Package 'BayesFBHborrow'

May 6, 2026

Title Bayesian Dynamic Borrowing with Flexible Baseline Hazard Function

Version 2.0.2

Description Allows Bayesian borrowing from a historical dataset for time-to-event data. A flexible baseline hazard function is achieved via a piecewise exponential likelihood with time varying split points and smoothing prior on the historic baseline hazards. The method is described in Scott and Lewin (2024) <doi:10.48550/arXiv.2401.06082>, and the software paper is in Axillus et al. (2024) <doi:10.48550/arXiv.2408.04327>.

License Apache License (>= 2)

Encoding UTF-8

Author Darren Scott [aut, cre], Sophia Axillus [aut]

RoxygenNote 7.3.1

Suggests tibble, readxl, testthat (>= 3.0.0), rmarkdown, ggfortify, condSURV

Config/testthat/edition 3

Imports dplyr, stats, survival, invgamma, mvtnorm, checkmate, magrittr, ggplot2

Depends R (>= 4.1)

LazyData true

NeedsCompilation no

Maintainer Darren Scott <darren.scott@astrazeneca.com>

Repository CRAN

Date/Publication 2024-09-16 11:00:06 UTC

Contents

.beta.MH.RW.glm 3
.beta_MH_MALA 3
.beta_MH_NR 4

.beta_MH_RW . . . 5
.beta_mom . . . 5
.beta_mom.NR.fun . . . 6
.birth_move . . . 6
.dataframe_fun . . . 7
.death_move . . . 7
.glmFit . . . 8
.ICAR_calc . . . 8
.input_check . . . 9
.J_RJMCMC . . . 10
.J_RJMCMC_NoBorrow . . . 12
.lambda_0_MH_cp . . . 13
.lambda_0_MH_cp_NoBorrow . . . 14
.lambda_conj_prop . . . 15
.lambda_MH_cp . . . 16
.lgamma_ratio . . . 17
.llikelihood_ratio_beta . . . 18
.llikelihood_ratio_lambda . . . 18
.logsumexp . . . 19
.log_likelihood . . . 19
.lprop.dens.beta.NR . . . 20
.lprop_density_beta . . . 20
.ltau_dprior . . . 21
.mu_update . . . 21
.normalize_prob . . . 22
.nu_sigma_update . . . 22
.plot_hist . . . 23
.plot_matrix . . . 24
.plot_trace . . . 25
.predictive_hazard . . . 25
.predictive_hazard_ratio . . . 26
.predictive_survival . . . 26
.set_hyperparameters . . . 27
.set_tuning_parameters . . . 27
.shuffle_split_point_location . . . 28
.shuffle_split_point_location_NoBorrow . . . 29
.sigma2_update . . . 30
.smooth_hazard . . . 31
.smooth_survival . . . 31
.tau_update . . . 32
BayesFBHborrow . . . 33
BayesFBHborrow.NoBorrow . . . 34
BayesFBHborrow.WBorrow . . . 36
coef.BayesFBHborrow . . . 38
GibbsMH . . . 39
GibbsMH.NoBorrow . . . 41
GibbsMH.WBorrow . . . 43
group_summary . . . 45

init_lambda_hyperparameters 46
piecewise_exp_cc 47
piecewise_exp_hist 47
plot.BayesFBHborrow 48
summary.BayesFBHborrow 49
weibull_cc 50
weibull_hist 50

Index **52**

`.beta.MH.RW.glm` *Beta MH RW sampler from freq PEM fit*

Description

Sample beta from RW sampler

Usage

```
.beta.MH.RW.glm(df, beta, beta_count, cprop_beta)
```

Arguments

`df`	Data frame with indicators
`beta`	vector of parameters
`beta_count`	count number of accepted proposals
`cprop_beta`	proposal scalar

Value

beta, either old or new move

`.beta_MH_MALA` *Proposal beta with a Metropolis Adjusted Langevin (MALA)*

Description

Proposal beta with a Metropolis Adjusted Langevin (MALA)

Usage

```
.beta_MH_MALA(df, beta, bp, cprop_beta, beta_count)
```

Arguments

df	Data frame with indicators
beta	vector of parameters
bp	number of covariates
cprop_beta	proposal variance standard deviation
beta_count	count number of accepts

Value

updated beta vector

.beta_MH_NR *Newton Raphson MH move*

Description

Sample beta from RW sampler

Usage

```
.beta_MH_NR(df, beta, bp, cprop_beta, beta_count)
```

Arguments

df	Data frame with indicators
beta	vector of parameters
bp	number of covariates
cprop_beta	proposal scalar
beta_count	count number of accepts

Value

updated beta

.beta_MH_RW *Beta Metropolis-Hastings random walk move*

Description

Update beta via a Metropolis-Hastings Random Walk move

Usage

```
.beta_MH_RW(df, beta, bp, cprop_beta, beta_count)
```

Arguments

df	data.frame from dataframe_fun()
beta	beta values
bp	number of covariates
cprop_beta	hyperparameter for beta proposal standard deviation
beta_count	number of moves done for beta

Value

beta, either old or new move

.beta_mom *Mean for MALA using derivative for beta proposal*

Description

Mean for MALA using derivative for beta proposal

Usage

```
.beta_mom(df, k, beta, bp, cprop_beta)
```

Arguments

df	Data frame with indicators
k	index for beta
beta	vector of parameters
bp	number of covariates
cprop_beta	proposal standard dev

Value

proposal mean

.beta_mom.NR.fun *First and second derivative of target for mode and variance of proposal*

Description

First and second derivative of target for mode and variance of proposal

Usage

```
.beta_mom.NR.fun(df, k, beta, bp, cprop_beta)
```

Arguments

df	Data frame with indicators
k	index
beta	vector of parameters
bp	number of covariates
cprop_beta	proposal variance standard deviation

Value

First and second derivative mode and variance

.birth_move *Birth move in RJMCMC*

Description

Calculates new values of x when proposing another split point, based on a weighted mean, as x_new/x <- (1-U)/U

Usage

```
.birth_move(U, sj, s_star, sjm1, x, j)
```

Arguments

U	uniform random number
sj	upcoming split point location, j
s_star	new split point location, *
sjm1	previous split point location, j-1
x	vector of parameter values, length J + 1
j	split point

Value

vector with adjusted parameter values after additional split point, length J + 2

.dataframe_fun *Create data.frame for piecewise exponential models*

Description

Construct a split data.frame for updated split points

Usage

```
.dataframe_fun(Y, I, X, s, lambda, bp, J)
```

Arguments

Y	time-to-event
I	censor indicator
X	design Matrix
s	split point locations, including start and end (length J + 2)
lambda	baseline Hazards (length J+1)
bp	number of covariates
J	number of split points

Value

data.frame with columns c(tstart, id, X1,..., Xp, Y, I, lambda)

.death_move *Death move in RJMCMC*

Description

Calculates new values of x when proposing the death of a split point

Usage

```
.death_move(sjp1, sj, sjm1, x, j)
```

Arguments

sjp1	upcoming split point location, J + 1
sj	split point location to be removed, j
sjm1	previous split point location, j-1
x	vector of parameter values, length J + 1
j	split point

Value

vector with adjusted parameter values after removal of split point, length J

.glmFit *Fit frequentist piecewise exponential model for MLE and information matrix of beta*

Description

Compute MLE for PEM

Usage

```
.glmFit(df)
```

Arguments

df	Data frame with time-to-event, censoring indicator and covariates

Value

beta MLE and inverse of information matrix

.ICAR_calc *Calculate covariance matrix in the MVN-ICAR*

Description

Calculate covariance matrix in the MVN-ICAR

Usage

```
.ICAR_calc(s, J, clam)
```

Arguments

s	split points, J + 2
J	number of split points
clam	controls neighbor interactions, in range (0, 1)

Value

Sigma_s = (I - W)^(-1) * Q, W, Q

.input_check *Input checker*

Description

Checks inputs before Gibbs sampler is run

Usage

```
.input_check(
  Y,
  Y_0,
  X,
  X_0,
  tuning_parameters,
  initial_values = NULL,
  hyperparameters
)
```

Arguments

Y	current time-to-event data
Y_0	historical time-to-event data
X	design Matrix
X_0	design Matrix for historical data
tuning_parameters	list of tuning parameters
initial_values	list of initial values (optional)
hyperparameters	list of hyperparameters

Value

a print statement

.J_RJMCMC *RJMCMC (with Bayesian Borrowing)*

Description

Metropolis-Hastings Green Reversible Jump move, with Bayesian Borrowing

Usage

```
.J_RJMCMC(
  df_hist,
  df_curr,
  Y,
  Y_0,
  I,
  I_0,
  X,
  X_0,
  lambda,
  lambda_0,
  beta,
  beta_0,
  mu,
  sigma2,
  tau,
  s,
  J,
  Jmax,
  bp,
  bp_0,
  clam_smooth,
  a_tau = NULL,
  b_tau = NULL,
  c_tau = NULL,
  d_tau = NULL,
  type,
  p_0 = NULL,
  phi,
  pi_b,
  maxSj
)
```

Arguments

df_hist	data_frame containing historical data.
df_curr	data_frame containing current trial data.

Y	data.
Y_0	historical data.
I	censoring indicator.
I_0	historical trial censoring indicator.
X	design matrix.
X_0	historical trial design matrix.
lambda	baseline hazard.
lambda_0	historical trial baseline hazard.
beta	current trial parameters.
beta_0	historical trial parameters.
mu	prior mean for baseline hazard.
sigma2	prior variance hyperparameter for baseline hazard.
tau	borrowing parameter.
s	split point locations, J + 2.
J	number of split points.
Jmax	maximum number of split points.
bp	number of covariates in current trial.
bp_0	number of covariates in historical trial.
clam_smooth	neighbor interactions, in range (0, 1), for ICAR update.
a_tau	tau hyperparameter.
b_tau	tau hyperparameter.
c_tau	tau hyperparameter.
d_tau	tau hyperparameter.
type	choice of borrowing, "mix", "uni", or any other string for borrowing on every baseline hazard without mixture.
p_0	mixture ratio.
phi	J hyperparameter.
pi_b	probability of birth move.
maxSj	maximal time point, either current or historic.

Value

list of proposed J and s, with adjusted values of lambda, lambda_0, tau, Sigma_s, and data_frames for historical and current trial data.

`.J_RJMCMC_NoBorrow` *RJMCMC (without Bayesian Borrowing)*

Description

Metropolis-Hastings Green Reversible Jump move, without Bayesian Borrowing

Usage

```
.J_RJMCMC_NoBorrow(
  df,
  Y_0,
  I_0,
  X_0,
  lambda_0,
  beta_0,
  mu,
  sigma2,
  s,
  J,
  Jmax,
  bp_0,
  clam_smooth,
  phi,
  pi_b
)
```

Arguments

`df`	data_frame
`Y_0`	data
`I_0`	censoring indicator
`X_0`	design matrix
`lambda_0`	baseline hazard
`beta_0`	historical trial parameters
`mu`	prior mean for baseline hazard
`sigma2`	prior variance hyperparameter for baseline hazard
`s`	split point locations, J + 2
`J`	number of split points
`Jmax`	maximum number of split points
`bp_0`	number of covariates in historical trial
`clam_smooth`	neighbor interactions, in range (0, 1), for ICAR update
`phi`	J hyperparameter
`pi_b`	probability of birth move

Value

list of proposed J and s, with adjusted values of lambda, lambda_0, tau, Sigma_s, and data_frames for historical and current trial data

---

.lambda_0_MH_cp *Lambda_0 MH step, proposal from conditional conjugate posterior*

Description

Lambda_0 MH step, proposal from conditional conjugate posterior

Usage

```
.lambda_0_MH_cp(
  df_hist,
  Y_0,
  I_0,
  X_0 = NULL,
  s,
  beta_0 = NULL,
  mu,
  sigma2,
  lambda,
  lambda_0,
  tau,
  bp_0 = 0,
  J,
  clam,
  a_lam = 0.01,
  b_lam = 0.01,
  lambda_0_count = 0,
  lambda_0_move = 0
)
```

Arguments

df_hist	data.frame from dataframe_fun()
Y_0	historical trial data
I_0	historical trial censoring indicator
X_0	historical trial design matrix
s	split point locations, (J+2)
beta_0	parameter value for historical covariates
mu	prior mean for baseline hazard
sigma2	prior variance hyperparameter for baseline hazard

lambda	baseline hazard
lambda_0	historical baseline hazard
tau	borrowing parameter
bp_0	number of covariates, length(beta_0)
J	number of split points
clam	controls neighbor interactions, in range (0, 1)
a_lam	lambda hyperparameter, default is 0.01
b_lam	lambda hyperparameter, default is 0.01
lambda_0_count	number of total moves for lambda_0
lambda_0_move	number of accepted moves for lambda_0

Value

list of updated (if accepted) lambda_0 and data.frames, as well as the number of accepted moves

---

.lambda_0_MH_cp_NoBorrow

Lambda_0 MH step, proposal from conditional conjugate posterior

---

Description

Lambda_0 MH step, proposal from conditional conjugate posterior

Usage

```
.lambda_0_MH_cp_NoBorrow(
  df_hist,
  Y_0,
  I_0,
  X_0 = NULL,
  s,
  beta_0 = NULL,
  mu,
  sigma2,
  lambda_0,
  bp_0 = 0,
  J,
  clam,
  a_lam = 0.01,
  b_lam = 0.01,
  lambda_0_count = 0,
  lambda_0_move = 0
)
```

Arguments

df_hist	data.frame from dataframe_fun()
Y_0	historical trial data
I_0	historical trial censoring indicator
X_0	historical trial design matrix
s	split point locations, (J+2)
beta_0	parameter value for historical covariates
mu	prior mean for baseline hazard
sigma2	prior variance hyperparameter for baseline hazard
lambda_0	baseline hazard
bp_0	number of covariates, length(beta_0)
J	number of split points
clam	controls neighbor interactions, in range (0, 1)
a_lam	lambda hyperparameter, default is 0.01
b_lam	lambda hyperparameter, default is 0.01
lambda_0_count	number of total moves for lambda_0
lambda_0_move	number of accepted moves for lambda_0

Value

list of updated (if accepted) lambda_0 and data.frames, as well as the number of accepted moves

---

.lambda_conj_prop *Propose lambda from a gamma conditional conjugate posterior proposal*

Description

Propose lambda from a gamma conditional conjugate posterior proposal

Usage

```
.lambda_conj_prop(df, beta, j, bp, alam = 0.01, blam = 0.01)
```

Arguments

df	data.frame from dataframe_fun()
beta	parameter value for beta
j	current split point
bp	number of covariates
alam	lambda hyperparameter, default set to 0.01
blam	lambda hyperparameter, default set to 0.01

Value

list containing proposed lambda, shape and rate parameters

`.lambda_MH_cp` *Lambda MH step, proposal from conditional conjugate posterior*

Description

Lambda MH step, proposal from conditional conjugate posterior

Usage

```
.lambda_MH_cp(
  df_hist,
  df_curr,
  Y,
  I,
  X,
  s,
  beta,
  beta_0 = NULL,
  mu,
  sigma2,
  lambda,
  lambda_0,
  tau,
  bp,
  bp_0 = 0,
  J,
  a_lam = 0.01,
  b_lam = 0.01,
  lambda_move = 0,
  lambda_count = 0,
  alpha = 0.3
)
```

Arguments

`df_hist`	data.frame from dataframe_fun()
`df_curr`	data.frame from dataframe_fun()
`Y`	data
`I`	censoring indicator
`X`	design matrix
`s`	split point locations, J + 2
`beta`	parameter value for covariates

beta_0	parameter value for historical covariates
mu	prior mean for baseline hazard
sigma2	prior variance hyperparameter for baseline hazard
lambda	baseline hazard
lambda_0	historical baseline hazard
tau	borrowing parameter
bp	number of covariates, length(beta)
bp_0	number of covariates, length(beta_0)
J	number of split points
a_lam	lambda hyperparameter
b_lam	lambda hyperparameter
lambda_move	number of accepted lambda moves
lambda_count	total number of lambda moves
alpha	power parameter

Value

list of updated (if accepted) lambda and data.frames, as well as the number of accepted moves

---

.lgamma_ratio *Calculate log gamma ratio for two different parameter values*

Description

Calculate log gamma ratio for two different parameter values

Usage

```
.lgamma_ratio(x1, x2, shape, rate)
```

Arguments

x1	old parameter value
x2	proposed parameter value
shape	shape parameter
rate	rate parameter

Value

log gamma ratio

.llikelihood_ratio_beta

Loglikelihood ratio calculation for beta parameters

Description

Compute log likelihood for beta update

Usage

```
.llikelihood_ratio_beta(df, beta, beta_new)
```

Arguments

df	data.frame from dataframe_fun()
beta	beta values
beta_new	proposed beta values

Value

likelihood ratio

.llikelihood_ratio_lambda

Log likelihood for lambda / lambda_0 update

Description

Log likelihood for lambda / lambda_0 update

Usage

```
.llikelihood_ratio_lambda(df, df_prop, beta)
```

Arguments

df	data.frame from dataframe_fun()
df_prop	proposal data.frame
beta	parameter value for beta

Value

log likelihood ratio for lambda

.logsumexp *Computes the logarithmic sum of an exponential*

Description

Computes the logarithmic sum of an exponential

Usage

```
.logsumexp(x)
```

Arguments

x	set of log probabilities

Value

the logarithmic sum of an exponential

.log_likelihood *Log likelihood function*

Description

Log likelihood function

Usage

```
.log_likelihood(df, beta)
```

Arguments

df	data.frame containing data, time split points, and lambda
beta	coefficients for covariates

Value

log likelihood given lambdas and betas

.lprop.dens.beta.NR *log Gaussian proposal density for Newton Raphson proposal*

Description

log Gaussian proposal density for Newton Raphson proposal

Usage

```
.lprop.dens.beta.NR(beta.prop, mu_old, var_old)
```

Arguments

beta.prop	beta proposal
mu_old	density mean
var_old	density variance

Value

log Gaussian density

.lprop_density_beta *Log density of proposal for MALA*

Description

Log density of proposal for MALA

Usage

```
.lprop_density_beta(beta_prop, mu, cprop_beta)
```

Arguments

beta_prop	proposal beta
mu	mean of proposal distribution
cprop_beta	proposal standard dev

Value

log density

.ltau_dprior *Calculate log density tau prior*

Description

Calculate log density tau prior

Usage

```
.ltau_dprior(tau, a_tau, b_tau, c_tau = NULL, d_tau = NULL, p_0 = NULL, type)
```

Arguments

tau	current value(s) of tau
a_tau	tau hyperparameter
b_tau	tau hyperparameter
c_tau	tau hyperparameter
d_tau	tau hyperparameter
p_0	mixture ratio
type	choice of borrowing, "mix", "uni", or any other string for borrowing on every baseline hazard without mixture

Value

log density of tau

.mu_update *Calculate mu posterior update*

Description

Calculate mu posterior update

Usage

```
.mu_update(Sigma_s, lambda_0, sigma2, J)
```

Arguments

Sigma_s	VCV matrix (j + 1) x (j + 1).
lambda_0	Baseline hazard.
sigma2	Scale variance.
J	Number of split point.

Value

mu update from Normal.

`.normalize_prob` *Normalize a set of probability to one, using the the log-sum-exp trick*

Description

Normalize a set of probability to one, using the the log-sum-exp trick

Usage

```
.normalize_prob(x)
```

Arguments

`x`	set of log probabilities

Value

normalized set of log probabilities

`.nu_sigma_update` *Calculates nu and sigma2 for the Gaussian Markov random field prior, for a given split point j*

Description

Calculates nu and sigma2 for the Gaussian Markov random field prior, for a given split point j

Usage

```
.nu_sigma_update(j, lambda_0, mu, sigma2, W, Q, J)
```

Arguments

`j`	current split point
`lambda_0`	historical baseline hazard
`mu`	prior mean for baseline hazard
`sigma2`	prior variance hyperparameter for baseline hazard
`W`	influence from right and left neighbors
`Q`	individual effect of neighborhood
`J`	number of split points

Value

nu and sigma2

`.plot_hist` *Plot histogram from MCMC samples*

Description

Plots a histogram of the given discrete MCMC samples

Usage

```
.plot_hist(
  samples,
  title = "",
  xlab = "Values",
  ylab = "Frequency",
  color = "black",
  fill = "blue",
  binwidth = 0.05,
  scale_x = FALSE
)
```

Arguments

`samples`	data.frame containing the discrete MCMC samples
`title`	title of the plot, default is none
`xlab`	x-label of the plot, default is "Values"
`ylab`	y-label of the plot, default is "Frequency"
`color`	outline color for the bars, default is "black"
`fill`	fill color, default is "blue"
`binwidth`	width of the histogram bins, default is 0.5
`scale_x`	option to scale the x-axis, suitable for discrete samples, default is FALSE

Value

a ggplot2 object

`.plot_matrix` *Plot smoothed baseline hazards*

Description

Plot mean and given quantiles of a matrix. Can also be used to plot derivatives of the baseline hazard, such as estimated cumulative hazard and survival function.

Usage

```
.plot_matrix(
  x_lim,
  y,
  percentiles = c(0.05, 0.95),
  title = "",
  xlab = "",
  ylab = "",
  color = "blue",
  fill = "blue",
  linewidth = 1,
  alpha = 0.2,
  y2 = NULL,
  color2 = "red",
  fill2 = "red"
)
```

Arguments

`x_lim`	time grid
`y`	samples
`percentiles`	percentiles to include in plot, default is c(0.025, 0.975)
`title`	optional, add title to plot
`xlab`	optional, add xlabel
`ylab`	optional, add ylabel
`color`	color of the mid line, default is blue
`fill`	color of the percentiles, default is blue
`linewidth`	thickness of the plotted line, default is 1
`alpha`	opacity of the percentiles, default is 0.2
`y2`	(optional) second set of samples for comparison
`color2`	(optional) color of the mid line, default is red
`fill2`	(optional) color of the percentiles, default is red

Value

a ggplot2 object

.plot_trace *Plot MCMC trace*

Description

Creates a trace plot of given MCMC samples.

Usage

```
.plot_trace(
  x_lim,
  samples,
  title = "",
  xlab = "",
  ylab = "",
  color = "black",
  linewidth = 1
)
```

Arguments

x_lim	x-axis of the plot
samples	samples from MCMC
title	optional, add title to plot
xlab	optional, add xlabel
ylab	optional, add ylabel
color	color of the mid line, default is black
linewidth	thickness of the plotted line, default is 1

Value

a ggplot2 object

.predictive_hazard *Predictive hazard from BayesFBHborrow object*

Description

Predictive hazard from BayesFBHborrow object

Usage

```
.predictive_hazard(out_slam, x_pred, beta_samples)
```

Arguments

out_slam	samples from the smoothed baseline hazard
x_pred	set of predictors to be used for calculating the predictive hazard
beta_samples	samples of the covariates

Value

matrix of the predictive hazard

.predictive_hazard_ratio *Predictive hazard ratio (HR) from BayesFBHborrow object*

Description

Predictive hazard ratio (HR) from BayesFBHborrow object

Usage

```
.predictive_hazard_ratio(x_pred, beta_samples)
```

Arguments

x_pred	set of predictors to be used for calculating the predictive HR
beta_samples	samples of the covariates

Value

posterior samples for expectation and credible intervals

.predictive_survival *Predictive survival from BayesFBHborrow object*

Description

Predictive survival from BayesFBHborrow object

Usage

```
.predictive_survival(grid_width, out_slam, x_pred, beta_samples)
```

Arguments

grid_width	size of time step
out_slam	samples from the smoothed baseline hazard
x_pred	set of predictors to be used for calculating the predictive survival
beta_samples	samples of the covariates

Value

matrix of the predictive survival

.set_hyperparameters *Set tuning parameters*

Description

Set tuning parameters

Usage

```
.set_hyperparameters(hyperparameters = NULL, model_choice)
```

Arguments

hyperparameters	list of hyperparameters, could contain any combination of the listed hyperparameters
model_choice	choice of model, could be either of 'mix', 'uni' or 'all'

Value

filled list of tuning_parameters

.set_tuning_parameters *Set tuning parameters*

Description

Set tuning parameters

Usage

```
.set_tuning_parameters(tuning_parameters = NULL, borrow, X, X_0 = NULL)
```

Arguments

tuning_parameters	list of tuning_parameters, could contain any combination of the listed tuning parameters
borrow	choice of borrow, could be TRUE or FALSE
X	design matrix for concurrent trial
X_0	design matrix for historical trial

Value

filled list of tuning_parameters

.shuffle_split_point_location

Metropolis Hastings step: shuffle the split point locations (with Bayesian borrowing)

Description

Metropolis Hastings step: shuffle the split point locations (with Bayesian borrowing)

Usage

```
.shuffle_split_point_location(
  df_hist,
  df_curr,
  Y_0,
  I_0,
  X_0,
  lambda_0,
  beta_0,
  Y,
  I,
  X,
  lambda,
  beta,
  s,
  J,
  bp_0,
  bp,
  clam_smooth,
  maxSj
)
```

Arguments

df_hist	dataframe containing historical trial data and parmaeters
df_curr	data.frame containing current trial data and parameters
Y_0	historical trial data
I_0	historical trial censoring indicator
X_0	historical trial design matrix
lambda_0	historical baseline hazard
beta_0	historical parameter vector
Y	data
I	censoring indicator
X	design matrix
lambda	baseline hazard
beta	parameter vector
s	split point locations, J + 2
J	number of split points
bp_0	number of covariates in historical trial
bp	number of covariates in current trial
clam_smooth	neighbor interactions, in range (0, 1), for ICAR update
maxSj	the smallest of the maximal time points, min(max(Y), max(Y_0))

Value

list containing new split points, updated Sigma_s and data.frames for historic and current trial data

---

.shuffle_split_point_location_NoBorrow

Metropolis Hastings step: shuffle the split point locations (without Bayesian borrowing)

Description

Metropolis Hastings step: shuffle the split point locations (without Bayesian borrowing)

Usage

```
.shuffle_split_point_location_NoBorrow(
  df,
  Y_0,
  I_0,
  X_0,
  lambda_0,
```

```
  beta_0,
  s,
  J,
  bp_0,
  clam_smooth
)
```

Arguments

df	dataframe containing trial data and parameters
Y_0	data
I_0	censoring indicator
X_0	design matrix
lambda_0	baseline hazard
beta_0	parameter vector
s	split point locations, J + 2
J	number of split points
bp_0	number of covariates in historical trial
clam_smooth	neighbor interactions, in range (0, 1), for ICAR update

Value

list containing new split points, updated Sigma_s and data.frames for historic and current trial data

---

.sigma2_update *Calculate sigma2 posterior update*

Description

Calculate sigma2 posterior update

Usage

```
.sigma2_update(mu, lambda_0, Sigma_s, J, a_sigma, b_sigma)
```

Arguments

mu	mean.
lambda_0	Baseline hazard.
Sigma_s	VCV matrix (j + 1) x (j + 1).
J	Number of split point.
a_sigma	Hyperparameter a.
b_sigma	Hyperparameter b.

Value

sigma2 draw from IG

.smooth_hazard *Smoothed hazard function*

Description

Smoothed hazard function

Usage

```
.smooth_hazard(out_slam, beta_samples = NULL)
```

Arguments

out_slam	samples from GibbsMH of the baseline hazard
beta_samples	samples from GibbsMH from the treatment effect

Value

smoothed function for the baseline hazard

.smooth_survival *Smoothed survival curve*

Description

Smoothed survival curve

Usage

```
.smooth_survival(grid_width, out_slam, beta_samples = NULL)
```

Arguments

grid_width	step size
out_slam	samples from GibbsMH of the baseline hazard
beta_samples	samples from GibbsMH from the treatment effect

Value

smoothed survival function

`.tau_update` *Sample tau from posterior distribution*

Description

Sample tau from posterior distribution

Usage

```
.tau_update(
  lambda_0,
  lambda,
  J,
  s,
  a_tau,
  b_tau,
  c_tau = NULL,
  d_tau = NULL,
  p_0 = NULL,
  type
)
```

Arguments

`lambda_0`	historical baseline hazard
`lambda`	baseline hazard
`J`	number of split points
`s`	split point locations, J + 2
`a_tau`	Inverse Gamma hyperparameter
`b_tau`	Inverse Gamma hyperparameter
`c_tau`	Inverse Gamma hyperparameter
`d_tau`	Inverse Gamma hyperparameter
`p_0`	mixture ratio
`type`	choice of borrowing, "mix", "uni", or any other string for borrowing on every baseline hazard without mixture

Value

list containing tau and new mixture ratio

BayesFBHborrow *BayesFBHborrow: Run MCMC for a piecewise exponential model*

Description

Main function of the BayesFBHborrow package. This generic function calls the correct MCMC sampler for time-to-event Bayesian borrowing.

Usage

```
BayesFBHborrow(
  data,
  data_hist = NULL,
  borrow = TRUE,
  model_choice,
  tuning_parameters,
  hyperparameters,
  lambda_hyperparameters,
  iter,
  warmup_iter,
  refresh,
  verbose,
  max_grid
)
```

Arguments

data	data.frame containing atleast three vectors of "tte" (time-to-event) and "event" (censoring), and covariates "X_i" (where i should be a number/ indicator of the covariate)
data_hist	data.frame containing atleast two vectors of "tte" (time-to-event) and "event" (censoring), with the option of adding covariates named "X_0_i" (where i should be a number/indicator of the covariate), for historical data
borrow	TRUE (default), will run the model with borrowing
model_choice	choice of which borrowing model to use out of "mix", "uni" or "all"
tuning_parameters	list of "cprop_beta" ("cprop_beta_0" for historical data), "alpha", "Jmax", and "pi_b". Default is list("Jmax" = 5, "clam_smooth" = 0.8, "cprop_beta" = 0.5, "cprop_beta_0" = 0.5, "pi_b" = 0.5, "alpha" = 0.4)
hyperparameters	list containing the hyperparameters ("a_tau", "b_tau", "c_tau", "d_tau","type", "p_0", "a_sigma", "b_sigma"). Default is list("a_tau" = 1, "b_tau" = 1,"c_tau" = 1, "d_tau" = 0.001, "type" = "mix", "p_0" = 0.5, "a_sigma" = 2, "b_sigma" = 2, "phi" = 3)

lambda_hyperparameters	contains two hyperparameters (a_lambda and b_lambda) used for the update of lambda and lambda_0. Default is c(0.01, 0.01)
iter	number of iterations for MCMC sampler
warmup_iter	number of warmup iterations (burn-in) for MCMC sampler.
refresh	number of iterations between printed screen updates
verbose	FALSE (default), choice of output, if TRUE will output intermittent results into console
max_grid	grid size for the smoothed baseline hazard

Value

a nested list of two items, 'out' and 'plots'. The list 'out' will contain all the samples of the MCMC chain, as well as acceptance ratios. The latter, 'plots', contains plots (and data) of the smoothed baseline hazard, smoothed survival, a histogram of the sampled number of split points, and the trace plot of the treatment effect beta_1

Examples

```
set.seed(123)
# Load the example data
data(piecewise_exp_cc, package = "BayesFBHborrow")
data(piecewise_exp_hist, package = "BayesFBHborrow")

# Set your tuning parameters
tuning_parameters <- list("Jmax" = 5,
                          "pi_b" = 0.5,
                          "cprop_beta" = 3.25,
                          "alpha" = 0.4)

# Set hyperparameters to default, with the borrowing model "mix"
out <- BayesFBHborrow(data = piecewise_exp_cc, data_hist = piecewise_exp_hist,
                      model_choice = 'mix', tuning_parameters = tuning_parameters,
                      iter = 2, warmup_iter = 0)

# Create a summary of the output
summary(out$out, estimator = "out_fixed")

# Plot the predictive curves for the treatment group
plots <- plot(out$out, out$out$time_grid, x_pred = c(1))
```

BayesFBHborrow.NoBorrow

Run the MCMC sampler without Bayesian Borrowing

Description

Main function of the BayesFBHborrow package. This generic function calls the correct MCMC sampler for time-to-event without Bayesian borrowing.

Usage

```
## S3 method for class 'NoBorrow'
BayesFBHborrow(
  data,
  data_hist = NULL,
  borrow = FALSE,
  model_choice = "no_borrow",
  tuning_parameters = NULL,
  hyperparameters = NULL,
  lambda_hyperparameters = list(a_lambda = 0.01, b_lambda = 0.01),
  iter = 2000,
  warmup_iter = 2000,
  refresh = 0,
  verbose = FALSE,
  max_grid = 2000
)
```

Arguments

data	data.frame containing atleast three vectors of "tte" (time-to-event) and "event" (event indicator), and covariates "X_i" (where i should be a number/ indicator of the covariate)
data_hist	NULL (not used)
borrow	FALSE (default), will run the model with borrowing
model_choice	'no_borrow' (default), for no borrowing
tuning_parameters	list of "cprop_beta", "Jmax", and "pi_b". Default is ("Jmax" = 5, "cprop_beta" = 0.5, "pi_b" = 0.5)
hyperparameters	list containing the hyperparameters c("a_sigma", "b_sigma", "phi", clam_smooth"). Default is list("a_sigma" = 2, "b_sigma" = 2, "phi" = 3 , "clam_smooth" = 0.8)
lambda_hyperparameters	contains two hyperparameters ("a_lambda" and "b_lambda") used for the update of lambda, default is c(0.01, 0.01)
iter	number of iterations for MCMC sampler. Default is 2000
warmup_iter	number of warmup iterations (burn-in) for MCMC sampler. Default is 2000
refresh	number of iterations between printed console updates. Default is 0
verbose	FALSE (default), choice of output, if TRUE will output intermittent results into console
max_grid	grid size for the smoothed baseline hazard. Default is 2000

Value

a nested list of two items, 'out' and 'plots'. The list 'out' will contain all the samples of the MCMC chain, as well as acceptance ratios. The latter, 'plots', contains plots (and data) of the smoothed baseline hazard, smoothed survival, a histogram of the sampled number of split points, and the trace plot of the treatment effect beta_1

Examples

```
set.seed(123)
# Load the example data
data(piecewise_exp_cc, package = "BayesFBHborrow")

# Set your tuning parameters
tuning_parameters <- list("Jmax" = 5,
                          "cprop_beta" = 3.25)

# Set initial values to default
out <- BayesFBHborrow(piecewise_exp_cc, NULL, borrow = FALSE,
                      tuning_parameters = tuning_parameters,
                      iter = 2, warmup_iter = 0)
```

BayesFBHborrow.WBorrow

Run the MCMC sampler with Bayesian Borrowing

Description

Main function of the BayesFBHborrow package. This generic function calls the correct MCMC sampler for time-to-event Bayesian borrowing.

Usage

```
## S3 method for class 'WBorrow'
BayesFBHborrow(
  data,
  data_hist,
  borrow = TRUE,
  model_choice = "mix",
  tuning_parameters = NULL,
  hyperparameters = NULL,
  lambda_hyperparameters = list(a_lambda = 0.01, b_lambda = 0.01),
  iter = 2000,
  warmup_iter = 2000,
  refresh = 0,
  verbose = FALSE,
  max_grid = 2000
)
```

Arguments

data	data.frame containing atleast three vectors called "tte" (time-to-event), "event" (censoring), and covariates "X_i" (where i should be a number/indicator of the covariate)

data_hist	data.frame containing atleast two vectors called "tte" (time-to-event) and "event" (censoring), with the option of adding covariates named "X_0_i" (where i should be a number/indicator of the covariate) for the historical data
borrow	TRUE (default), will run the model with borrowing
model_choice	choice of which borrowing model to use out of 'mix', 'uni' or 'all'
tuning_parameters	list of "cprop_beta" ("cprop_beta_0" for historical data), "alpha", "Jmax", and "pi_b". Default is list("Jmax" = 5, "clam_smooth" = 0.8, "cprop_beta" = 0.5, cprop_beta_0" = 0.5, "pi_b" = 0.5, "alpha" = 0.4)
hyperparameters	list containing the hyperparameters ("a_tau", "b_tau", "c_tau", "d_tau","type", "p_0", "a_sigma", "b_sigma"). Default is list("a_tau" = 1, "b_tau" = 1,"c_tau" = 1, "d_tau" = 0.001, "type" = "mix", "p_0" = 0.5, "a_sigma" = 2, "b_sigma" = 2, "phi" = 3)
lambda_hyperparameters	contains three hyperparameters (a_lambda, b_lambda) used for the update of lambda and lambda_0. Default is c(0.01, 0.01)
iter	number of iterations for MCMC sampler. Default is 2000
warmup_iter	number of warmup iterations (burn-in) for MCMC sampler. Default is 2000
refresh	number of iterations between printed console updates. Default is 0
verbose	FALSE (default), choice of output, if TRUE will output intermittent results into console
max_grid	grid size for the smoothed baseline hazard. Default is 2000

Value

a nested list of two items, 'out' and 'plots'. The list 'out' will contain all the samples of the MCMC chain, as well as acceptance ratios. The latter, 'plots', contains plots (and data) of the smoothed baseline hazard, smoothed survival, a histogram of the sampled number of split points, and the trace plot of the treatment effect beta_1

Examples

```
set.seed(123)
# Load the example data
data(piecewise_exp_cc, package = "BayesFBHborrow")
data(piecewise_exp_hist, package = "BayesFBHborrow")

# Set your tuning parameters
tuning_parameters <- list("Jmax" = 5,
                          "pi_b" = 0.5,
                          "cprop_beta" = 3.25,
                          "alpha" = 0.4)

# Set hyperparameters to default, with the borrowing model "mix"
out <- BayesFBHborrow(data = piecewise_exp_cc, data_hist = piecewise_exp_hist,
                      model_choice = 'mix', tuning_parameters = tuning_parameters,
```

```
                      iter = 2, warmup_iter = 0)

# Create a summary of the output
summary(out$out, estimator = "out_fixed")

# Plot the predictive curves for the treatment group
plots <- plot(out$out, out$out$time_grid, x_pred = c(1))
```

`coef.BayesFBHborrow` *Extract mean posterior values*

Description

S3 method for class "BayesFBHborrow", returns the mean posterior values for the fixed parameters

Usage

```
## S3 method for class 'BayesFBHborrow'
coef(object, ...)
```

Arguments

`object`	MCMC sample object from BayesFBHborrow()
`...`	other arguments, see coef.default()

Value

mean values of given samples

Examples

```
data(weibull_cc, package = "BayesFBHborrow")

# Set your tuning parameters
tuning_parameters <- list("Jmax" = 5,
                          "pi_b" = 0.5,
                          "cprop_beta" = 0.5)

# run the MCMC sampler
out <- BayesFBHborrow(weibull_cc, NULL, tuning_parameters = tuning_parameters,
                      iter = 3, warmup_iter = 1)

# Plot the posterior mean values of the fixed parameters
coef(out$out)
```

GibbsMH *S3 generic, calls the correct GibbsMH sampler*

Description

An MCMC sampler for Bayesian borrowing with time-to-event data. We obtain a flexible baseline hazard function by making the split points random within a piecewise exponential model and using a Gaussian Markov random field prior to smooth the baseline hazards. Only calls the sampler and does not run any input checks. Best practice is to call BayesFBHborrow(), if the user is not familiar with the model at hand.

Usage

```
GibbsMH(
  Y,
  I,
  X,
  Y_0 = NULL,
  I_0 = NULL,
  X_0 = NULL,
  tuning_parameters,
  hyperparameters,
  lambda_hyperparameters,
  iter,
  warmup_iter,
  refresh,
  max_grid
)
```

Arguments

Y	data
I	event indicator
X	design matrix
Y_0	historical data, default is NULL
I_0	historical event indicator, default is NULL
X_0	historical design matrix, default is NULL
tuning_parameters	list of "cprop_beta", "cprop_beta_0", "alpha", "Jmax", and "pi_b"
hyperparameters	list containing the hyperparameters c("a_tau", "b_tau", "c_tau", "d_tau","type", "p_0", "a_sigma", "b_sigma", "Jmax", "clam_smooth", "cprop_beta", "phi", "pi_b"). Default is list("a_tau" = 1,"b_tau" = 1,"c_tau" = 1, "d_tau" = 0.001, "type" = "mix", "p_0" = 0.5, "a_sigma" = 2, "b_sigma" = 2, "Jmax" = 20, "clam_smooth" = 0.8, "cprop_beta" = 0.5, "phi" = 3, "pi_b" = 0.5)

lambda_hyperparameters
: contains two hyperparameters (a_lambda and b_lambda) used for the update of lambda and lambda_0

iter
: number of iterations for MCMC sampler, excluding warmup, default is 2000

warmup_iter
: number of warmup iterations (burn-in) for MCMC sampler, default is 2000

refresh
: number of iterations between printed screen updates, default is 500

max_grid
: grid size for the smoothed baseline hazard, default is 2000

Value

depending on if the user wishes to borrow; returns a list with values after each iteration for parameters: out_fixed (J, mu, sigma2, beta), lambda, lambda_0, tau, s, as well as tuning values of the total number of accepts: lambda_move, lambda_0_move and beta_move. Also included is the out_slam which contains the shrunk estimate of the baseline hazard.

Examples

```
set.seed(123)
# Load example data and set your initial values and hyper parameters
data(weibull_cc, package = "BayesFBHborrow")
data(weibull_hist, package = "BayesFBHborrow")

# The datasets consists of 3 (2) columns named "tte", "event" and "X"
# (only for concurrent). To explicitly run the sampler, extract the samples as
# following
Y <- weibull_cc$tte
I <- weibull_cc$event
X <- matrix(weibull_cc$X_trt)

Y_0 <- weibull_hist$tte
I_0 <- weibull_hist$event
X_0 <- NULL

# Specify hyperparameters and tuning parameters
hyper <-  list("a_tau" = 1,
               "b_tau" = 0.001,
               "c_tau" = 1,
               "d_tau" = 1,
               "type" = 'all',
               "p_0" = 0.5,
               "a_sigma" = 2,
               "b_sigma" = 2,
               "clam_smooth" = 0.5,
               "phi" = 3)

tuning_parameters <- list("Jmax" = 5,
                          "pi_b" = 0.5,
                          "cprop_beta" = 0.5,
                          "alpha" = 0.4)

output <- GibbsMH(Y, I, X, Y_0, I_0, X_0,
```

```
                    tuning_parameters, hyper,
                    iter = 5, warmup_iter = 1)
```

GibbsMH.NoBorrow *GibbsMH sampler, without Bayesian Borrowing*

Description

An MCMC sampler for time-to-event data, without Bayesian Borrowing. We obtain a flexible baseline hazard function by making the split points random within a piecewise exponential model and using a Gaussian Markov random field prior to smooth the baseline hazards. Only calls the sampler and does not run any input checks. Best practice is to call BayesFBHborrow(), if the user is not familiar with the model at hand.

Usage

```
## S3 method for class 'NoBorrow'
GibbsMH(
  Y,
  I,
  X = NULL,
  Y_0 = NULL,
  I_0 = NULL,
  X_0 = NULL,
  tuning_parameters,
  hyperparameters = list(a_sigma = 1, b_sigma = 1, phi = 3, clam_smooth = 0.8),
  lambda_hyperparameters = list(a_lambda = 0.01, b_lambda = 0.01),
  iter = 1500L,
  warmup_iter = 10L,
  refresh = 0,
  max_grid = 2000L
)
```

Arguments

Y	data
I	event indicator
X	design matrix
Y_0	historical data, default is NULL
I_0	historical event indicator, default is NULL
X_0	historical design matrix, default is NULL
tuning_parameters	list of "cprop_beta", "Jmax", and "pi_b"

`hyperparameters`
: list containing the hyperparameters c("a_sigma", "b_sigma", "Jmax", "clam_smooth", "cprop_beta", "phi"). Default is list("a_sigma" = 2, "b_sigma" = 2, "Jmax" = 20, "clam_smooth" = 0.8, "cprop_beta" = 0.5, "phi" = 3)

`lambda_hyperparameters`
: contains two hyperparameters ("a" and "b") used for the update of lambda, default is c(0.01, 0.01)

`iter`
: number of iterations for MCMC sampler, excluding warmup, default is 2000

`warmup_iter`
: number of warmup iterations (burn-in) for MCMC sampler, default is 2000

`refresh`
: number of iterations between printed screen updates, default is 500

`max_grid`
: grid size for the smoothed baseline hazard, default is 2000

Value

list with values after each iteration for parameters: out_fixed (J, mu, sigma2, beta), lambda, s, as well as tuning values of the total number of accepts: lambda_move and beta_move. Also included is the out_slam which contains the shrunk estimate of the baseline hazard.

Examples

```
set.seed(123)
# Load example data and set your hyper parameters
data(weibull_cc, package = "BayesFBHborrow")
data(weibull_hist, package = "BayesFBHborrow")

# The datasets consists of 3 (2) columns named "tte", "event" and "X".
# To explicitly run the sampler, extract the samples as following
Y <- weibull_cc$tte
I <- weibull_cc$event
X <- matrix(weibull_cc$X_trt)

# Specify hyperparameters and tuning parameters
hyper <-  list("a_sigma" = 2,
               "b_sigma" = 2,
               "clam_smooth" = 0.5,
               "phi" = 3)

tuning_parameters <- list("Jmax" = 5,
                          "pi_b" = 0.5,
                          "cprop_beta" = 0.5)

# Set initial values to 'NULL' for default settings
output <- GibbsMH(Y, I, X, NULL, NULL, NULL,
                  tuning_parameters = tuning_parameters, hyperparameters = hyper,
                  iter = 5, warmup_iter = 1)
```

GibbsMH.WBorrow *GibbsMH sampler, with Bayesian Borrowing*

Description

An MCMC sampler for Bayesian borrowing with time-to-event data. We obtain a flexible baseline hazard function by making the split points random within a piecewise exponential model and using a Gaussian Markov random field prior to smooth the baseline hazards. Only calls the sampler and does not run any input checks. Best practice is to call BayesFBHborrow(), if the user is not familiar with the model at hand.

Usage

```
## S3 method for class 'WBorrow'
GibbsMH(
  Y,
  I,
  X,
  Y_0,
  I_0,
  X_0,
  tuning_parameters = NULL,
 hyperparameters = list(a_tau = 1, b_tau = 0.001, c_tau = 1, d_tau = 1, type = "mix",
    p_0 = 0.8, a_sigma = 1, b_sigma = 1, phi = 3, clam_smooth = 0.8),
  lambda_hyperparameters = list(a_lambda = 0.01, b_lambda = 0.01),
  iter = 150L,
  warmup_iter = 10L,
  refresh = 0,
  max_grid = 2000L
)
```

Arguments

Y	data
I	event indicator
X	design matrix
Y_0	historical data
I_0	historical event indicator
X_0	historical design matrix
tuning_parameters	list of "cprop_beta", "cprop_beta_0", "alpha", "Jmax", and "pi_b"
hyperparameters	list containing the hyperparameters c("a_tau", "b_tau", "c_tau", "d_tau","type", "p_0", "a_sigma", "b_sigma", "Jmax", "clam_smooth", "cprop_beta", "phi", "pi_b"). Default is list("a_tau" = 1,"b_tau" = 1,"c_tau" = 1, "d_tau" = 0.001,

"type" = "mix", "p_0" = 0.5, "a_sigma" = 2, "b_sigma" = 2, "Jmax" = 20, "clam_smooth" = 0.8, "cprop_beta" = 0.5, "phi" = 3, "pi_b" = 0.5)

`lambda_hyperparameters`
: contains two hyperparameters (a_lambda and b_lambda) used for the update of lambda and lambda_0. Default is c(0.01, 0.01)

`iter`
: number of iterations for MCMC sampler, excluding warmup, default is 2000

`warmup_iter`
: number of warmup iterations (burn-in) for MCMC sampler, default is 2000

`refresh`
: number of iterations between printed screen updates, default is 500

`max_grid`
: grid size for the smoothed baseline hazard, default is 2000

Value

list with values after each iteration for parameters: out_fixed (J, mu, sigma2, beta), lambda, lambda_0, tau, s, as well as tuning values of the total number of accepts: lambda_move, lambda_0_move and beta_move. Also included is the out_slam which contains the shrunk estimate of the baseline hazard.

Examples

```
set.seed(123)
# Load example data and set your initial values and hyper parameters
data(weibull_cc, package = "BayesFBHborrow")
data(weibull_hist, package = "BayesFBHborrow")

# The datasets consists of 3 (2) columns named "tte", "event" and "X"
# (only for concurrent). To explicitly run the sampler, extract the samples as
# following
Y <- weibull_cc$tte
I <- weibull_cc$event
X <- matrix(weibull_cc$X_trt)

Y_0 <- weibull_hist$tte
I_0 <- weibull_hist$event
X_0 <- NULL

# Specify hyperparameters and tuning parameters
hyper <-  list("a_tau" = 1,
               "b_tau" = 0.001,
               "c_tau" = 1,
               "d_tau" = 1,
               "type" = "all",
               "p_0" = 0.5,
               "a_sigma" = 2,
               "b_sigma" = 2,
               "clam_smooth" = 0.5,
               "phi" = 3)

tuning_parameters <- list("Jmax" = 5,
                          "pi_b" = 0.5,
                          "cprop_beta" = 0.5,
```

```
                          "alpha" = 0.4)

output <- GibbsMH(Y, I, X, Y_0, I_0, X_0, tuning_parameters = tuning_parameters,
                  hyperparameters = hyper, iter = 5, warmup_iter = 1)
```

---

group_summary *Create group level data*

Description

Aggregate individual level data into group level data

Usage

```
group_summary(Y, I, X, s)
```

Arguments

Y	data
I	censoring indicator
X	design matrix
s	split points, J + 2

Value

list of group level data

Examples

```
set.seed(111)
# Load example data and set your initial values and hyper parameters
data(weibull_cc, package = "BayesFBHborrow")
data(weibull_hist, package = "BayesFBHborrow")

Y <- weibull_cc$tte
I <- weibull_cc$event
X <- weibull_cc$X_trt

# Say we want to know the group level data for the following split points
s <- quantile(Y, c(0, 0.45, 0.65, 1), names = FALSE)

group_summary(Y, I, X, s)
```

`init_lambda_hyperparameters` *Initialize lambda hyperparameters*

Description

Propose lambda hyperparameters for the choice of initial values for lambda

Usage

```
init_lambda_hyperparameters(group_data, s, w = 0.5)
```

Arguments

`group_data`	group level data
`s`	split points
`w`	weight

Value

shape and rate for the estimated lambda distribution

Examples

```
set.seed(111)
# Load example data and set your initial values and hyper parameters
data(weibull_cc, package = "BayesFBHborrow")
data(weibull_hist, package = "BayesFBHborrow")

Y <- weibull_cc$tte
I <- weibull_cc$event
X <- weibull_cc$X_trt

# Say we want to know the group level data for the following split points
s <- quantile(Y, c(0, 0.45, 0.65, 1), names = FALSE)

group_data <- group_summary(Y, I, NULL, s)
init_lambda_hyperparameters(group_data, s)
```

piecewise_exp_cc *Example data, simulated from a piecewise exponential model.*

Description

Data is simulated for a concurrent trial with three columns named "tte" (time-to-event), "event" (event indicator), and "X_trt" (treatment indicator). It was simulated using the following parameters:

Usage

```
data(piecewise_exp_cc)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 250 rows and 3 columns.

Examples

```
data(piecewise_exp_cc)
survival_model <- survival::survfit(survival::Surv(tte, event) ~ X_trt, data = piecewise_exp_cc)
line_colors <- c("blue", "red")  # Adjust colors as needed
line_types <- 1:length(unique(piecewise_exp_cc$X_trt))
plot(survival_model, col = line_colors, lty = line_types,
     xlab = "Time (tte)", ylab = "Survival Probability",
     main = "Kaplan-Meier Survival Curves by Treatment")
```

piecewise_exp_hist *Example data, simulated from a piecewise exponential model.*

Description

Data is simulated for a historical trial with two columns named "tte" (time-to-event) and "event" (event indicator). It was simulated using the following parameters:

Usage

```
data(piecewise_exp_hist)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 100 rows and 2 columns.

Examples

```
data(piecewise_exp_cc)
data(piecewise_exp_hist)
piecewise_exp_hist$X_trt <- 0
survival_model <- survival::survfit(survival::Surv(tte, event) ~ X_trt,
                                    data = rbind(piecewise_exp_cc,
                                    piecewise_exp_hist))
line_colors <- c("blue", "red", "green")  # Adjust colors as needed
line_types <- 1:length(unique(piecewise_exp_cc$X_trt))
plot(survival_model, col = line_colors, lty = line_types,
     xlab = "Time (tte)", ylab = "Survival Probability",
     main = "Kaplan-Meier Survival Curves by Treatment")
```

---

plot.BayesFBHborrow *Plot the MCMC results*

Description

S3 object which produces predictive probabilities of the survival, hazard, and hazard ratio for a given set of predictors

Usage

```
## S3 method for class 'BayesFBHborrow'
plot(x, x_lim, x_pred = NULL, ...)
```

Arguments

x	object of class "BayesFBHborrow" to be visualized
x_lim	x-axis to be used for plot, set to NULL to use default from MCMC sampling
x_pred	vector of chosen predictors
...	other plotting arguments, see .plot_matrix() for more information

Value

nested list of 'plots' (posterior predictive hazard, survival, and hazard ratio) as well as their samples.

Examples

```
data(weibull_cc, package = "BayesFBHborrow")

# Set your tuning parameters
tuning_parameters <- list("Jmax" = 5,
                          "pi_b" = 0.5,
                          "cprop_beta" = 0.5)

# run the MCMC sampler
out <- BayesFBHborrow(weibull_cc, NULL, tuning_parameters = tuning_parameters,
```

```
                      iter = 3, warmup_iter = 1)

# for the treatment group
plots <- plot(out$out, out$out$time_grid, x_pred = c(1))
```

---

summary.BayesFBHborrow

Summarize fixed MCMC results

Description

S3 method for with borrowing. Returns summary of mean, median and given percentiles for the one dimensional parameters.

Usage

```
## S3 method for class 'BayesFBHborrow'
summary(
  object,
  estimator = NULL,
  percentiles = c(0.025, 0.25, 0.75, 0.975),
  ...
)
```

Arguments

object	MCMC sample object from BayesFBHborrow()
estimator	The type of estimator to summarize, could be "fixed", "lambda", "lambda_0" or "s". The default is NULL and will print a summary of the output list.
percentiles	Given percentiles to output, default is c(0.025, 0.25, 0.75, 0.975)
...	other arguments, see summary.default

Value

summary of the given estimator

Examples

```
data(piecewise_exp_cc, package = "BayesFBHborrow")

# Set your tuning parameters
tuning_parameters <- list("Jmax" = 5,
                          "pi_b" = 0.5,
                          "cprop_beta" = 0.5)

# run the MCMC sampler
out <- BayesFBHborrow(piecewise_exp_cc, NULL, tuning_parameters = tuning_parameters,
                      iter = 3, warmup_iter = 1)
```

```
# Create a summary of the output
summary(out$out, estimator = "out_fixed")
```

`weibull_cc` *Example data, simulated from a Weibull distribution.*

Description

Data is simulated for a concurrent trial with three columns named "tte" (time-to-event), "event" (event indicator), and "X_trt" (treatment indicator). It was simulated by drawing samples from a Weibull with kappa = 1.5 (shape) and nu = 0.4 (scale)

Usage

```
data(weibull_cc)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 250 rows and 3 columns.

Examples

```
data(weibull_cc)
survival_model <- survival::survfit(survival::Surv(tte, event) ~ X_trt, data = weibull_cc)
line_colors <- c("blue", "red")  # Adjust colors as needed
line_types <- 1:length(unique(weibull_cc$X_trt))
plot(survival_model, col = line_colors, lty = line_types,
     xlab = "Time (tte)", ylab = "Survival Probability",
     main = "Kaplan-Meier Survival Curves by Treatment")
```

`weibull_hist` *Example data, simulated from a Weibull distribution*

Description

Data is simulated for a historical trial with two columns named "tte" (time-to-event) and "event" (event indicator). It was simulated using the following parameters:

Usage

```
data(weibull_hist)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 100 rows and 2 columns.

Examples

```
data(weibull_cc)
data(weibull_hist)
weibull_hist$X_trt <- 0
survival_model <- survival::survfit(survival::Surv(tte, event) ~ X_trt,
                                    data = rbind(weibull_cc,
                                    weibull_hist))
line_colors <- c("blue", "red", "green")  # Adjust colors as needed
line_types <- 1:length(unique(weibull_cc$X_trt))
plot(survival_model, col = line_colors, lty = line_types,
     xlab = "Time (tte)", ylab = "Survival Probability",
     main = "Kaplan-Meier Survival Curves by Treatment")
```

Index

∗ **datasets**
piecewise_exp_cc, 47
piecewise_exp_hist, 47
weibull_cc, 50
weibull_hist, 50
.ICAR_calc, 8
.J_RJMCMC, 10
.J_RJMCMC_NoBorrow, 12
.beta.MH.RW.glm, 3
.beta_MH_MALA, 3
.beta_MH_NR, 4
.beta_MH_RW, 5
.beta_mom, 5
.beta_mom.NR.fun, 6
.birth_move, 6
.dataframe_fun, 7
.death_move, 7
.glmFit, 8
.input_check, 9
.lambda_0_MH_cp, 13
.lambda_0_MH_cp_NoBorrow, 14
.lambda_MH_cp, 16
.lambda_conj_prop, 15
.lgamma_ratio, 17
.llikelihood_ratio_beta, 18
.llikelihood_ratio_lambda, 18
.log_likelihood, 19
.logsumexp, 19
.lprop.dens.beta.NR, 20
.lprop_density_beta, 20
.ltau_dprior, 21
.mu_update, 21
.normalize_prob, 22
.nu_sigma_update, 22
.plot_hist, 23
.plot_matrix, 24
.plot_trace, 25
.predictive_hazard, 25
.predictive_hazard_ratio, 26
.predictive_survival, 26
.set_hyperparameters, 27
.set_tuning_parameters, 27
.shuffle_split_point_location, 28
.shuffle_split_point_location_NoBorrow, 29
.sigma2_update, 30
.smooth_hazard, 31
.smooth_survival, 31
.tau_update, 32

BayesFBHborrow, 33
BayesFBHborrow.NoBorrow, 34
BayesFBHborrow.WBorrow, 36

coef.BayesFBHborrow, 38

GibbsMH, 39
GibbsMH.NoBorrow, 41
GibbsMH.WBorrow, 43
group_summary, 45

init_lambda_hyperparameters, 46

piecewise_exp_cc, 47
piecewise_exp_hist, 47
plot.BayesFBHborrow, 48

summary.BayesFBHborrow, 49

weibull_cc, 50
weibull_hist, 50