Package ‘welchADF’

September 8, 2019

Title  Welch-James Statistic for Robust Hypothesis Testing under Heterocedasticity and Non-Normality


Version  0.3.2

Date  2019-09-08

Author  Pablo J. Villacorta <pjvi@decsai.ugr.es>

Maintainer  Pablo J. Villacorta <pjvi@decsai.ugr.es>

License  LGPL (>= 3)

URL  <http://decsai.ugr.es/~pjvi/r-packages.html>

Imports  lme4

LazyData  true

RoxygenNote  6.0.1

Suggests  testthat

NeedsCompilation  no

Repository  CRAN

Date/Publication  2019-09-08 07:50:02 UTC

R topics documented:

  adhdData ................................................................. 2
  adhdData2 ............................................................... 3
  confint.welchADFt ....................................................... 4
**adhdData**

- **miceData**
- **perceptionData**
- **summary.welchADPf**
- **welchADF.test**
- **womenStereotypeData**

**Index**

adhdData  
Children’s reaction times (milliseconds) to stimuli of different nature, arranged with four response columns.

**Description**

The data (Keselman et al., 2003) represent the reaction times in milliseconds of children with attention-deficit hyperactivity (ADHD) and normal children when they are presented four kinds of inputs: a target alone or an arrow stimuli incongruent, congruent and neutral to the target. According to the authors, the dataset was artificially generated from the summary measures given in the original study by Jonkman et al. (1999), in groups of 20 and 10 children to create an unbalanced design.

**Usage**

adhdData

**Format**

A data frame with 30 rows and 5 variables:

- **Group** whether the child has ADHD or is healthy (normal).
- **TargetAlone** Reaction time (milliseconds) to a target alone.
- **Congruent** Reaction time (milliseconds) to a congruent stimulus.
- **Neutral** Reaction time (milliseconds) to a neutral stimulus.
- **Incongruent** Reaction time (milliseconds) to an incongruent stimulus.

**Source**


**References**

Examples

# Assuming a multivariate response
omnibus_LSM_multi <- welchADF.test(adhdData, response = c("TargetAlone", "Congruent", "Neutral", "Incongruent"), between.s = "Group", within.s = "multivariate", contrast = "omnibus")
# The same using the S3 method for class formula
omnibus_LSM_multi_form <- welchADF.test(cbind(TargetAlone, Congruent, Neutral, Incongruent) ~ Group, data = adhdData)

# Pairwise comparisons of the implicit within-subjects effect present in the multivariate response
pairwise_LSM_multi <- update(omnibus_LSM_multi, contrast = "all.pairwise", effect = "multivariate")
summary(omnibus_LSM_multi)
summary(pairwise_LSM_multi)

adhdData2

Children’s reaction times (milliseconds) to stimuli of different nature, arranged with one single response column and taking the multi-variate response as an explicit within-subjects factor.

Description

Exactly the same data explained in "adhdData" but reshaped as follows.

Usage

adhdData2

Format

A data frame with 120 rows and 4 variables:

- **Group**: whether the child has ADHD or is healthy (normal).
- **Stimulus**: the stimulus to which the reaction time in this row corresponds.
- **Subject**: an integer ID that identifies the subject to which the reaction time corresponds.
- **Milliseconds**: reaction time (milliseconds) to of the aforementioned subject to the aforementioned stimulus.

Examples

# Omnibus test of a mixed between x within subjects model,
# using trimmed means and Winsorized variances
omnibus_trimmer <- welchADF.test(adhdData2, response = "Milliseconds", between.s = "Group", within.s = "Stimulus", subject = "Subject", contrast = "omnibus", trimming = TRUE)

# The same using S3 method for class formula. The data can be in separate variables in
# the scope of the call, not necessarily in a data.frame
millis <- adhdData2$Milliseconds
gr <- adhdData2$Group
st <- adhdData2$Stimulus
confint.welchADFt

Confidence interval for the effect size of model effects

Description

For the effects indicated by the user, extracts confidence intervals for the effect size from an object of class `confint.welchADFt` returned by a call to `welchADF.test`.

Usage

```r
## S3 method for class 'confint.welchADFt'
confint(object, parm, level = 0.95, ...)
```

Arguments

- `object`: a object of (S3 class) `confint.welchADFt`
- `parm`: a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
- `level`: confidence level (completely ignored, since the confidence threshold had been in the call to `welchADF.test`).
- `...`: additional argument(s) for methods.

Value

If `effect.size` was set to `TRUE` in the call to `welchADF.test`, a matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as \((1\text{-level})/2\) and \(1\cdot(1\text{-level})/2\) in `confint.welchADFt`.

See Also

`welchADF.test`, `confint.lm`
miceData

Number of visits and time spent in different tunnels of laboratory mice

Description

Wild strain house mice were, at birth, cross fostered onto house mouse (Mus), deer mouse (Peromyscus) or rat (Rattus) nursing mothers. Ten days after weaning, each subject was tested in an apparatus that allowed it to enter four different tunnels: one scented with clean pine shavings, and the other three tunnels with shavings bearing the scent of Mus, Peromyscus, or Rattus respectively. Three variables were measured for each tunnel: the number of visits to the tunnel during a twenty minute test, the time spent by each subject in each of the four tunnels and the latency to first visit of each tunnel.

Usage

miceData

Format

A data frame with 144 rows and 6 variables:

- **Subject**: an ID that identifies the mouse to which the measurements of the row correspond.
- **nurs**: type of nursing mother this mouse had at birth.
- **tunnel**: type of tunnel to which the measurements of this row correspond.
- **visits**: number of visits to the aforementioned tunnel.
- **time**: total time spent by the mouse at this tunnel.
- **latency**: seconds passed before the mouse first visited this tunnel.

Source

http://core.ecu.edu/psyc/wuenschk/SPSS/TUNNEL4b.sav

References


Examples

```r
omnibus_LSM <- welchADF.test(miceData, response = c("visits", "time", "latency"),
       between.s = "nurs", within.s = "tunnel", subject = "Subject", contrast = "omnibus")
# Formula interface, using cbind() to specify a multivariate response.
omnibus_LSM_formula <- welchADF.test( 
       cbind(visits, time, latency) ~ nurs*tunnel + (tunnel|Subject), data = miceData)
pairwise_LSM_nurs <- welchADF.test(miceData, response = c("visits", "time", "latency"),
       between.s = "nurs", within.s = "tunnel", subject = "Subject",
       effect = "nurs", contrast = "all.pairwise")
```
perceptionData

Number of puzzles 42 students were able to solve

Description

An artificial dataset created by Lix et al. which recreates data reported by a real study on perception and concentration, on which 42 students were given several puzzles to be solved. The students are divided into three balanced groups as they had previously been asked to imagine solving puzzles in the distant future, near future, or not to imagine anything at all (control group).

Usage

perceptionData

Format

A data frame with 42 rows and 2 variables:

- **Group** group of the student (what the student was asked to imagine)
- **y** number of puzzles the student was able to solve, out of 12

References


Examples

```r
omnibus_LSM <- welchADF.test(perceptionData, response = "y", between.s = "Group")
omnibus_trimmed <- update(omnibus_LSM, trimming = TRUE)
pairwise_LSM <- update(omnibus_LSM, effect = "Group", contrast = "all.pairwise")
pairwise_trimmed <- update(pairwise_LSM, trimming = TRUE, effect.size = TRUE)
summary(omnibus_LSM)
## Not run:
pairwise_trimmed_boot <- update(pairwise_trimmed, bootstrap = TRUE, seed = 12345, numsim_b = 600)
summary(pairwise_trimmed_boot, digits = 6) # digits defaults to max(4, getOption("digits"))
## End(Not run)
```
**summary.welchADFt**

*Summary: Welch Approximate Degrees of Freedom test results*

**Description**

Summary and print methods for class "welchADF".

**Usage**

```r
## S3 method for class 'welchADFt'
summary(object, verbose = FALSE, digits = max(4,
    getOption("digits")), ...)

## S3 method for class 'welchADFt'
print(x, digits = getOption("digits"), ...)
```

**Arguments**

- `object`: an object of class "welchADFt" returned by a call to `welchADF.test`.
- `verbose`: whether the summary table should be preceded by an extended description of the Welch ADF test performed, including the kind of means (trimmed or not), kind of test (omnibus or pairwise) and whether bootstrap was used or not. Defaults to FALSE.
- `digits`: the number of significant digits to use when printing.
- `...`: further arguments passed to or from other methods.
- `x`: an object of class "welchADFt".

**Value**

A string which summarizes the info, with class "summary.welchADFt" ready to be printed by specific method `print.summary.welchADFt`.

**welchADF.test**

*General Approximate Degrees of Freedom Solution for Inference and Estimation*

**Description**

Computes the Welch-James statistic with Approximate Degrees of Freedom, based on the SAS code by H.J. Keselman, R.R. Wilcox and L.M. Lix.
Usage
welchADF.test(formula, ...)

## S3 method for class 'formula'
welchADF.test(formula, data, subset, ...)

## S3 method for class 'lm'
welchADF.test(formula, ...)

## S3 method for class 'aov'
welchADF.test(formula, ...)

## S3 method for class 'lmer'
welchADF.test(formula, ...)

## Default S3 method:
welchADF.test(formula, response, between.s = NULL,
              within.s = NULL, subject = NULL, contrast = c("omnibus",
              "all.pairwise"), effect = NULL, correction = c("hochberg",
              "holm"),
              trimming = FALSE, per = 0.2, bootstrap = FALSE, numsim_b = 999,
              effect.size = FALSE, numsim_es = 999, scaling = TRUE,
              standardize.effsz = TRUE, alpha = 0.05, seed = 0, ...)

Arguments

formula A data frame or a formula or an lm object returned by lm. If formula is a data
          frame, it must have the format described for argument data of WelchADF.test
          for class formula. If formula is a formula, it must be a two-sided linear
          formula object describing the between-subjects effects of the model, with the
          response variable(s) (surrounded by cbind()) if it is a multivariate response, as
          cbind(r1, r2, r3) on the left of the ~ operator and the terms on the right.
          Within-subject factors must be specified as a parenthesis term on the rhs with
          the form (w1*w2*...|subject) with 'subject' being the name of the subject ID
          column, which must be present in the data (cannot be omitted like in the data.frame
          version) e.g. cbind(y1, y2) ~ A*B*C + (B*C|subject). In this example, A would
          be a between-subject factor because it is the only variable not present again on
          the left side of the parenthesis term (where the within-subjects factors are
          specified). Multi-level or customized structures are not supported, e.g. those having
          strata or more than one random factor. The formula does not affect the results
          of the tests as no 'model' is actually fit to the data. It only affects the terms
          displayed as a result of an omnibus contrast when no effect argument is given
          (which means run an omnibus test for every effect or interaction (up to second
          order)). In such case, the factors or interactions not specified in the formula will
          not be applied an omnibus test. In case of a pairwise contrast, the structure in
          the formula is ignored beyond interpreting which variables are between-subject
          factors and which are within-subjects factors.

... Further arguments to be passed to specialized methods. Should be named arguments.
data  A data.frame with the data, formatted as follows: one row per observation of the response variable(s), and as many columns as needed to indicate the factor combination to which the observation corresponds. If necessary, an extra column with the subject ID for designs having within-subjects factors (can be omitted if there is only one within-subjects factor, see the vignette).

subset A specification of the rows to be used as in `model.frame`: defaults to all rows. This can be any valid indexing vector (see `[.data.frame) for the rows of data or if that is not supplied, a data frame made up of the variables used in formula.

response A string or vector of strings with the name(s) of the column(s) of data corresponding to the response variable(s). If a vector of strings is provided, the responses are taken as a set of repeated measurements or dependent variables.

between.s Vector of strings with the columns that correspond to between-subjects factors, if any (defaults to NULL).

within.s Vector of strings with the columns that correspond to within-subjects factors, if any (defaults to NULL).

subject Name of the column (if any) containing the subject ID (defaults to NULL).

contrast One of "omnibus", "all.pairwise" indicating the type of contrast to be performed with the data. "omnibus" stands for omnibus contrasts on either all the variables found in the data, including the within-subjects factor if any, or on a single effect specified in the `effect` argument. "all.pairwise" stands for pairwise comparisons of the response(s) variable(s) over all combinations of values of one effect, indicated in the `effect` parameter. Defaults to "omnibus".

effect If contrast = "omnibus", then `effect` must be a string or vector of strings with the name of the effect(s) for which the "omnibus" contrasts will be applied. If contrast = "all.pairwise", then `effect` must be a tagged list in which the names are the factors, and the elements must be vectors of two strings each, indicating If contrast = "omnibus" but `effect` is not specified, then an omnibus contrast will be done separately for each of the effects and also all possible interactions. If contrast = "omnibus" and `effect` is a vector of strings, then only one contrast will be done, involving the simultaneous interaction of all effects indicated in the `effect` vector (i.e. no other interaction or main effect contrasts will be performed).

correction The type of p-value correction when applying multiple pairwise comparisons (i.e. when contrast = "all.pairwise"). Defaults to "hochberg".

trimming Boolean to control the use of robust estimators. FALSE indicates the ADF test statistic will be computed with the usual Least-Squares Estimators. TRUE indicates that trimmed means and Winsorized variances/covariances will be used to compute the ADF test statistic.

per (Only if trimming is TRUE) The proportion of trimming in EACH tail of the data distribution. Must be between 0 and .49 (i.e., 49% trimming in each tail). Recommended per = 0.2 symmetric trimming (i.e., 20% of the observations in each tail are trimmed).

bootstrap Boolean; whether bootstrap should be used to compute a critical value for the test statistic produced by WJGLM. FALSE means a theoretical critical value for the test statistic TWJ/c (or TWJu/c) will be computed. TRUE indicates the non-parametric percentile bootstrap method will be used to compute a critical value.
When combined with `contrast = "all.pairwise"`, then Family-Wise Error Rate (FWER) control is achieved using bootstrapping to compute an adjusted empirical critical value (refer to Keselman, Wilcox & Lix 2003, page 589 for details).

**numsim_b**  
If `bootstrap = TRUE`, this is a positive integer that defines the number of bootstrap samples to generate a critical value (defaults to 999). Carpenter and Bithell (2000) recommend between 1000 and 2000 samples for setting confidence intervals around a parameter estimate, but Efron and Tibshirani (1986) note that for other applications of the bootstrap, such as computing the standard error, as few as 100 bootstrap samples will be sufficient. Choosing a value of B such that $(B + 1)\alpha$ is an integer value is recommended, because it avoids the need for interpolation.

**effect.size**  
Boolean; whether effect size estimates should be computed. TRUE means effect size estimates together with its confidence intervals will be computed (defaults to FALSE).

**numsim_es**  
Positive integer defining the number of bootstrap samples to generate a CI for the effect size estimate (defaults to 999). Ignored if `effect.size = FALSE`.

**scaling**  
Boolean; whether a scaling factor for the effect size estimator should be used (0.642 for 20% symmetric trimming) when robust estimators are adopted. FALSE means no scaling factor will be used. TRUE indicates that a scaling factor will be adopted. Defaults to TRUE.

**standardize.effsz**  
Boolean: whether the effect size should be standardized by the average of variances or not. Defaults to TRUE.

**alpha**  
Significance threshold for the confidence interval calculation, where $1 - \alpha$ is the probability coverage. Defaults to 0.05.

**seed**  
Initial seed (positive integer) for the (R default) random number generator. Defaults to NULL (seed taken from the current time).

**Value**

An object of class "welchADFt" which is a list of lists (one sub-list per effect, even if there is only one). There are methods `print.welchADFt` and `summary.welchADFt`.

**Methods (by class)**

- **formula**: Specialized method that accepts a formula
- **lm**: Specialized method that accepts a linear model object of class `lm`
- **aov**: Specialized method that accepts an Analysis of Variance Model of class `aov`
- **lmer**: Specialized method that accepts a Linear Mixed-Effects Model of class `lmer`
- **default**: Default method that accepts a data.frame (see argument `formula`) and where factors are passed as strings corresponding to column names
welchADF.test

References


Website with the original SAS code and examples: [http://supp.apa.org/psycarticles/supplemental/met_13_2_110/met_13_2_110_supp.html](http://supp.apa.org/psycarticles/supplemental/met_13_2_110/met_13_2_110_supp.html)


See Also

`print.welchADFt, summary.welchADFt`

`p.adjust.methods perceptionData adhdData adhdData2 womenStereotypeData miceData`

Examples

# Omnibus contrast only of those effects included, namely condition and sex (no interaction)
omnibus_LSM_formula <- welchADF.test(y ~ condition + sex, data = womenStereotypeData)

# Works well with update.default() method
omnibus_interact_formula <- update(omnibus_LSM_formula, . ~ condition*sex)

summary(omnibus_LSM_formula)
summary(omnibus_interact_formula)

# Fit a linear model using the built-in function stats::lm
lm.women <- lm(y ~ condition + sex, womenStereotypeData)

# Fit an Analysis of Variance model using the built-in function stats::aov
aov.women <- aov(lm.women)

# Now use the this object to apply a welchADF test to the same formula and data
omnibus_no_interact <- welchADF.test(lm.women, contrast = "omnibus")

# Integrates well with the update.default() method
omnibus_interact <- update(omnibus_no_interact, . ~ condition*sex)

summary(omnibus_no_interact)
summary(omnibus_interact)

# Two-way factorial design using the default interface. See the vignette.
omnibus_LSM <- welchADF.test(womenStereotypeData, response = "y", between.s = c("condition", "sex"), contrast = "omnibus")

# Method update() also works with the welchADF.test.default interface
omnibus_trimmed <- update(omnibus_LSM, effect = "condition", trimming = TRUE)

pairwise_LSM <- update(omnibus_LSM, contrast = "all.pairwise", effect = c("condition", "sex"))

pairwise_trimmed <- update(pairwise_LSM, trimming = TRUE) # just trimming
womenStereotypeData

Students’ scores (men and women) on an arithmetic test

Description

An artificial dataset created by Lix et al. from summary data presented by Wicherts et al. (2005) (see the vignette). These authors examined the effects of stereotype threat on women’s mathematics ability. Originally there were four different tests administered to study participants (arithmetic, number series, word problems, and sums tests). The dataset contains only scores for the arithmetic test because these scores exhibited a greater magnitude of variance heterogeneity than scores for the other tests. It is an unbalanced design with cell sizes ranging from 45 to 50 participants, and a total sample size of 283.

Usage

womenStereotypeData

Format

A data frame with 283 rows and 3 variables:

- **condition** test condition (control, nullified, stereotype threat)
- **sex** the individual’s sex (male, female)
- **y** score on the arithmetic test, out of 40

Source


References

Examples

omnibus_LSM <- welchADF.test(womenStereotypeData, response = "y", between.s =
  c("condition", "sex"), contrast = "omnibus", effect = "condition")
omnibus_trimmed <- update(omnibus_LSM, trimming = TRUE, effect = NULL) # unset value of "effect"
pairwise_LSM <- update(omnibus_LSM, contrast = "all.pairwise", effect = c("condition", "sex"))
pairwise_trimmed <- update(pairwise_LSM, trimming = TRUE)
summary(omnibus_LSM)
summary(omnibus_trimmed)
summary(pairwise_trimmed)
## Not run:
  pairwise_trimmed_boot <- update(pairwise_trimmed, bootstrap = TRUE, seed = 12345)
  summary(pairwise_trimmed_boot)

## End(Not run)
Index

*Topic datasets
  adhdData, 2
  adhdData2, 3
  miceData, 5
  perceptionData, 6
  womenStereotypeData, 12

adhdData, 2, 11
adhdData2, 3, 11
aov, 10

confint.lm, 4
confint.welchADFt, 4

lm, 8, 10
lmer, 10

miceData, 5, 11
model.frame, 9

p.adjust.methods, 11
perceptionData, 6, 11
print.welchADFt, 10, 11
print.welchADFt(summary.welchADFt), 7

summary.welchADFt, 7, 10, 11

welchADF.test, 4, 7, 7
womenStereotypeData, 11, 12