

Package ‘familial’

November 13, 2021

Type Package

Title Familial Inference

Version 0.1.0

Author Ryan Thompson

Maintainer Ryan Thompson <ryan.thompson@monash.edu>

Description Provides functionality for testing familial hypotheses. Currently supports tests of the Huber family of centers. Testing is carried out using the Bayesian bootstrap. One- and two-sample tests are supported, as are directional tests. Methods for visualizing output are provided.

URL <https://github.com/ryan-thompson/familial>

BugReports <https://github.com/ryan-thompson/familial/issues>

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports parallel, ggplot2, DepthProc

RoxygenNote 7.1.2

Suggests testthat, knitr, rmarkdown, MASS

VignetteBuilder knitr

Config/testthat/edition 3

R topics documented:

bayes.boot	2
center.test	3
fit.family	4
plot.center.test	5
plot.fit.family	6
print.center.test	7
rudirichlet	7
weighted	8

Index**9**

bayes.boot	<i>Bayesian bootstrap</i>
------------	---------------------------

Description

Performs a Bayesian bootstrap for a statistic defined via a suitable function.

Usage

```
bayes.boot(x, fun, nboot = 1000, cluster = NULL, ...)
```

Arguments

x	a numeric vector to be passed as the first argument to fun
fun	the function to bootstrap; must accept data x and weights w (in that order), and return a data frame
nboot	the number of bootstraps to perform
cluster	an optional cluster for running bootstraps in parallel; must be set up using <code>parallel::makeCluster</code>
...	any other arguments for fun

Value

An object of class `bayes.boot`; a data frame with the following columns:

boot.id	the bootstrap iteration index
...	any columns returned by fun

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

Examples

```
set.seed(1)

# Bootstrap
boot <- bayes.boot(MASS::galaxies, fun = fit.family, nboot = 100)
head(boot)
```

center.test

*Center test***Description**

Performs a one- or two-sample test for a family of centers.

Usage

```
center.test(
  x,
  y = NULL,
  family = "huber",
  alternative = c("two.sided", "less", "greater"),
  mu = 0,
  paired = FALSE,
  nboot = 1000,
  loss = NULL,
  cluster = NULL,
  ...
)
```

Arguments

x	a numeric vector of data
y	an optional numeric vector of data
family	the family of centers; currently only allows 'huber' for Huber family
alternative	the form of the alternative hypothesis; must be one of 'two.sided' (default), 'greater', or 'less'
mu	the null value of the center for a one-sample test, or the null value of the center of differences for a paired two-sample test, or the null value of the difference in centers for an independent two-sample test; can be an interval
paired	a logical indicating whether to treat x and y as paired
nboot	the number of bootstraps to perform
loss	an optional c×2 matrix of losses incurred from an incorrect decision, where c is the number of candidate choices (typically c=3: H0, H1, or indeterminate)
cluster	an optional cluster for running bootstraps in parallel; must be set up using <code>parallel::makeCluster</code>
...	any other arguments

Details

Uses the Bayesian bootstrap to compute posterior probabilities for the hypotheses $H_0 : \mu(\lambda) = \mu_0$ for some $\lambda \in \Lambda$ vs. $H_1 : \mu(\lambda) \neq \mu_0$ for all $\lambda \in \Lambda$, where $\{\mu(\lambda) : \lambda \in \Lambda\}$ is a family of centers. The default loss matrix results in a decision whenever the posterior probability for one of the hypotheses is greater than 0.95 and otherwise is indeterminate.

Value

An object of class `center.test`; a list with the following components:

<code>expected.loss</code>	the expected loss, calculated by post-multiplying <code>loss</code> with <code>prob</code>
<code>decision</code>	the optimal decision given the expected loss
<code>loss</code>	the loss matrix
<code>prob</code>	the estimated posterior probabilities of the null and alternative
<code>boot</code>	the bootstrap output from <code>bayes.boot</code>
<code>x</code>	the <code>x</code> that was supplied
<code>y</code>	the <code>y</code> that was supplied
<code>mu</code>	the <code>mu</code> that was supplied
<code>family</code>	the family that was supplied

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

Examples

```
set.seed(1)

# Familial test using Huber family with point null
test <- center.test(MASS::galaxies, mu = 21000, nboot = 100)
print(test)
plot(test)

# Familial test using Huber family with interval null
test <- center.test(MASS::galaxies, mu = c(20500, 21500), nboot = 100)
print(test)

# Familial test in parallel
cl <- parallel::makeCluster(2)
test <- center.test(MASS::galaxies, mu = c(20500, 21500), nboot = 100, cluster = cl)
parallel::stopCluster(cl)
print(test)
```

fit.family

Fit family

Description

Fits a family of centers.

Usage

```
fit.family(
  x,
  w = rep(1, length(x)),
  family = "huber",
  spread.fun = weighted.mad,
  eps = .Machine$double.eps
)
```

Arguments

x	a numeric vector of data
w	a numeric vector of weights
family	the location family; currently only allows 'huber' for Huber family
spread.fun	a function used for the spread of x; must accept data x and weights w (in that order), and return a numeric
eps	a numerical tolerance parameter

Value

An object of class `fit.family`; a data frame with the following columns:

mu.hat	the fitted values
lambda	the thresholding parameter

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

Examples

```
fit <- fit.family(MASS::galaxies)
plot(fit)
```

plot.center.test	<i>Plot function for center.test object</i>
------------------	---

Description

Plot the posterior distribution for the family of centers using a functional box plot.

Usage

```
## S3 method for class 'center.test'
plot(x, band = c(0.5, 0.75, 0.95), ninterp = 25, ...)
```

Arguments

x	an object of class center.test
band	a vector of band limits for the functional box plot
ninterp	the number of interpolation points for the functional box plot; more points lead to finer resolution of the plot at the expense of additional computation
...	any other arguments

Value

A plot of the posterior distribution.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

plot.fit.family	<i>Plot function for fit.family object</i>
-----------------	--

Description

Plot a fitted family.

Usage

```
## S3 method for class 'fit.family'  
plot(x, y = NULL, ...)
```

Arguments

x	an object of class fit.family
y	an object of class fit.family
...	any other arguments

Value

A plot of the fitted family.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

print.center.test	<i>Print function for center.test object</i>
-------------------	--

Description

Print objects of class center.test.

Usage

```
## S3 method for class 'center.test'
print(x, ...)
```

Arguments

x	an object of class center.test
...	any other arguments

Value

The argument x.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

rudirichlet	<i>Uniform Dirichlet distribution</i>
-------------	---------------------------------------

Description

Random number generation for the uniform Dirichlet distribution (having all concentration parameters set to one).

Usage

```
rudirichlet(n, d)
```

Arguments

n	the number of observations
d	the number of dimensions

Value

A matrix; each row is a random draw and each column is a dimension.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

weighted

Weighted statistics

Description

Assorted weighted statistics unavailable in base R.

Usage

`weighted.median(x, w)`

`weighted.mad(x, w)`

Arguments

<code>x</code>	a numeric vector of data
<code>w</code>	a numeric vector of weights

Value

A length-one numeric vector.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

Index

`bayes.boot`, [2](#)

`center.test`, [3](#)

`fit.family`, [4](#)

`plot.center.test`, [5](#)

`plot.fit.family`, [6](#)

`print.center.test`, [7](#)

`rudirichlet`, [7](#)

`weighted`, [8](#)