Package 'incubate'

October 13, 2022

Title Parametric Time-to-Event Analysis with Variable Incubation Phases

Version 1.2.0

Date 2022-07-25

Description Fit parametric models for time-to-event data that show an initial 'incubation period', i.e., a variable delay phase where the hazard is zero. The delayed Weibull distribution serves as the foundational data model. The specific method of 'MPSE' (maximum product of spacings estimation) is used for parameter estimation. Bootstrap confidence intervals for parameters and significance tests in a two group setting are provided.

License LGPL (>= 3)

Encoding UTF-8

LazyData true

Imports future (>= 1.21), future.apply (>= 1.6), glue (>= 1.4), MASS, purrr (>= 0.3), rlang (>= 0.4), stats, survival, tibble

Suggests boot, future.callr, ggplot2 (>= 3.3), knitr, testthat (>= 3.0.0)

URL https://gitlab.com/imb-dev/incubate/

BugReports https://gitlab.com/imb-dev/incubate/-/issues/

RoxygenNote 7.2.0

Config/testthat/edition 3

Depends R (>= 2.10)

NeedsCompilation no

Author Matthias Kuhn [aut, cre] (<https://orcid.org/0000-0003-2868-5155>)

Maintainer Matthias Kuhn <matthias.kuhn@tu-dresden.de>

Repository CRAN

Date/Publication 2022-07-25 13:30:08 UTC

R topics documented:

`as_percent` *Format a number as percentage.*

Description

Internal helper function that is not exported.

Usage

```
as_percent(x, digits = 1)
```

Arguments

`x`	numeric vector to be formated as percentage
`digits`	requested number of decimal digits of the percentage

Value

number formatted as percentage character

bsDataStep — *Generate bootstrap distribution of model parameters to fitted incubate model.*

Description

Bootstrap data are here estimated coefficients from models fitted to bootstrap samples. The bootstrap data is used to make bootstrap inference in the second step. It is an internal function, the main entry point is `confint.incubate_fit()`.

Usage

```
bsDataStep(
  object,
  bs_data = c("parametric", "ordinary"),
  R,
  useBoot = FALSE,
  smd_factor = 0.25
)
```

Arguments

`object`	an `incubate_fit`-object
`bs_data`	character. Which type of bootstrap method to generate data?
`R`	integer. Number of bootstrapped model coefficient estimates
`useBoot`	flag. Do you want to use the boot-package? Default value is `FALSE`.
`smd_factor`	numeric. smooth-delay factor: influence the amount of smoothing. 0 means no smoothing at all. Default is 0.25 (as was optimal in simulation for log-quantile together with log-delay-shift = 5)

Value

bootstrap data, either as matrix or of class `boot` (depending on the `useBoot`-flag)

coef.incubate_fit — *Coefficients of a delay-model fit.*

Description

Coefficients of a delay-model fit.

Usage

```
## S3 method for class 'incubate_fit'
coef(object, group = NULL, ...)
```

Arguments

object	object that is a incubate_fit
group	character string to request the canonical parameter for one group
...	further arguments, currently not used.

Value

named coefficient vector

---

confint.incubate_fit *Confidence intervals for parameters of incubate-model fits.*

Description

Bias-corrected bootstrap confidence limits (either quantile-based or normal-approximation based) are generated. Optionally, there are also variants that use a log-transformation first. At least R=1000 bootstrap replications are recommended. Default are quantile-based confidence intervals that internally use a log-transformation.

Usage

```
## S3 method for class 'incubate_fit'
confint(
  object,
  parm,
  level = 0.95,
  R = 199L,
  bs_data,
  bs_infer = c("logquantile", "lognormal", "quantile", "quantile0", "normal",
    "normal0"),
  useBoot = FALSE,
  ...
)
```

Arguments

object	object of class incubate_fit
parm	character. Which parameters to get confidence interval for?
level	numeric. Which is the requested confidence level for the interval? Default value is 0.95
R	number of bootstrap replications. Used only if not bs_data-object is provided.
bs_data	character or bootstrap data object. If character, it specifies which type of bootstrap is requested and the bootstrap data will be generated. Data can also be provided here directly. If missing it uses parametric bootstrap.

bs_infer	character. Which type of bootstrap inference is requested to generate the confidence interval?
useBoot	logical. Delegate bootstrap confint calculation to the boot-package?
...	further arguments, currently not used.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter.

DelayedExponential *Delayed Exponential Distribution*

Description

Density, distribution function, quantile function and random generation for the delayed exponential distribution with rate-parameter.

Usage

```
dexp_delayed(x, delay, rate = 1, ...)

pexp_delayed(q, delay, rate = 1, ...)

qexp_delayed(p, delay, rate = 1, ...)

rexp_delayed(n, delay, rate = 1)
```

Arguments

x	A numeric vector of values for which to get the density.
delay	numeric. The delay, must be non-negative.
rate	numeric. The event rate, must be non-negative.
...	further arguments are passed on to the underlying non-delayed function, e.g., stats::dexp()
q	A numeric vector of quantile values.
p	A numeric vector of probabilities.
n	integer. Number of random observations requested.

Details

Additional arguments are forwarded via ... to the underlying functions of the exponential distribution in the stats-package. The numerical arguments other than n are recycled to the length of the result. Only the first elements of the logical arguments are used.

Value

dexp_delayed gives the density, pexp_delayed gives the distribution function, qexp_delayed gives the quantile function, and rexp_delayed generates a pseudo-random sample from the delayed exponential distribution.

The length of the result is determined by n for rexp_delayed, and is the maximum of the lengths of the numerical arguments for the other functions.

DelayedWeibull *Delayed Weibull Distribution*

Description

Density, distribution function, quantile function and random generation for the delayed Weibull distribution with parameters as in the Weibull distribution functions in R's stats-package, namely:

- delay
- shape
- scale (inverse of rate)

Usage

```
dweib_delayed(x, delay, shape, scale = 1, ...)

pweib_delayed(q, delay, shape, scale = 1, ...)

qweib_delayed(p, delay, shape, scale = 1, ...)

rweib_delayed(n, delay, shape, scale = 1)
```

Arguments

x	A numeric vector of values for which to get the density.
delay	numeric. The delay, must be non-negative.
shape	numeric. Shape parameter, must be positive.
scale	numeric. Scale parameter (inverse of rate), must be positive.
...	further arguments are passed on to the underlying non-delayed function, e.g., stats::dweibull()
q	A numeric vector of quantile values.
p	A numeric vector of probabilities.
n	integer. Number of random observations requested.

Details

Additional arguments are forwarded via ... to the underlying functions of the exponential distribution in the stats-package.

The numerical arguments other than n are recycled to the length of the result. Only the first elements of the logical arguments are used.

Value

dweib_delayed gives the density, pweib_delayed gives the distribution function, qweib_delayed gives the quantile function, and rweib_delayed generates a pseudo-random sample from the delayed Weibull distribution.

The length of the result is determined by n for rweib_delayed, and is the maximum of the lengths of the numerical arguments for the other functions.

---

delay_fit *Fit optimal parameters according to the objective function (either MPSE or MLE0).*

Description

The objective function carries the given data in its environment and it is to be minimized. R's standard routine stats::optim does the numerical optimization, using numerical derivatives. or the analytical solution is returned directly if available.

Usage

```
delay_fit(objFun, optim_args = NULL, verbose = 0)
```

Arguments

objFun	objective function to be minimized
optim_args	list of own arguments for optimization. If NULL it uses the default optim arguments associated to the objective function.
verbose	integer that indicates the level of verboseness. Default 0 is quiet.

Value

optimization object including a named parameter vector or NULL in case of errors during optimization

`delay_model` *Fit a delayed Exponential or Weibull model to one or two given sample(s).*

Description

Maximum product spacing is used to fit the parameters. Numerical optimization is done by `stats::optim`.

Usage

```
delay_model(
  x = stop("Specify observations!", call. = FALSE),
  y = NULL,
  distribution = c("exponential", "weibull"),
  method = c("MPSE", "MLE0"),
  bind = NULL,
  ties = c("density", "equidist", "random", "error"),
  optim_args = NULL,
  verbose = 0
)
```

Arguments

`x`	numeric. observations of 1st group. Can also be a list of data from two groups.
`y`	numeric. observations from 2nd group
`distribution`	character. Which delayed distribution is assumed? Exponential or Weibull.
`method`	character. Which method to fit the model? 'MPSE' = maximum product of spacings estimation *or* 'MLE0' = standard maximum likelihood estimation
`bind`	character. parameter names that are bind together in 2-group situation.
`ties`	character. How to handle ties.
`optim_args`	list. optimization arguments to use. Use `NULL` to use the data-dependent default values.
`verbose`	integer. level of verboseness. Default 0 is quiet.

Value

`incubate_fit` the delay-model fit object. Or `NULL` if optimization failed (e.g. too few observations).

`estimRoundingError` *Estimate rounding error based on given sample of metric values The idea is to check at which level of rounding the sample values do not change.*

Description

Estimate rounding error based on given sample of metric values The idea is to check at which level of rounding the sample values do not change.

Usage

```
estimRoundingError(obs, roundDigits = seq.int(-4L, 6L), maxObs = 100L)
```

Arguments

`obs`	numeric. Metric values from a sample to estimate the corresponding rounding error
`roundDigits`	integer. Which level of rounding to test? Negative numbers round to corresponding powers of 10
`maxObs`	integer. How many observations to consider at most? If the provided sample has more observations a sub-sample is used.

Value

estimated rounding error

`getDist` *Get delay distribution function*

Description

Get delay distribution function

Usage

```
getDist(
  distribution = c("exponential", "weibull"),
  type = c("cdf", "prob", "density", "random", "param"),
  twoGroup = FALSE,
  bind = NULL
)
```

Arguments

distribution	character(1). delay distribution.
type	character(1). type of function, cdf: cumulative distribution function, density or random function
twoGroup	logical(1). Do we have two groups?
bind	character. Names of parameters that are bind between the two groups.

Value

selected distribution function or parameter names

---

getPars *Extract the parameters for the specified group.*

Description

The parameters of the requested group are named using the canonical parameter names of the distribution.

Usage

```
getPars(par, group = "x", twoGroup, oNames, bind)
```

Arguments

par	named parameters (as simple vector or as list)
group	character. Which group to extract parameters for?
twoGroup	flag. Is it a two-group setting?
oNames	character. Original parameter names from distribution.
bind	character. Which parameters are bind together in a two-group setting?

Details

For a one-group setting or when group=NULL it simply returns the given parameter. This is an internal helper function used in coef.incubate_fit(), bsDataStep() and in the factory method objFunFactory() below.

Value

named vector of parameters from the relevant group

incubate *Incubate Package*

Description

Estimation and statistical tests on parameters in parametric time-to-event analyses with delay.

objFunFactory *Factory method for objective function, either according to maximum product of spacings estimation ('MPSE') or according to standard maximum likelihood estimation ('MLE0').*

Description

Given the observed data this factory method produces an MPSE objective function implementation which is the negative of the MPSE-criterion H or the negative log-likelihood for MLE.

Usage

```
objFunFactory(
  x,
  y = NULL,
  method = c("MPSE", "MLE0"),
  distribution = c("exponential", "weibull"),
  bind = NULL,
  ties = c("density", "equidist", "random", "error"),
  verbose = 0L
)
```

Arguments

x	numeric. observations
y	numeric. observations in second group.
method	character(1). Specifies the method for which to build the objective function. Default value is MPSE. MLE0 is the standard MLE-method, calculating the likelihood function as the product of density values
distribution	character(1). delayed distribution family
bind	character. parameter names that are bind together (i.e. equated) between both groups
ties	character. How to handle ties within data of a group.
verbose	integer flag. How much verbosity in output? The higher the more output. Default value is 0 which is no output.

Details

From the observations, negative or infinite values are discarded. In any case, the objective function is to be minimized.

Value

the objective function (e.g., the negative MPSE criterion) for given choice of model parameters or `NULL` upon errors

---

`power_diff` *Power simulation function for a two-group comparison of the delay parameter.*

Description

There are two ways of operation:

1. `power=NULL` Given sample size `n` it simulates the power.
2. `n=NULL` Given a power an iterative search is started to find a suitable `n` within a specified range.

Usage

```
power_diff(
  distribution = c("exponential", "weibull"),
  param = "delay",
  test = c("bootstrap", "pearson", "moran", "lr", "lr_pp"),
  eff = stop("Provide parameters for both group that reflect the effect!"),
  n = NULL,
  r = 1,
  sig.level = 0.05,
  power = NULL,
  nPowerSim = 1600,
  R = 201,
  nRange = c(5, 50)
)
```

Arguments

`distribution`	character. Which assumed distribution is used for the power calculation.
`param`	character. Parameter name(s) for which to simulate the power.
`test`	character. Which test to use for this power estimation?
`eff`	list. The two list elements contain the model parameters (as understood by the delay-distribution functions provided by this package) for the two groups.
`n`	integer. Number of observations per group for the power simulation or `NULL` when n is to be estimated for a given power.

r	numeric. Ratio of both groups sizes, ny / nx. Default value is 1, i.e., balanced group sizes. Must be positive.
sig.level	numeric. Significance level. Default is 0.05.
power	numeric. NULL when power is to be estimated for a given sample size or a desired power is specified (and n is estimated).
nPowerSim	integer. Number of simulation rounds. Default value 1600 yields a standard error of 0.01 for power if the true power is 80%.
R	integer. Number of bootstrap samples for test of difference in parameter within each power simulation. It affects the resolution of the P-value for each simulation round. A value of around R=200 gives a resolution of 0.5% which might be enough for power analysis.
nRange	integer. Admissible range for sample size when power is pre-specified and sample size is requested.

Details

In any case, the distribution, the parameters that are tested for, the type of test and the effect size (eff=) need to be specified. The more power simulation rounds (parameter nPowerSim=) the more densely the space of data according to the specified model is sampled.

Note that this second modus (when n is estimated) is computationally quite heavy. The iterative search for n uses some heuristics and the estimated sample size might actually give a different power-level. It is important to check the stated power in the output. The search algorithm comes to results closer to the power aimed at when the admissible range for sample size (nRange=) is chosen sensibly. In case the estimated sample size and the achieved power is too high it might pay off to rerun the function with an adapted admissible range.

Value

List of results of power simulation. Or NULL in case of errors.

---

stankovic *Survival of mice with glioma under different treatments.*

Description

A dataset from an animal experiment described in Stankovic (2018), shown in Figure 6J and 6K.

Usage

```
stankovic
```

Format

Figure The figure in the publication where the data is shown

Time Survival in days

Status Right-censor status: 1 means observed event

Group Experimental group identifier

Colour Colour used in the Stankovic publication to mark this group

Details

The data were read directly from the survival plots in the publication with the help of Plot Digitizer, version 2.6.9.

Source

Dudvarski Stankovic N, Bicker F, Keller S, et al. EGFL7 enhances surface expression of integrin a5b1 to promote angiogenesis in malignant brain tumors. EMBO Mol Med. 2018;10(9):e8420. doi:10.15252/emmm.201708420 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6127886/

`test_diff` *Test the difference for delay model parameter(s) between two uncorrelated groups, based on maximum product of spacings estimation (MPSE).*

Description

It is in fact a model comparison between a null model where the parameters are enforced to be equal and an unconstrained full model. As test statistic we use twice the difference in best (=lowest) objective function value, i.e. 2 * (`val_0` - `val_1`). This is reminiscent of a likelihood ratio test statistic albeit the objective function is not a negative log-likelihood but the negative of the maximum product spacing metric.

Usage

```
test_diff(
  x,
  y = stop("Provide data for group y!"),
  distribution = c("exponential", "weibull"),
  param = "delay",
  R = 400,
  ties = c("density", "equidist", "random", "error"),
  type = c("all", "bootstrap", "gof", "moran", "pearson", "lr", "lr_pp"),
  verbose = 0
)
```

Arguments

x	data from reference/control group.
y	data from the treatment group.
distribution	character(1). Name of the parametric delay distribution to use.
param	character. Names of parameters to test difference for. Default value is 'delay'.
R	numeric(1). Number of bootstrap samples to evaluate the distribution of the test statistic.
ties	character. How to handle ties in data vector of a group?
type	character. Which type of tests to perform?
verbose	numeric. How many details are requested? Higher value means more details. 0=off, no details.

Details

High values of this difference speak against the null-model (i.e. high val_0 indicates bad fit under 0-model and low values of val_1 indicate a good fit under the more general model1. The test is implemented as a parametric bootstrap test, i.e. we

1. take given null-model fit as ground truth
2. regenerate data according to this model.
3. recalculate the test statistic
4. appraise the observed test statistic in light of the generated distribution under H0

Value

list with the results of the test. Element P contains the different P-values, for instance from parametric bootstrap

---

test_GOF *Goodness-of-fit (GOF) test statistic.*

Description

The GOF-test is performed for a fitted delay-model. There are different GOF-tests implemented:

- **Moran GOF** is based on spacings, like the MPSE-criterion itself.
- **Pearson GOF** uses categories and compares observed to expected frequencies.

Usage

```
test_GOF(delayFit, method = c("moran", "pearson"))
```

Arguments

delayFit	delay_model fit
method	character(1). which method to use for GOF. Default is 'moran'.

Value

An htest-object containing the GOF-test result

---

transform.incubate_fit

Transform observed data to unit interval

Description

The transformation is the probability integral transform. It uses the cumulative distribution function with the estimated parameters of the model fit. All available data in the model fit is transformed.

Usage

```
## S3 method for class 'incubate_fit'
transform(`_data`, ...)
```

Arguments

_data	a fitted model object of class incubate_fit
...	currently ignored

Value

The transformed data, either a vector (for single group) or a list with entries x and y (in two group scenario)

Note

This S3-method implementation is quite different from its default method that allows for non-standard evaluation on data frames, primarily for interactive use. But the name transform just fits so nicely to the intended purpose that it is re-used for the probability integral transform.

update.incubate_fit *Refit an* incubate_fit*-object with specified optimization arguments. If more things need to be changed use* delay_model.

Description

Refit an incubate_fit-object with specified optimization arguments. If more things need to be changed use delay_model.

Usage

```
## S3 method for class 'incubate_fit'
update(object, optim_args, verbose = 0, ...)
```

Arguments

object	incubate_fit-object
optim_args	optimization arguments
verbose	integer flag. Requested verbosity during delay_fit
...	further arguments, currently not used.

Value

The updated fitted object of class incubate_fit

Index