Package: cpsurvsim (via r-universe)

October 30, 2024

Type Package

Title Simulating Survival Data from Change-Point Hazard Distributions

Version 1.2.2.9000

Date 2023-09-05

Author Camille Hochheimer [aut, cre]

Maintainer Camille Hochheimer <dochoch19@gmail.com>

Description Simulates time-to-event data with type I right censoring using two methods: the inverse CDF method and our proposed memoryless method. The latter method takes advantage of the memoryless property of survival and simulates a separate distribution between change-points. We include two parametric distributions: exponential and Weibull. Inverse CDF method draws on the work of Rainer Walke (2010), <https://www.demogr.mpg.de/papers/technicalreports/tr-2010-003.pdf>.

Depends R (>= 3.6.0)

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports plyr (>= 1.8.5), stats, Hmisc (>= 4.3.0), knitr (>= 1.27)

Suggests rmarkdown, testthat

RoxygenNote 7.2.3

VignetteBuilder knitr

URL https://github.com/camillejo/cpsurvsim

BugReports https://github.com/camillejo/cpsurvsim/issues

Repository https://camillejo.r-universe.dev

RemoteUrl https://github.com/camillejo/cpsurvsim

RemoteRef HEAD

RemoteSha 920224475c0671fee409416dd54f6c505f1592fb

Contents

`cpsurvsim` *cpsurvsim: Simulating Survival Data from Change-Point Hazard Distributions*

Description

The cpsurvsim package simulates time-to-event data with type I right censoring using two methods: the inverse CDF method and a memoryless method (for more information on simulation methods, see the vignette). We include two parametric distributions: exponential and Weibull.

cpsurvsim functions

For the exponential distribution, the `exp_icdf` function simulates values from the inverse exponential distribution. `exp_cdfsim` and `exp_memsim` return time-to-event datasets simulated using the inverse CDF and memoryless methods respectively.

For the Weibull distribution, the `weib_icdf` function simulates values from the inverse Weibull distribution. `weib_cdfsim` and `weib_memsim` return time-to-event datasets simulated using the inverse CDF and memoryless methods respectively.

`exp_cdfsim` *Inverse CDF simulation for the exponential change-point hazard distribution*

Description

`exp_cdfsim` simulates time-to-event data from the exponential change-point hazard distribution by implementing the inverse CDF method.

Usage

```
exp_cdfsim(n, endtime, theta, tau = NA)
```

Arguments

`n`	Sample size
`endtime`	Maximum study time, point at which all participants are censored
`theta`	Scale parameter θ
`tau`	Change-point(s) τ

Details

This function simulates data for the exponential change-point hazard distribution with K change-points by simulating values of the exponential distribution and substituting them into the inverse hazard function. This method applies Type I right censoring at the endtime specified by the user. This function allows for up to four change-points.

Value

Dataset with n participants including a survival time and censoring indicator (0 = censored, 1 = event).

Examples

```
nochangepoint <- exp_cdfsim(n = 10, endtime = 20, theta = 0.05)
onechangepoint <- exp_cdfsim(n = 10, endtime = 20,
  theta = c(0.05, 0.01), tau = 10)
twochangepoints <- exp_cdfsim(n = 10, endtime = 20,
  theta = c(0.05, 0.01, 0.05), tau = c(8, 12))

# Pay attention to how you parameterize your model!
# This simulates a decreasing hazard
set.seed(7830)
decreasingHazard <- exp_cdfsim(n = 10, endtime = 20,
  theta = c(0.5, 0.2, 0.01), tau = c(8, 12))
# This tries to fit an increasing hazard, resulting in biased estimates
cp2.nll <- function(par, tau = tau, dta = dta){
  theta1 <- par[1]
  theta2 <- par[2]
  theta3 <- par[3]
  ll <- log(theta1) * sum(dta$time < tau[1])+
        log(theta2) * sum((tau[1] <= dta$time) * (dta$time < tau[2])) +
        log(theta3) * sum((dta$time >= tau[2]) * dta$censor) -
        theta1 * sum(dta$time * (dta$time < tau[1]) +
          tau[1] * (dta$time >= tau[1])) -
        theta2 * sum((dta$time - tau[1]) * (dta$time >= tau[1]) *
          (dta$time < tau[2]) + (tau[2] - tau[1]) * (dta$time >= tau[2])) -
        theta3 * sum((dta$time - tau[2]) * (dta$time >= tau[2]))
  return(-ll)
}
optim(par = c(0.001, 0.1, 0.5), fn = cp2.nll,
      tau = c(8, 12), dta = decreasingHazard)
```

exp_icdf *Inverse CDF for the exponential distribution*

Description

exp_icdf simulates values from the inverse CDF of the exponential distribution.

Usage

```
exp_icdf(n, theta)
```

Arguments

n	Number of output exponential values
theta	Scale parameter θ

Details

This function uses the exponential distribution of the form

$$f(t) = \theta exp(-\theta t)$$

to get the inverse CDF

$$F^(-1)(u) = (-log(1-u))/\theta$$

where u is a uniform random variable. It can be implemented directly and is also called by the function exp_memsim.

Value

Output is a value or a vector of values from the exponential distribution.

Examples

```
simdta <- exp_icdf(n = 10, theta = 0.05)
```

`exp_memsim` *Memoryless simulation for the exponential change-point hazard distribution*

Description

`exp_memsim` simulates time-to-event data from the exponential change-point hazard distribution by implementing the memoryless method.

Usage

```
exp_memsim(n, endtime, theta, tau = NA)
```

Arguments

`n`	Sample size
`endtime`	Maximum study time, point at which all participants are censored
`theta`	Scale parameter θ
`tau`	Change-point(s) τ

Details

This function simulates time-to-event data between K change-points from independent exponential distributions using the inverse CDF implemented in `exp_icdf`. This method applies Type I right censoring at the endtime specified by the user.

Value

Dataset with n participants including a survival time and censoring indicator (0 = censored, 1 = event).

Examples

```
nochangepoint <- exp_memsim( n = 10, endtime = 20, theta = 0.05)
onechangepoint <- exp_memsim(n = 10, endtime = 20,
  theta = c(0.05, 0.01), tau = 10)
twochangepoints <- exp_memsim(n = 10, endtime = 20,
  theta = c(0.05, 0.01, 0.05), tau = c(8, 12))

# Pay attention to how you parameterize your model!
# This simulates a decreasing hazard
set.seed(1245)
decreasingHazard <- exp_memsim(n = 10, endtime = 20,
  theta = c(0.05, 0.02, 0.01), tau = c(8, 12))
# This tries to fit an increasing hazard, resulting in biased estimates
cp2.nll <- function(par, tau = tau, dta = dta){
  theta1 <- par[1]
  theta2 <- par[2]
```

```
  theta3 <- par[3]
  ll <- log(theta1) * sum(dta$time < tau[1])+
        log(theta2) * sum((tau[1] <= dta$time) * (dta$time < tau[2])) +
        log(theta3) * sum((dta$time >= tau[2]) * dta$censor) -
        theta1 * sum(dta$time * (dta$time < tau[1]) +
          tau[1] * (dta$time >= tau[1])) -
        theta2 * sum((dta$time - tau[1]) * (dta$time >= tau[1]) *
          (dta$time < tau[2]) + (tau[2] - tau[1]) * (dta$time >= tau[2])) -
        theta3 * sum((dta$time - tau[2]) * (dta$time >= tau[2]))
  return(-ll)
}
optim(par = c(0.001, 0.1, 0.5), fn = cp2.nll,
      tau = c(8, 12), dta = decreasingHazard)
```

`weib_cdfsim` *Inverse CDF simulation for the Weibull change-point hazard distribution*

Description

`weib_cdfsim` simulates time-to-event data from the Weibull change-point hazard distribution by implementing the inverse CDF method.

Usage

```
weib_cdfsim(n, endtime, gamma, theta, tau = NA)
```

Arguments

`n`	Sample size
`endtime`	Maximum study time, point at which all participants are censored
`gamma`	Shape parameter γ
`theta`	Scale parameter θ
`tau`	Change-point(s) τ

Details

This function simulates data from the Weibull change-point hazard distribution with K change-points by simulating values of the exponential distribution and substituting them into the inverse hazard function. This method applies Type I right censoring at the endtime specified by the user. This function allows for up to four change-points and γ is held constant.

Value

Dataset with n participants including a survival time and censoring indicator (0 = censored, 1 = event).

Examples

```
nochangepoint <- weib_cdfsim(n = 10, endtime = 20, gamma = 2,
  theta = 0.5)
onechangepoint <- weib_cdfsim(n = 10, endtime = 20, gamma = 2,
  theta = c(0.05, 0.01), tau = 10)
twochangepoints <- weib_cdfsim(n = 10, endtime = 20, gamma = 2,
  theta = c(0.05, 0.01, 0.05), tau = c(8, 12))

#' # Pay attention to how you parameterize your model!
# This simulates an increasing hazard
set.seed(9945)
increasingHazard <- weib_cdfsim(n = 100, endtime = 20, gamma = 2,
  theta = c(0.001, 0.005, 0.02), tau = c(8, 12))
# This tries to fit a decreasing hazard, resulting in biased estimates
cp2.nll <- function(par, tau = tau, gamma = gamma, dta = dta){
  theta1 <- par[1]
  theta2 <- par[2]
  theta3 <- par[3]
  ll <- (gamma - 1) * sum(dta$censor * log(dta$time)) +
    log(theta1) * sum((dta$time < tau[1])) +
    log(theta2) * sum((tau[1] <= dta$time) * (dta$time < tau[2])) +
    log(theta3) * sum((dta$time >= tau[2]) * dta$censor) -
    (theta1/gamma) * sum((dta$time^gamma) * (dta$time < tau[1]) +
      (tau[1]^gamma) * (dta$time >= tau[1])) -
    (theta2/gamma) * sum((dta$time^gamma - tau[1]^gamma) *
      (dta$time >= tau[1]) * (dta$time<tau[2]) +
      (tau[2]^gamma - tau[1]^gamma) * (dta$time >= tau[2])) -
    (theta3/gamma) * sum((dta$time^gamma - tau[2]^gamma) *
      (dta$time >= tau[2]))
  return(-ll)
}
optim(par = c(0.2, 0.02, 0.01), fn = cp2.nll,
  tau = c(8, 12), gamma = 2,
  dta = increasingHazard)
```

`weib_icdf` *Inverse CDF value generation for the Weibull distribution*

Description

`weib_icdf` returns a value from the Weibull distribution by using the inverse CDF.

Usage

```
weib_icdf(n, gamma, theta)
```

Arguments

n	Number of output Weibull values
gamma	Shape parameter γ
theta	Scale parameter θ

Details

This function uses the Weibull density of the form

$$f(t) = \theta t^(\gamma - 1) exp(-\theta/\gamma t^(\gamma))$$

to get the inverse CDF

$$F^(- 1)(u) = (-\gamma/\theta log(1-u))^(1/\gamma)$$

where u is a uniform random variable. It can be implemented directly and is also called by the function weib_memsim.

Value

Output is a value or vector of values from the Weibull distribution.

Examples

```
simdta <- weib_icdf(n = 10, theta = 0.05, gamma = 2)
```

---

weib_memsim *Memoryless simulation for the Weibull change-point hazard distribution*

Description

weib_memsim simulates time-to-event data from the Weibull change-point hazard distribution by implementing the memoryless method.

Usage

```
weib_memsim(n, endtime, gamma, theta, tau = NA)
```

Arguments

n	Sample size
endtime	Maximum study time, point at which all participants are censored
gamma	Shape parameter γ
theta	Scale parameter θ
tau	Change-point(s) τ

Details

This function simulates time-to-event data between K change-points τ from independent Weibull distributions using the inverse Weibull CDF implemented in weib_icdf. This method applies Type I right censoring at the endtime specified by the user. γ is held constant.

Value

Dataset with n participants including a survival time and censoring indicator (0 = censored, 1 = event).

Examples

```
nochangepoint <- weib_memsim(n = 10, endtime = 20, gamma = 2,
  theta = 0.05)
onechangepoint <- weib_memsim(n = 10, endtime = 20, gamma = 2,
  theta = c(0.05, 0.01), tau = 10)
twochangepoints <- weib_memsim(n = 10, endtime = 20, gamma = 2,
  theta = c(0.05, 0.01, 0.05), tau = c(8, 12))

# Pay attention to how you parameterize your model!
# This simulates an increasing hazard
set.seed(5738)
increasingHazard <- weib_memsim(n = 100, endtime = 20, gamma = 2,
  theta = c(0.001, 0.005, 0.02), tau = c(8, 12))
# This tries to fit a decreasing hazard, resulting in biased estimates
cp2.nll <- function(par, tau = tau, gamma = gamma, dta = dta){
  theta1 <- par[1]
  theta2 <- par[2]
  theta3 <- par[3]
  ll <- (gamma - 1) * sum(dta$censor * log(dta$time)) +
    log(theta1) * sum((dta$time < tau[1])) +
    log(theta2) * sum((tau[1] <= dta$time) * (dta$time < tau[2])) +
    log(theta3) * sum((dta$time >= tau[2]) * dta$censor) -
    (theta1/gamma) * sum((dta$time^gamma) * (dta$time < tau[1]) +
      (tau[1]^gamma) * (dta$time >= tau[1])) -
    (theta2/gamma) * sum((dta$time^gamma - tau[1]^gamma) *
      (dta$time >= tau[1]) * (dta$time<tau[2]) +
      (tau[2]^gamma - tau[1]^gamma) * (dta$time >= tau[2])) -
    (theta3/gamma) * sum((dta$time^gamma - tau[2]^gamma) *
      (dta$time >= tau[2]))
  return(-ll)
}
optim(par = c(0.2, 0.02, 0.01), fn = cp2.nll,
  tau = c(8, 12), gamma = 2,
  dta = increasingHazard)
```

Index