Package 'oncomsm'

April 17, 2023

Type Package

Title Bayesian Multi-State Models for Early Oncology

Version 0.1.4

Description Implements methods to fit a parametric Bayesian multi-state model to tumor response data.
The model can be used to sample from the predictive distribution to impute missing data and calculate probability of success for custom decision criteria in early clinical trials during an ongoing trial.
The inference is implemented using 'stan'.

Language en-US

License Apache License 2.0

Encoding UTF-8

RoxygenNote 7.2.3

Biarch true

Depends R (>= 3.6)

Imports methods, Rcpp, RcppNumerical (>= 0.4), rstan (>= 2.18), rlang (>= 0.4), magrittr, tibble, dplyr, tidyr, purrr, furrr, stringr, ggplot2, checkmate, rstantools

SystemRequirements GNU make

LinkingTo BH (>= 1.66.0), Rcpp, RcppEigen (>= 0.3), RcppNumerical (>= 0.4), rstan (>= 2.18), StanHeaders (>= 2.18), RcppParallel

Suggests rmarkdown, knitr, testthat (>= 3.0.0), patchwork, bhmbasket, vdiffr, DiagrammeR, future, doFuture, doRNG, rjags

VignetteBuilder knitr

Config/testthat/edition 3

URL https://boehringer-ingelheim.github.io/oncomsm/, https://github.com/Boehringer-Ingelheim/oncomsm

BugReports https://github.com/Boehringer-Ingelheim/oncomsm/issues

NeedsCompilation yes

Author Kevin Kunzmann [aut, cre] (<https://orcid.org/0000-0002-1140-7143>),
Karthik Ananthakrishnan [ctb],
Boehringer Ingelheim Ltd. [cph, fnd]

Maintainer Kevin Kunzmann <kevin.kunzmann@boehringer-ingelheim.com>

Repository CRAN

Date/Publication 2023-04-17 07:00:02 UTC

R topics documented:

`oncomsm-package` *The oncomsm package*

Description

This package implements methods to dynamically predict response and progression of individuals in early oncology trials using parametric multi-state models and Bayesian inference. This allows the dynamic computation of Probability of Success for a wide The inference is implemented using 'rstan'.

References

Stan Development Team (2021). "RStan: the R interface to Stan". R package version 2.21.3. https://mc-stan.org

check_data *Check a visits data set for correct format*

Description

Raises specific errors when encountering issues in the data.

Usage

```
check_data(data, model)
```

Arguments

data	data.frame to check
model	srpmodel object used to fit data

Value

data.frame, same as input but all censoring events after terminal states are removed.

Examples

```
tbl <- data.frame(group_id = "A", subject_id = "A1", t = 0, state = "stable")
mdl <- create_srpmodel(A = define_srp_prior())
check_data(tbl, mdl)
```

compute_pfs *Compute progression-free-survival rate given sample*

Description

`compute_pfs()` computes the progression-free-survival rate at specified times given a paramter sample.

Usage

```
compute_pfs(
  model,
  t,
  parameter_sample = NULL,
  warmup = 500L,
  nsim = 1000L,
  seed = NULL,
  ...
)
```

Arguments

model	an object of class srpmodel containing prior information
t	a vector of time-points at which the PFS rate should be computed
parameter_sample	a stanfit object with samples from the respective model.
warmup	integer, number of warm-up samples for the MCMC sampler before retaining samples; see warmup parameter in rstan::stan().
nsim	integer, number of samples to draw
seed	integer, fixed random seed; NULL for no fixed seed
...	further arguments passed to method implementations

Value

a data frame with samples of PFS rates at each of the time points in the vector t.

Examples

```
mdl <- create_srpmodel(A = define_srp_prior())
smpl <- sample_prior(mdl, nsim = 500, seed = 34L)
dplyr::filter(
  compute_pfs(mdl, t = seq(0, 12), parameter_sample = smpl),
  iter == 1
)
```

---

impute *Sample visits from predictive distribution*

Description

impute() samples visits for individuals in data and potentially missing individuals up to a maximum of n_per_group from the posterior predictive distribution of the given model.

sample_predictive() draws samples from the predictive distribution of a model given a parameter sample.

Usage

```
impute(
  model,
  data,
  nsim,
  n_per_group = NULL,
  sample = NULL,
  p = NULL,
  shape = NULL,
```

```
  scale = NULL,
  now = NULL,
  seed = NULL,
  nsim_parameters = 1000L,
  warmup_parameters = 250L,
  nuts_control = list(),
  as_mstate = FALSE,
  ...
)

sample_predictive(
  model,
  nsim,
  n_per_group,
  sample = NULL,
  p = NULL,
  shape = NULL,
  scale = NULL,
  seed = NULL,
  nsim_parameters = 1000L,
  warmup_parameters = 250,
  nuts_control = list(),
  as_mstate = FALSE,
  ...
)
```

Arguments

model	an object of class srpmodel containing prior information
data	a data frame with variables subject_id<chr> (subject identifier), group_id<chr> (group identifier), t<dbl> (time of visit, relative to first visit in study), state<chr> (state recorded at visit). Allowed states are "stable", "response", "progression" (or death), and "EOF" (end of follow-up). The EOF state marks the end of an individual's follow-up before the absorbing state "progression".
nsim	integer, number of samples to draw
n_per_group	integer vector with number of individuals per group.
sample	a stanfit object with samples from the respective model.
p	numeric, vector of optional fixed response probabilities to use for sampling
shape	numeric, matrix of optional fixed Weibull shape parameters to use for sampling must be a matrix of dim c(n_groups, 3) where the second dimension corresponds to the transitions between s->r, s->p, r->p
scale	numeric, matrix of optional fixed Weibull scale parameters to use for sampling must be a matrix of dim c(n_groups, 3) where the second dimension corresponds to the transitions between s->r, s->p, r->p
now	numeric, time since first visit in data if not last recorded visit time
seed	integer, fixed random seed; NULL for no fixed seed

`nsim_parameters`
: integer, number of parameter samples

`warmup_parameters`
: integer, number of warmup samples for the rstan sampler before retaining samples of the parameters.

`nuts_control`
: list, parameters for NUTS algorithm see `control` argument in`rstan::stan()`

`as_mstate`
: logical, return data in mstate format?

`...`
: further arguments passed to method implementations

Value

a data frame with variables `subject_id<chr>` (subject identifier), `group_id<chr>` (group identifier), `t<dbl>` (time of visit, relative to first visit in study), `state<chr>` (state recorded at visit) `iter<int>` (re-sample indicator). Allowed states are "stable", "response", "progression" (or death), and "EOF" (end of follow-up). The EOF state marks the end of an individual's follow-up before the absorbing state "progression".

See Also

`sample_prior()` `sample_posterior()`

Examples

```
mdl <- create_srpmodel(A = define_srp_prior())
tbl <- tibble::tibble(
  subject_id = c("A1", "A1"),
  group_id = c("A", "A"),
  t = c(0, 1.5),
  state = c("stable", "stable")
)
impute(mdl, tbl, 1L, seed = 38L)

sample_predictive(mdl, 1L, 20L, seed = 38L)
```

`parameter_sample_to_tibble`
Convert parameter sample to data table

Description

`parameter_sample_to_tibble()` takes a rstan::stanfit parameter sample of a model, extracts the paramters values and returns them in a data frame.

Usage

```
parameter_sample_to_tibble(model, sample, ...)
```

Arguments

model	an object of class srpmodel containing prior information
sample	a stanfit object with samples from the respective model.
...	further arguments passed to method implementations

Value

a tibble with the sampled parameters, in long format

See Also

sample_prior() sample_posterior()

Examples

```
mdl <- create_srpmodel(A = define_srp_prior())
smpl <- sample_prior(mdl, seed = 3647L)
parameter_sample_to_tibble(mdl, smpl)
```

---

plot.srpmodel *Summary plot of model prior*

Description

Summary plot of model prior

Usage

```
## S3 method for class 'srpmodel'
plot(
  x,
  parameter_sample = NULL,
  seed = 42L,
  nsim = 500L,
  warmup = 250,
  nuts_control = list(),
  dt_interval = NULL,
  dt_n_grid = 25,
  dt_expand = 1.1,
  dt_grid = NULL,
  confidence = NULL,
  ...
)
```

Arguments

x	the model to plot
parameter_sample	a stanfit object with samples from the respective model.
seed	integer, fixed random seed; NULL for no fixed seed
nsim	integer, number of samples to draw
warmup	integer, number of warm-up samples for the MCMC sampler before retaining samples; see warmup parameter in rstan::stan().
nuts_control	list, parameters for NUTS algorithm see control argument inrstan::stan()
dt_interval	numeric vector of length two with minimal and maximal time (relative to individual first visit) to use for plotting
dt_n_grid	number of grid points to use when automatically choosing plotting interval
dt_expand	expansion factor for upper plotting limit when using automatic interval detection
dt_grid	numeric vector of time points to use for plotting
confidence	numeric in (0, 1) confidence level for point-wise confidence bands around mean; none plotted if NULL.
...	further arguments passed to method implementations

Value

A patchwork object, see patchwork::patchwork

See Also

plot_pfs() plot_transition_times() plot_response_probability()

Examples

```
## Not run:
mdl <- create_srpmodel(A = define_srp_prior())
plot(mdl)

## End(Not run)
```

---

plot_mstate *Swimmer plot of multi-state data*

Description

plot_mstate() plots data in 'multi-state-format' as swimmer plot.

Usage

```
plot_mstate(
  data,
  model,
  now = max(tbl_mstate$t_max),
  relative_to_sot = TRUE,
  ...
)
```

Arguments

data	a data frame with multi-state data; variables are subject_id<chr>, group_id<chr>, subject_id<chr>, from<chr>, to<chr>, t_min<dbl>, t_max<dbl>, t_sot<dbl>, where to and from indicate the state from which and into which the transitions occurs (stable, response, progression), t_max and t_min specify the interval in which the transition occurred relative to t_sot (start of treatment).
model	an object of class srpmodel containing prior information
now	the current time relative to the start of the trial
relative_to_sot	logical, should the timeline be relative to the start of trial or the start of treatment for each individual
...	further arguments passed to method implementations

Value

a ggplot2::ggplot object

See Also

visits_to_mstate()

Examples

```
mdl <- create_srpmodel(A = define_srp_prior())
tbl_visits <- sample_predictive(mdl, n_per_group = 5L, nsim = 1, seed = 468L)
tbl_mstate <- visits_to_mstate(tbl_visits, mdl)
plot_mstate(tbl_mstate, mdl)
```

---

plot_pfs *Plot progression-free-survival function*

Description

plot_pfs() plots the progression-free-survival function of a model.

Usage

```
plot_pfs(
  model,
  parameter_sample = NULL,
  seed = 42L,
  nsim = 500L,
  warmup = 250,
  nuts_control = list(),
  dt_interval = NULL,
  dt_n_grid = 25,
  dt_expand = 1.1,
  dt_grid = NULL,
  confidence = NULL,
  ...
)
```

Arguments

model	an object of class srpmodel containing prior information
parameter_sample	a stanfit object with samples from the respective model.
seed	integer, fixed random seed; NULL for no fixed seed
nsim	integer, number of samples to draw
warmup	integer, number of warm-up samples for the MCMC sampler before retaining samples; see warmup parameter in rstan::stan().
nuts_control	list, parameters for NUTS algorithm see control argument inrstan::stan()
dt_interval	numeric vector of length two with minimal and maximal time (relative to individual first visit) to use for plotting
dt_n_grid	number of grid points to use when automatically choosing plotting interval
dt_expand	expansion factor for upper plotting limit when using automatic interval detection
dt_grid	numeric vector of time points to use for plotting
confidence	numeric in (0, 1) confidence level for point-wise confidence bands around mean; none plotted if NULL.
...	further arguments passed to method implementations

Value

a ggplot2::ggplot object

See Also

plot_transition_times() plot_response_probability()

Examples

```
## Not run:
mdl <- create_srpmodel(A = define_srp_prior())
plot_pfs(mdl)

## End(Not run)
```

`plot_response_probability` *Plot the response probability distributions*

Description

`plot_response_probability()` plots the distribution over the response probability parameter in the specified model.

Usage

```
plot_response_probability(
  model,
  parameter_sample = NULL,
  seed = 42L,
  nsim = 500L,
  warmup = 250,
  nuts_control = list(),
  ...
)
```

Arguments

`model`	an object of class srpmodel containing prior information
`parameter_sample`	a stanfit object with samples from the respective model.
`seed`	integer, fixed random seed; NULL for no fixed seed
`nsim`	integer, number of samples to draw
`warmup`	integer, number of warm-up samples for the MCMC sampler before retaining samples; see `warmup` parameter in `rstan::stan()`.
`nuts_control`	list, parameters for NUTS algorithm see `control` argument in`rstan::stan()`
`...`	further arguments passed to method implementations

Value

a ggplot2::ggplot object

See Also

plot_transition_times() plot_pfs()

Examples

```
mdl <- create_srpmodel(A = define_srp_prior())
plot_response_probability(mdl)
```

---

plot_transition_times *Plot the transition times of a model*

Description

plot_transition_times() plots a the survival functions for the transition times in a multi-state model.

Usage

```
plot_transition_times(
  model,
  parameter_sample = NULL,
  seed = 42L,
  nsim = 500L,
  warmup = 250,
  nuts_control = list(),
  dt_interval = NULL,
  dt_n_grid = 25,
  dt_expand = 1.1,
  dt_grid = NULL,
  confidence = NULL,
  ...
)
```

Arguments

model	an object of class srpmodel containing prior information
parameter_sample	a stanfit object with samples from the respective model.
seed	integer, fixed random seed; NULL for no fixed seed
nsim	integer, number of samples to draw
warmup	integer, number of warm-up samples for the MCMC sampler before retaining samples; see warmup parameter in rstan::stan().
nuts_control	list, parameters for NUTS algorithm see control argument inrstan::stan()
dt_interval	numeric vector of length two with minimal and maximal time (relative to individual first visit) to use for plotting

dt_n_grid	number of grid points to use when automatically choosing plotting interval
dt_expand	expansion factor for upper plotting limit when using automatic interval detection
dt_grid	numeric vector of time points to use for plotting
confidence	numeric in (0, 1) confidence level for point-wise confidence bands around mean; none plotted if NULL.
...	further arguments passed to method implementations

Value

a ggplot2::ggplot object

See Also

plot_pfs() plot_response_probability()

Examples

```
## Not run:
mdl <- create_srpmodel(A = define_srp_prior())
plot_transition_times(mdl)

## End(Not run)
```

---

print.srpmodel *Print an srpmodel*

Description

Print an srpmodel

Usage

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

## S3 method for class 'srpmodel'
format(x, ...)
```

Arguments

x	model to print
...	further arguments passed to method implementations

Value

format() returns a character string representation of the object, print() prints to the console and returns the object itself invisibly.

Examples

```
print(create_srpmodel(A = define_srp_prior()))
format(create_srpmodel(A = define_srp_prior()))
```

`sample_posterior` *Sample parameters from a model*

Description

`sample_posterior()` draws samples from the posterior distribution of the specified model given a data set with visit data.

`sample_prior()` draws samples from the prior distribution of the specified model object.

Usage

```
sample_posterior(
  model,
  data,
  now = NULL,
  nsim = 2000L,
  seed = NULL,
  warmup = 500L,
  nuts_control = list(),
  acceptable_divergent_transition_fraction = 0.1,
  ...
)

sample_prior(
  model,
  nsim = 2000L,
  seed = NULL,
  warmup = 500L,
  nuts_control = list(),
  ...
)
```

Arguments

`model`	an object of class srpmodel containing prior information
`data`	a data frame with variables `subject_id<chr>` (subject identifier), `group_id<chr>` (group identifier), `t<dbl>` (time of visit, relative to first visit in study), `state<chr>` (state recorded at visit). Allowed states are "stable", "response", "progression" (or death), and "EOF" (end of follow-up). The EOF state marks the end of an individual's follow-up before the absorbing state "progression".
`now`	numeric, time from first visit in data if different form last recorded visit
`nsim`	integer, number of samples to draw

seed	integer, fixed random seed; NULL for no fixed seed
warmup	integer, number of warm-up samples for the MCMC sampler before retaining samples; see warmup parameter in rstan::stan().
nuts_control	list, parameters for NUTS algorithm see control argument inrstan::stan()
acceptable_divergent_transition_fraction,	numeric between 0 and 1 giving the acceptable fraction of divergent transitions before throwing an error
...	further arguments passed to method implementations

Value

A rstan::stanfit object with posterior samples.

See Also

rstan::stan() parameter_sample_to_tibble() sample_predictive() impute()

Examples

```
mdl <- create_srpmodel(A = define_srp_prior())
tbl <- tibble::tibble(
  subject_id = c("A1", "A1"),
  group_id = c("A", "A"),
  t = c(0, 1.5),
  state = c("stable", "response")
)
sample_posterior(mdl, tbl, seed = 42L)

sample_prior(mdl, seed = 42L)
```

---

simulate_decision_rule

Simulate results under a custom decision rule

Description

simulate_decision_rule() simulates from the prior or posterior predictive distribution of a model and applies a custom decision rule to each simulated data set.

Usage

```
simulate_decision_rule(
  model,
  n_per_group,
  decision_rule,
  data = NULL,
```

```
  parameter_sample = NULL,
  seed = NULL,
  nsim = 1L
)
```

Arguments

model	model to use for sampling
n_per_group	group size
decision_rule	a function with signature rule(mdl, data, ...) returning a data frame with results from a applying the decision rule to data setdata, typically contains a column group_id and a one column per decision/result.
data	a data frame with visit data to condition on
parameter_sample	an optional parameter sample to reuse
seed	optional fixed seed
nsim	the number of resamples to draw from the predictive distribution

Details

The sampling is implementing using `furrr::future_map()` and thus supports parallel execution when specifying a `future::plan()`.

Value

A data frame with columns `iter` (the resample index) and any columns returned by `decision_rule` applied to each of the `nsim` datasets sampled from the predictive distribution.

Examples

```
mdl <- create_srpmodel(A = define_srp_prior())
rule <- function(model, data) {
  tibble::tibble(decision = sample(c(0,1), 1))
}
simulate_decision_rule(mdl, 5, rule, nsim = 3)
```

---

srpmodel *A stable-response-progression model*

Description

`create_model()` takes one or more prior-specifications for an SRP multi-state model and combines them into a single model object. Groups are still treated as independent.

Usage

```
define_srp_prior(
  p_mean = 0.5,
  p_n = 3,
  p_eta = 0,
  p_min = 0,
  p_max = 1,
  median_t_q05 = c(1, 1, 1),
  median_t_q95 = c(60, 60, 60),
  shape_q05 = rep(0.9, 3),
  shape_q95 = rep(2.5, 3),
  visit_spacing = 1,
  recruitment_rate = 1
)

create_srpmodel(
  ...,
  maximal_time = 10 * 12,
  states = c("stable", "response", "progression"),
  censored = "EOF"
)
```

Arguments

p_mean	numeric, mean of the beta prior for the response probability
p_n	numeric, beta prior equivalent sample size (a + b)
p_eta	numeric, robustification parameter for beta prior; actual prior is (1 - eta) beta + eta; i.e., eta is the non-informative weight.
p_min	numeric, minimal response probability
p_max	numeric, maximal response probability
median_t_q05	numeric of length three, 5% quantiles of the log-normal distributions for the median time-to-next-event for the three transitions s->r, s->p, r->p.
median_t_q95	numeric of length three, 95% quantiles of the log-normal distributions for the median time-to-next-event for the three transitions s->r, s->p, r->p.
shape_q05	numeric of length three, 5% quantiles of the log-normal distributions for the shapes of the time-to-next-event distributions for the three transitions s->r, s->p, r->p.
shape_q95	numeric of length three, 95% quantiles of the log-normal distributions for the shapes of the time-to-next-event distributions for the three transitions s->r, s->p, r->p.
visit_spacing	numeric, fixed duration between visits
recruitment_rate	numeric, constant recruitment rate
...	named srp_prior objects; the argument names serve as group labels

maximal_time	the maximal overall runtime of the trial as measured from the first visit of any group. No visits past this point are sampled.
states	character vector of three states (initial, intermediate, terminal)
censored	string, indicator of premature censoring events; no data is imputed after this point.

Details

`define_srp_prior()` specifies a prior distribution for a three state model (stable, response, progression) for a single group.

Value

`define_srp_prior()` returns an object of class `srp_prior`, all inputs are accessible via `$x` where `x` is the name of the input argument in the function call except for the two parameters `visit_spacing` and `recruitment_rate`. These two parameters are saved as attributes and can be retrieved directly using `attr(mdl, "visit_spacing")` and `attr(mdl, "recruitment_rate")`.

`create_srpmodel()` returns an object of class `c("srpmodel", "list")` that holds information about potentially multiple groups in a compact format and can be accessed using the list operator `$name`. `group_id` is a character vector with the group names, `maximal_time` is the maximal follow-up time since the first visit in the study, `visit_spacing` is the vector of per-group difference between visits (only relevant for forward sampling), `recruitment_rate` is the vector of per-group recruitment rates, `stan_model` is the pre-compiled 'stan' model used for inference, `states` is the vector of state names in the multi-state model, and `prior` is a list of hyperparamters for the model prior with elements `p`, vector, for the response probability per group, `median_t` is an `c(n_groups, 3, 2)` dimensional array where `median_t[i,j,1]` holds the 5% quantile of the the lognormal prior on median transition time for group `i` and transition `j` and `median_t[i,j,2]` the corresponding upper 95% quantile. The `shape` hyperparamter has the same format and specified the corresponding quantiles for the Weibull shape parameter.

Examples

```
# a model with prior 25% response rate and variance equivalent to
# 10 data points (i.e. a Beta(2.5, 7.5) distribution).
grp <- define_srp_prior(p_mean = 0.25, p_n = 10)
attr(grp, "recruitment_rate")

# a model with two groups and different priors on the respective response
# probabilities
mdl <- create_srpmodel(
  A = define_srp_prior(),
  B = define_srp_prior(p_mean = 0.33, p_n = 10)
)
mdl$median_t
```

visits_to_mstate *Convert cross-sectional visit data to multi-state format*

Description

visits_to_mstate() converts visits to interval-censored multi-state data where each row corresponds to a transition between states. The conversion assumes that visit spacing is tight enough to not miss any transitions.

Usage

```
visits_to_mstate(tbl_visits, model, now = max(tbl_visits$t))
```

Arguments

tbl_visits	data frame, visit data in long format
model	an object of class srpmodel containing prior information
now	time point since start of trial (might be later than last recorded visit)

Value

A data frame with multi-state data; variables are subject_id<chr>, group_id<chr>, subject_id<chr>, from<chr>, to<chr>, t_min<dbl>, t_max<dbl>, t_sot<dbl>, where to and from indicate the state from which and into which the transitions occurs, t_max and t_min specify the interval in which the transition occurred relative to t_sot (start of treatment).

Examples

```
mdl <- create_srpmodel(A = define_srp_prior())
tbl_visits <- sample_predictive(mdl, n_per_group = 5L, nsim = 1, seed = 468L)
visits_to_mstate(tbl_visits, mdl)
```

Index