Package ‘eventTrack’

February 22, 2025

Type Package

Title Event Prediction for Time-to-Event Endpoints

Version 1.0.4

Date 2025-02-21

Maintainer Kaspar Rufibach <kaspar.rufibach@gmail.com>

Depends survival, muhaz, R (>= 3.5.0)

Suggests knitr, rmarkdown, rpact (>= 3.3.1), fitdistrplus, gestate

VignetteBuilder knitr

Description Implements the hybrid framework for event prediction described in Fang & Zheng (2011, <doi:10.1016/j.cct.2011.05.013>). To estimate the survival function the event prediction is based on, a piecewise exponential hazard function is fit to the time-to-event data to infer the potential change points. Prior to the last identified change point, the survival function is estimated using Kaplan-Meier, and the tail after the change point is fit using piecewise exponential.

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Author Kaspar Rufibach [aut, cre]

Repository CRAN

Date/Publication 2025-02-22 11:00:02 UTC

Contents

`eventTrack-package` *Event Prediction for Time-to-Event Endpoints*

Description

Implements the hybrid framework for event prediction described in Fang & Zheng (2011). To estimate the survival function the event prediction is based on, a piecewise Exponential hazard function is fit to the time-to-event data to infer the potential change points. Prior to the last identified change point, the survival function is estimated using Kaplan-Meier, and the tail after the change point is fit using piecewise Exponential. The Weibull version described in Fang and Zheng (2011) is not implemented here.

An example has been presented in this talk: `https://baselbiometrics.github.io/home/docs/talks/20160428/1_Rufibach.pdf`.

Details

Package:	eventTrack
Type:	Package
Version:	1.0.4
Date:	2025-02-21
License:	GPL (>=2)
LazyLoad:	yes

Validation status

The functions in this package have been written by Kaspar Rufibach and are generally to be considered experimental.

Author(s)

Kaspar Rufibach (maintainer)
`<kaspar.rufibach@gmail.com>`

Examples

```
# --------------------------------------------------
# simulate data
# --------------------------------------------------
set.seed(2021)
n <- 600
time0 <- rexp(n, rate = log(2) / 20)
cens <- rexp(n, rate = log(2) / 50)
time <- pmin(time0, cens)
event <- as.numeric(time0 < cens)
accrual_after_ccod <- 1:(n - length(time)) / 30

# --------------------------------------------------
# compute hybrid estimate and predict timepoint
# --------------------------------------------------
plot(survfit(Surv(time, event) ~ 1), mark = "", xlim = c(0, 200),
     ylim = c(0, 1), conf.int = FALSE, xaxs = "i", yaxs = "i",
     main = "estimated survival functions", xlab = "time",
     ylab = "survival probability", col = grey(0.75), lwd = 5)

# how far out should we predict monthly number of events?
future.units <- 15
tab <- matrix(NA, ncol = 2, nrow = future.units)
tab[, 1] <- 1:nrow(tab)
ts <- seq(0, 100, by = 0.01)

# --------------------------------------------------
# starting from a piecewise Exponential hazard with
# K = 5 change points, infer the last "significant"
# change point
# --------------------------------------------------
pe5 <- piecewiseExp_MLE(time = time, event = event, K = 5)
pe5.tab <- piecewiseExp_test_changepoint(peMLE = pe5, alpha = 0.05)
cp.select <- max(c(0, as.numeric(pe5.tab[, "established change point"])), na.rm = TRUE)

# the function predictEvents takes as an argument any survival function
# hybrid exponential with cp.select
St1 <- function(t0, time, event, cp){
     return(hybrid_Exponential(t0 = t0, time = time, event = event,
     changepoint = cp))}

pe1 <- predictEvents(time = time, event = event,
          St = function(t0){St1(t0, time = time,
          event = event, cp.select)}, accrual_after_ccod,
          future.units = future.units)
tab[, 2] <- pe1[, 2]
lines(ts, St1(ts, time, event, cp.select), col = 2, lwd = 2)

# --------------------------------------------------
# compute exact date when we see targeted number of events
# for hybrid Exponential model, through linear interpolation
# --------------------------------------------------
```

```
exactDatesFromMonths(predicted = tab, 450)
```

---

bootstrapTimeToEvent *Bootstrap the predicted time when a given number of events is reached, for hybrid Exponential model*

Description

Bootstrap the predicted time when a given number of events is reached, based on the hybrid Exponential model.

Usage

```
bootstrapTimeToEvent(time, event, interim.gates, future.units, n, M0 = 1000,
                             K0 = 5, alpha = 0.05, accrual, seed = 2014)
```

Arguments

`time`	Event times.
`event`	Censoring indicator, 0 = censored, 1 = event.
`interim.gates`	Number of events for which confidence intervals should be computed. May be a vector.
`future.units`	Number of months for which predictions are to be made.
`n`	Size of the bootstrap samples to be drawn from `surv.obj`.
`M0`	Number of bootstrap samples to be drawn.
`K0`	Number of changepoints for the piecewise constant hazard.
`alpha`	Familywise error rate for the sequential test.
`accrual`	Vector of dates when future patients enter the study. As for the specification, assume 50 pts are to be recruited in Dec 2013 and 20pts in Jan 2014, then use `accrual <- c(rep(as.Date("2013-12-01"), 50), rep(as.Date("2014-01-01"), 20))`. Leave as `NULL` if accrual for the study is completed.
`seed`	Seed for generation of bootstrap samples.

Value

A list containing the following objects:

`pe.MLEs`	Piecewise Exponential MLE object for each bootstrap sample, output of function `piecewiseExp_MLE`.
`pe.tabs`	Result of sequential test for each bootstrap sample, output of function `piecewiseExp_test_changepoint`
`changepoints`	For each bootstrap sample, changepoint as resulting from sequential test.
`estS`	Estimated survival function for each bootstrap sample.
`event.dates`	Event dates for each bootstrap sample, where columns relate to the number of events in `interim.gates`.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

References

Rufibach, K. (2016). *Event projection: quantify uncertainty and manage expectations of broader teams.* Slides for talk given in Basel Biometric Section Seminar on 28th April 2016. https://baselbiometrics.github.io/home/docs/talks/20160428/1_Rufibach.pdf.

Examples

```
## Not run:
# --------------------------------------------------
# simulate data for illustration
# --------------------------------------------------
set.seed(2021)
n <- 600
time <- rexp(n, rate = log(2) / 20)
event <- sample(round(runif(n, 0, 1)))
accrual_after_ccod <- 1:(n - length(time)) / 30

# --------------------------------------------------
# run bootstrap, for M0 = 3 only, for illustration
# tune parameters for your own example
# --------------------------------------------------
boot1 <- bootstrapTimeToEvent(time, event,
          interim.gates = c(330, 350), future.units = 50, n = length(time),
          M0 = 3, K0 = 5, alpha = 0.05, accrual = accrual_after_ccod,
          seed = 2014)

# median of bootstrap samples:
apply(boot1$event.dates, 2, median)

## End(Not run)
```

---

bootSurvivalSample *Bootstrap survival data*

Description

Generate bootstrap samples from a survival object, for right-censored data. See Efron (1981) and Akritas (1986) for details.

Usage

```
bootSurvivalSample(surv.obj, n, M = 1000)
```

Arguments

surv.obj	Surv object containing survival times and censoring indicator.
n	Size of the bootstrap samples to be drawn from surv.obj.
M	Number of bootstrap samples to be drawn.

Value

Matrix with bootstrap samples as columns.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

References

Akritas, M.G. (1986). Bootstrapping the Kaplan-Meier Estimator. *JASA*, **81**, 1032-1038

Efron, B. (1981). Censored Data and the Bootstrap. *JASA*, **76**, 312-319.

Examples

```
# --------------------------------------------------
# simulate data for illustration
# --------------------------------------------------
set.seed(2021)
n <- 600
time <- rexp(n, rate = log(2) / 20)
event <- sample(round(runif(n, 0, 1)))

# --------------------------------------------------
# draw 20 bootstrap samples of size 10
# --------------------------------------------------
bootSurvivalSample(surv.obj = Surv(time, event), n = 10, M = 20)
```

---

exactDatesFromMonths *Compute exact timepoint when a certain number of events is reached, based on monthly number of events*

Description

Based on monhtly number of events, compute exact day when the pre-specified number of events happens, using simple linear interpolation.

Usage

```
exactDatesFromMonths(predicted, nevent)
```

Arguments

predicted	`data.frame`, with first column monthly dates and second column the number of events in each month.
nevent	Number of targeted events.

Value

The exact date.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

---

hybrid_Exponential *Estimate survival function, as hybrid between Kaplan-Meier and Exponential tail*

Description

This function estimates the values of the survival function as a hybrid of Kaplan-Meier for times smaller than the specified change point and an Exponential fit to the tail of the survival function. The Exponential tail fit is computed assuming a piecewise constant hazard with one change point.

Usage

```
hybrid_Exponential(t0, time = time, event = event, changepoint)
```

Arguments

t0	Value at which to compute value of survival function. Can be a vector.
time	Event times, censored or observed, in months.
event	Censoring indicator, 1 for event, 0 for censored.
changepoint	Pre-specified change point.

Value

A vector of the same dimension as `t0` containing the values of the estimated survival function at `t0`.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

References

Fang, L., Zheng, S. (2011). A hybrid approach to predicting events in clinical trials with time-to-event outcomes. *Contemp. Clin. Trials*, **32**, 755–759.

Goodman, M.S., Li, Y., Tiwari, R.C. (2011). Detecting multiple change points in piecewise constant hazard functions. *J. Appl. Stat*, **38(11)**, 2523–2532.

Rufibach, K. (2016). *Event projection: quantify uncertainty and manage expectations of broader teams.* Slides for talk given in Basel Biometric Section Seminar on 28th April 2016. https://baselbiometrics.github.io/home/docs/talks/20160428/1_Rufibach.pdf.

Examples

```
# ---------------------------------------------------
# simulate data
# ---------------------------------------------------
set.seed(2021)
n <- 600
time0 <- rexp(n, rate = log(2) / 20)
cens <- rexp(n, rate = log(2) / 50)
time <- pmin(time0, cens)
event <- as.numeric(time0 < cens)
accrual_after_ccod <- 1:(n - length(time)) / 30

# ---------------------------------------------------
# compute hybrid estimate and predict timepoint
# ---------------------------------------------------
plot(survfit(Surv(time, event) ~ 1), mark = "", xlim = c(0, 200),
     ylim = c(0, 1), conf.int = FALSE, xaxs = "i", yaxs = "i",
     main = "estimated survival functions", xlab = "time",
     ylab = "survival probability", col = grey(0.75), lwd = 5)

# how far out should we predict monthly number of events?
future.units <- 15
tab <- matrix(NA, ncol = 2, nrow = future.units)
tab[, 1] <- 1:nrow(tab)
ts <- seq(0, 100, by = 0.01)

# ---------------------------------------------------
# starting from a piecewise Exponential hazard with
# K = 5 change points, infer the last "significant"
# change point
# ---------------------------------------------------
pe5 <- piecewiseExp_MLE(time = time, event = event, K = 5)
pe5.tab <- piecewiseExp_test_changepoint(peMLE = pe5, alpha = 0.05)
cp.select <- max(c(0, as.numeric(pe5.tab[, "established change point"])), na.rm = TRUE)

# the function predictEvents takes as an argument any survival function
# hybrid exponential with cp.select
St1 <- function(t0, time, event, cp){
     return(hybrid_Exponential(t0 = t0, time = time, event = event,
     changepoint = cp))}
```

```
pe1 <- predictEvents(time = time, event = event,
          St = function(t0){St1(t0, time = time,
          event = event, cp.select)}, accrual_after_ccod,
          future.units = future.units)
tab[, 2] <- pe1[, 2]
lines(ts, St1(ts, time, event, cp.select), col = 2, lwd = 2)

# -------------------------------------------------
# compute exact date when we see targeted number of events
# for hybrid Exponential model, through linear interpolation
# -------------------------------------------------
exactDatesFromMonths(predicted = tab, 450)
```

---

kaplanMeier_at_t0 *Compute value of Kaplan-Meier estimate at a given time*

Description

Compute value of Kaplan-Meier estimate at a given time t_0.

Usage

```
kaplanMeier_at_t0(time, event, t0)
```

Arguments

time	Event times, censored or observed.
event	Censoring indicator, 1 for event, 0 for censored.
t0	Vector (or single number) of time points to compute confidence interval for.

Value

Matrix with values of Kaplan-Meier estimate at t_0.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

Examples

```
# use Acute Myelogenous Leukemia survival data contained in package 'survival'
time <- leukemia[, 1]
status <- leukemia[, 2]

tmp <- Surv(time, status) ~ 1
plot(survfit(tmp, conf.type = "none"), mark = "/", col = 1:2)
kaplanMeier_at_t0(time, status, t0 = c(10, 25, 50))
```

`lambda_j_Exp` *Compute lambda_j*

Description

For given change points $\tau_1, \ldots, \tau_k$, compute the profile maximum likelihood estimates $\lambda_1, \ldots, \lambda_{k+1}$ for the values of the piecewise constant hazard function in a piecewise Exponential survival model. Standard errors for these estimates are provided as well, based on standard maximum (profile) maximum likelihood theory.

Usage

```
lambda_j_Exp(tau, time, event)
```

Arguments

`tau`	Single number or vector with values of change points.
`time`	Event times, censored or observed, in months.
`event`	Censoring indicator, 1 for event, 0 for censored.

Value

A list containing the following elements:

`aj`	The esimtates of the λ_j.
`hess.diag`	The diagonal of the corresponding Hessian matrix. Note that the off-diagonal elements of the Hessian are all equal to 0.

Note

This function is not intended to be invoked by the end user.

Author(s)

Kaspar Rufibach (maintainer)
`<kaspar.rufibach@roche.com>`

References

Fang, L., Zheng, S. (2011). A hybrid approach to predicting events in clinical trials with time-to-event outcomes. *Contemp. Clin. Trials*, **32**, 755–759.

Goodman, M.S., Li, Y., Tiwari, R.C. (2011). Detecting multiple change points in piecewise constant hazard functions. *J. Appl. Stat*, **38(11)**, 2523–2532.

`piecewiseExp_MLE` *Estimate hazard function in piecewise Exponential survival model*

Description

This function estimates the values of the hazard function and the change points in a piecewise Exponential survival model. The number of change points K needs to be pre-specified.

Usage

```
piecewiseExp_MLE(time, event, K)
```

Arguments

`time`	Event times, censored or observed, in months.
`event`	Censoring indicator, 1 for event, 0 for censored.
`K`	Number of change points to be used in the model.

Value

A list containing the following objects:

`tau`	The maximum likelihood estimates of the change points.
`lambda`	The maximum likelihood estimates of the value of the hazard function.
`lambda.SE`	The standard errors of `lambda`.
`K`	Number of change points to be used in the model.

Author(s)

Kaspar Rufibach (maintainer)
`<kaspar.rufibach@roche.com>`

References

Fang, L., Zheng, S. (2011). A hybrid approach to predicting events in clinical trials with time-to-event outcomes. *Contemp. Clin. Trials*, **32**, 755–759.

Goodman, M.S., Li, Y., Tiwari, R.C. (2011). Detecting multiple change points in piecewise constant hazard functions. *J. Appl. Stat*, **38(11)**, 2523–2532.

Examples

```
# see vignette
```

piecewiseExp_profile_loglik_tau

Profile maximum log-likelihood function for change points in piecewise Exponential survival model

Description

In a piecewise Exponential survival model, the estimates for the value of the hazard function can be given explicitly, see Fang and Su (2011), and can be computed using `lambda_j_Exp`. The values of the change points can then be computed using the profile log-likelihood function. The function `piecewiseExp_profile_loglik_tau` computes the value of this profile log-likelihood function and can be used together with `optim` to compute the change points $\tau_1, \ldots, \tau_k$.

Usage

```
piecewiseExp_profile_loglik_tau(tau, time, event)
```

Arguments

`tau`	Single number or vector with values of change points.
`time`	Event times, censored or observed, in months.
`event`	Censoring indicator, 1 for event, 0 for censored.

Value

Value of the profile likelihood function.

Note

This function is not intended to be invoked by the end user.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

References

Fang, L., Zheng, S. (2011). A hybrid approach to predicting events in clinical trials with time-to-event outcomes. *Contemp. Clin. Trials*, **32**, 755–759.

Goodman, M.S., Li, Y., Tiwari, R.C. (2011). Detecting multiple change points in piecewise constant hazard functions. *J. Appl. Stat*, **38(11)**, 2523–2532.

---

piecewiseExp_test_changepoint

Wald test to infer change point in piecewise Exponential survival model

---

Description

This function implements the Wald test described in Goodman et al (2011) and applied in Fang & Zheng (2011) to infer a change point in an estimated piecewise Exponential hazard function. Adjusts for sequential testing.

Usage

```
piecewiseExp_test_changepoint(peMLE, alpha = 0.05)
```

Arguments

peMLE	Piecewise Exponential hazard function estimate, as generated by piecewiseExp_MLE.
alpha	Overall significance level. Will be adjusted for sequential testing internally, see function output.

Value

A data.frame containing the sequential test results.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

References

Fang, L., Zheng, S. (2011). A hybrid approach to predicting events in clinical trials with time-to-event outcomes. *Contemp. Clin. Trials*, **32**, 755–759.

Goodman, M.S., Li, Y., Tiwari, R.C. (2011). Detecting multiple change points in piecewise constant hazard functions. *J. Appl. Stat*, **38(11)**, 2523–2532.

Examples

```
# see vignette
```

predictEvents	*Compute timepoint when a certain number of events in a time-to-event study is reached*

Description

Based on a specified survival function and potential future accrual, compute for each month in the future the number of expected events reached by then.

Usage

```
predictEvents(time, event, St, accrual, future.units = 50)
```

Arguments

`time`	Event times, censored or observed, in months.
`event`	Censoring indicator, 1 for event, 0 for censored.
`St`	Function that specifies the survival function to be used.
`accrual`	`NULL` if trial is fully accrued. For potential future accural, see example in `hybrid_Exponential` of how to specify it.
`future.units`	Number of future months to compute prediction for.

Value

A `data.frame` with the months and corresponding expected events.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

References

Fang, L., Zheng, S. (2011). A hybrid approach to predicting events in clinical trials with time-to-event outcomes. *Contemp. Clin. Trials*, **32**, 755–759.

Goodman, M.S., Li, Y., Tiwari, R.C. (2011). Detecting multiple change points in piecewise constant hazard functions. *J. Appl. Stat*, **38(11)**, 2523–2532.

Examples

```
# see vignette
```

`predictEventsUncond` *compute expected number of events based on a fixed survival function and with no recruited patients yet*

Description

Based on a specified survival function and future accrual, compute for each month in the future the number of expected events reached by then, based on a fixed pre-specified survival function.

Usage

```
predictEventsUncond(St, accrual, future.units = 50)
```

Arguments

`St`	Function that specifies the survival function to be used.
`accrual`	`NULL` if trial is fully accrued. For potential future accural, see the example below how to specify it.
`future.units`	Number of future months to compute prediction for.

Value

A `data.frame` with the months and corresponding expected events.

Author(s)

Kaspar Rufibach (maintainer)
<kaspar.rufibach@roche.com>

Examples

```
# compute date when 380 events are reached:
nevent <- 380
n <- 800
accrual.month <- 50
accrual <- rep(seq(1, 16), each = accrual.month)
rate <- log(2) / 12

St1 <- function(t0){
      res <- 1 - pexp(t0, rate = rate)
      return(res)
}
pred1 <- predictEventsUncond(St = St1, accrual, future.units = 25)
pred1
exactDatesFromMonths(predicted = pred1, nevent)
```

Index