Package ‘marp’

October 13, 2022

Version  0.1.0
Type     Package
Title    Model-Averaged Renewal Process
Maintainer Jie Kang <jkang@maths.otago.ac.nz>

Description To implement a model-averaging approach with different renewal
models, with a primary focus on forecasting large earthquakes. Based on
six renewal models (i.e., Poisson, Gamma, Log-Logistics, Weibull,
Log-Normal and BPT), model-averaged point estimates are calculated using
AIC (or BIC) weights. Additionally, both percentile and studentized
bootstrapped model-averaged confidence intervals are constructed. In
comparison, point and interval estimation from the individual or ``best"
model (determined via model selection) can be retrieved.

URL   https://github.com/kanji709/marp

BugReports https://github.com/kanji709/marp/issues

Depends R (>= 2.15)
Imports stats, gtools, statmod, VGAM,
Suggests knitr, devtools, roxygen2, testthat (>= 3.0.0)
License MIT + file LICENSE
Encoding UTF-8
RoxygenNote 7.1.2
Config/testthat/edition 3

Date/Publication 2022-08-11 15:20:02 UTC
R topics documented:

- `bpt_bstrp` .......................... 2
- `bpt_logl` .......................... 4
- `bpt_rp` .......................... 5
- `dllog` .......................... 6
- `gamma_bstrp` ...................... 7
- `gamma_logl` ....................... 8
- `gamma_rp` ........................ 9
- `loglogis_bstrp` .................. 10
- `loglogis_logl` ................... 12
- `loglogis_rp` ..................... 13
- `lognorm_bstrp` ................... 14
- `lognorm_rp` ...................... 15
- `lowerT` ........................ 17
- `marp` ........................ 18
- `marp_bstrp` ..................... 19
- `marp_confint` .................... 21
- `percent_confint` ................. 24
- `pllog` ........................ 26
- `poisson_bstrp` .................. 27
- `poisson_rp` ..................... 28
- `student_confint` ................. 29
- `upperT` ........................ 32
- `weibull_bstrp` .................. 33
- `weibull_logl` ................... 35
- `weibull_rp` ..................... 36

Index .............................. 38

**bpt_bstrp**

A function to generate (double) bootstrap samples and fit BPT renewal model

**Description**

A function to generate (double) bootstrap samples and fit BPT renewal model

**Usage**

`bpt_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)`

**Arguments**

- `n` .......................... number of inter-event times
- `t` .......................... user-specified time intervals (used to compute hazard rate)
- `B` .......................... number of bootstrap samples
number of double-bootstrap samples
m the number of iterations in nlm
par_hat estimated parameters
mu_hat estimated mean inter-event times
pr_hat estimated time to event probability
haz_hat estimated hazard rates
y user-specified time point (used to compute time-to-event probability)

Value

returns list of estimates after fitting BPT renewal model on (double) bootstrap samples, containing:

mu_star Estimated mean from bootstrapped samples
pr_star Estimated probability from bootstrapped samples
haz_star Estimated hazard rates from bootstrapped samples
mu_var_hat Variance of estimated mean
pr_var_hat Variance of estimated probability
haz_var_hat Variance of estimated hazard rates
mu_var_double Variance of estimated mean of bootstrapped samples (via double-bootstrapping)
pr_var_double Variance of estimated probability of bootstrapped samples (via double-bootstrapping)
haz_var_double Variance of estimated hazard rates of bootstrapped samples (via double-bootstrapping)
mu_Tstar Pivot quantity of the estimated mean
pr_Tstar Pivot quantity of the estimated probability
haz_Tstar Pivot quantity of the estimated hazard rates

Examples

# set some parameters
n <- 30 # sample size
t <- seq(100, 200, by = 10) # time intervals
B <- 100 # number of bootstraps
BB <- 100 # number of double-bootstraps
m <- 10 # number of iterations for MLE optimization
par_hat <- c(
  3.41361e-03, 2.76268e+00, 2.60378e+00, 3.30802e+02, 5.48822e+00, 2.92945e+02, NA,
  9.43071e-03, 2.47598e+02, 1.80102e+00, 6.50845e-01, 7.18247e-01
)
mu_hat <- c(292.94512, 292.94513, 319.72017, 294.16945, 298.87286, 292.94512)
pr_hat <- c(0.60039, 0.42155, 0.53434, 0.30780, 0.56416, 0.61795)
haz_hat <- matrix(c(
  -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
  -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
  -5.99679, -5.91174, -5.83682, -5.77031, -5.71085, -5.65738,
  -5.60904, -5.56512, -5.52504, -5.48833, -6.09902, -5.97017,
  -5.85769, -5.75939, -5.67350, -5.59856, -5.53336, -5.47683,
  -5.60904, -5.56512, -5.52504, -5.48833, -6.09902, -5.97017,
  -5.85769, -5.75939, -5.67350, -5.59856, -5.53336, -5.47683,
  -5.60904, -5.56512, -5.52504, -5.48833, -6.09902, -5.97017,
  -5.85769, -5.75939, -5.67350, -5.59856, -5.53336, -5.47683,
  -5.60904, -5.56512, -5.52504, -5.48833, -6.09902, -5.97017,
  -5.85769, -5.75939, -5.67350, -5.59856, -5.53336, -5.47683,
  -5.60904, -5.56512, -5.52504, -5.48833, -6.09902, -5.97017,
  -5.85769, -5.75939, -5.67350, -5.59856, -5.53336, -5.47683,
), nrow = 6, byrow = TRUE)

A function to calculate the log-likelihood of BPT model

Usage

bpt_logl(param, x)

Arguments

param parameters of BPT model
x input data for BPT model

Value

returns the value of negative log-likelihood of the BPT model

Examples

set.seed(42)
data <- rgamma(30,3,0.01)

# set some parameters
par_hat <- c(292.945125794581, 0.718247184450307) # estimated parameters
param <- c(log(par_hat[1]),log(par_hat[2]^2)) # input parameters for logl function

# calculate log-likelihood
result <- marp::bpt_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)

# print result
cat("-logl = ", result, "\n")
**bpt_rp**

### A function to fit BPT renewal model

**Description**

A function to fit BPT renewal model

**Usage**

```r
bpt_rp(data, t, m, y)
```

**Arguments**

- **data**: input inter-event times
- **t**: user-specified time intervals (used to compute hazard rate)
- **m**: the number of iterations in nlm
- **y**: user-specified time point (used to compute time-to-event probability)

**Value**

returns list of estimates after fitting BPT renewal model

- **par1**: Estimated parameter (mu) of the BPT model
- **par2**: Estimated parameter (alpha) of the BPT model
- **logL**: Negative log-likelihood
- **AIC**: Akaike information criterion (AIC)
- **BIC**: Bayesian information criterion (BIC)
- **mu_hat**: Estimated mean
- **pr_hat**: Estimated (logit) probabilities
- **haz_hat**: Estimated (log) hazard rates

**Examples**

```r
set.seed(42)
data <- rgamma(30, 3, 0.01)

# set some parameters
m <- 10 # number of iterations for MLE optimization
t <- seq(100, 200, by=10) # time intervals
y <- 304 # cut-off year for estimating probability

# fit BPT renewal model
result <- marp::bpt_rp(data, t, m, y)

# print result
cat("par1 = ", result$par1, "\n")
```
dllog

Density function of Log-Logistics model

Description

Density function of Log-Logistics model

Usage

dllog(x, shape = 1, scale = 1, log = FALSE)

Arguments

x          input data for Log-Logistics model
shape      shape parameter of Log-Logistics model
scale      scale parameter of Log-Logistics model
log        logic function to determine whether log of logistics to be returned

Value

returns the density of the Log-Logistics model

Examples

x <- as.numeric(c(350., 450., 227., 352., 654.))
# set parameters
shape <- 5
scale <- 3
log <- FALSE
result_1 <- marp::dllog(x, shape, scale, log)

# alternatively, set log == TRUE
log <- TRUE
result_2 <- marp::dllog(x, shape, scale, log)
Description

A function to generate (double) bootstrap samples and fit Gamma renewal model

Usage

gamma_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)

Arguments

- **n**: number of inter-event times
- **t**: user-specified time intervals (used to compute hazard rate)
- **B**: number of bootstrap samples
- **BB**: number of double-bootstrap samples
- **m**: the number of iterations in nlm
- **par_hat**: estimated parameters
- **mu_hat**: estimated mean inter-event times
- **pr_hat**: estimated time to event probability
- **haz_hat**: estimated hazard rates
- **y**: user-specified time point (used to compute time-to-event probability)

Value

returns list of estimates after fitting Gamma renewal model on (double) bootstrap samples

- **mu_star**: Estimated mean from bootstrapped samples
- **pr_star**: Estimated probability from bootstrapped samples
- **haz_star**: Estimated hazard rates from bootstrapped samples
- **mu_var_hat**: Variance of estimated mean
- **pr_var_hat**: Variance of estimated probability
- **haz_var_hat**: Variance of estimated hazard rates
- **mu_var_double**: Variance of estimated mean of bootstrapped samples (via double-bootstrapping)
- **pr_var_double**: Variance of estimated probability of bootstrapped samples (via double-bootstrapping)
- **haz_var_double**: Variance of estimated hazard rates of bootstrapped samples (via double-bootstrapping)
- **mu_Tstar**: Pivot quantity of the estimated mean
- **pr_Tstar**: Pivot quantity of the estimated probability
- **haz_Tstar**: Pivot quantity of the estimated hazard rates
Examples

```r
# set some parameters
n <- 30 # sample size
t <- seq(100, 200, by = 10) # time intervals
B <- 100 # number of bootstraps
BB <- 100 # number of double-bootstraps
m <- 10 # number of iterations for MLE optimization
par_hat <- c(
  3.41361e-03, 2.76268e+00, 3.30802e+02, 5.48822e+00, 2.92945e+02, NA,
  9.43071e-03, 2.47598e+02, 1.80102e+00, 6.50845e-01, 7.18247e-01
)
mu_hat <- c(292.94512, 292.94513, 319.72017, 294.16945, 298.87286, 292.94512)
pr_hat <- c(0.60039, 0.42155, 0.53434, 0.30780, 0.56416, 0.61795)
haz_hat <- matrix(c(
  -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
  -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -6.09420,
  -5.99679, -5.91174, -5.83682, -5.77031, -5.71085, -5.65738,
  -5.60904, -5.56512, -5.52504, -5.48833, -6.09902, -5.97017,
  -5.85769, -5.75939, -5.67350, -5.59586, -5.53336, -5.47683,
  -5.42805, -5.38621, -5.35060, -6.17146, -6.05512, -6.02542,
  -5.36171, -5.30194, -5.38468, -5.79490, -5.74642, -5.70064,
  -5.65733, -5.61624, -5.92355, -5.80239, -5.70475, -5.62524,
  -5.55994, -5.50595, -5.46106, -5.42359, -5.39222, -5.36591,
  -5.34383, -5.79111, -5.67660, -5.58924, -5.52166, -5.46879,
  -5.42707, -5.39394, -5.36751, -5.34637, -5.32946, -5.31596
), length(t), 6)
y <- 304 # cut-off year for estimating probability

# generate bootstrapped samples then fit renewal model
res <- marp::gamma_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)
```

---

**gamma_logl**

A function to calculate the log-likelihood of Gamma model

**Description**

A function to calculate the log-likelihood of Gamma model

**Usage**

```r
gamma_logl(param, x)
```

**Arguments**

- `param` : parameters of Gamma model
- `x` : input data for Gamma model
**Value**

returns the value of negative log-likelihood of the Gamma model

**Examples**

```r
set.seed(42)
data <- rgamma(30, 3, 0.01)

# set some parameters
par_hat <- c(2.7626793657057762, 0.0094307059277139432)  # estimated parameters
param <- log(par_hat)  # input parameters for logl function

# calculate log-likelihood
result <- marp::gamma_logl(param, data)

# print result
cat("-logl = ", result, "\n")
```

**Description**

A function to fit Gamma renewal model

**Usage**

```r
gamma_rp(data, t, m, y)
```

**Arguments**

- `data` input inter-event times
- `t` user-specified time intervals (used to compute hazard rate)
- `m` the number of iterations in nlm
- `y` user-specified time point (used to compute time-to-event probability)

**Value**

returns list of estimates after fitting Gamma renewal model

- `par1` Estimated shape parameter of the Gamma model
- `par2` Estimated scale parameter of the Gamma model
- `logL` Negative log-likelihood
- `AIC` Akaike information criterion (AIC)
- `BIC` Bayesian information criterion (BIC)
mu_hat Estimated mean

pr_hat Estimated (logit) probabilities

haz_hat Estimated (log) hazard rates

Examples

set.seed(42)
data <- rgamma(100,3,0.01)

# set some parameters
m = 10 # number of iterations for MLE optimization
t = seq(100, 200, by=10) # time intervals
y = 304 # cut-off year for estimating probability

# fit Gamma renewal model
result <- marp::gamma_rp(data, t, m, y)

# print result
cat("par1 = ", result$par1, "\n")
cat("par2 = ", result$par2, "\n")
cat("logL = ", result$logL, "\n")
cat("AIC = ", result$AIC, "\n")
cat("BIC = ", result$BIC, "\n")
cat("mu_hat = " , result$mu_hat, "\n")
cat("pr_hat = " , result$pr_hat, "\n")

loglogis_bstrp A function to generate (double) bootstrap samples and fit Log-Logistic renewal model

Description

A function to generate (double) bootstrap samples and fit Log-Logistic renewal model

Usage

loglogis_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)

Arguments

n number of inter-event times
t user-specified time intervals (used to compute hazard rate)
B number of bootstrap samples
BB number of double-bootstrap samples
m the number of iterations in nlm
par_hat estimated parameters
`loglogis_bstrap`

- `mu_hat`  estimated mean inter-event times
- `pr_hat`   estimated time to event probability
- `haz_hat`  estimated hazard rates
- `y`        user-specified time point (used to compute time-to-event probability)

**Value**

returns list of estimates after fitting Log-Logistic renewal model on (double) bootstrap samples

- `mu_star` Estimated mean from bootstrapped samples
- `pr_star` Estimated probability from bootstrapped samples
- `haz_star` Estimated hazard rates from bootstrapped samples
- `mu_var_hat` Variance of estimated mean
- `pr_var_hat` Variance of estimated probability
- `haz_var_hat` Variance of estimated hazard rates
- `mu_var_double` Variance of estimated mean of bootstrapped samples (via double-bootstrapping)
- `pr_var_double` Variance of estimated probability of bootstrapped samples (via double-bootstrapping)
- `haz_var_double` Variance of estimated hazard rates of bootstrapped samples (via double-bootstrapping)
- `mu_Tstar` Pivot quantity of the estimated mean
- `pr_Tstar` Pivot quantity of the estimated probability
- `haz_Tstar` Pivot quantity of the estimated hazard rates

**Examples**

```r
# set some parameters
n <- 30 # sample size
t <- seq(100, 200, by = 10) # time intervals
B <- 100 # number of bootstraps
BB <- 100 # number of double-bootstraps
m <- 10 # number of iterations for MLE optimization
par_hat <- c(
  3.41361e-03, 2.76268e+00, 2.60370e+00, 3.30802e+02, 5.48822e+00, 2.92945e+02, NA,
  9.43071e-03, 2.47598e+02, 1.80102e+00, 6.50845e-01, 7.18247e-01
)
mu_hat <- c(292.94512, 292.94513, 319.72017, 294.16945, 298.87286, 292.94512)
pr_hat <- c(0.60039, 0.42155, 0.53434, 0.30780, 0.56416, 0.61795)
```

```r
doublenesthat <- c(0.60039, 0.42155, 0.53434, 0.30780, 0.56416, 0.61795)
mu_hat <- c(292.94512, 292.94513, 319.72017, 294.16945, 298.87286, 292.94512)
```
-5.34383, -5.79111, -5.67660, -5.58924, -5.52166, -5.46879, -5.42707, -5.39394, -5.36751, -5.34637, -5.32946, -5.31596

y <- 304 # cut-off year for estimating probability

# generate bootstrapped samples then fit renewal model
res <- marp::loglogis_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)

loglogis_logl A function to calculate the log-likelihood of Log-Logistics model

Description

A function to calculate the log-likelihood of Log-Logistics model

Usage

loglogis_logl(param, x)

Arguments

param parameters of Log-Logistics model
x input data for Log-Logistics model

Value

returns the value of negative log-likelihood of the Log-Logistics model

Examples

set.seed(42)
data <- rgamma(30,3,0.01)

# set some parameters
par_hat <- c(2.6037079185931518, 247.59811806509711) # estimated parameters
param <- c(log(par_hat[2]),log(par_hat[1])) # input parameters for logl function

# calculate log-likelihood
result <- marp::loglogis_logl(param, data)

# print result
cat("-logl = ", result, "\n")
**loglogis_rp**  

*Description*

A function to fit Log-Logistics renewal model

**Usage**

`loglogis_rp(data, t, m, y)`

**Arguments**

- `data`: input inter-event times
- `t`: user-specified time intervals (used to compute hazard rate)
- `m`: the number of iterations in nlm
- `y`: user-specified time point (used to compute time-to-event probability)

**Value**

returns list of estimates after fitting Log-Logistics renewal model

- `par1`: Estimated shape parameter of the Log-Logistics model
- `par2`: Estimated scale parameter of the Log-Logistics model
- `logL`: Negative log-likelihood
- `AIC`: Akaike information criterion (AIC)
- `BIC`: Bayesian information criterion (BIC)
- `mu_hat`: Estimated mean
- `pr_hat`: Estimated (logit) probabilities
- `haz_hat`: Estimated (log) hazard rates

**Examples**

```r
set.seed(42)
data <- rgamma(100, 3, 0.01)

# set some parameters
m = 10  # number of iterations for MLE optimization
t = seq(100, 200, by=10)  # time intervals
y = 304  # cut-off year for estimating probability

# fit Log-Logistic renewal model
result <- marp::loglogis_rp(data, t, m, y)

# print result
print(result)
```
lognorm_bstrp

A function to generate (double) bootstrap samples and fit Log-Normal renewal model

Description

A function to generate (double) bootstrap samples and fit Log-Normal renewal model

Usage

lognorm_bstrp(n, t, B, BB, par_hat, mu_hat, pr_hat, haz_hat, y)

Arguments

n      number of inter-event times

#  user-specified time intervals (used to compute hazard rate)
B      number of bootstrap samples
BB     number of double-bootstrap samples
par_hat estimated parameters
mu_hat estimated mean inter-event times
pr_hat estimated time to event probability
haz_hat estimated hazard rates
y      user-specified time point (used to compute time-to-event probability)

Value

returns list of estimates after fitting Log-Normal renewal model on (double) bootstrap samples

mu_star Estimated mean from bootstrapped samples
pr_star Estimated probability from bootstrapped samples
haz_star Estimated hazard rates from bootstrapped samples
mu_var_hat Variance of estimated mean
pr_var_hat Variance of estimated probability
haz_var_hat Variance of estimated hazard rates
mu_var_double Variance of estimated mean of bootstrapped samples (via double-bootstrapping)
**Examples**

```r
# set some parameters
n <- 30 # sample size
t <- seq(100, 200, by = 10) # time intervals
B <- 100 # number of bootstraps
BB <- 100 # number of double-bootstraps
# m <- 10 # number of iterations for MLE optimization
par_hat <- c(
  3.41361e-03, 2.76268e+00, 2.60370e+00, 3.30802e+02, 5.48822e+00, 2.92945e+02, NA,
  9.43071e-03, 2.47598e+02, 1.80102e+00, 6.50845e-01, 7.18247e-01
)
mu_hat <- c(292.94512, 292.94513, 319.72017, 294.16945, 298.87286, 292.94512)
pr_hat <- c(0.60039, 0.42155, 0.53434, 0.30780, 0.56416, 0.61795)
# generate bootstrapped samples then fit renewal model
res <- marp::lognorm_bstrp(n, t, B, BB, par_hat, mu_hat, pr_hat, haz_hat, y)
```

---

**lognorm_rp**

A function to fit Log-Normal renewal model

**Description**

A function to fit Log-Normal renewal model
lognorm_rp

Usage

lognorm_rp(data, t, y)

Arguments

- **data**: as input inter-event times
- **t**: as user-specified time intervals (used to compute hazard rate)
- **y**: as user-specified time point (used to compute time-to-event probability)

Value

returns list of estimates after fitting Log-Normal renewal model

- **par1**: Estimated mean (on the log scale) of the Log-Normal model
- **par2**: Estimated standard deviation (on the log scale) of the Log-Normal model
- **logL**: Negative log-likelihood
- **AIC**: Akaike information criterion (AIC)
- **BIC**: Bayesian information criterion (BIC)
- **mu_hat**: Estimated mean
- **pr_hat**: Estimated (logit) probabilities
- **haz_hat**: Estimated (log) hazard rates

Examples

```r
set.seed(42)
data <- rgamma(100, 3, 0.01)

# set some parameters
t = seq(100, 200, by=10)  # time intervals
y = 304  # cut-off year for estimating probability

# fit Log-Normal renewal model
result <- marp::lognorm_rp(data, t, y)

# print result
cat("par1 = ", result$par1, "\n")
cat("par2 = ", result$par2, "\n")
cat("logL = ", result$logL, "\n")
cat("AIC = ", result$AIC, "\n")
cat("BIC = ", result$BIC, "\n")
cat("mu_hat = ", result$mu_hat, "\n")
cat("pr_hat = ", result$pr_hat, "\n")
```


lowerT

An utility function to calculate upper limit of T statistic

Description

An utility function to calculate upper limit of T statistic

Usage

lowerT(low, hat, sigmasq, Tstar, weights, B, alpha)

Arguments

- **low**: lower limit
- **hat**: estimates
- **sigmasq**: variance
- **Tstar**: T statistics estimated from bootstrap samples
- **weights**: model weights
- **B**: number of bootstraps
- **alpha**: confidence level

Value

returns upper limit of T-statistic

Examples

```r
# set some parameters
low <- 100  # lower bound
hat <- rep(150, 6)  # estimates obtained from each model
sigmasq <- 10  # variance
Tstar <- matrix(rep(100,600),6,100)  # T statistics estimated from bootstrap samples
weights <- rep(1/6, 6)  # model weights
B <- 100  # number of bootstrapped samples
alpha <- 0.05  # confidence level

# calculate the upper limit of T statistics
res <- marp::lowerT(low, hat, sigmasq, Tstar, weights, B, alpha)

# print result
cat("res = ", res, "\n")
```
marp

A function to apply model-averaged renewal process

Description
A function to apply model-averaged renewal process

Usage
marp(data, t, m, y, which.model = 1)

Arguments
data input inter-event times
t user-specified time intervals (used to compute hazard rate)
m the number of iterations in nlm
y user-specified time point (used to compute time-to-event probability)
which.model user-specified generating (or true underlying if known) model

Value
returns list of estimates obtained from different renewal processes and after applying model-averaging

par1 Estimated scale parameters (if applicable) of all six renewal models
par2 Estimated shape parameters (if applicable) of all six renewal models
logL Negative log-likelihood
AIC Akaike information criterion (AIC)
BIC Bayesian information criterion (BIC)
mu_hat Estimated mean
pr_hat Estimated (logit) probabilities
haz_hat Estimated (log) hazard rates
weights_AIC Model weights calculated based on AIC
weights_BIC Model weights calculated based on BIC
model_best Model selected based on the lowest AIC
mu_best Estimated mean obtained from the model with the lowest AIC
pr_best Estimated probability obtained from the model with the lowest AIC
haz_best Estimated hazard rates obtained from the model with the lowest AIC
mu_gen Estimated mean obtained from the (true or hypothetical) generating model
pr_gen Estimated probability obtained from the (true or hypothetical) generating model
haz_gen Estimated hazard rates obtained from the (true or hypothetical) generating model
mu_aic Estimated mean obtained from model-averaging (using AIC weights)
pr_aic Estimated probability obtained from model-averaging (using AIC weights)
haz_aic Estimated hazard rates obtained from model-averaging (using AIC weights)
Examples

```r
set.seed(42)
data <- rgamma(100,3,0.01)

# set some parameters
m = 10 # number of iterations for MLE optimization
t = seq(100, 200, by=10) # time intervals
y = 304 # cut-off year for estimating probability
which.model <- 2 # specify the generating model

# model selection and averaging
result <- marp::marp(data, t, m, y, which.model)
```

marp_bstrp

A function to fit model-averaged renewal process

Description

A function to fit model-averaged renewal process

Usage

```r
marp_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)
```

Arguments

- **n**: number of inter-event times
- **t**: user-specified time intervals (used to compute hazard rate)
- **B**: number of bootstrap samples
- **BB**: number of double-bootstrap samples
- **m**: the number of iterations in nlm
- **par_hat**: estimated parameters
- **mu_hat**: estimated mean inter-event times
- **pr_hat**: estimated time to event probability
- **haz_hat**: estimated hazard rates
- **y**: user-specified time point (used to compute time-to-event probability)

Value

returns list of estimates after fitting different renewal models on (double) bootstrap samples

- **mu_star**: Estimated mean from bootstrapped samples
- **pr_star**: Estimated probability from bootstrapped samples
- **haz_star**: Estimated hazard rates from bootstrapped samples
mu_var_hat  Variance of estimated mean
pr_var_hat  Variance of estimated probability
haz_var_hat  Variance of estimated hazard rates
mu_var_double  Variance of estimated mean of bootstrapped samples (via double-bootstrapping)
pr_var_double  Variance of estimated probability of bootstrapped samples (via double-bootstrapping)
haz_var_double  Variance of estimated hazard rates of bootstrapped samples (via double-bootstrapping)
mu_Tstar  Pivot quantity of the estimated mean
pr_Tstar  Pivot quantity of the estimated probability
haz_Tstar  Pivot quantity of the estimated hazard rates

Examples

# set some parameters
n <- 30 # sample size
t <- seq(100, 200, by = 10) # time intervals
B <- 100 # number of bootstraps
BB <- 100 # number of double-bootstraps
m <- 10 # number of iterations for MLE optimization
par_hat <- c(3.41361e-03, 2.76268e+00, 2.60370e+00, 3.30802e+02, 5.48822e+00, 2.92945e+02, NA,
            9.43071e-03, 2.47598e+02, 1.80102e+00, 6.50845e-01, 7.18247e-01)
mu_hat <- c(292.94512, 292.94513, 319.72017, 294.16945, 298.87286, 292.94512)
pr_hat <- c(0.60039, 0.42155, 0.53434, 0.30780, 0.56416, 0.61795)
haz_hat <- matrix(c(-5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
                     -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
                     -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
                     -5.9679,  -5.91174,  -5.83682,  -5.77031,  -5.71085,  -5.65738,
                     -5.60904,  -5.56512,  -5.52504,  -5.48833,  -6.09902,  -5.97017,
                     -5.85769,  -5.75939,  -5.67350,  -5.59856,  -5.53336,  -5.47683,
                     -5.42805,  -5.38621,  -5.35060,  -6.17146,  -6.09512,  -6.02542,
                     -5.96131,  -5.90194,  -5.84668,  -5.79498,  -5.74642,  -5.70064,
                     -5.65733,  -5.61624,  -5.92355,  -5.80239,  -5.70475,  -5.62524,
                     -5.55994,  -5.50595,  -5.46106,  -5.42359,  -5.39222,  -5.36591,
                     -5.34383,  -5.79111,  -5.67660,  -5.58924,  -5.52166,  -5.46879,
                     -5.42707,  -5.39394,  -5.36751,  -5.34637,  -5.32946,  -5.31596
                     ),length(t),6)
y <- 304 # cut-off year for estimating probability

# generate bootstrapped samples then fit renewal model
res <- marp::marp_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)
marp_confint

Description

A function to apply model-averaged renewal process

Usage

marp_confint(data, m, t, B, BB, alpha, y, which.model)

Arguments

data      input inter-event times
m         the number of iterations in nlm
m         user-specified time intervals (used to compute hazard rate)
B         number of bootstrap samples
BB        number of double-bootstrap samples
alpha     significance level
y         user-specified time point (used to compute time-to-event probability)
which.model user-specified generating (or true underlying if known) model

Value

returns list of point and interval estimation obtained from different renewal models (including
model-averaged confidence intervals).

par1      Estimated scale parameters (if applicable) of all six renewal models
par2      Estimated shape parameters (if applicable) of all six renewal models
logL      Negative log-likelihood
AIC       Akaike information criterion (AIC)
BIC       Bayesian information criterion (BIC)
mu_hat    Estimated mean
pr_hat     Estimated (logit) probabilities
haz_hat    Estimated (log) hazard rates
weights_AIC Model weights calculated based on AIC
weights_BIC Model weights calculated based on BIC
model_best Model selected based on the lowest AIC
mu_best    Estimated mean obtained from the model with the lowest AIC
pr_best    Estimated probability obtained from the model with the lowest AIC
haz_best   Estimated hazard rates obtained from the model with the lowest AIC
**mu_gen**  Estimated mean obtained from the (true or hypothetical) generating model

**pr_gen**  Estimated probability obtained from the (true or hypothetical) generating model

**haz_gen**  Estimated hazard rates obtained from the (true or hypothetical) generating model

**mu_aic**  Estimated mean obtained from model-averaging (using AIC weights)

**pr_aic**  Estimated probability obtained from model-averaging (using AIC weights)

**haz_aic**  Estimated hazard rates obtained from model-averaging (using AIC weights)

**mu_bstrp**  Estimated mean obtained from model-averaging (using bootstrapped weights)

**pr_bstrp**  Estimated probability obtained from model-averaging (using bootstrapped weights)

**haz_bstrp**  Estimated hazard rates obtained from model-averaging (using bootstrapped weights)

**weights_bstp**  Model weights calculated by bootstrapping, that is, the frequency of each model being selected as the best model is divided by the total number of bootstraps

**mu_gen**  Median of the percentile bootstrap confidence interval of the estimated mean based on the generating model

**mu_gen_lower**  Lower limit of the percentile bootstrap confidence interval of the estimated mean based on the generating model

**mu_gen_upper**  Upper limit of the percentile bootstrap confidence interval of the estimated mean based on the generating model

**mu_best**  Median of the percentile bootstrap confidence interval of the estimated mean based on the best model

**mu_best_lower**  Lower limit of the percentile bootstrap confidence interval of the estimated mean based on the best model

**mu_best_upper**  Upper limit of the percentile bootstrap confidence interval of the estimated mean based on the best model

**pr_gen**  Median of the percentile bootstrap confidence interval of the estimated probabilities based on the generating model

**pr_gen_lower**  Lower limit of the percentile bootstrap confidence interval of the estimated probabilities based on the generating model

**pr_gen_upper**  Upper limit of the percentile bootstrap confidence interval of the estimated probabilities based on the generating model

**pr_best**  Median of the percentile bootstrap confidence interval of the estimated probabilities based on the best model

**pr_best_lower**  Lower limit of the percentile bootstrap confidence interval of the estimated probabilities based on the best model

**pr_best_upper**  Upper limit of the percentile bootstrap confidence interval of the estimated probabilities based on the best model

**haz_gen**  Median of the percentile bootstrap confidence interval of the estimated hazard rates based on the generating model

**haz_gen_lower**  Lower limit of the percentile bootstrap confidence interval of the estimated hazard rates based on the generating model

**haz_gen_upper**  Upper limit of the percentile bootstrap confidence interval of the estimated hazard rates based on the generating model
\textbf{haz\_best}  Median of the percentile bootstrap confidence interval of the estimated hazard rates based on the best model

\textbf{haz\_best\_lower}  Lower limit of the percentile bootstrap confidence interval of the estimated hazard rates based on the best model

\textbf{haz\_best\_upper}  Upper limit of the percentile bootstrap confidence interval of the estimated hazard rates based on the best model

\textbf{mu\_lower\_gen}  Lower limit of the studentized bootstrap confidence interval of the estimated mean based on the generating model

\textbf{mu\_upper\_gen}  Upper limit of the studentized bootstrap confidence interval of the estimated mean based on the generating model

\textbf{mu\_lower\_best}  Lower limit of the studentized bootstrap confidence interval of the estimated mean based on the best model

\textbf{mu\_upper\_best}  Upper limit of the studentized bootstrap confidence interval of the estimated mean based on the best model

\textbf{pr\_lower\_gen}  Lower limit of the studentized bootstrap confidence interval of the estimated probabilities based on the generating model

\textbf{pr\_upper\_gen}  Upper limit of the studentized bootstrap confidence interval of the estimated probabilities based on the generating model

\textbf{pr\_lower\_best}  Lower limit of the studentized bootstrap confidence interval of the estimated probabilities based on the best model

\textbf{pr\_upper\_best}  Upper limit of the studentized bootstrap confidence interval of the estimated probabilities based on the best model

\textbf{haz\_lower\_gen}  Lower limit of the studentized bootstrap confidence interval of the estimated hazard rates based on the generating model

\textbf{haz\_upper\_gen}  Upper limit of the studentized bootstrap confidence interval of the estimated hazard rates based on the generating model

\textbf{haz\_lower\_best}  Lower limit of the studentized bootstrap confidence interval of the estimated hazard rates based on the best model

\textbf{haz\_upper\_best}  Upper limit of the studentized bootstrap confidence interval of the estimated hazard rates based on the best model

\textbf{mu\_lower\_ma}  Lower limit of model-averaged studentized bootstrap confidence interval of the estimated mean

\textbf{mu\_upper\_ma}  Upper limit of model-averaged studentized bootstrap confidence interval of the estimated mean

\textbf{pr\_lower\_ma}  Lower limit of model-averaged studentized bootstrap confidence interval of the estimated probabilities

\textbf{pr\_upper\_ma}  Upper limit of model-averaged studentized bootstrap confidence interval of the estimated probabilities

\textbf{haz\_lower\_ma}  Lower limit of model-averaged studentized bootstrap confidence interval of the estimated hazard rates

\textbf{haz\_upper\_ma}  Upper limit of model-averaged studentized bootstrap confidence interval of the estimated hazard rates
Examples

# generate random data
set.seed(42)
data <- rgamma(30, 3, 0.01)

# set some parameters
m <- 10 # number of iterations for MLE optimization
t <- seq(100,200,by=10) # time intervals
alpha <- 0.05 # confidence level
y <- 304 # cut-off year for estimating probability
B <- 100 # number of bootstraps
BB <- 100 # number of double bootstraps
which.model <- 2 # specify the generating model

# construct confidence intervals
res <- marp::marp_confint(data,m,t,B, BB, alpha, y, which.model)

percent_confint

A function to calculate percentile bootstrap confidence interval

Description

A function to calculate percentile bootstrap confidence interval

Usage

percent_confint(data, B, t, m, y, which.model = 1)

Arguments

data input inter-event times
B number of bootstrap samples
t user-specified time intervals (used to compute hazard rate)
m the number of iterations in nlm
y user-specified time point (used to compute time-to-event probability)
which.model user-specified generating (or true underlying if known) model

Value

returns list of percentile bootstrap intervals (including the model-averaged approach).

weights_bstp Model weights calculated by bootstrapping, that is, the frequency of each model being selected as the best model is divided by the total number of bootstraps
percent_confint

\textbf{mu_gen} Median of the percentile bootstrap confidence interval of the estimated mean based on the generating model

\textbf{mu_gen_lower} Lower limit of the percentile bootstrap confidence interval of the estimated mean based on the generating model

\textbf{mu_gen_upper} Upper limit of the percentile bootstrap confidence interval of the estimated mean based on the generating model

\textbf{mu_best} Median of the percentile bootstrap confidence interval of the estimated mean based on the best model

\textbf{mu_best_lower} Lower limit of the percentile bootstrap confidence interval of the estimated mean based on the best model

\textbf{mu_best_upper} Upper limit of the percentile bootstrap confidence interval of the estimated mean based on the best model

\textbf{pr_gen} Median of the percentile bootstrap confidence interval of the estimated probabilities based on the generating model

\textbf{pr_gen_lower} Lower limit of the percentile bootstrap confidence interval of the estimated probabilities based on the generating model

\textbf{pr_gen_upper} Upper limit of the percentile bootstrap confidence interval of the estimated probabilities based on the generating model

\textbf{pr_best} Median of the percentile bootstrap confidence interval of the estimated probabilities based on the best model

\textbf{pr_best_lower} Lower limit of the percentile bootstrap confidence interval of the estimated probabilities based on the best model

\textbf{pr_best_upper} Upper limit of the percentile bootstrap confidence interval of the estimated probabilities based on the best model

\textbf{haz_gen} Median of the percentile bootstrap confidence interval of the estimated hazard rates based on the generating model

\textbf{haz_gen_lower} Lower limit of the percentile bootstrap confidence interval of the estimated hazard rates based on the generating model

\textbf{haz_gen_upper} Upper limit of the percentile bootstrap confidence interval of the estimated hazard rates based on the generating model

\textbf{haz_best} Median of the percentile bootstrap confidence interval of the estimated hazard rates based on the best model

\textbf{haz_best_lower} Lower limit of the percentile bootstrap confidence interval of the estimated hazard rates based on the best model

\textbf{haz_best_upper} Upper limit of the percentile bootstrap confidence interval of the estimated hazard rates based on the best model

\textbf{Examples}

# generate random data
set.seed(42)
data <- rgamma(30, 3, 0.01)
# set some parameters
m <- 10 # number of iterations for MLE optimization
t <- seq(100,200,by=10) # time intervals
y <- 304 # cut-off year for estimating probability
B <- 100 # number of bootstraps
BB <- 100 # number of double bootstraps
which.model <- 2 # specify the generating model

# construct percentile bootstrap confidence intervals
marp::percent_confint(data, B, t, m, y, which.model)

pllog

\textit{Probability function of Log-Logistics model}

\textbf{Description}

Probability function of Log-Logistics model

\textbf{Usage}

\texttt{pllog(q, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)}

\textbf{Arguments}

- \texttt{q}  
  input quantile for Log-Logistics model
- \texttt{shape}  
  shape parameter of Log-Logistics model
- \texttt{scale}  
  scale parameter of Log-Logistics model
- \texttt{lower.tail}  
  logic function to determine whether lower tail probability to be returned
- \texttt{log.p}  
  logic function to determine whether log of logistics to be returned

\textbf{Value}

returns the probability of the Log-Logistics model

\textbf{Examples}

\begin{verbatim}
q <- c(1, 2, 3, 4)
# set parameters
shape <- 5
scale <- 3
log <- FALSE
result_1 <- marp::pllog(q, shape, scale, log)

# alternatively, set log == TRUE
log <- TRUE
result_2 <- marp::pllog(q, shape, scale, log)
\end{verbatim}
A function to generate (double) bootstrap samples and fit Poisson renewal model

Usage

poisson_bstrp(n, t, B, BB, par_hat, mu_hat, pr_hat, haz_hat, y)

Arguments

n number of inter-event times

\( t \) user-specified time intervals (used to compute hazard rate)

B number of bootstrap samples

BB number of double-bootstrap samples

par_hat estimated parameters

\( \mu \_\text{hat} \) estimated mean inter-event times

pr_hat estimated time to event probability

haz_hat estimated hazard rates

\( y \) user-specified time point (used to compute time-to-event probability)

Value

returns list of estimates after fitting Poisson renewal model on (double) bootstrap samples

\textbf{mu\_star} Estimated mean from bootstrapped samples

\textbf{pr\_star} Estimated probability from bootstrapped samples

\textbf{haz\_star} Estimated hazard rates from bootstrapped samples

\textbf{mu\_var\_hat} Variance of estimated mean

\textbf{pr\_var\_hat} Variance of estimated probability

\textbf{haz\_var\_hat} Variance of estimated hazard rates

\textbf{mu\_var\_double} Variance of estimated mean of bootstrapped samples (via double-bootstrapping)

\textbf{pr\_var\_double} Variance of estimated probability of bootstrapped samples (via double-bootstrapping)

\textbf{haz\_var\_double} Variance of estimated hazard rates of bootstrapped samples (via double-bootstrapping)

\textbf{mu\_Tstar} Pivot quantity of the estimated mean

\textbf{pr\_Tstar} Pivot quantity of the estimated probability

\textbf{haz\_Tstar} Pivot quantity of the estimated hazard rates
Examples

# set some parameters
n <- 30 # sample size
t <- seq(100, 200, by = 10) # time intervals
B <- 100 # number of bootstraps
BB <- 100 # number of double-bootstraps
# m <- 10 # number of iterations for MLE optimization
par_hat <- c(
  3.41361e-03, 2.76268e+00, 2.60370e+00, 3.30802e+02, 5.48822e+00, 2.92945e+02, NA,
  9.43071e-03, 2.47598e+02, 1.80102e+00, 6.50845e-01, 6.17951e-01
)
mu_hat <- c(292.94512, 292.94513, 319.72017, 294.16945, 298.87286, 292.94512)
pr_hat <- c(0.60039, 0.42155, 0.53434, 0.30780, 0.56416, 0.61795)
haz_hat <- matrix(c(
  -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
  -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
  -5.96979, -5.91174, -5.83682, -5.77031, -5.71085, -5.65738,
  -5.60904, -5.56512, -5.52504, -5.48833, -6.09902, -5.97017,
  -5.85769, -5.75939, -5.67350, -5.59856, -5.53336, -5.47683,
  -5.42805, -5.38621, -5.35060, -6.17146, -6.09512, -6.02542,
  -5.96131, -5.90194, -5.84668, -5.79498, -5.74642, -5.70664,
  -5.65733, -5.61624, -5.52355, -5.50239, -5.70475, -5.62524,
  -5.59994, -5.50595, -5.46106, -5.42359, -5.39222, -5.36591,
  -5.34383, -5.79111, -5.67660, -5.58924, -5.52166, -5.46879,
  -5.42707, -5.39394, -5.36751, -5.34637, -5.32946, -5.31596
),nrow=length(t),6)
y <- 304 # cut-off year for estimating probability

# generate bootstrapped samples then fit renewal model
res <- marp::poisson_bstrp(n, t, B, BB, par_hat, mu_hat, pr_hat, haz_hat, y)

---

poisson_rp

A function to fit Poisson renewal model

Description

A function to fit Poisson renewal model

Usage

poisson_rp(data, t, y)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>input inter-event times</td>
</tr>
<tr>
<td>t</td>
<td>user-specified time intervals (used to compute hazard rate)</td>
</tr>
<tr>
<td>y</td>
<td>user-specified time point (used to compute time-to-event probability)</td>
</tr>
</tbody>
</table>
Value

returns list of estimates after fitting Poisson renewal model

- **par1**: Estimated parameter of the Poisson model
- **par2**: N/A, only keep it as a place holder for output formatting purpose
- **logL**: Negative log-likelihood
- **AIC**: Akaike information criterion (AIC)
- **BIC**: Bayesian information criterion (BIC)
- **mu_hat**: Estimated mean
- **pr_hat**: Estimated (logit) probabilities
- **haz_hat**: Estimated (log) hazard rates

Examples

```r
set.seed(42)
data <- rgamma(100, 3, 0.01)
# set some parameters
t = seq(100, 200, by=10) # time intervals
y = 304 # cut-off year for estimating probability

# fit Poisson renewal model
result <- marp::poisson_rp(data, t, y)

# print result
cat("par1 = ", result$par1, "\n")
cat("par2 = ", result$par2, "\n")
cat("logL = ", result$logL, "\n")
cat("AIC = ", result$AIC, "\n")
cat("BIC = ", result$BIC, "\n")
cat("mu_hat = ", result$mu_hat, "\n")
cat("pr_hat = ", result$pr_hat, "\n")
```

---

**student_confint**: A function to calculate Studentized bootstrap confidence interval

**Description**

A function to calculate Studentized bootstrap confidence interval
Usage

student_confint(
    n,
    B,
    t,
    m,
    BB,
    par_hat,
    mu_hat,
    pr_hat,
    haz_hat,
    weights,
    alpha,
    y,
    best.model,
    which.model = 1
)

Arguments

- **n**: number of inter-event times
- **B**: number of bootstrap samples
- **t**: user-specified time intervals (used to compute hazard rate)
- **m**: the number of iterations in nlm
- **BB**: number of double-bootstrap samples
- **par_hat**: estimated parameters
- **mu_hat**: estimated mean inter-event times
- **pr_hat**: estimated time to event probability
- **haz_hat**: estimated hazard rates
- **weights**: model weights
- **alpha**: significance level
- **y**: user-specified time point (used to compute time-to-event probability)
- **best.model**: best model based on information criterion (i.e. AIC)
- **which.model**: user-specified generating (or true underlying if known) model

Value

returns list of Studentized bootstrap intervals (including the model-averaged approach).

- **mu_lower_gen**: Lower limit of the studentized bootstrap confidence interval of the estimated mean based on the generating model
- **mu_upper_gen**: Upper limit of the studentized bootstrap confidence interval of the estimated mean based on the generating model
- **mu_lower_best**: Lower limit of the studentized bootstrap confidence interval of the estimated mean based on the best model
**Examples**

```r
# generate random data
set.seed(42)
data <- rgamma(30, 5, 0.01)

# set some parameters
n <- 30 # sample size
m <- 10 # number of iterations for MLE optimization
t <- seq(100, 200, by=10) # time intervals
y <- 304 # cut-off year for estimating probability
B <- 100 # number of bootstraps
BB <- 100 # number of double bootstraps
```
par_hat <- c(
  3.41361e-03, 2.76268e+00, 2.60370e+00, 3.30802e+02, 5.48822e+00, 2.92945e+02, NA,
  9.43071e-03, 2.47598e+02, 1.80102e+00, 6.50845e-01, 7.18247e-01
)
mu_hat <- c(292.94512, 292.94513, 319.72017, 294.16945, 298.87286, 292.94512)
pr_hat <- c(0.60039, 0.42155, 0.53434, 0.30780, 0.56416, 0.61795)
haz_hat <- matrix(c(
  -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
  -5.67999, -5.67999, -5.67999, -5.67999, -5.67999, -5.67999,
  -5.99679, -5.91174, -5.83682, -5.77031, -5.71085, -5.65738,
  -5.60904, -5.56512, -5.52504, -5.48833, -5.09902, -5.97017,
  -5.85769, -5.75939, -5.67350, -5.59856, -5.53336, -5.47683,
  -5.42805, -5.38621, -5.35060, -6.17146, -6.09512, -6.02542,
  -5.96131, -5.90194, -5.84668, -5.79498, -5.74642, -5.70064,
  -5.65733, -5.61624, -5.92355, -5.80239, -5.70475, -5.62524,
  -5.55994, -5.50595, -5.46106, -5.42359, -5.39222, -5.36591,
  -5.34383, -5.79111, -5.67660, -5.58924, -5.52166, -5.46879,
  -5.42707, -5.39394, -5.36751, -5.34637, -5.32946, -5.31596
),length(t),6)
weights <- c(0.00000, 0.21000, 0.02000, 0.55000, 0.00000, 0.22000) # model weights
alpha <- 0.05 # confidence level
y <- 304 # cut-off year for estimating probability
best.model <- 2
which.model <- 2 # specify the generating model#

# construct Studentized bootstrap confidence interval
marp::student_confint(
  n,B,t,m,BB,par_hat,mu_hat,pr_hat,haz_hat,weights,alpha,y,best.model,which.model
)

upperT An utility function to calculate lower limit of T statistic

Description
An utility function to calculate lower limit of T statistic

Usage
upperT(up, hat, sigmasq, Tstar, weights, B, alpha)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>up</td>
<td>upper limit</td>
</tr>
<tr>
<td>hat</td>
<td>estimates</td>
</tr>
<tr>
<td>sigmasq</td>
<td>variance</td>
</tr>
<tr>
<td>Tstar</td>
<td>T statistics estimated from bootstrap samples</td>
</tr>
</tbody>
</table>
weibull_bstrp

weights model weights
B number of bootstraps
alpha confidence level

Value
returns lower limit of T statistic

Examples

# set some parameters
up <- 100 # upper bound
hat <- rep(150, 6) # estimates obtained from each model
sigmasq <- 10 # variance
Tstar <- matrix(rep(100, 6), 6, 100) # T statistics estimated from bootstrap samples
weights <- rep(1/6, 6) # model weights
B <- 100 # number of bootstrapped samples
alpha <- 0.05 # confidence level

# calculate the upper limit of T statistics
res <- marp::upperT(up, hat, sigmasq, Tstar, weights, B, alpha)

# print result
cat("res = ", res, "\n")

weibull_bstrp A function to generate (double) bootstrap samples and fit Weibull renewal model

Description
A function to generate (double) bootstrap samples and fit Weibull renewal model

Usage
weibull_bstrp(n, t, B, BB, m, par_hat, mu_hat, pr_hat, haz_hat, y)

Arguments

n number of inter-event times
t user-specified time intervals (used to compute hazard rate)
B number of bootstrap samples
BB number of double-bootstrap samples
m the number of iterations in nlm
par_hat estimated parameters
mu_hat estimated mean inter-event times
weibull_bstrp

pr_hat  estimated time to event probability
haz_hat  estimated hazard rates
y  user-specified time point (used to compute time-to-event probability)

Value

returns list of estimates after fitting Weibull renewal model on (double) bootstrap samples

mu_star  Estimated mean from bootstrapped samples
pr_star  Estimated probability from bootstrapped samples
haz_star  Estimated hazard rates from bootstrapped samples
mu_var_hat  Variance of estimated mean
pr_var_hat  Variance of estimated probability
haz_var_hat  Variance of estimated hazard rates
mu_var_double  Variance of estimated mean of bootstrapped samples (via double-bootstrapping)
pr_var_double  Variance of estimated probability of bootstrapped samples (via double-bootstrapping)
haz_var_double  Variance of estimated hazard rates of bootstrapped samples (via double-bootstrapping)
mu_Tstar  Pivot quantity of the estimated mean
pr_Tstar  Pivot quantity of the estimated probability
haz_Tstar  Pivot quantity of the estimated hazard rates

Examples

# set some parameters
n <- 30 # sample size
t <- seq(100, 200, by = 10) # time intervals
B <- 100 # number of bootstraps
BB <- 100 # number of double-bootstraps
m <- 10 # number of iterations for MLE optimization
par_hat <- c(
  3.4136086430979953e-03, 2.7626793657657762e+00, 2.6037039674870583e+00, 3.3080162440951688e+02,
  5.4882183783786558e+00, 2.9294512422957868e+02, NA, 9.438705977139432e-03,
  2.475979659031687e+02, 1.801018350766513e+00, 9.4307059277139432e-03,
)
mu_hat <- c(
  292.94512187913182, 292.94512912200048, 319.72017228620746, 294.16945213908519,
  298.87285747700128, 292.94512422957860
)
pr_hat <- c(
  0.60038574701819891, 0.42154974433034809, 0.53433568234281148, 0.30779792692414687,
  0.56416103510057725, 0.61794524610544410
)
haz_hat <- matrix(c(
  -5.679852941338829, -5.679852941338829, -5.679852941338829, -5.679852941338829,
  -5.679852941338829, -5.679852941338829, -5.679852941338829, -5.679852941338829,
  -5.679852941338829, -5.679852941338829, -5.679852941338829, -5.679852941338829,
  -5.679852941338829, -5.679852941338829, -5.679852941338829, -5.679852941338829,
  -5.679852941338829, -5.679852941338829, -5.679852941338829, -5.679852941338829,
), ncol = 20)
A function to calculate the log-likelihood of Weibull model

**Usage**

```r
weibull_logl(param, x)
```

**Arguments**

- `param`: parameters of Weibull model
- `x`: input data for Weibull model

**Value**

returns the value of negative log-likelihood of the Weibull model

**Examples**

```r
set.seed(42)
data <- rgamma(30,3,0.01)

# set some parameters
par_hat <- c(330.801103808081, 1.80101338777944) # estimated parameters
```
weibull_rp <- log(par_hat) # input parameters for logl function

# calculate log-likelihood
result <- marp::weibull_logl(param, data)

# print result
cat("-logl = ", result, "\n")

weibull_rp A function to fit Weibull renewal model #' @import weibull_logl

Description

A function to fit Weibull renewal model #' @import weibull_logl

Usage

weibull_rp(data, t, m, y)

Arguments

data input inter-event times
t user-specified time intervals (used to compute hazard rate)
m the number of iterations in nlm
y user-specified time point (used to compute time-to-event probability)

Value

returns list of estimates after fitting Weibull renewal model

par1 Estimated scale parameter of the Weibull model
par2 Estimated shape parameter of the Weibull model
logL Negative log-likelihood
AIC Akaike information criterion (AIC)
BIC Bayesian information criterion (BIC)
mu_hat Estimated mean
pr_hat Estimated (logit) probabilities
haz_hat Estimated (log) hazard rates
Examples

```r
set.seed(42)
data <- rgamma(100,3,0.01)

# set some parameters
m = 10  # number of iterations for MLE optimization
t = seq(100, 200, by=10)  # time intervals
y = 304  # cut-off year for estimating probability

# fit Weibull renewal model
result <- marp::weibull_rp(data, t, m, y)

# print result
cat("par1 = ", result$par1, "\n")
cat("par2 = ", result$par2, "\n")
cat("logL = ", result$logL, "\n")
cat("AIC = ", result$AIC, "\n")
cat("BIC = ", result$BIC, "\n")
cat("mu_hat = ", result$mu_hat, "\n")
cat("pr_hat = ", result$pr_hat, "\n")
```
Index

bpt_bstrp, 2
bpt_logl, 4
bpt_rp, 5
dllog, 6
gamma_bstrp, 7
gamma_logl, 8
gamma_rp, 9
loglogis_bstrp, 10
loglogis_logl, 12
loglogis_rp, 13
lognorm_bstrp, 14
lognorm_rp, 15
lowerT, 17

marp, 18
marp_bstrp, 19
marp_confint, 21

percent_confint, 24
pllog, 26
poisson_bstrp, 27
poisson_rp, 28

student_confint, 29

upperT, 32

weibull_bstrp, 33
weibull_logl, 35
weibull_rp, 36