Package ‘TukeyC’

May 24, 2026

Version 1.4-0

Date 2026-05-23

Title Conventional Tukey Test

Depends R (>= 4.0.0)

Imports emmeans, xtable

Suggests pbkrtest (>= 0.4-6), lme4, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Description Performs multiple comparison analyses using Tukey's Honestly Significant Difference (HSD) test, with intuitive letter grouping of means for balanced and unbalanced designs. Accepts input from 'formula', 'aov', 'lm', 'aovlist', and 'lmerMod' objects, including straightforward handling of interactions. For more details see Tukey (1949) <doi:10.2307/3001913>.

License GPL (>= 2)

URL `https://github.com/jcfaria/TukeyC`,

`https://lec.pro.br/software/pac-r/tukeyc`

BugReports `https://github.com/jcfaria/TukeyC/issues`

Encoding UTF-8

LazyData true

NeedsCompilation no

Author J. C. Faria [aut],
E. G. Jelihovschi [aut],
I. B. Allaman [aut, cre]

Maintainer I. B. Allaman <`ivanalaman@gmail.com`>

Repository CRAN

Date/Publication 2026-05-24 11:40:02 UTC

Contents

`boxplot.TukeyC` *Boxplot TukeyC Objects*

Description

S3 method to plot `TukeyC` objects.

Usage

```
## S3 method for class 'TukeyC'
boxplot(x,
          mean.type   = c('line', 'point', 'none'),
          xlab        = NULL,
          mean.col    = 'gray',
          mean.pch    = 1,
          mean.lwd    = 1,
          mean.lty    = 1,
          args.legend = NULL, ...)
```

Arguments

`x`	A `TukeyC` object.
`mean.type`	The type of mean must be plotted. The default is "line".
`xlab`	A label for the ‘x’ axis.

`mean.col`	A vector of colors for the means representation.
`mean.pch`	A vector of plotting symbols or characters. Only when `mean.type` is `"point"`.
`mean.lwd`	Line width of mean.
`mean.lty`	Line type of mean. Only when `mean.type` is `"line"`.
`args.legend`	List of additional arguments to be passed to `legend`; The default is `NULL`.
`...`	Optional plotting parameters.

Details

The `boxplot.TukeyC` function is an S3 method to plot `TukeyC` objects. Unlike the generic `boxplot`, it displays Tukey group letters and overlays treatment means on the boxes.

Value

'NULL' (invisibly). The main purpose of this function is to produce a plot.

Author(s)

Faria, J. C. (<`joseclaudio.faria@gmail.com`>)
Jelihovschi, E. G. (<`eniojelihovs@gmail.com`>)
Allaman, I. B. (<`ivanalaman@gmail.com`>)

References

Murrell, P. (2005) *R Graphics*. Chapman and Hall/CRC Press.

See Also

`boxplot`

Examples

```
##
## Examples: Completely Randomized Design (CRD)
## More details: demo(package='TukeyC')
##

library(TukeyC)
data(CRD1)

## From: formula
# Simple!
tk1 <- TukeyC(y ~ x,
              data=CRD1$dfm,
              which='x')
boxplot(tk1)

# A little more elaborate!
boxplot(tk1,
        mean.lwd=1.3,
```

```
        mean.col='red')

# A little more!
boxplot(tk1,
        mean.lwd=1.3,
        mean.lty=2,
        mean.col='red',
        args.legend=list(x='bottomleft'))

# With point type!
boxplot(tk1,
        mean.type='point')

boxplot(tk1,
        mean.type='point',
        mean.pch=19,
        cex=1.5,
        mean.col='red')

# With other point
boxplot(tk1,
        mean.type='point',
        mean.pch='+',
        cex=2,
        mean.col='blue',
        args.legend=list(x='bottomleft'))
```

`CRD1` *Completely Randomized Design (CRD)*

Description

A `list` illustrating the resources of `TukeyC` package related to Completely Randomized Design (‘`CRD`’).

Usage

```
data(CRD1)
```

Details

A simulated data to model a Completely Randomized Design (‘`CRD`’) of 4 factor levels and 6 repetitions.

`CRD2` *Completely Randomized Design (‘`CRD`’)*

Description

A `list` illustrating the resources of `TukeyC` package related to Completely Randomized Design (‘`CRD`’).

Usage

```
data(CRD2)
```

Details

A simulated data to model a Completely Randomized Design (‘`CRD`’) of 45 factor levels and 4 repetitions.

`cv` *Coefficient of variation*

Description

Returns the coefficient of variation from models `lm`, `aov` and `aovlist`.

Usage

```
cv(x,
   round=2)
```

Arguments

`x`	An object of class `lm`, `aov`, or `aovlist`.
`round`	An integer value indicating the number of decimal places to be used. The default value is 2.

Details

sqrt(MSerror)*100/mean(x)

Value

`x`	named numeric vector

Author(s)

Faria, J. C. (<`joseclaudio.faria@gmail.com`>)
Jelihovschi, E. G. (<`eniojelihovs@gmail.com`>)
Allaman, I. B. (<`ivanalaman@gmail.com`>)

Examples

```
library(TukeyC)

## Completely Randomized Design (CRD - aov)
data(CRD1)
av1 <- with(CRD1,
            aov(y ~ x,
                data=dfm))
summary(av1)
cv(av1)

## Randomized Complete Block Design (RCBD - aov)
data(RCBD)
av2 <- with(RCBD,
            aov(y ~ blk + tra,
                data=dfm))
summary(av2)
cv(av2)

## Split-plot experiment (SPE - aovlist)
data(SPE)
av3 <- with(SPE,
            aov(y ~ blk + P*SP + Error(blk/P),
                data=dfm))
summary(av3)
cv(av3)

## Split-split-plot experiment (SSPE - aovlist)
data(SSPE)
av4 <- with(SSPE,
            aov(y ~  blk + P*SP*SSP + Error(blk/P/SP),
                data=dfm))
summary(av4)
cv(av4)

## storing
res <- cv(av4)
res[2:3]
```

`FE` *Factorial Experiment (FE)*

Description

A `list` illustrating the resources of `TukeyC` package related to Factorial Experiment (‘`FE`’).

Usage

```
data(FE)
```

Details

A simulated data to model a Factorial Experiment (‘`FE`’) with 3 factors, 2 levels per factor and 4 blocks.

`LSD` *Latin Squares Design (LSD)*

Description

A `list` illustrating the resources of `TukeyC` package related to Latin Squares Design (‘`LSD`’).

Usage

```
data(LSD)
```

Details

A simulated data to model a Latin Squares Design (‘`LSD`’) with 5 5 treatment levels, 5 rows, and 5 columns.

`plot.TukeyC` *Plot TukeyC and TukeyC.nest Objects*

Description

S3 method to plot `TukeyC` and `TukeyC.nest` objects.

Usage

```
## S3 method for class 'TukeyC'
plot(x,
     result         = TRUE,
     replicates     = TRUE,
     pch            = 19,
     col            = NULL,
     xlab           = NULL,
     ylab           = NULL,
```

```
         xlim           = NULL,
         ylim           = NULL,
         axisx          = TRUE,
         axisy          = TRUE,
         id.lab         = NULL,
         id.las         = 1,
         yl             = TRUE,
         yl.lty         = 3,
         yl.col         = 'gray',
         dispersion     = c('mm','sd','ci','cip'),
         d.lty          = 1,
         d.col          = 'black',
         title          = '', ...)
```

Arguments

`x`	A `TukeyC` object.
`result`	The result of the test (letters) should be visible.
`replicates`	The number of replicates should be visible.
`pch`	A vector of plotting symbols or characters.
`col`	A vector of colors for the means representation.
`xlab`	A label for the ‘`x`’ axis.
`ylab`	A label for the ‘`y`’ axis.
`xlim`	The ‘`x`’ limits of the plot.
`ylim`	The ‘`y`’ limits of the plot.
`axisx`	Axis x? If ‘`TRUE`’ you must accept the default, otherwise, you must customize.
`axisy`	Axis y? If ‘`TRUE`’ you must accept the default, otherwise, you must customize.
`id.lab`	Factor level names at ‘`x`’ axis.
`id.las`	Factor level names written either horizontally or vertically.
`yl`	Horizontal (reference) line connecting the circle to the ‘`y`’ axis.
`yl.lty`	Line type of ‘`yl`’.
`yl.col`	Line color of ‘`yl`’.
`dispersion`	Type of dispersion bar drawn through each mean point. Options: ‘`mm`’ (min-max range), ‘`sd`’ (standard deviation), ‘`ci`’ (individual confidence interval), ‘`cip`’ (pooled confidence interval). Default is ‘`mm`’.
`d.lty`	Line type of dispersion.
`d.col`	A vector of colors for the line type of dispersion.
`title`	A title for the plot.
`...`	Optional plotting parameters.

Details

The `plot.TukeyC` function is an S3 method to plot `TukeyC` and `TukeyC.nest` objects. It generates a series of points (the means) and a vertical line showing the dispersion of the values corresponding to each group mean. With `dispersion = "ci"`, intervals use each treatment variance as an estimate of the population variance. With `dispersion = "cip"`, intervals use the mean square error (MSE).

Value

‘NULL‘ (invisibly). The main purpose of this function is to produce a plot.

Author(s)

Faria, J. C. (<`joseclaudio.faria@gmail.com`>)
Jelihovschi, E. G. (<`eniojelihovs@gmail.com`>)
Allaman, I. B. (<`ivanalaman@gmail.com`>)

References

Murrell, P. (2005) *R Graphics*. Chapman and Hall/CRC Press.

See Also

`plot`

Examples

```
##
## Examples: Completely Randomized Design (CRD)
## More details: demo(package='TukeyC')
##

library(TukeyC)
data(CRD2)

## From: formula
tk1 <- with(CRD2,
            TukeyC(y ~ x,
                   data=dfm,
                   which='x'))

old.par <- par(mar=c(6, 3, 6, 2))
plot(tk1,
     id.las=2)

plot(tk1,
     yl=FALSE,
     dispersion='sd',
     id.las=2)

## From: aov
av <- with(CRD2,
```

```
           aov(y ~ x,
               data=dfm))
summary(av)

tk2 <- TukeyC(x=av,
              which='x')
plot(tk2,
     dispersion='sd',
     yl=FALSE,
     id.las=2)

# From: lm
av_lm <- with(CRD2,
           lm(y ~ x,
              data=dfm))

tk3 <- TukeyC(x=av_lm,
              which='x')

par(mfrow=c(2, 1))
plot(tk3,
     dispersion='ci',
     id.las=2,
     yl=FALSE)

plot(tk3,
     dispersion='cip',
     id.las=2,
     yl=FALSE)

par(mfrow=c(1, 1))
par(old.par)
```

`print.TukeyC` *Print Method for* `TukeyC` *objects.*

Description

Returns (and prints) a list for objects of class `TukeyC`.

Usage

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

Arguments

`x`	A given object of the class `TukeyC`.
`...`	Further arguments (required by the generic).

Value

A list with the following elements:

`a list of length 5`
: In the first position of the list there is a data.frame with the means and the groupings. In the second position of the list there is a scalar with the significance level. In the third position there is a matrix with the p-values obtained in each mean comparison. In the fourth position there is another matrix with the values obtained from the minimum significant difference. In the fifth position there is a vector with the number of replicates per treatment.

`a list of length 5`
: In the first position there is a data.frame with the names of the treatments and the means. In the second position there is another data.frame with the means, minimum and maximum of the data. In the third position there are the means with the lower and upper limits of the confidence interval using the standard deviation to calculate the margin of error. In the fourth position there is also a data.frame with the means and the lower and upper limits of a confidence interval using the standard error of the mean of each treatment to calculate the margin of error. In the fifth position there is also a data.frame with the means and the lower and upper limits of the confidence interval using the standard error of the experimental error to calculate the margin of error.

`a list of length 1`
: A call object.

Author(s)

Faria, J. C. (<`joseclaudio.faria@gmail.com`>)
Jelihovschi, E. G. (<`eniojelihovs@gmail.com`>)
Allaman, I. B. (<`ivanalaman@gmail.com`>)

See Also

`TukeyC`

Examples

```
data(RCBD)

tk <- with(RCBD,
   TukeyC(y ~ blk + tra,
  data=dfm,
  which='tra'))
tk
```

`RCBD` *Randomized Complete Block Design (RCBD)*

Description

A `list` illustrating the resources of `TukeyC` package related to Randomized Complete Block Design (‘`RCBD`’).

Usage

```
data(RCBD)
```

Details

A simulated data to model a Randomized Complete Block Design (‘`RCBD`’) of 5 factor levels, 4 blocks and 4 one replicate of each treatment per block.

`sorghum` *Completely Randomized Design (CRD)*

Description

The experiment consists of 16 treatments (cultivars) of sorghum conducted in a balanced squared lattice design and the yield by plot (kg/plot).

Usage

```
data(sorghum)
```

Format

An incomplete balanced block design with 4 blocks, 16 treatments, and 5 repetitions, that is, the yield of each treatment is measured 5 times. `sorghum` is a list with 4 elements. The first ‘`x`’ is a factor of lenght 80 with 16 levels describing the treatments. The second ‘`dm`’ is data.frame describing the design matrix. Its columns are ‘`x`’, ‘`bl`’ (blocks) and ‘`r`’ repetitions. The third ‘`y`’ is a numeric vector the yields. The fourth ‘`dfm`’ is a data frame with four columns. The first tree columns are the design matrix and the fourth is ‘`y`’.

Details

The experiment was conducted at EMBRAPA Milho e Sorgo (The Brazilian Agricultural Research Corporation, Corn and Sorghum section).

Source

Ramalho, M.A.P. and Ferreira and D.F. and Oliveira, A.C. (2000) *Experimentacao em Genetica e Melhoramento de Plantas*. Editora UFLA, Lavras, Brazil, page 167.

Examples

```
library(TukeyC)

data(sorghum)

av <- aov(y ~ r/bl + x,
          data=sorghum$dfm)

tk <- TukeyC(av,
             which='x',
             sig.level=0.05)

summary(tk)

plot(tk)
```

`SPE` *Split-plot Experiment (SPE)*

Description

A `list` to illustrate the resources of `TukeyC` package related to Split-plot Experiment (‘`SPE`’).

Usage

```
data(SPE)
```

Details

A simulated data to model a Split-plot Experiment (‘`SPE`’) with 3 plots, each one split 4 times and 6 repetitions per split.

`SPET` *Split-plot Experiment in Time (SPET)*

Description

The experiment consists of 8 treatments (7 leguminous cover crops and maize) in a Randomized Complete Block Design (‘`RCBD`’) and the yield by plot (kg/plot).

Usage

```
data(SPET)
```

Source

Gomes, F.P. (1990). Curso de Estatistica Experimental. 13 ed. Editora NOBEL, Piracicaba, Brazil, page 157.

`SSPE` *Split-split-plot Experiment (SSPE)*

Description

A `list` to illustrate the resources of `TukeyC` package related to Split-split-plot Experiment (‘`SSPE`’).

Usage

```
data(SSPE)
```

Details

A simulated data to model a Split-split-plot Experiment (‘`SSPE`’) with 3 plots, each one split 3 times, each split, split again 5 times and 4 repetitions per split-split.

`summary` *Summary Method for TukeyC and TukeyC.nest Objects*

Description

Returns (and prints) a summary list for `TukeyC` objects.

Usage

```
  ## S3 method for class 'TukeyC'
summary(object,
        complete=TRUE, ...)
```

Arguments

`object`	A given object of the class `TukeyC`.
`complete`	A logical value indicating if the summary is complete (mean difference and p-value) or only the groups.
`...`	Potential further arguments (required by generic).

Value

A `matrix` if `complete` is `TRUE`, or a `data.frame` if `complete` is `FALSE`.

Author(s)

Faria, J. C. (<`joseclaudio.faria@gmail.com`>)
Jelihovschi, E. G. (<`eniojelihovs@gmail.com`>)
Allaman, I. B. (<`ivanalaman@gmail.com`>)

References

Chambers, J.M. and Hastie, T.J. (1992) *Statistical Models in S*. Wadsworth and Brooks/Cole.

See Also

`TukeyC`

Examples

```
##
## Examples: Completely Randomized Design (CRD)
## More details: demo(package='TukeyC')
##

## The parameters can be: formula, aov, lm and aovlist
data(CRD2)

## From: formula
tk1 <- with(CRD2,
            TukeyC(y ~ x,
                   data=dfm,
                   which='x',
                   id.trim=5))
summary(tk1)

##
## Example: Randomized Complete Block Design (RCBD)
## More details: demo(package='TukeyC')
##

## The parameters can be: formula, aov, lm and aovlist
data(RCBD)

## From: formula
tk2 <- with(RCBD,
            TukeyC(y ~ blk + tra,
                   data=dfm,
                   which='tra'))
summary(tk2)

##
## Example: Latin Squares Design (LSD)
## More details: demo(package='TukeyC')
##

## The parameters can be: design matrix and the response variable,
## data.frame or aov

data(LSD)

## From: formula
tk3 <- with(LSD,
```

```
            TukeyC(y ~ rows + cols + tra,
                   data=dfm,
                   which='tra'))
summary(tk3)

##
## Example: Factorial Experiment (FE)
## More details: demo(package='TukeyC')
##

## The parameters can be: design matrix and the response variable,
## data.frame or aov

data(FE)
## From: design matrix (dm) and response variable (y)
## Main factor: N
tk4 <- with(FE,
            TukeyC(y ~ blk + N*P*K,
                   data=dfm,
                   which='N'))
summary(tk4)

## Nested: p1/N
## Testing N inside of level one of P
ntk1 <- with(FE,
             TukeyC(y ~ blk + N*P*K,
                    data=dfm,
                    which='P:N',
                    fl1=1))
summary(ntk1)

## Nested: k1/p1/N
## Testing N inside of level one of K and level one of P
ntk2 <- with(FE,
             TukeyC(y ~ blk + N*P*K,
                    data=dfm,
                    which='K:P:N',
                    fl1=1,
                    fl2=1))
summary(ntk2)

## Nested: k2/n2/P
ntk3 <- with(FE,
             TukeyC(y ~ blk + N*P*K,
                    data=dfm,
                    which='K:N:P',
                    fl1=2,
                    fl2=2))
summary(ntk3)

## Nested: p1/n1/K
ntk4 <- with(FE,
             TukeyC(y ~ blk + N*P*K,
```

```
                     data=dfm,
                     which='P:N:K',
                     fl1=1,
                     fl2=1))
summary(ntk4)

##
## Example: Split-plot Experiment (SPE)
## More details: demo(package='TukeyC')
##

data(SPE)
## From: formula
## Main factor: P
## Specify the appropriate error term
tk1 <- with(SPE,
            TukeyC(y ~ blk + P*SP + Error(blk/P),
                   data=dfm,
                   which='P',
                   error='blk:P'))
summary(tk1)

## Nested: p1/SP
tkn1 <- with(SPE,
             TukeyC(y ~ blk + P*SP + Error(blk/P),
                    data=dfm,
                    which='P:SP',
                    fl1=1 ))
summary(tkn1)

## From: formula
## Main factor: P
## Specify the appropriate error term
data(SSPE)

tk1 <- with(SSPE,
            TukeyC(y ~ blk + P*SP*SSP + Error(blk/P/SP),
                   data=dfm,
                   which='P',
                   error='blk:P'))
summary(tk1)

## Main factor: SP
## Specify the appropriate error term
tk2 <- with(SSPE,
            TukeyC(y ~ blk + P*SP*SSP + Error(blk/P/SP),
                   data=dfm,
                   which='SP',
                   error='blk:P:SP'))
summary(tk2)

## Main factor: SSP
tk3 <- with(SSPE,
```

```
            TukeyC(y ~ blk + P*SP*SSP + Error(blk/P/SP),
                   data=dfm,
                   which='SSP'))
summary(tk3)

## Nested: p1/SSP
tkn1 <- with(SSPE,
             TukeyC(y ~ blk + P*SP*SSP + Error(blk/P/SP),
                    data=dfm,
                    which='P:SSP',
                    fl1=1))
summary(tkn1)

## From: aovlist
av <- with(SSPE,
           aov(y ~  blk + P*SP*SSP + Error(blk/P/SP),
               data=dfm))
summary(av)

## Nested: P1/SP1/SSP
tkn2 <- TukeyC(av,
               which='P:SP:SSP',
               fl1=1,
               fl2=1)
summary(tkn2)

## Nested: P2/SP1/SSP
tkn3 <- TukeyC(av,
               which='P:SP:SSP',
               fl1=2,
               fl2=1)
summary(tkn3)

## Nested: SSP2/P1/SP - specify how to combine error terms
tkn4 <- TukeyC(av,
               which='SSP:P:SP',
               fl1=2,
               fl2=1,
               error='Within/blk:P/blk:P:SP')
summary(tkn4)
```

`Tukey test` *Conventional Tukey Test*

Description

This package performs what is known as the Tukey HSD test in the conventional way. It also uses an algorithm which divides the set of all means in groups and assigns letters to the different groups, allowing for overlapping. This is done for simple experimental designs and schemes. The most usual designs are: Completely Randomized Design (‘`CRD`’), Randomized Complete Block Design (‘`RCBD`’) and Latin Squares Design (‘`LSD`’). The most usual schemes are: Factorial Experiment (‘`FE`’), Split-Plot Experiment (‘`SPE`’) and Split-Split-Plot Experiment (‘`SSPE`’).

The package can be used for balanced and unbalanced experiments (when possible).

R has some functions (`TukeyHSD` provided by `stats`, `glht` provided by `multcomp`, `HSD.test` provided by `agricolae` and `cld` provided by `multcomp`) that also perform the Tukey test. The `TukeyHSD` returns intervals based on the range of the sample means rather than the individual differences. Those intervals are based on Studentized range statistics and are, in essence, confidence intervals. This approach has two advantages: p-values are shown, allowing the user to allow flexible inferential decisions and make it possible to plot the result of the test. However, it has one disadvantage, since the final result is more difficult to understand and summarize. Others (`glht`, `cld`) are also useful but difficult to manage. Additionally, many users of other statistical software are accustomed to letters that group the means of the factor tested, which makes it unattractive or difficult to adapt to common R workflows.

So, the main aim of this package is to make available in the R environment the conventional approach to the Tukey test with a set of flexible functions and S3 methods.

Author(s)

Faria, J. C. (<`joseclaudio.faria@gmail.com`>)
Jelihovschi, E. G. (<`eniojelihovs@gmail.com`>)
Allaman, I. B. (<`ivanalaman@gmail.com`>)

References

Miller, R.G. (1981) *Simultaneous Statistical Inference*. Springer.

Ramalho M.A.P, Ferreira D.F and Oliveira A.C. (2000) *Experimentacao em Genetica e Melhoramento de Plantas*. Editora UFLA.

Steel, R.G., Torrie, J.H and Dickey D.A. (1997) *Principles and procedures of statistics: a biometrical approach*. Third Edition.

Yandell, B.S. (1997) *Practical Data Analysis for Designed Experiments*. Chapman & Hall.

`TukeyC` *The TukeyC Test for Single Experiments*

Description

These are methods for objects of class `formula`, `lm`, `aov`, `aovlist` and `lmerMod` for single, factorial, split-plot and split-split-plot experiments.

Usage

```
TukeyC(x,...)

## S3 method for class 'formula'
TukeyC(formula,
       data           = NULL,
       which          = NULL,
       fl1            = NULL,
       fl2            = NULL,
       error          = NULL,
       sig.level      = .05,
       round          = 2,
       adjusted.pvalue = 'none',
       ...)

## S3 method for class 'lm'
TukeyC(x,
       which          = NULL,
       fl1            = NULL,
       fl2            = NULL,
       error          = NULL,
       sig.level      = .05,
       round          = 2,
       adjusted.pvalue = 'none',
       ...)

## S3 method for class 'aovlist'
TukeyC(x,
       which          = NULL,
       fl1            = NULL,
       fl2            = NULL,
       error          = NULL,
       sig.level      = .05,
       round          = 2,
       adjusted.pvalue = 'none',
       ...)

## S3 method for class 'lmerMod'
TukeyC(x,
       which          = NULL,
       fl1            = NULL,
       fl2            = NULL,
       error          = NULL,
       sig.level      = .05,
       round          = 2,
       adjusted.pvalue = 'none',
       ...)
```

Arguments

`x, formula`	A `formula`, `lm`, `aov`, `aovlist`, or `lmerMod` object. For the `formula` method, use `response ~ predictors`.
`data`	An object of class `data.frame` class. Use only objects of `formula` class.
`which`	The name of the treatment to be used in the comparison. The name must be inside quoting marks.
`fl1`	A vector of length 1 giving the level of the first factor in nesting order tested.
`fl2`	A vector of length 1 giving the level of the second factor in nesting order tested.
`error`	The error to be considered. For split-plot or split-split-plot designs, see Details.
`sig.level`	Level of Significance used in the TukeyC algorithm to create the groups of means. The default value is 0.05.
`round`	Integer indicating the number of decimal places.
`adjusted.pvalue`	Method for adjusting p values (see `p.adjust` for details). The possible values are: `"holm"`, `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"` and `"none"`. The default is `"none"`.
`...`	Potential further arguments (required by generic).

Details

The function `TukeyC` returns an object of class `TukeyC` containing the groups of means plus other necessary variables for summary and plot.

The generic functions `summary` and `plot` are used to obtain and print a summary and a plot of the results.

The error arguments may be used when the user wants a specific error other than the experimental error. In split-plot and split-split-plot designs, error terms may be combined with "/" in the sequence of the `which` argument. For example, an `aovlist` object, a possible combination would be `error = 'Within/blk:plot'` in a blocked split-plot design with `which = 'subplot:plot'` argument.

Value

The function `TukeyC` returns a list of the class `TukeyC` with the slots:

`Result`	A `data.frame` storing the result of Tukey test.
`Sig.level`	A scalar giving the level of significance of the test.
`Diff_Prob`	A `matrix` at the lower diagonal with p-values and upper diagonal with means differences.
`MSD`	A `matrix` with minimum significant differences by Tukey methodology. If balanced data, then all values are equal.

Author(s)

Faria, J. C. (`<joseclaudio.faria@gmail.com>`)
Jelihovschi, E. G. (`<eniojelihovs@gmail.com>`)
Allaman, I. B. (`<ivanalaman@gmail.com>`)

References

Miller, R.G. (1981) *Simultaneous Statistical Inference*. Springer.

Ramalho M.A.P, Ferreira D.F and Oliveira A.C. (2000) *Experimentacao em Genetica e Melhoramento de Plantas*. Editora UFLA.

Steel, R.G., Torrie, J.H and Dickey D.A. (1997) *Principles and procedures of statistics: a biometrical approach*. Third Edition.

Yandell, B.S. (1997) *Practical Data Analysis for Designed Experiments*. Chapman and Hall.

Examples

```
##
## Examples: Randomized Complete Block Design (RCBD)
## More details: demo(package='TukeyC')
##

## The parameters can be: formula, aov, lm, aovlist and lmerMod

data(RCBD)

## From: formula
tk1 <- with(RCBD,
            TukeyC(y ~ blk + tra,
                   data=dfm,
                   which='tra'))
summary(tk1)

## From: lmerMod
## This class is specific of the lme4 package.

  if(require(lme4)){
    lmer1 <- with(RCBD,
                  lmer(y ~ (1|blk) + tra,
                       data=dfm))

    tk2 <-  TukeyC(lmer1,
                   which='tra')
    summary(tk2)
  }

##
## Example: Latin Squares Design (LSD)
## More details: demo(package='TukeyC')
##

data(LSD)

## From: formula
tk3 <- with(LSD,
            TukeyC(y ~ rows + cols + tra,
                   data=dfm,
                   which='tra'))
```

```
summary(tk3)

## From: aov
av1 <- with(LSD,
            aov(y ~ rows + cols + tra,
                data=dfm))

tk4 <- TukeyC(av1,
              which='tra')
summary(tk4)

## From: lm
lm1 <- with(LSD,
            lm(y ~ rows + cols + tra,
               data=dfm))

tk5 <- TukeyC(lm1,
              which='tra')
summary(tk5)

##
## Example: Factorial Experiment (FE)
## More details: demo(package='TukeyC')
##

data(FE)
## From: formula
## Main factor: N
tk6 <- with(FE,
            TukeyC(y ~ blk + N*P*K,
                   data=dfm,
                   which='N'))
summary(tk6)

## Nested: p1/N
# From: formula
n_tk1 <- with(FE,
              TukeyC(y ~ blk + N*P*K,
                     data=dfm,
                     which='P:N',
                     fl1=1))
summary(n_tk1)

## Nested: p2/N
# From: lm
lm2 <- with(FE,
            lm(y ~ blk + N*P*K,
               dfm))

n_tk2 <- with(FE,
              TukeyC(lm2,
                     which='P:N',
                     fl1=2))
```

```
summary(n_tk2)

## Nested: n1/P
# From: aov
av2 <- with(FE,
            aov(y ~ blk + N*P*K,
                dfm))

n_tk3 <- with(FE,
              TukeyC(av2,
                     which='N:P',
                     fl1=1))
summary(n_tk3)

# From: lmerMod

  if(require(lme4)){
    lmer2 <- with(FE,
                  lmer(y ~ (1|blk) + N*P*K,
                       dfm))

    n_tk4 <- with(FE,
                  TukeyC(lmer2,
                         which='N:P',
                         fl1=1))
    summary(n_tk4)
  }


##
## Example: Split-plot Experiment (SPET)
## More details: demo(package='TukeyC')
##
data(SPET)

## From lm
lm3 <- with(SPET,
            lm(y ~ blk*tra + tra*year,
               dfm))

# crotjuncea/year
sp_tk1 <- TukeyC(lm3,
                 which='tra:year',
                 fl1=1)
summary(sp_tk1)

# year1/tra
# Specify the year error combined with the treatment error in the order of the which argument.
sp_tk2 <-  TukeyC(lm3,
                  which='year:tra',
                  error='Residuals/blk:tra',
                  fl1=1)
summary(sp_tk2)
```

```
# From: lmerMod
# Only tra

  if(require(lme4)){
    lmer3 <- with(SPET,
                  lmer(y ~ blk + (1|blk:tra) + tra*year,
                       dfm))

    # comparison only tra
    sp_tk3 <- TukeyC(lmer3,
                     which = 'tra',
                     error = 'blk:tra')
    summary(sp_tk3)

    # year1/tra
    sp_tk4 <- TukeyC(lmer3,
                     which='year:tra',
                     error='Residual/blk:tra',
                     fl1=1)
    summary(sp_tk4)
  }


## Example: Split-split-plot Experiment (SSPE)
## More details: demo(package='TukeyC')
##

data(SSPE)
## From: formula
## Main factor: P
## Specify the appropriate error term
ssp_tk1 <- with(SSPE,
                TukeyC(y ~ blk + P*SP*SSP + Error(blk/P/SP),
                       data=dfm,
                       which='P',
                       error='blk:P'))
summary(ssp_tk1)

## Main factor: SP
## Specify the appropriate error term
ssp_tk2 <- with(SSPE,
                TukeyC(y ~ blk + P*SP*SSP + Error(blk/P/SP),
                       data=dfm,
                       which='SP',
                       error='blk:P:SP'))
summary(ssp_tk2)

## Main factor: SSP
ssp_tk3 <- with(SSPE,
                TukeyC(y ~ blk + P*SP*SSP + Error(blk/P/SP),
                       data=dfm,
                       which='SSP'))
```

```
summary(ssp_tk3)

## From: aov
## Main factor: SSP
av3 <- with(SSPE,
            aov(y ~ blk + P*SP*SSP + Error(blk/P/SP),
                data=dfm))

ssp_tk4 <- TukeyC(av3,
                  which='SSP')
summary(ssp_tk4)

## Nested: p1/SP
## Specify the appropriate error term
ssp_tk5 <- TukeyC(av3,
                  which='P:SP',
                  error='blk:P:SP',
                  fl1=1)
summary(ssp_tk5)

## Nested: p1/SSP
ssp_tk6 <- TukeyC(av3,
                  which='P:SSP',
                  fl1=1)
summary(ssp_tk6)

## Nested: p1/sp1/SSP
## Testing SSP inside of level one of P and level one of SP
ssp_tk7 <- TukeyC(av3,
                  which='P:SP:SSP',
                  fl1=1,
                  fl2=1)
summary(ssp_tk7)

## Nested: p2/sp1/SSP
ssp_tk8 <- TukeyC(av3,
                  which='P:SP:SSP',
                  fl1=2,
                  fl2=1)
summary(ssp_tk8)

## Nested: sp1/P
## Specify the appropriate error term
ssp_tk9 <- TukeyC(av3,
                  which='SP:P',
                  error='blk:P:SP/blk:P',
                  fl1=1)

summary(ssp_tk9)

## Nested: ssp1/SP
ssp_tk10 <- TukeyC(av3,
                   which='SSP:SP',
```

```
                   error='Within/blk:P:SP',
                   fl1=1)
summary(ssp_tk10)

## Nested: ssp1/sp1/P
## Specify the appropriate error term
ssp_tk11 <- TukeyC(av3,
                   which='SSP:SP:P',
                   error='Within/blk:P:SP/blk:P',
                   fl1=1,
                   fl2=1)
summary(ssp_tk11)

## UNBALANCED DATA
## Means are adjusted using least-squares means.
## From: formula
data(CRD2)

uCRD2 <- CRD2$dfm
uCRD2[c(3, 5, 10, 44, 45), 3] <- NA

utk1 <-  TukeyC(y ~ x,
                data=uCRD2,
                which='x')
summary(utk1)

## From: lm
ulm1 <- lm(y ~ x,
           data=uCRD2)

utk2 <- TukeyC(ulm1,
               which='x')
summary(utk2)


## Factorial Experiments
## Nested: p1/N
# From: lm

uFE <- FE$dfm
uFE[c(3, 6, 7, 20, 31, 32), 5] <- NA

ulm2 <- lm(y ~ blk + N*P*K,
           uFE)

## Nested: p1/N
utk3 <- TukeyC(ulm2,
               data=uFE,
               which='P:N',
               fl1=1)
summary(utk3)

## Nested: p2/n2/K
```

```
utk4 <- TukeyC(ulm2,
               data=uFE,
               which='P:N:K',
               fl1=2,
               fl2=2)
summary(utk4)
```

`xtable` *Create a Table for Export*

Description

This function is re-exported from the **xtable** package so that `xtable()` is available after `library(TukeyC)` without requiring a separate `library(xtable)` call.

For `TukeyC` objects the S3 method `xtable.TukeyC` is dispatched automatically. For full documentation of the generic see `help("xtable", package = "xtable")`.

See Also

`xtable.TukeyC`, `xtable`

`xtable.TukeyC` *xtable method for* `TukeyC` *objects.*

Description

Convert a `TukeyC` object to an `xtable.TukeyC` object, which can then be printed as a LaTeX or HTML table. This function provides an additional method for the `xtable` function from the **xtable** package.

Usage

```
## S3 method for class 'TukeyC'
xtable(x, ...)

## S3 method for class 'xtable.TukeyC'
print(x, include.rownames = FALSE, ...)
```

Arguments

`x`	A given object of the class `TukeyC` (for `xtable`) or `xtable.TukeyC` (for `print`).
`include.rownames`	Logical; passed to `xtable::print.xtable`. Treatment names are already in the `Treatment` column, so the default is `FALSE`.
`...`	Further arguments (required by `xtable::xtable` or `xtable::print.xtable`).

Author(s)

Faria, J. C. (<`joseclaudio.faria@gmail.com`>)
Jelihovschi, E. G. (<`eniojelihovs@gmail.com`>)
Allaman, I. B. (<`ivanalaman@gmail.com`>)

See Also

`xtable`

Examples

```
data(RCBD)

lm1 <- with(RCBD,
            lm(y ~ blk + tra,
               data = dfm))

tk1 <- TukeyC(lm1,
              which = 'tra')

tb <- xtable(tk1)
## Not run:
  print(tb)

## End(Not run)
```

Index