Package ‘bdots’

August 16, 2016

Type Package

Title Bootstrapped Differences of Time Series

Version 0.1.9

Date 2016-8-15

Author Michael Seedorff, Jacob Oleson, Grant Brown, Joseph Cavanaugh, and Bob McMurray

Maintainer Michael Seedorff <michael-seedorff@uiowa.edu>

Depends nlme, mvtnorm

Imports doParallel, doRNG, foreach

LazyData FALSE


License GPL (>= 3)

NeedsCompilation no

Repository CRAN

Date/Publication 2016-08-16 09:55:58

R topics documented:

bdots.write.csv ........................................ 2
Bootstrap Step ........................................... 3
ci ......................................................... 4
ests.plot ................................................. 5
Fitting Step ................................................ 6
Print Fits .................................................. 7
Refitting Step .............................................. 8
replot ..................................................... 9
subs.delete .............................................. 10
subs.plot ................................................. 11

Index 13
bdots.write.csv  Write to CSV

Description

Write bootstrapped estimates and confidence intervals to csv

Usage

bdots.write.csv(part1.list, part2.list, file, agg = TRUE, ...)

Arguments

part1.list  list. Output from doubleGauss.fit
part2.list  list. Output from doubleGauss.boot
file  Name of file to write to
agg  Whether to aggregate the data.
...  Further arguments to write.csv

Details

Write raw group averages, bootstrapped estimates, and confidence intervals to csv

Value

NULL

Note

There are no further notes

Examples

```r
## Not run:
data(ci)
ci.1 <- subset(ci, ci$LookType == "Target")
ci.1$Group <- ci.1$protocol
out.1 <- logistic.fit(ci.1, 4)
out.2 <- logistic.boot(out.1)
bdots.write.csv(out.1, out.2, "CIOutput.csv", row.names = FALSE)

## End(Not run)
```
Bootstrap Step

Bootstrap Step

Bootstrap on the Fitted Parameters

Description

Bootstrap on the fitted parameters, plot the estimates, and highlight the significant regions

Usage

doubleGauss.boot(part1.list, seed = new.seed(), alpha = 0.05, paired = FALSE, N.iter = 1000, cores = 1, p.adj = "oleson", test.spots = NULL, time.test = NULL, test.params = FALSE)

logistic.boot(part1.list, seed = new.seed(), alpha = 0.05, paired = FALSE, N.iter = 1000, cores = 1, p.adj = "oleson", test.spots = NULL, time.test = NULL, test.params = FALSE)

Arguments

part1.list list. Output from doubleGauss.fit
seed integer. What to set seed at
alpha numeric (Between 0 and 1). Probability of familywise Type I Error
paired boolean. Whether the same subjects are in both data sets
N.iter numeric (positive integer). Number of bootstrap iterations to run
cores integer. Number of cores on the localhost to use
p.adj Options: oleson, fdr, none
test.spots numeric. Specify specific x-values for testing at
time.test numeric. Specify individual time points to conduct t-tests at without any p-value correction
test.params boolean. Whether to test for significant differences in group mean parameter estimates. Performs a 2-sample t-test with equal variance assumption if paired = FALSE. If paired = TRUE, performs a paired t-test.

Details

Bootstrap on the fitted parameters, plot the estimates, and highlight the significant regions

Value

List: for input in replot

Note

There are no further notes
Examples

```r
## Not run:
data(ci)
ci.1 <- subset(ci, ci$LookType == "Target")
ci.1$Group <- ci.1$protocol
out.1 <- logistic.fit(ci.1, 4)
out.2 <- logistic.boot(out.1)
replot(out.2, bucket.lim = c(0, 1))

ci.2 <- subset(ci, ci$LookType == "Cohort" | ci$LookType == "Unrelated")
ci.2$Group <- ci.2$protocol
ci.2$Curve <- ifelse(ci.2$LookType == "Cohort", 1, 2)
out.1 <- doubleGauss.fit(ci.2, 4, diff = TRUE)
out.1 <- doubleGauss.refit(out.1, subj = c(13, 23), group = c(2, 2),
curves = c(2, 2), cor = c(FALSE, FALSE))
out.2 <- doubleGauss.boot(out.1)
replot(out.2, ylim = c(-0.01, 0.1), bucket.lim = c(0, 0.08))

## End(Not run)
```

---

ci

*Eyetracking Data from Normal Hearing Individuals and those with Cochlear Implants*

Description

This data set has stuff on CIs and NHs. Provide more info here.

Usage

`ci`

Format

A matrix of values

References

Description

Plots of a histogram of the parameter estimates

Usage

ests.plot(part1.list)

Arguments

part1.list  Output from doubleGauss.fit or logistic.fit

Details

Plots of a histogram of the parameter estimates

Value

NULL

Note

There are no further notes

Examples

## Not run:
data(ci)
ci.1 <- subset(ci, ci$LookType == "Target")
ci.1$Group <- ci.1$protocol
out.1 <- logistic.fit(ci.1, 4)ests.plot(out.1)
ests.plot(out.1)
## End(Not run)
Fitting Step

Fit Subjects Individual Curves

Description

Fit Subjects from 2 groups with the 6-parameter Double Gaussian or the 4-parameter Logistic

Usage

doublegauss.fit(data, col, concave = TRUE, diffs = FALSE, rho.0 = 0.9, cor = TRUE, cores = 1)
logistic.fit(data, col, diffs = FALSE, rho.0 = 0.9, cor = TRUE, cores = 1)

Arguments

data
  data.frame. A data.frame with the columns 'Subject', 'Time', and 'Group'. 'Subject' designates the subject number (numeric), 'Time' designates the time (numeric, should be ordered from low to high), and 'Group' designates the group (should be 2 unique groups)

col	numeric. The column in the data.frame that corresponds to the eyetracking

concave
  boolean. TRUE indicates concave UP, FALSE indicates concave DOWN. Only for Double Gaussian.

diffs
  boolean. If the each group is calculating the difference of 2 logistic curves, set to TRUE. In this case, there needs to be a numeric 'Curve' column (with only 1s and 2s) designating the secondary curve (2) to subtract from the primary curve (1).

rho.0
  numeric (Between 0 and 1). Assumed autocorrelation of errors for individual subject’s curve

cor
  boolean. If TRUE assumes an AR1 autocorrelation structure among the residuals for fitting

cores
  integer. Number of cores on the localhost to use

Details

Fit Subjects from 2 groups with the 6-parameter Double Gaussian or the 4-parameter Logistic

Value

List: for input into refit, boot, plot.est, and plot.subjs functions

Note

There are no further notes
Examples

```r
## Not run:
data(ci)
ci.1 <- subset(ci, ci$LookType == "Target")
ci.1$Group <- ci.1$protocol
ciN1 <- logistic.fit(ci.1, 4)
out1 <- logistic.boot(outN1)
replot(out1, bucket.lim = c(0, 1))

ci.2 <- subset(ci, ci$LookType == "Cohort" | ci$LookType == "Unrelated")
ci.2$Group <- ci.2$protocol
ci.2$Curve <- ifelse(ci.2$LookType == "Cohort", 1, 2)
out1 <- doubleGauss.fit(ci.2, 4, diffs = TRUE)
out1 <- doubleGauss.refit(out1, subj = c(13, 23), group = c(2, 2),
curves = c(2, 2), cor = c(FALSE, FALSE))
out2 <- doubleGauss.boot(out1)
replot(out2, ylim = c(-0.01, 0.1), bucket.lim = c(0, 0.08))

## End(Not run)
```

---

**Print Fits**

*Print summary of quality of curve fits*

**Description**

Print summary of quality of curve fits

**Usage**

`printFits(part1.list)`

**Arguments**

- `part1.list`: list. Output from `doubleGauss.fit`

**Details**

Print summary of quality of curve fits

**Note**

There are no further notes
Examples

```r
## Not run:
data(ci)
ci.2 <- subset(ci, ci$LookType == "Cohort" | ci$LookType == "Unrelated")
ci.2$Group <- ci.2$protocol
ci.2$Curve <- ifelse(ci.2$LookType == "Cohort", 1, 2)
out.1 <- doubleGauss.fit(ci.2, 4, diffs = TRUE)
out.1 <- doubleGauss.refit(out.1, subj = c(13, 23), group = c(2, 2),
curves = c(2, 2), cor = c(FALSE, FALSE))
printfits(out.1)
out.2 <- doubleGauss.boot(out.1)
replot(out.2, ylim = c(-0.01, 0.1), bucket.lim = c(0, 0.08))
## End(Not run)
```

### Refitting Step

**Description**

Refit Subjects from 2 groups with the 6-parameter Double Gaussian or the 4-parameter Logistic. Can specify starting parameters.

**Usage**

```r
doubleGauss.refit(part1.list, subj = NULL, group = NULL, curves = NULL,
params = NULL, cor=NULL, rho.0 = NULL, info.matrix = NULL)
logistic.refit(part1.list, subj = NULL, group = NULL, curves = NULL,
params = NULL, cor = NULL, rho.0 = NULL, info.matrix = NULL)
```

**Arguments**

- `part1.list`: Output from fitting step
- `info.matrix`: numeric matrix. Can put subj/group/curves/params information here for convenience. Columns should have the following order: subj, group, curves, params. This matrix cannot contain any NA/NaN values. If diffs=FALSE, this information is not used – simply use filler values.
- `subj`: numeric vector. Subject numbers (within their group) that you want to refit
- `group`: numeric vector. Group numbers corresponding to subject vector
- `curves`: numeric vector. Curve numbers if it’s a difference of fits (i.e. diffs = TRUE)
- `params`: list of numeric vectors (length 4 or 6). Parameter estimates for the 6 parameter DoubleGauss in the order mu, height, sd 1, sd 2, base 1, base 2. Parameter estimate for the 4 parameter Logistic in the order min, peak, slope, crossover
- `cor`: logical vector. If TRUE assumes an correlation structure of AR(rho) and if FALSE assumes no correlation structure
- `rho.0`: numeric. assumed autocorrelation of errors for subject’s curve
Details

Refit Subjects from 2 groups with the 6-parameter Double Gaussian or the 4-parameter Logistic. Can specify starting parameters

Value

List

Note

There are no further notes

Examples

```r
## Not run:
data(ci)

## Not run:
ci.2 <- subset(ci, ci$LookType == "Cohort" | ci$LookType == "Unrelated")

## Not run:
ci.Group <- ci.2$protocol
ci.2$Curve <- ifelse(ci.2$LookType == "Cohort", 1, 2)

## Not run:
out.1 <- doubleGauss.fit(ci.2, 4, diffs = TRUE)

## Not run:
out.1 <- doubleGauss.refit(out.1, subj = c(13, 23), group = c(2, 2),
                          curves = c(2, 2), cor = c(FALSE, FALSE))

## Not run:
out.2 <- doubleGauss.boot(out.1)

## Not run:
replot(out.2, ylim = c(-0.01, 0.1), bucket.lim = c(0, 0.08))

## Not run:

## End(Not run)
```

---

**replot**

*Replot Boostrapped Output*

Description

Plot the boostrapped output with different parameters than the default ones.

Usage

```
replot(part2.list, xlim = NULL, ylim = c(0, 1), main = "Curve",
       legend.location = "topleft", bucket.lim = c(0, .9))
```

Arguments

- `part2.list`: list. Output from `doubleGauss.boot` or `logistic.boot`
- `xlim`: numeric vector (length = 2). Start and end point of x-axis. If NULL, takes the full time course
- `ylim`: numeric vector (length = 2). Start and end point of y-axis
- `main`: string. Title
Delete individual subjects from both data and fits

**Usage**

```r
subs.delete(part1.list, subj, group)
```

**Arguments**

- `part1.list` 
  Output from `doubleGauss.fit` or `logistic.fit`
- `subj` 
  numeric. Subject numbers to remove.
- `group` 
  numeric. Corresponding group numbers to subjects.
Details

Delete individual subjects from both data and fits

Value

part1.list

Examples

```r
## Not run:
data(ci)
ci.1 <- subset(ci, ci$LookType == "Target")
ci.1$Group <- ci.1$protocol
out.1 <- logistic.fit(ci.1, 4)ests.plot(out.1)
#Remove subject 1 from group 1 and subject 10 from group 2
out.1 <- subjs.delete(out.1, subj = c(1, 10), group = c(1, 2))
## End(Not run)
```

---

**subs.plot**

*Plot subjects raw data along with function fits*

Description

Plot subjects raw data along with function fits

Usage

```r
subs.plot(part1.list, legend.spot = "topright", ylim = NULL)
```

Arguments

- `part1.list`: Output from doubleGauss.fit or logistic.fit
- `legend.spot`: string. Location of the legend
- `ylim`: If NULL, takes the min and max of all observed curves

Details

Plot subjects raw data along with function fits

Value

NULL

Note

Options for legend.location include "topleft", "top", "topright", "right", "bottomright", "bottom", "bottomleft", "left"
Examples

```r
## Not run:
data(ci)
ci.1 <- subset(ci, ci$LookType == "Target")
ci.1$Group <- ci.1$protocol
out.1 <- logistic.fit(ci.1, 4)ests.plot(out.1)
subs.plot(out.1)

## End(Not run)
```
Index

*Topic datasets
  ci, 4

*Topic htest
  bdots.write.csv, 2
  Bootstrap Step, 3
  ests.plot, 5
  Fitting Step, 6
  Print Fits, 7
  Refitting Step, 8
  replot, 9
  subs.delete, 10
  subs.plot, 11

bdots.write.csv, 2
Bootstrap Step, 3

ci, 4

doubleGauss.boot (Bootstrap Step), 3
doubleGauss.fit (Fitting Step), 6
doubleGauss.refit (Refitting Step), 8

ests.plot, 5
Fitting Step, 6

logistic.boot (Bootstrap Step), 3
logistic.fit (Fitting Step), 6
logistic.refit (Refitting Step), 8

Print Fits, 7
printFits (Print Fits), 7

Refitting Step, 8
replot, 9

subs.delete, 10
subs.plot, 11