Package: intRvals (via r-universe)

September 3, 2024

Type Package

Title Analysis of Time-Ordered Event Data with Missed Observations

Version 1.0.1

Date 2022-05-02

Author Adriaan M. Dokter

Maintainer Adriaan M. Dokter <amd427@cornell.edu>

Description Calculates event rates and compares means and variances of groups of interval data corrected for missed arrival observations.

License GNU General Public License

Imports plyr,lme4

LazyData TRUE

Encoding UTF-8

RoxygenNote 7.1.2

NeedsCompilation no

Repository CRAN

Date/Publication 2022-05-02 22:52:12 UTC

Contents

`intRvals-package` *Analysing time-ordered event data with missed observations*

Description

intRvals calculates means and variances of arrival intervals (and arrival rates) corrected for missed arrival observations, and compares means and variances of groups of interval data.

Details

General: The central function of package **intRvals** is `estinterval`, which is used to estimate the mean arrival interval (and its standard deviation) from interval data with missed arrivals. This is achieved by fitting the theoretical probability density `intervalpdf` to the interval data

The package can be used to analyse general interval data where intervals are derived from distinct arrival observations. For example, the authors have used it to analyze dropping intervals of grazing geese for estimating their faecal output.

Intervals are defined as the time between observed arrival events (e.g. the time between one excreted droppings to the next) The package provides a way of taking into account missed observations (e.g. defecations), which lead to occasional observed intervals at integer multiples of the true arrival interval.

Typical workflow:

1. Fit interval model `m` to an interval dataset `d` using `estinterval`, as in `m=estinterval(d)`.
2. Visually inspect model fits using `plot.intRvals`, as in `plot(m)`.
3. Use `anova.intRvals` to check whether the missed event probability was signficantly different from zero, as in `anova(m)`.
4. Also use `anova.intRvals` to perform model selection between competing models `m1`,`m2` for the same interval dataset `d`, as in `anova(m1,m2)`.
5. Compare means and variances between different interval datasets `d1`,`d2` using `ttest` and `vartest`.

Other useful functionality: `fold` provides functionality to fold observed intervals back to their fundamental interval

`fundamental` tests which intervals are fundamental, i.e. intervals not containing a missed arrival observation

`interval2rate` converts interval estimates to rates

`partition` estimates and tests for the presence of within-subject variation

`intervalsim` simulates a set of observed intervals

The package comes with a example interval dataset `goosedrop`

Please cite this package using the publication "Analysing time-ordered event data with missed observations, Ecology and Evolution, 2017" by Dokter et al.

References

Dokter, A.M., van Loon, E.E., Fokkema, W., Lameris, T.K, , Nolet, B.A. and van der Jeugd, H.P. 2017. Analysing time-ordered event data with missed observations, Ecology and Evolution, 2017, in press.

B\'edard, J. & Gauthier, G. 1986. Assessment of faecal output in geese. Journal of Applied Ecology, 23, 77-90.

Owen, M. 1971. The Selection of Feeding Site by White-Fronted Geese in Winter. Journal of Applied Ecology 8: 905-917.

`anova.intRvals` *Compares model fits of* `intRvals` *objects*

Description

Compare model fits of `intRvals` objects estimated on the same data. If one object is provided, the results of a deviance test against a model without a missed event probability 'p' is reported. If two objects are provided, the results of a deviance test between the model fits of the two objects is given.

Usage

```
## S3 method for class 'intRvals'
anova(
  object,
  y = NULL,
  conf.level = 0.95,
  digits = max(3L, getOption("digits") - 3L),
  ...
)
```

Arguments

`object`	an object of class `intRvals`, usually a result of a call to estinterval
`y`	an (optional) object of class `intRvals`, usually a result of a call to estinterval
`conf.level`	confidence level for the deviance test
`digits`	the number of digits for printing to screen
`...`	other arguments to be passed to low level functions

Value

A list of class "`anova.intRvals`" with the best model (1 or 2), deviance statistic and test results

`best.model` the index of the best model (1 is first argument, 2 is second)

`deviance` the deviance between the two tested models

`p.value` p-value for the deviance (likelihood-ratio) test `conf.level` assumed confidence level for the test

`model1.call` call that generated model 1

`model2.call` call that generated model 2

`AIC` numeric 2-vector containg the AIC value for model 1 (first element) and model 2 (second element)

`loglik` numeric 2-vector containg the log-likelihood value for model 1 (first element) and model 2 (second element)

Examples

```
data(goosedrop)
model1=estinterval(goosedrop$interval,fun="gamma")
# visually inspect model1 fit:
plot(model1)
# The observed distribution has intervals near zero.
# We allow a small random baseline to reduce the effect
# of intervals near zero on the fit result.
model2=estinterval(goosedrop$interval,fun="gamma",fpp.method='auto')
# model2 performs better than model1:
anova(model1,model2)
```

estinterval *Estimate interval model accounting for missed arrival observations*

Description

Estimate interval mean and variance accounting for missed arrival observations, by fitting the probability density function intervalpdf to the interval data.

Usage

```
estinterval(
  data,
  mu = median(data),
  sigma = sd(data)/2,
  p = 0.2,
  N = 5L,
  fun = "gamma",
  trunc = c(0, Inf),
  fpp = (if (fpp.method == "fixed") 0 else 0.1),
  fpp.method = "auto",
  p.method = "auto",
  conf.level = 0.9,
  group = NA,
  sigma.within = NA,
  iter = 10,
  tol = 0.001,
  silent = F,
  ...
)
```

Arguments

`data` A numeric list of intervals.

`mu` Start value for the numeric optimization for the mean arrival interval.

`sigma` Start value for the numeric optimization for the standard deviation of the arrival interval.

`p` Start value for the numeric optimization for the probability to not observe an arrival.

`N` Maximum number of missed observations to be taken into account (default N=5).

`fun` Assumed distribution for the intervals, one of `"normal"` or `"gamma"`, corresponding to the Normal and GammaDist distributions

`trunc` Use a truncated probability density function with range `trunc`

`fpp` Baseline proportion of intervals distributed as a random poisson process with mean arrival interval `mu`

`fpp.method` A string equal to ’fixed’ or ’auto’. When ’auto’ `fpp` is optimized as a free model parameter, in which case `fpp` is taken as start value in the optimisation

`p.method` A string equal to ’fixed’ or ’auto’. When ’auto’ `p` is optimized as a free model parameter, in which case `p` is taken as start value in the optimisation

`conf.level` Confidence level for deviance test that checks whether model with nonzero missed event probability `p` significantly outperforms a model without a missed event probability (`p=0`).

`group` optional vector of equal length as data, indicating the group or subject in which the interval was observed

`sigma.within` optional within-subject standard deviation. When equal to default ’NA’, assumes no additional between-subject effect, with `sigma.within` equal to `sigma`. When equal to ’auto’ an estimate is provided by iteratively calling partition

`iter` maximum number of iterations in numerical iteration for `sigma.within`

`tol` tolerance in the iteration, when `sigma.within` changes less than this value in one iteration step, the optimization is considered converged.

`silent` logical. When `TRUE` print no information to console

`...` Additional arguments to be passed to optim

Details

The probability density function for observed intervals intervalpdf is fit to `data` by maximization of the associated log-likelihood using optim.

Within-group variation `sigma.within` may be separated from the total variation `sigma` in an iterative fit of intervalpdf on the interval data. In the iteration partition is used to (1) determine which intervals according to the fit are a fundamental interval at a confidence level conf.level, and (2) to partition the within-group variation from the total variation in interval length.

Within- and between-group variation is estimated on the subset of fundamental intervals with repeated measures only. As the set of fundamental interval depends on the precise value of sigma.within, the fit of intervalpdf and the subsequent estimation of sigma.within using partition is iterated until both converge to a stable solution. Parameters tol and iter set the threshold for convergence and the maximum number of iterations.

We note that an exponential interval model can be fitted by setting fpp=1 and fpp.method=fixed.

Value

This function returns an object of class intRvals, which is a list containing the following:

data the interval data

mu the modelled mean interval

mu.se the modelled mean interval standard error

sigma the modelled interval standard deviation

p the modelled probability to not observe an arrival

fpp the modelled fraction of arrivals following a random poisson process, see intervalpdf

N the highest number of consecutive missed arrivals taken into account, see intervalpdf

convergence convergence field of optim

counts counts field of optim

loglik vector of length 2, with first element the log-likelihood of the fitted model, and second element the log-likelihood of the model without a missed event probability (i.e. p=0)

df.residual degrees of freedom, a 2-vector (1, number of intervals - n.param)

n.param number of optimized model parameters

p.chisq p value for a likelihood-ratio test of a model including a miss probability relative against a model without a miss probability

distribution assumed interval distribution, one of 'gamma' or 'normal'

trunc interval range over which the interval pdf was truncated and normalized

fpp.method A string equal to 'fixed' or 'auto'. When 'auto' fpp has been optimized as a free model parameter

p.method A string equal to 'fixed' or 'auto'. When 'auto' p has been optimized as a free model parameter

Examples

```
data(goosedrop)
# calculate mean and standard deviation of arrival intervals, accounting for missed observations:
dr=estinterval(goosedrop$interval)
# plot some summary information
summary(dr)
# plot a histogram of the intervals and fit:
plot(dr)
```

```
# test whether the mean arrival interval is greater than 200 seconds:
ttest(dr,mu=200,alternative="greater")

# let's estimate mean and variance of dropping intervals by site
# (schiermonnikoog vs terschelling) for time period 5.
# first prepare the two datasets:
set1=goosedrop[goosedrop$site=="schiermonnikoog" & goosedrop$period == 5,]
set2=goosedrop[goosedrop$site=="terschelling"  & goosedrop$period == 5,]
# allowing a fraction of intervals to be distributed randomly (fpp='auto')
dr1=estinterval(set1$interval,fpp.method='auto')
dr2=estinterval(set2$interval,fpp.method='auto')
# plot the fits:
plot(dr1,xlim=c(0,1000))
plot(dr2,xlim=c(0,1000))
# mean dropping interval are not significantly different
# at the two sites (on a 0.95 confidence level):
ttest(dr1,dr2)
# now compare this test with a t-test not accounting for unobserved intervals:
t.test(set1$interval,set2$interval)
# not accounting for missed observations leads to a (spurious)
# larger difference in means, which also increases
# the apparent statistical significance of the difference between means
```

fold — *Folds observed arrival intervals to a fundamental interval*

Description

Folds observed arrival intervals with missed observations back to their most likely fundamental interval

Usage

```
fold(object, take.sample = F, sigma.within = NA, silent = F)
```

Arguments

`object` an object of class `intRvals`, usually a result of a call to estinterval

`take.sample` when `TRUE` the number of folds of the fundamental interval is sampled randomly, taking into account the probability weight of each possibility. When `FALSE` the fold with the highest probability weight is taken.

`sigma.within` (optional) numeric value with an assumed within-group/subject standard deviation, or ’`auto`’ to estimate it automatically using partition.

`silent` logical, if `TRUE` print no text to console

Details

Arrival intervals containing missed observations are folded to their most likely fundamental interval according to a fit of the distribution of intervals by estinterval.

There is inherent uncertainty on how many missed arrival events an observed interval contains, and therefore to which fundamental interval it should be folded. Intervals folded to the fundamental can therefore introduce extra unexplained variance.

The default is to fold intervals to the fundamental with the highest probability weight (`take.sample = F`). Alternatively, randomly sampled intervals can be generated, that take into account the probability weights of each possible fold (`take.sample = T`).

Intervals `x` are transformed to their fundamental interval according to

$$\mu + (x - i * \mu)/\sqrt{i}$$

with `i-1` the estimated number of missed observations within the interval. This transformation scales appropriately with the expected broadening of the standard distributions $\phi(x|i\mu, \sqrt{i}\sigma)$ with `i` in intervalpdf.

When no `sigma.within` is provided, μ equals the mean arrival rate, estimated by estinterval.

When `sigma.within` is '`auto`', `sigma.within` is estimated using partition.

When `sigma.within` is a user-specified numeric value or '`auto`', μ is estimated for each group (as specified in the group argument of estinterval), by maximizing the log-likelihood of intervalpdf, with its `data` argument equals to the intervals of the group, its `sigma` argument equal to `sigma.within`, and its remaining arguments taken from `object`.

Intervals assigned to the `fpp` component (see estinterval) are not folded, and return as `NA` values.

Value

numeric vector with intervals folded into the fundamental interval

Examples

```
dr=estinterval(goosedrop$interval,group=goosedrop$bout_id)
# fold assuming no within-group variation:
interval.fundamental=fold(dr)
# test whether there is evidence for within-group variation:
partition(dr)$`p<alpha`   #> TRUE
# there is evidence, therefore better to fold
# while accounting for within-group variation:
interval.fundamental=fold(dr,sigma.within='auto')
```

`fundamental` *Estimate which intervals are fundamental*

Description

Estimates which intervals in a dataset are fundamental intervals, i.e. an interval not containing a missed arrival observation

Usage

```
fundamental(x, conf.level = 0.9)
```

Arguments

`x`	object inheriting from class `intRvals`, usually a result of a call to `estinterval`
`conf.level`	confidence level for identifying intervals as fundamental

Details

This functions thus determines for each interval `x$data` whether it has a probabiliy > `conf.level` to be a fundamental interval, given the model fit generated by estinterval for object `x`.

The fit of an `intRvals` object gives the decomposition of the likelihood of an interval observation into partial likelihoods $\phi_{obs}(x, i|\mu, \sigma, p)$ (see intervalpdf). If the amplitude of the partial likelihood with i=0 (i.e. the likelihood component without missed observations) is at least a proportion `conf.level` of the sum of all terms i=0..N, an interval is considered to be fundamental (not containing a missed event observation).

Value

logical atomic vector of the same length as `x$data`

`goosedrop` *Dataset with dropping intervals observed for foraging Brent Geese (Branta bernicla bernicla)*

Description

The dataset contains observations from two sites: the island of Schiermonnikoog (saltmarsh) and Terschelling (agricultural grassland). Brent geese were observed continuously with spotting scopes, and the time when geese excreted a dropping was written down. The time in seconds between wo subsequently observed dropping arrivals of a single individual refers to one dropping interval. The variables are as follows:

`date` observation start time of the interval

`interval` length of the interval in seconds

`bout_length` total observation time of individual

`prop_abdomen_seen` proportion of total observation time when abdomen could be observed

`bout_id` intervals belonging to the same observation bout of an individual have the same `bout_id`

`site` observation site. One of 'terschelling' (agricultural grassland) or 'schiermonnikoog' (salt marsh)

`period` two-week observation period (1-5)

Usage

```
goosedrop
```

Format

An object of class `data.frame` with 705 rows and 7 columns.

Author(s)

Adriaan Dokter <`a.m.dokter@uva.nl`>

`interval2rate` *Conversion of interval estimates to rates*

Description

Conversion of interval estimates to rates

Usage

```
interval2rate(
  data,
  minint = data$mu/100,
  maxint = data$mu + 3 * data$sigma,
  digits = max(3L, getOption("digits") - 3L),
  method = "exact"
)
```

Arguments

`data` An object of class `intRvals`, usually a result of a call to estinterval

`minint` the minimum interval value from which numerical integrations converting to rates are started

`maxint` the maximum interval value up to which numerical integrations converting to rates are continued

`digits` the number of digits for printing to screen

`method` A string equal to 'exact' or 'taylor'. When 'exact' exact formula or numeric integration is used. When 'taylor' a Taylor approximation is used as in standard propagation of uncertainty in the case of division.

Details

Gamma-distributed intervals: When inter-arrival times (intervals) follow a gamma distribution with mean μ and standard deviation σ, i.e. follow the probability density function `GammaDist(shape=`$\alpha = \mu^2/\sigma^2$`, scale=`$\beta = \sigma^2/\mu$`)`, then the associated distribution of rates is given by an inverse gamma distribution with shape parameter α and scale parameter $1/\beta$.

The mean of this inverse gamma distribution is given by the formula

$$\mu_{rate} = \mu/(\mu^2 - \sigma^2)$$

provided that $\alpha > 1$, i.e. $\mu > \sigma$.

The variance of this inverse gamma distribution is given by the formula

$$\sigma_{rate}^2 = \mu^2\sigma^2/((\mu^2 - \sigma^2)(\mu^2 - 2\sigma^2)^2$$

provided that $\alpha > 2$, i.e. $\mu > sqrt(2) * \sigma$.

Values μ and σ are estimated on the interval data, and above formula are used to calculate the estimated mean and variance of the arrival rate.

If these formula cannot be used (because the provisions on the value of α are not met), numerical integration is used instead, analagous to the procedure for normal-distributed intervals, see below.

Normal-distributed intervals: When inter-arrival times (intervals) x follow a normal distribution with mean μ and standard deviation σ, i.e. follow the probability density function `Normal(mean=`μ`, sd=`σ`)`, then the mean rate (μ_{rate}) can be calculated numerically by:

$$\mu_{rate} = \int_0^\infty (1/x) * \phi(x|\mu, \sigma)$$

and the variance of the rate (σ_{rate}^2) by:

$$\sigma_{rate}^2 = \int_0^\infty (1/x^2) * \phi(x|\mu, \sigma) - \mu_{rate}^2$$

This approximation is only valid for distributions that have a negligable density near $x = 0$, such that the distribution can be effectively truncated before x approaches zero, where the integral is not defined. For interval data with intervals x near zero, use of a gamma distribution is recommended instead.

Value

The function `interval2rate` computes and returns a named vector with the rate mean and standard deviation

Examples

```
data(goosedrop)
dr=estinterval(goosedrop$interval)
interval2rate(dr)
```

`intervalpdf` *Probability density function of an observed interval distribution*

Description

Observed intervals are assumed to be sampled through observation of continuous distinct arrivals in time. Two subsequently observed arrivals mark the start and end of an interval. The probability that an arrival is not observed can be nonzero, leading to observed intervals at integer multiples of the true interval.

Usage

```
intervalpdf(
  data = seq(0, 1000),
  mu = 200,
  sigma = 40,
  p = 0.3,
  N = 5L,
  fun = "gamma",
  trunc = c(0, Inf),
  fpp = 0,
  sigma.within = NA
)
```

Arguments

`data` A list of intervals for which to calculate the probability density

`mu` The mean of the true interval distribution

`sigma` The standard deviation of the true interval distribution

`p` The probability that an arrival that marks the start or end of an interval is not observed

`N` The maximum number of consecutive missed arrivals to take into consideration

`fun` assumed distribution family of the true interval distribution, one of "`normal`" or "`gamma`", corresponding to the Normal and GammaDist distributions.

`trunc` Use a truncated probability density function with range `trunc`

`fpp` Baseline proportion of intervals distributed as a random poisson process with mean arrival rate `mu`

`sigma.within` within-subject standard deviation, only available when `fun` is "normal"

Details

General: intervals x are assumed to follow a standard distribution (either a normal or gamma distribution) with probability density function $\phi(x|\mu, \sigma)$ with μ the mean arrival interval and σ its associated standard deviation. The probability density function ϕ_{obs} of observed arrival intervals in a scenario where the probability to not observe an arrival is nonzero, will be a superposition of several standard distributions, at multiples of the fundamental mean arrival interval. Standard distribution i will correspond to those intervals where i arrivals have been missed consecutively. If p equals this probability of not observing an arrival, then the probability $P(i)$ to miss i consecutive arrivals equals

$$P(i) = p^i - p^{i+1}$$

The width of standard distribution i will be broadened relative to the fundamental, according to standard uncertainty propagation in the case of addition. Both in the case of normal and gamma-distributed intervals (see next subsections) we may write for the observed probability density function, ϕ_{obs}:

$$\phi_{obs}(x|\mu, \sigma, p) = \sum_{i=1}^{\infty} \phi_{obs}(x, i|\mu, \sigma, p)$$

with

$$\phi_{obs}(x, i|\mu, \sigma, p) = P(i-1)\phi(x|i\mu, \sqrt{i}\sigma)$$

In practice, this probability density function is well approximate when the infinite sum is capped at a finite integer N. Be default the sum is ran up to N=5.

Gamma-distributed intervals: By default intervals x are assumed to follow a Gamma (GammaDist) distribution $Gamma(\mu, \sigma)$~dgamma(shape=μ^2/σ^2, scale=σ^2/μ) with a probability density function $\phi(x)$:

$$\phi(x|\mu, \sigma)\ Gamma(\mu, \sigma)$$

which has a mean μ and standard deviation σ.

Normal-distributed intervals: intervals x may also be assumed to follow a Normal distribution $N(\mu, \sigma)$~dnorm(mean=μ,sd=σ), with a probability density function $\phi(x)$:

$$\phi(x|\mu, \sigma)\ N(\mu, \sigma)$$

which also has a mean μ and standard deviation σ. Because intervals are by definition non-negative, the Normal distribution is always truncated at zero. In the limit that $\mu > \sigma$ the gamma distribution tends to the normal distribution.

Within and between-subject variation: To account for witin-subject and between-subject differences in mean interval length we define σ_w as within-subject standard deviation in interval length, and σ_b as between-subject standard deviation in interval length, with $\sigma^2 = \sigma_b^2 + \sigma_w^2$. In the normal limit ($\mu > \sigma$) the population pdf will be a convolution between $\phi(x|\mu, \sigma_b)$ and $\phi(x|\mu, \sigma_w)$ equal to:

$$\phi_{obs}(x|\mu, \sigma, \sigma_w, p) = \sum_{i=1}^{\infty} P(i-1)\phi(x|i\mu, \sqrt{i}\sigma)$$

Value

This function returns a list of points describing the interval distribution

Examples

```
# a low probability of not observing an arrival
# results in an observed PDF with primarily
# a single peak, with a mean and standard
# deviation almost identical to the true interval
# distribution:
plot(intervalpdf(mu=200,sigma=40,p=0.01),type='l',col='red')

# a higher probability to miss an arrival
# results in an observed PDF with multiple
# peaks at integer multiples of the mean of the true
# interval distribution
plot(intervalpdf(mu=200,sigma=40,p=0.4),type='l',col='red')
```

`intervalsim` *Simulate a set of observed intervals*

Description

Simulate a set of observed intervals

Usage

```
intervalsim(
  n = 500,
  mu = 200,
  sigma = 40,
  p = 0.3,
  fun = ”gamma”,
  trunc = c(0, 600),
  fpp = 0,
  n.ind = NA,
  sigma.within = NA
)
```

Arguments

`n` Number of simulated interval observations.

`mu` Mean arrival interval.

`sigma` Standard deviation of the arrival interval.

`p` Probability to not observe an arrival.

`fun` Assumed distribution for the intervals, one of `"normal"` or `"gamma"`, corresponding to the Normal and GammaDist distributions

`trunc` Observational range of intervals (intervals outside this range won't be observed)

`fpp` Baseline proportion of intervals distributed as a random poisson process with mean arrival interval `mu`

`n.ind` Number of intervals per group. Ignored without a numeric value for `sigma.within`.

`sigma.within` The within-group standard-deviation. When a numeric value is given for `sigma.within`, `sigma` denotes the total (within+between subject) standard deviation

Details

Simulates the observations process of arrival intervals.

The default is to not differentiate between within- and between-group variance.

If both `n.ind` and `sigma.within` have numeric values, intervals are simulated with separate within-group variation (`sigma.within`) and between-group variation, for groups of size `n.ind`. Intervals belonging to the same group have:

- a within-group mean interval length that has been randomly drawn from a distribution with mean mu and between-group standard deviation $\sqrt{sigma^2 - sigma.within^2}$
- a within-group standard deviation in interval length equal to sigma.within

Value

This function returns a dataframe containing the following:

interval the simulated interval data

group_id a group identifier

Examples

```
# simulate observed intervals:
intervals=intervalsim(n=50,mu=200,sigma=40,trunc=c(0,600),fpp=0.1)
# check whether we retrieve the simulation parameters:
estinterval(goosedrop$interval)
```

loglikinterval *log-likelihood of an observed interval distribution*

Description

log-likelihood of an observed interval distribution

Usage

```
loglikinterval(
  data,
  mu,
  sigma,
  p,
  N = 5L,
  fun = "gamma",
  trunc = c(0, Inf),
  fpp = 0
)
```

Arguments

data	A numeric list of intervals.
mu	mean arrival interval.
sigma	standard deviation of the arrival interval.
p	chance to not observe an arrival.
N	Maximum number of missed observations to be taken into account (default N=5).

`fun` Assumed distribution for the intervals, one of "`normal`" or "`gamma`", corresponding to the Normal and GammaDist distributions

`trunc` Use a truncated probability density function with range `trunc`

`fpp` Baseline proportion of intervals distributed as a random poisson process with mean arrival interval `mu`

Details

Refer to intervalpdf for details on the functional form of the probability density function of an observed interval distribution ϕ_{obs}. The log-likelihood L given a set of intervals x_j in `data` is given by

$$L(\mu, \sigma, p) = \log \sum_j \phi_{obs}(x_j|\mu, \sigma, p)$$

The function is provided to allow likelihood maximisation by other optimization tools than the default by optim.

Value

returns the value of the loglikelihood

Examples

```
data(goosedrop)
loglikinterval(goosedrop$interval,mu=200,sigma=50,p=.3)
```

partition *Estimate within-group variation*

Description

Estimate within-group variation in interval length

Usage

```
partition(x, conf.level = 0.9, alpha = 0.05, silent = F)
```

Arguments

`x` object inheriting from class `intRvals`

`conf.level` confidence level passed to function fundamental, used in selecting fundamental intervals

`alpha` significance level for differences within and between groups or subjects

`silent` logical, if `TRUE` print no text to console

Details

Within- and between-group variation is estimated on the subset of fundamental intervals only.

The subset of fundamental intervals is selected using fundamental.

We calculate $sigma.within = s_w n_{ind}/(n_{ind}+1)$ with s_w the uncorrected sample standard deviation of within-group centered values (obtained from subtracting the group's mean value from each observation value), and $n_{ind}/(n_{ind}+1)$ Bessel's correction with n_{ind} the average number of repeated measures per group. Significance of within-group variation is determined by testing for a random effect of group against a constant null model (van de Pol & Wright 2009), using the R-package lme4 (Bates et al. 2015).

Value

A logical atomic vector indicating which intervals are fundamental.

`sigma.within` within-group standard deviation in interval length, estimated on fundamental intervals with repeated measures only

`sigma` the total standard deviation in interval length, copied from `x$sigma`

`p.within` p-value form a likelihood-ratio test indicating whether there is evidence for a random effect of group or subject

`n.within` average number of intervals per group

`n.total` total number of intervals

`n.repeat` number of fundamental intervals with repeated measures, the size of the dataset on which `sigma.within` was estimated

`p<alpha` logical. Whether there was significant evidence for a difference in within- and between-group/subject variance

References

van de Pol, M. & Wright, J. (2009). A simple method for distinguishing within- versus between-subject effects using mixed models. Animal Behaviour, 77, 753-758.

Bates, D., M\"achler, M., Bolker, B.M. & Walker, S.C. (2015). Fitting linear mixed-effects models using lme4. Journal of Statistical Software, 67, 1-48.

Examples

```
# select the group of intervals observed on Terschelling island
dropset=goosedrop[goosedrop$site=="terschelling",]
# estimate an interval model, with separate within- and between-group variation:
dr=estinterval(data=dropset$interval,group = dropset$bout_id)
# plot the model fit:
plot(dr)
# estimate within-group variation, and its significance:
output=partition(dr)
# print within-group standard deviation:
output$sigma.within
# is the model including within-group standard deviation signicant,
# relative to a null model without separate within-group sd,
# at the specified confidence level alpha?
output$`p<alpha`  #> TRUE
```

plot.intRvals *Plot an interval histogram and fit of intRvals object*

Description

Plot an interval histogram and fit of intRvals object

Usage

```
## S3 method for class 'intRvals'
plot(
  x,
  binsize = 20,
  xlab = "Interval",
  ylab = "Density",
  main = "Interval histogram and fit",
  line.col = "red",
  line.lwd = 1,
  ...
)
```

Arguments

`x`	An intRvals class object
`binsize`	Width of the histogram bins
`xlab`	a title for the x axis
`ylab`	a title for the y axis
`main`	an overall title for the plot
`line.col`	Color of the plotted curve for the model fit
`line.lwd`	Line width of the plotted curve for the model fit
`...`	Additional arguments to be passed to the low level hist plotting function

Value

This function returns a list with data, corresponding to the model fit

Examples

```
data(goosedrop)
dr=estinterval(goosedrop$interval)
plot(dr)
plot(dr,binsize=10,line.col='blue')
```

summary.intRvals *summary method for class* `intRvals`

Description

summary method for class `intRvals`

Usage

```
## S3 method for class 'intRvals'
summary(object, ...)
```

Arguments

`object`	An object of class `intRvals`, usually a result of a call to estinterval
`...`	further arguments passed to or from other methods.

Value

The function `summary.intRvals` computes and returns a list of summary statistics

`data` the interval data

`mu` the modelled mean interval

`mu.se` the modelled mean interval standard error

`sigma` the modelled interval standard deviation

`p` the modelled probability to not observe an arrival

`fpp` the modelled fraction of arrivals following a random poisson process, see intervalpdf

`N` the highest number of consecutive missed arrivals taken into account, see intervalpdf

`convergence` convergence field of optim

`counts` counts field of optim

`loglik` vector of length 2, with first element the log-likelihood of the fitted model, and second element the log-likelihood of the model without a missed event probability (i.e. p=0)

`df.residual` degrees of freedom, a 2-vector (1, number of intervals - `n.param`)

`n.param` number of optimized model parameters

`distribution` assumed interval distribution, one of 'gamma' or 'normal'

`trunc` interval range over which the interval pdf was truncated and normalized

`fpp.method` A string equal to 'fixed' or 'auto'. When 'auto' fpp has been optimized as a free model parameter. When 'fixed' the model is fitted with a fixed value set by parameter `fpp`

`deviance` deviance between the fitted model and a model without a missed event probability (i.e. p=0)

`p.value` numeric vector with two elements. First element contains the p.value for a likelihood ratio (deviance) test between the fitted model and a model without a missed event probability (i.e. p=0). Second element contains the p.value for a likelihood ratio (deviance) test between the fitted model and a saturated null model.

Examples

```
data(goosedrop)
dr=estinterval(goosedrop$interval)
summary(dr)
```

ttest *Student's t-test to compare two means of objects of class* `intRvals`

Description

Performs one and two sample t-tests on objects of class `intRvals`

Usage

```
ttest(
  x,
  y = NULL,
  alternative = c("two.sided", "less", "greater"),
  mu = 0,
  var.equal = FALSE,
  conf.level = 0.95
)
```

Arguments

`x`	an object of class `intRvals`, usually a result of a call to estinterval
`y`	an (optional) object of class `intRvals`, usually a result of a call to estinterval
`alternative`	a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter.
`mu`	a number indicating the true value of the mean (or difference in means if you are performing a two sample test).
`var.equal`	a logical variable indicating whether to treat the two variances as being equal. If TRUE then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.
`conf.level`	confidence level of the interval

Details

`alternative = "greater"` is the alternative that `x` has a larger mean than `y`.

If the input data are effectively constant (compared to the larger of the two means) an error is generated.

Value

A list with class `"htest"` containing the same components as in t.test

Examples

```
data(goosedrop)
dr=estinterval(goosedrop$interval)
# perform a one-sample t-test
ttest(dr,mu=200)  #> FALSE, true mean not equal to 200
# two sample t-test
data.beforeMay=goosedrop[goosedrop$date<as.POSIXct('2013-05-01'),]
data.afterMay=goosedrop[goosedrop$date>as.POSIXct('2013-05-01'),]
dr.beforeMay=estinterval(data.beforeMay$interval)
dr.afterMay=estinterval(data.afterMay$interval)
ttest(dr.beforeMay,dr.afterMay)
```

vartest *F Test to compare two variances of objects of class* intRvals

Description

Performs an F test to compare the variances of objects of class intRvals

Usage

```
vartest(
  x,
  y,
  ratio = 1,
  alternative = c("two.sided", "less", "greater"),
  conf.level = 0.95
)
```

Arguments

x an object of class intRvals, usually a result of a call to estinterval

y an (optional) object of class intRvals, usually a result of a call to estinterval

ratio the hypothesized ratio of the population variances of x and y.

alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

conf.level confidence level for the returned confidence interval

Details

The null hypothesis is that the ratio of the variances of the data to which the models x and y were fitted, is equal to ratio.

Value

A list with class "htest" containing the same components as in var.test

Examples

```
data(goosedrop)
dr=estinterval(goosedrop$interval)
# split the interval data into two periods
data.beforeMay=goosedrop[goosedrop$date<as.POSIXct('2013-05-01'),]
data.afterMay=goosedrop[goosedrop$date>as.POSIXct('2013-05-01'),]
dr.beforeMay=estinterval(data.beforeMay$interval)
dr.afterMay=estinterval(data.afterMay$interval)
# perform an F test
vartest(dr.beforeMay,dr.afterMay)
```

Index