.

Description

This method simply forwards to [prob_success](#).

Usage

```
## S3 method for class 'augbin_2t_1a_fit'
predict(
  object,
  y1_lower = -Inf,
  y1_upper = Inf,
  y2_lower = -Inf,
  y2_upper = log(0.7),
  probs = c(0.025, 0.975),
  newdata = NULL,
  ...
)
```

Arguments

object	Object of class <code>augbin_2t_1a_fit</code> .
y1_lower	numeric, minimum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 1 to baseline. Defaults to negative infinity.
y1_upper	numeric, maximum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 1 to baseline. Defaults to positive infinity.
y2_lower	numeric, minimum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 2 to baseline.
y2_upper	numeric, maximum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 2 to baseline. Defaults to <code>log(0.7)</code> .
probs	pair of probabilities to use to calculate the credible interval for the probability of success.
newdata	data for which to infer the probability of success. A dataframe-like object with baseline tumour sizes in first column, and first and second post-baseline tumour sizes in columns 2 and 3. Omitted by default. When omitted, <code>newdata</code> is set to be the <code>object\$tumour_size</code> .
...	Extra args passed onwards.

Value

Object of class [tibble](#)

print.augbin_fit *Print augbin_fit object.*

Description

Print augbin_fit object.

Usage

```
## S3 method for class 'augbin_fit'
print(
  x,
  pars = c("alpha", "beta", "gamma", "Omega", "sigma", "alphaD1", "gammaD1", "alphaD2",
           "gammaD2"),
  ...
)
```

Arguments

x	augbin_fit object to print.
pars	parameters in model to summarise.
...	Extra parameters, passed onwards.

print.crm_fit *Print crm_fit object.*

Description

Print crm_fit object.

Usage

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

Arguments

x	crm_fit object to print.
...	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.

print.nbg_fit *Print nbg_fit object.*

Description

Print nbg_fit object.

Usage

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

Arguments

x nbg_fit object to print.
... Extra parameters, passed onwards.

prior_predictive_augbin_2t_1a

Sample data from the Augmented Binary model prior predictive distribution.

Description

Sample data from the prior predictive distributions of the two-period, single arm Augmented Binary model, subject to chosen prior parameters.

Usage

```
prior_predictive_augbin_2t_1a(
  num_samps,
  alpha_mean,
  alpha_sd,
  beta_mean,
  beta_sd,
  gamma_mean,
  gamma_sd,
  sigma_mean,
  sigma_sd,
  omega_lkj_eta,
  alpha_d1_mean,
  alpha_d1_sd,
  gamma_d1_mean,
  gamma_d1_sd,
  alpha_d2_mean,
  alpha_d2_sd,
  gamma_d2_mean,
  gamma_d2_sd
)
```

Arguments

num_samps	Number of samples.
alpha_mean	Prior mean of alpha parameter.
alpha_sd	Prior sd of alpha parameter.
beta_mean	Prior mean of beta parameter.
beta_sd	Prior sd of beta parameter.
gamma_mean	Prior mean of gamma parameter.
gamma_sd	Prior sd of gamma parameter.
sigma_mean	Prior mean of sigma parameter.
sigma_sd	Prior sd of sigma parameter.

omega_lkj_eta	Prior eta parameter for LKJ prior on covariance matrix of log tumour sizes.
alpha_d1_mean	Prior mean of alpha_D1 parameter.
alpha_d1_sd	Prior sd of alpha_D1 parameter.
gamma_d1_mean	Prior mean of gamma_D1 parameter.
gamma_d1_sd	Prior sd of gamma_D1 parameter.
alpha_d2_mean	Prior mean of alpha_D2 parameter.
alpha_d2_sd	Prior sd of alpha_D2 parameter.
gamma_d2_mean	Prior mean of gamma_D2 parameter.
gamma_d2_sd	Prior sd of gamma_D2 parameter.

Value

Object of class [tibble](#)

See Also

[stan_augbin](#)

Examples

```
prior_predictive_augbin_2t_1a(num_samps = 1000,  
  alpha_mean = 0, alpha_sd = 1,  
  beta_mean = 0, beta_sd = 1,  
  gamma_mean = 0, gamma_sd = 1,  
  sigma_mean = 0, sigma_sd = 1,  
  omega_lkj_eta = 1,  
  alpha_d1_mean = 0, alpha_d1_sd = 1,  
  gamma_d1_mean = 0, gamma_d1_sd = 1,  
  alpha_d2_mean = 0, alpha_d2_sd = 1,  
  gamma_d2_mean = 0, gamma_d2_sd = 1)
```

prob_success

Calculate the probability of success.

Description

Calculate the probability of success.

Calculate the probability of success for an `augbin_2t_1a_fit` object.

Usage

```

prob_success(x, ...)

## S3 method for class 'augbin_2t_1a_fit'
prob_success(
  x,
  y1_lower = -Inf,
  y1_upper = Inf,
  y2_lower = -Inf,
  y2_upper = log(0.7),
  probs = c(0.025, 0.975),
  newdata = NULL,
  ...
)

```

Arguments

x	an R object of class "augbin_fit"
...	arguments passed to other methods
y1_lower	numeric, minimum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 1 to baseline. Defaults to negative infinity.
y1_upper	numeric, maximum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 1 to baseline. Defaults to positive infinity.
y2_lower	numeric, minimum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 2 to baseline.
y2_upper	numeric, maximum threshold to constitute success, scrutinising the log of the tumour size ratio comparing time 2 to baseline. Defaults to log(0.7).
probs	pair of probabilities to use to calculate the credible interval for the probability of success.
newdata	data for which to infer the probability of success. A dataframe-like object with baseline tumour sizes in first column, and first and second post-baseline tumour sizes in columns 2 and 3. Omitted by default. When omitted, newdata is set to be the fit\$tumour_size.

Value

Object of class `tibble`

Examples

```

## Not run:
fit <- stan_augbin_demo()
prob_success(fit, y2_upper = log(0.7))

## End(Not run)

```

prob_tox_exceeds	<i>Calculate the probability that the rate of toxicity exceeds some threshold</i>
------------------	---

Description

Calculate the probability that the rate of toxicity exceeds some threshold

Calculate the probability that the rate of toxicity exceeds some threshold

Usage

```
prob_tox_exceeds(x, ...)

## S3 method for class 'dose_finding_fit'
prob_tox_exceeds(x, threshold, ...)
```

Arguments

x	an R object of class "dose_finding_fit"
...	arguments passed to other methods
threshold	numeric, threshold value.

Value

numerical vector of probabilities

numerical vector of probabilities

Examples

```
## Not run:
# CRM example
target <- 0.2
fit <- stan_crm('1N 2N 3T', skeleton = c(0.1, 0.2, 0.35, 0.6),
               target = target, model = 'empiric', beta_sd = sqrt(1.34),
               seed = 123)
prob_tox_exceeds(fit, target)

## End(Not run)
```

ranBin2	<i>Sample pairs of correlated binary events</i>
---------	---

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 80 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
```

rlkjcorr	<i>Sample LKJ correlation matrices.</i>
----------	---

Description

This function was copied from Richard McElreath's `rethinking` package hosted at <https://github.com/rmcelreath/rethinking>. In turn, he appears to have copied it from Ben Bolker's `rLKJ` function from the `emdbook` package, although I cannot find it there (else I would have imported it).

Usage

```
rlkjcorr(n, K, eta = 1)
```

Arguments

n	Number of matrices to sample.
K	dimension of matrix to sample.
eta	Distribution parameter

Value

matrix

spread_paths *Spread the information in dose_finding_paths object to a wide data.frame format.*

Description

Spread the information in dose_finding_paths object to a wide data.frame format.

Usage

```
spread_paths(df = NULL, dose_finding_paths = NULL, max_depth = NULL)
```

Arguments

df	Optional data.frame like that returned by as_tibble(dose_finding_paths). Columns .depth, .node, .parent are required. All other columns are spread with a suffix reflecting depth.
dose_finding_paths	Optional instance of dose_finding_paths. Required if 'df' is null.
max_depth	integer, maximum depth of paths to traverse.

Value

A data.frame

Examples

```
## Not run:
target <- 0.25
skeleton <- c(0.05, 0.15, 0.25, 0.4, 0.6)
paths <- crm_dtpts(skeleton = skeleton, target = target, model = 'empiric',
                  cohort_sizes = c(1, 1), next_dose = 3, beta_sd = 1)
spread_paths(dose_finding_paths = paths)

df <- as_tibble(paths)
spread_paths(df)
spread_paths(df %>% select(-fit, -parent_fit, -dose_index))

## End(Not run)
```


stan_augbin

*Fit Wason & Seaman's Augmented Binary model for tumour response.***Description**

Phase II clinical trials in oncology commonly assess response as a key outcome measure. Patients achieve a RECIST response if their tumour size post-baseline has changed in size by some threshold amount and they do not experience non-shrinkage failure. An example of non-shrinkage failure is the appearance of new lesions. As a dichotomisation of the underlying continuous tumour size measurement, RECIST response is inefficient. Wason & Seaman introduced the Augmented Binary method to incorporate mechanisms for non-shrinkage failure whilst modelling the probability of response based on the continuous tumour size measurements. See model-specific sections below, and the references.

Usage

```
stan_augbin(
  tumour_size,
  non_shrinkage_failure,
  arm = NULL,
  model = c("2t-1a"),
  prior_params = list(),
  ...
)
```

Arguments

tumour_size	matrix-like object containing tumour size measures, with rows representing patients and columns representing chronological standardised assessment points. Column one is baseline.
non_shrinkage_failure	matrix-like object containing logical indicators of non-shrinkage failure, with rows representing patients and columns representing chronological standardised assessment points.
arm	optional vector of integers representing the allocated treatment arms for patients, assumed in the same order as tumour_size and non_shrinkage_failure. NULL to fit the augbin variant for single-arm trials. NULL is the default.
model	Character string to denote the desired model. Currently, only 2t-1a is supported, representing the model variant with two post-baseline assessments in a single arm trial. Multi-period and multi-arm versions will be added in future releases. The model choice determines the prior parameters that must be provided. See sections below.
prior_params	list of prior parameters. These are combined with the data and passed to <code>rstan::sampling</code> . The parameters required depend on the model form being fit. See sections below.
...	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> . See sampling .

Value

an instance or subclass of type [augbin_fit](#).

Single-arm model with two post-baseline assessments

The complete model form is:

$$(y_{1i}, y_{2i})^T \sim N((\mu_{1i}, \mu_{2i})^T, \Sigma)$$

$$\mu_{1i} = \alpha + \gamma z_{0i}$$

$$\mu_{2i} = \beta + \gamma z_{0i}$$

$$\text{logit}(\Pr(D_{1i} = 1 | Z_{0i})) = \alpha_{D1} + \gamma_{D1} z_{0i}$$

$$\text{logit}(\Pr(D_{2i} = 1 | D_{1i} = 0, Z_{0i}, Z_{1i})) = \alpha_{D2} + \gamma_{D2} z_{1i}$$

where z_{0i}, z_{1i}, z_{2i} are tumour sizes at baseline, period 1, and period 2, for patient i ; y_{1i}, y_{2i} are the log-tumour-size ratios with respect to baseline; D_{1i}, D_{2i} are indicators of non-shrinkage failure; and Σ is assumed to be unstructured covariance matrix, with associated correlation matrix having an LKJ prior.

The following prior parameters are required:

- alpha_mean & alpha_sd for normal prior on α .
- beta_mean & beta_sd for normal prior on β .
- gamma_mean & gamma_sd for normal prior on γ .
- sigma_mean & sigma_sd for normal priors on diagonal elements of Σ ;
- omega_lkj_eta for a LKJ prior on the two-period correlation matrix associated with Sigma. omega_lkj_eta = 1 is uniform, analogous to a Beta(1,1) prior on a binary probability.
- alpha_d1_mean & alpha_d1_sd for normal prior on α_{D1} .
- gamma_d1_mean & gamma_d1_sd for normal prior on γ_{D1} .
- alpha_d2_mean & alpha_d2_sd for normal prior on α_{D2} .
- gamma_d2_mean & gamma_d2_sd for normal prior on γ_{D2} .

Author(s)

Kristian Brock

References

Wason JMS, Seaman SR. Using continuous data on tumour measurements to improve inference in phase II cancer studies. *Statistics in Medicine*. 2013;32(26):4639-4650. doi:10.1002/sim.5867

Eisenhauer EA, Therasse P, Bogaerts J, et al. New response evaluation criteria in solid tumours: Revised RECIST guideline (version 1.1). *European Journal of Cancer*. 2009;45(2):228-247. doi:10.1016/j.ejca.2008.10.026

See Also

[augbin_fit prior_predictive_augbin_2t_1a sampling](#)

Examples

```

priors <- list(alpha_mean = 0, alpha_sd = 1,
              beta_mean = 0, beta_sd = 1,
              gamma_mean = 0, gamma_sd = 1,
              sigma_mean = 0, sigma_sd = 1,
              omega_lkj_eta = 1,
              alpha_d1_mean = 0, alpha_d1_sd = 1,
              gamma_d1_mean = 0, gamma_d1_sd = 1,
              alpha_d2_mean = 0, alpha_d2_sd = 1,
              gamma_d2_mean = 0, gamma_d2_sd = 1)
# Scenario 1 of Table 1 in Wason & Seaman (2013)
N <- 50
sigma <- 1
delta1 <- -0.356
mu <- c(0.5 * delta1, delta1)
Sigma = matrix(c(0.5 * sigma^2, 0.5 * sigma^2, 0.5 * sigma^2, sigma^2),
              ncol = 2)
alphaD <- -1.5
gammaD <- 0
set.seed(123456)
y <- MASS::mvrnorm(n = N, mu, Sigma)
z0 <- runif(N, min = 5, max = 10)
z1 <- exp(y[, 1]) * z0
z2 <- exp(y[, 2]) * z0
d1 <- rbinom(N, size = 1, prob = gtools::inv.logit(alphaD + gammaD * z0))
d2 <- rbinom(N, size = 1, prob = gtools::inv.logit(alphaD + gammaD * z1))
tumour_size <- data.frame(z0, z1, z2) # Sizes in cm
non_shrinkage_failure <- data.frame(d1, d2)
# Fit
## Not run:
fit <- stan_augbin(tumour_size, non_shrinkage_failure,
                  prior_params = priors, model = '2t-1a', seed = 123)

## End(Not run)

```

stan_augbin_demo

Simple helper function to demonstrate fitting of an Augmented Binary model.

Description

This function exist mostly to demonstrate things you can do to instances of `augbin_fit` without having to paste into each example the not inconsiderable blob of code to sample outcomes and fit the model.

Usage

```
stan_augbin_demo()
```

Value

instance of `augbin_fit`

See Also

`stan_augbin` `augbin_fit` `prior_predictive_augbin_2t_1a` `sampling`

Examples

```
## Not run:
fit <- stan_augbin_demo()
# I told you it was simple.

## End(Not run)
```

stan_crm

Fit a CRM model

Description

Fit a continual reassessment method (CRM) model for dose-finding using Stan for full Bayesian inference. There are several likelihood and prior combinations supported. See model-specific sections below.

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,
  weights = 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.
weights	An optional vector of numeric weights for the observations for patients 1:num_patients, thus facilitating the TITE-CRM design. Can be used with outcome_str, or with doses_given and tox. It is generally tidier to specify doses_given, tox and weights when a TITE-CRM analysis is desired.
...	Extra parameters are passed to rstan::sampling. Commonly used options are iter, chains, warmup, cores, and control.

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 choices require that different parameters are provided. See sections below.

Value

An object of class `crm_fit`

The empiric model

The model form is:

$$F(x_i, \beta) = x_i^{\exp \beta}$$

and the required parameters are:

- `beta_sd`

The logistic model

The model form is:

$$F(x_i, \beta) = 1/(1 + \exp(-a_0 - \exp(\beta)x_i))$$

and the required parameters are:

- `a0`
- `beta_mean`
- `beta_sd`

The logistic_gamma model

The model form is:

$$F(x_i, \beta) = 1/(1 + \exp(-a_0 - \exp(\beta)x_i))$$

and the required parameters are:

- `a0`
- `beta_shape`
- `beta_inverse_scale`

The logistic2 model

The model form is:

$$F(x_i, \alpha, \beta) = 1/(1 + \exp(-\alpha - \exp(\beta)x_i))$$

and the required parameters are:

- `alpha_mean`
- `alpha_sd`
- `beta_mean`
- `beta_sd`

Author(s)

Kristian Brock

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://www.jstor.org/stable/2531628>
- Cheung, Y.K. (2011). *Dose Finding by the Continual Reassessment Method*. CRC Press. ISBN 9781420091519

See Also

[crm_fit sampling](#)

Examples

```
## Not run:
# CRM example
fit1 <- 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)

fit2 <- stan_crm('1NNN 2NNN 3TTT', skeleton = c(0.1, 0.2, 0.35, 0.6),
               target = 0.2, model = 'logistic', a0 = 3, beta_mean = 0,
               beta_sd = sqrt(1.34), 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.

# TITE-CRM example, p.124 of Dose Finding by the CRM, Cheung (2010)
fit3 <- stan_crm(skeleton = c(0.05, 0.12, 0.25, 0.40, 0.55), target = 0.25,
               doses_given = c(3, 3, 3, 3),
               tox = c(0, 0, 0, 0),
               weights = c(73, 66, 35, 28) / 126,
               model = 'empiric', beta_sd = sqrt(1.34), seed = 123)
fit3$recommended_dose

## 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,
  priors = NULL,
  alpha_mean = NULL,
  alpha_sd = NULL,
  beta_mean = NULL,
  beta_sd = NULL,
  gamma_mean = NULL,
  gamma_sd = NULL,
  zeta_mean = NULL,
  zeta_sd = NULL,
  eta_mean = NULL,
  eta_sd = NULL,
  psi_mean = NULL,
  psi_sd = NULL,
  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).
priors	instance of class <code>efftox_priors</code> , the hyperparameters for normal priors on the six model parameters.
alpha_mean	Optional, the prior normal mean of the intercept term in the toxicity logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
alpha_sd	Optional, the prior normal standard deviation of the intercept term in the toxicity logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
beta_mean	Optional, the prior normal mean of the slope term in the toxicity logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
beta_sd	Optional, the prior normal standard deviation of the slope term in the toxicity logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
gamma_mean	Optional, The prior normal mean of the intercept term in the efficacy logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
gamma_sd	Optional, the prior normal standard deviation of the intercept term in the efficacy logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
zeta_mean	Optional, the prior normal mean of the slope term in the efficacy logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
zeta_sd	Optional, the prior normal standard deviation of the slope term in the efficacy logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
eta_mean	Optional, the prior normal mean of the squared term coefficient in the efficacy logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
eta_sd	Optional, the prior normal standard deviation of the squared term coefficient in the efficacy logit model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
psi_mean	Optional, the prior normal mean of the association term in the combined efficacy-toxicity model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
psi_sd	Optional, the prior normal standard deviation of the association term in the combined efficacy-toxicity model. A number. You should prioritise specifying this value via <code>priors</code> but this option is provided for backwards-compatibility.
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 <code>outcome_str</code> 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 <code>outcome_str</code> 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 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_priors](#) [get_efftox_priors](#) [efftox_fit](#) [stan_efftox_demo](#)

Examples

```
## Not run:
# This model is presented in Thall et al. (2014).
p <- efftox_priors(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)
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,
  priors = p,
  seed = 123)

# The above is a longhad version of:
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](#)

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_hierarchical_response_thall

Fit the hierarchical response model described by Thall et al. (2003).

Description

Fit the hierarchical response model to exchangeable groups described by Thall *et al.* (2003).

Usage

```
stan_hierarchical_response_thall(
  group_responses,
  group_sizes,
  mu_mean,
  mu_sd,
  tau_alpha,
  tau_beta,
  ...
)
```

Arguments

group_responses	vector of integers, number of responses in each group
group_sizes	vector of integers, number of patients in each group
mu_mean	mean parameter of normal prior distribution on mu. See details.
mu_sd	standard deviation parameter of normal prior distribution on mu. See details.
tau_alpha	parameter alpha of inverse gamma prior distribution on tau. See details.
tau_beta	beta parameter of inverse gamma prior distribution on tau. See details.
...	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> , and <code>control</code> .

Details

Thall *et al.* (2003) describe hierarchical methods for analysing treatment effects of a common intervention in several sub-types of a disease. The treatment effects are assumed to be different but exchangeable and correlated. Observing efficacy in one cohort, for example, increases one's expectations of efficacy in others. They demonstrate the hierarchical approach in a trial with binary response outcomes and in another with time-to-event outcomes. This function fits their model for binary response outcomes.

Let the probability of response in group i be $\pi[i]$ for $i = 1, \dots, N$. They assume a logistic model so that $\theta_i = \log \pi_i / (1 - \pi_i)$ is the log-odds of response in group i . They assume that $\theta_i \sim N(\mu, \sigma^2)$.

The authors implemented their model in BUGS. As is the convention in BUGS, the authors define normal distributions by a precision parameter τ as opposed to the standard deviation parameter σ used here. We have re-specified their model to comply with the Stan convention of using standard deviation. The authors use a normal prior on μ , and a gamma prior on τ , equivalent to an inverse gamma prior on $\tau^{-1} = \sigma^2$.

The authors provide WinBUGS code in their publication. We implement their model here in Stan.

Value

Object of class `rstan::stanfit` returned by `rstan::sampling`

References

Thall, Wathen, Bekele, Champlin, Baker, and Benjamin. 2003. "Hierarchical Bayesian approaches to phase II trials in diseases with multiple subtypes." *Statistics in Medicine* 22 (5): 763–80. <https://doi.org/10.1002/sim.1399>.

See Also

`rstan::stanfit`, `rstan::sampling`

Examples

```
## Not run:
# Example from p.778 of Thall et al. (2003)
mod0 <- stan_hierarchical_response_thall(
  group_responses = c(0, 0, 1, 3, 5, 0, 1, 2, 0, 0),
```

```

group_sizes = c(0, 2, 1, 7, 5, 0, 2, 3, 1, 0),
mu_mean = -1.3863,
mu_sd = sqrt(1 / 0.1),
tau_alpha = 2,
tau_beta = 20)

## End(Not run)

```

stan_nbg

Fit a Neuenschwander, Branson & Gsponer logit dose-finding model

Description

Fit Neuenschwander, Branson & Gsponer logit model for dose-finding using Stan for full Bayesian inference.

Usage

```

stan_nbg(
  outcome_str = NULL,
  real_doses,
  d_star,
  target,
  alpha_mean = NULL,
  alpha_sd = NULL,
  beta_mean = NULL,
  beta_sd = NULL,
  doses_given = NULL,
  tox = NULL,
  weights = 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.
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).
d_star	d_star, numeric reference dose-level. The linear covariate in this logit model is dose / d_star.
target	the target toxicity probability, a number between 0 and 1.
alpha_mean	Prior mean of intercept variable for normal prior. See Details.

alpha_sd	Prior standard deviation of intercept variable for normal prior. See Details.
beta_mean	Prior mean of gradient variable for normal prior. See Details.
beta_sd	Prior standard deviation of slope variable for normal prior. 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.
weights	An optional vector of numeric weights for the observations for patients 1:num_patients, thus facilitating a time-to-event (TITE) design. Can be used with outcome_str, or with doses_given and tox. It is generally tidier to specify doses_given, tox and weights when a TITE-analysis is desired.
...	Extra parameters are passed to rstan::sampling. Commonly used options are iter, chains, warmup, cores, and control.

Details

The quickest and easiest way to fit this 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.

The two-parameter model form is:

$$F(x_i, \alpha, \beta) = 1 / (1 + \exp -(\alpha + \exp(\beta) \log(x_i / d_{star})))$$

and the required parameters are:

- alpha_mean
- alpha_sd
- beta_mean
- beta_sd

Value

An object of class `nbg_fit`, which inherits behaviour from [crm_fit](#).

Author(s)

Kristian Brock <kristian.brock@gmail.com>

References

Neuenschwander, B., Branson, M., & Gsponer, T. (2008). Critical aspects of the Bayesian approach to phase I cancer trials. *Statistics in Medicine*, 27, 2420–2439. <https://doi.org/10.1002/sim>

See Also

[crm_fit](#) [sampling](#)

Examples

```
## Not run:
# Non-TITE example:
fit1 <- stan_nbg('1NNN 2NNN 3TTT', real_doses = c(10, 20, 50, 100, 200),
               d_star = 200, target = 0.25,
               alpha_mean = -1, alpha_sd = 2,
               beta_mean = 0, beta_sd = 1,
               seed = 123)
fit1$recommended_dose

# 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.

# TITE-CRM example
fit2 <- stan_nbg(real_doses = c(10, 20, 50, 100, 200), d_star = 200,
               target = 0.25,
               doses_given = c(3, 3, 3, 3),
               tox = c(0, 0, 0, 0),
               weights = c(73, 66, 35, 28) / 126,
               alpha_mean = -1, alpha_sd = 2,
               beta_mean = 0, beta_sd = 1,
               seed = 123)
fit2$recommended_dose

## End(Not run)
```

stan_peps2

Fit the P2TNE model developed for the PePS2 trial to some outcomes.

Description

The PePS2 trial investigates pembrolizumab in non-small-cell lung cancer. Patients may be previously treated (PT) or treatment naive (TN). 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 categories <1 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

```
stan_peps2(
  eff,
```



```

    tox,
    cohorts,
    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>eff</code>	A vector of efficacy outcomes for the patients, where 1=efficacy and 0=no efficacy.
<code>tox</code>	A vector of toxicity outcomes for the patients, where 1=toxicity and 0=no toxicity.
<code>cohorts</code>	A vector of integers from 1 to 6, denoting the cohorts to which the patients belong.
<code>alpha_mean</code>	The prior mean of alpha. Alpha is the efficacy model intercept.
<code>alpha_sd</code>	The prior standard deviation of alpha. Alpha is the efficacy model intercept.
<code>beta_mean</code>	The prior mean of beta. Beta is the efficacy model term for being previously treated.
<code>beta_sd</code>	The prior standard deviation of beta. Beta is the efficacy model term for being previously treated.
<code>gamma_mean</code>	The prior mean of gamma. Gamma is the efficacy model term for being PD-L1 score = Low.
<code>gamma_sd</code>	The prior standard deviation of gamma. Gamma is the efficacy model term for being PD-L1 score = Low.
<code>zeta_mean</code>	The prior mean of zeta. Zeta is the efficacy model term for being PD-L1 score = Medium.
<code>zeta_sd</code>	The prior standard deviation of zeta. Zeta is the efficacy model term for being PD-L1 score = Medium.
<code>lambda_mean</code>	The prior mean of lambda. Lambda is the toxicity model intercept.
<code>lambda_sd</code>	The prior standard deviation of lambda. Lambda is the toxicity model intercept.
<code>psi_mean</code>	The prior mean of psi. Psi is the joint model association parameter.
<code>psi_sd</code>	The prior standard deviation of psi. Psi is the joint model association parameter.
<code>...</code>	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> , and <code>control</code> .

Value

Object of class `rstan::stanfit` returned by `rstan::sampling`

Examples

```
## Not run:
fit <- stan_peps2(
  eff = c(0, 1, 0, 1, 0, 0),
  tox = c(0, 0, 1, 1, 0, 0),
  cohorts = c(3, 1, 1, 4, 5, 6)
)

## End(Not run)
```

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	<code>crm_fit</code> object to summarise.
...	Extra parameters, passed onwards.

Value

A summary object.

See Also

[stan_crm](#)

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.

total_weight_at_dose *Get the total weight of patient outcomes at the doses under investigation.*

Description

Get the total weight of patient outcomes at the doses under investigation.

Usage

```
total_weight_at_dose(x, dose, ...)

## Default S3 method:
total_weight_at_dose(x, dose = NULL, ...)
```

Arguments

x An R object of class "dose_finding_fit"
 dose Optional integer, at which dose-level? Omit to get data on all doses.
 ... arguments passed to other methods

Value

numerical vector

Examples

```
## Not run:
# CRM example
fit <- stan_crm(skeleton = c(0.1, 0.2, 0.35, 0.6), target = 0.2,
               model = 'empiric', beta_sd = sqrt(1.34), seed = 123,
               doses = c(1, 1, 2, 2, 2),
               tox = c(0, 0, 0, 0, 0),
               weights = c(1, 1, 0.9, 0.1, 0.1))

total_weight_at_dose(fit)           # c(2, 1.1, 0, 0)
total_weight_at_dose(fit, dose = 2) # 1.1

## End(Not run)
```

tox_at_dose	<i>Get the number of toxicity events seen at the doses under investigation.</i>
-------------	---

Description

Get the number of toxicity events seen at the doses under investigation.

Usage

```
tox_at_dose(x, dose, ...)

## S3 method for class 'dose_finding_fit'
tox_at_dose(x, dose = NULL, ...)
```

Arguments

x	An R object of class "dose_finding_fit"
dose	Optional integer, at which dose-level? Omit to get data on all doses.
...	arguments passed to other methods

Value

integer vector

Examples

```
## Not run:
# CRM example
target <- 0.2
fit <- stan_crm('1N 2N 3T', skeleton = c(0.1, 0.2, 0.35, 0.6),
               target = target, model = 'empiric', beta_sd = sqrt(1.34),
               seed = 123)

tox_at_dose(fit)           # c(0, 0, 1, 0)
tox_at_dose(fit, dose = 3) # 1

## End(Not run)
```

trialr_simulate	<i>Run a simulation study.</i>
-----------------	--------------------------------

Description

This function is a fairly flexible way of running simulation studies in trialr, and beyond. It essentially uses delegates to perform this pattern:

```
for i in 1:N:
  data = get_data_func()
  fit = fit_model_func(data)
  if summarise_func is null:
    sims[i] = fit
  else
    sims[i] = summarise_func(data, fit)
  end
loop
return sims
```

Usage

```
trialr_simulate(
  N,
  get_data_func,
  fit_model_func,
  summarise_func = NULL,
  num_logs = 10,
  num_saves = NULL,
  save_func = NULL
)
```

Arguments

N	integer, number of simulated iterations to run.
get_data_func	Function that takes no parameters and returns a sampled dataset to be analysed. I.e. the call signature is f().
fit_model_func	Function that accepts the output of get_data_func as the sole parameter and fits the model or performs the analysis, returning an object of arbitrary type.
summarise_func	Optional. If provided, this function should accept the outputs of get_data_func and fit_model_func as parameters 1 & 2 and perform some post-fit processing or simplification. The result of this call is the output from iteration i. If omitted, the fit object from fit_model_func is simply used as the output from iteration i.
num_logs	Number of log messages to receive about progress. NULL to suppress logging. E.g. if N=100 and num_logs=10, you will get log messages when i=10, 20, 30, etc.

num_saves	Number of intermittent saves to attempt. NULL to suppress saving E.g. if N=100 and num_saves=10, the save_func delegate will be called after iteration i=10, 20, 30, etc.
save_func	Optional. Function that takes the interim list of simulated objects as the sole parameter and saves them somehow. This, combined with num_saves, allows periodic saving of in-progress results to avoid complete data loss if the simulation study fails for some reason.

Value

list of length N. The items in the list are as returned by summarise_func or fit_model_func.

Examples

```
get_data_func <- function() {
  group_sizes <- rbinom(n = 5, size = 50, prob = c(0.1, 0.3, 0.3, 0.2, 0.1))
  group_responses <- rbinom(n = 5, size = group_sizes,
                           prob = c(0.2, 0.5, 0.2, 0.2, 0.2))
  list(
    group_responses = group_responses, group_sizes = group_sizes,
    mu_mean = gtools::logit(0.1), mu_sd = 1, tau_alpha = 2, tau_beta = 20
  )
}
fit_model_func <- function(data) {
  data <- append(data, list(refresh = 0))
  do.call(stan_hierarchical_response_thall, args = data)
}
summarise_func <- function(data, fit) {
  # Probability that estimate response rate exceeds 30%
  unname(colMeans(as.data.frame(fit, 'prob_response') > 0.3))
}
## Not run:
sims <- trialr_simulate(N = 20, get_data_func, fit_model_func, summarise_func)
# Posterior probabilities that the response rate in each cohort exceeds 30%:
do.call(rbind, sims)
# Cohorts are in columns; simulated iterations are in rows.

## End(Not run)
```

weights_at_dose

Get the weights of patient outcomes at the doses under investigation.

Description

Get the weights of patient outcomes at the doses under investigation.

Usage

```
weights_at_dose(x, dose, ...)  
  
## Default S3 method:  
weights_at_dose(x, dose = NULL, ...)  
  
## S3 method for class 'crm_fit'  
weights_at_dose(x, dose = NULL, ...)
```

Arguments

x	An R object of class "dose_finding_fit"
dose	Optional integer, at which dose-level? Omit to get data on all doses.
...	arguments passed to other methods

Value

list if dose omitted, numerical vector if dose provided.

Examples

```
## Not run:  
# CRM example  
fit <- stan_crm(skeleton = c(0.1, 0.2, 0.35, 0.6), target = 0.2,  
              model = 'empiric', beta_sd = sqrt(1.34), seed = 123,  
              doses = c(1, 1, 2, 2, 2),  
              tox = c(0, 0, 0, 0, 0),  
              weights = c(1, 1, 0.9, 0.1, 0.1))  
l <- weights_at_dose(fit)  
  
length(l) # 4  
l[[1]] # c(1, 1)  
l[[2]] # c(0.9, 0.1, 0.1)  
l[[3]] # c()  
  
weights_at_dose(fit, dose = 2) # c(0.9, 0.1, 0.1)  
  
## End(Not run)
```

Index

as.data.frame.crm_fit, 4
as.data.frame.efftox_fit, 4
as.mcmc.list.crm_fit, 5
as.mcmc.list.efftox_fit, 5
as_tibble.augbin_2t_1a_fit, 6
as_tibble.dose_finding_paths, 6
augbin_2t_1a_fit, 7
augbin_fit, 7, 8, 57, 66–68

binary_prob_success, 8

careful_escalation, 10
closest_to_target, 11
crm_codified_dose_logistic, 11
crm_dtpts, 12
crm_fit, 4, 5, 13, 21, 22, 24, 55, 57, 70, 71, 79, 82
crm_fit (crm_fit-class), 15
crm_fit-class, 15
crm_params, 16, 22, 24
crm_params (crm_params-class), 16
crm_params-class, 16
crm_path_analysis, 14, 18
crm_prior_beliefs, 20
crm_process, 22

df_parse_outcomes, 13, 14, 18, 19, 23, 25, 69, 78, 79
dose_finding_fit, 10
dose_finding_fit
(dose_finding_fit-class), 24
dose_finding_fit-class, 24
dose_finding_path_node, 13, 14, 18, 19, 29, 38
dose_finding_path_node
(dose_finding_path_node-class), 25
dose_finding_path_node-class, 25

eff_at_dose, 46

efftox_analysis_to_df, 26
efftox_contour_plot, 27
efftox_dtpts, 28, 31
efftox_dtpts_to_dataframe, 28, 31
efftox_fit, 4, 5, 24, 26–28, 41, 55, 58, 74–76, 83
efftox_fit (efftox_fit-class), 32
efftox_fit-class, 32
efftox_get_tox, 34
efftox_parameters_demo, 31, 35, 41, 42
efftox_params, 31, 33, 35, 41, 42
efftox_params (efftox_params-class), 36
efftox_params-class, 36
efftox_parse_outcomes, 28, 29, 37, 38, 72, 74, 75
efftox_path_analysis, 29, 38
efftox_priors, 36, 37, 39, 48, 73, 74
efftox_process, 41
efftox_simulate, 41
efftox_solve_p, 34, 43, 45
efftox_superiority, 44
efftox_utility, 44
efftox_utility_density_plot, 45

get_efftox_priors, 37, 47, 74

list, 13, 18, 38

mcmc.list, 5

n_at_dose, 48

parse_dose_finding_outcomes, 49
parse_eff_tox_dose_finding_outcomes, 50

peps2_get_data, 51, 54
peps2_process, 53
plot.crm_fit, 55
plot.efftox_fit, 55
predict.augbin_2t_1a_fit, 56
print.augbin_fit, 57

print.crm_fit, 57
print.efftox_fit, 58
print.nbg_fit, 58
prior_predictive_augbin_2t_1a, 59, 66,
68
prob_success, 56, 60
prob_tox_exceeds, 62

ranBin2, 63
rlkjcorr, 63
rstan::sampling, 77, 82
rstan::stanfit, 77, 82

sampling, 16, 24, 33, 65, 66, 68, 71, 74, 75, 79
spread_paths, 64
stan_augbin, 7, 8, 60, 65, 68
stan_augbin_demo, 67
stan_crm, 13, 14, 16–22, 68, 82
stan_efftox, 26, 27, 29, 34, 37, 38, 71, 76
stan_efftox_demo, 34, 37, 74, 75
stan_hierarchical_response_thall, 76
stan_nbg, 78
stan_peps2, 80
stanfit, 7, 8, 16, 24, 33
summary.crm_fit, 82
summary.efftox_fit, 83

tibble, 6, 56, 60, 61
total_weight_at_dose, 83
tox_at_dose, 84
trialr (trialr-package), 3
trialr-package, 3
trialr_simulate, 85

weights_at_dose, 86