Package ‘ScottKnott’

June 7, 2018

Type Package

Title The ScottKnott Clustering Algorithm

Version 1.2-7

Date 2018-06-07

Author Enio Jelihovschi, José Cláudio Faria and Ivan Bezerra Allaman

Maintainer Ivan Bezerra Allaman <ivanalaman@gmail.com>

Depends R (>= 2.6.0), stats, base

Description Division of an ANOVA experiment treatment means into homogeneous distinct groups using the clustering method of Scott & Knott.

License GPL (>= 2)

URL https://github.com/jcfaria/ScottKnott,
http://nbcgib.uesc.br/lec/software/pac-r/scottknott

Encoding latin1

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2018-06-07 17:43:41 UTC

R topics documented:

`ScottKnott-package` *The ScottKnott Clustering Algorithm*

Description

The Scott & Knott clustering algorithm is a very useful clustering algorithm widely used as a multiple comparison method in the Analysis of Variance context, as for example Gates and Bilbro (1978), Bony et al. (2001), Dilson et al. (2002) and Jyotsna et al. (2003).

It was developed by Scott, A.J. and Knott, M. (Scott and Knott, 1974). All methods used up to that date as, for example, the t-test, Tukey, Duncan, Newman-Keuls procedures, have overlapping problems. By overlapping we mean the possibility of one or more treatments to be classified in more than one group, in fact, as the number of treatments reach a number of twenty or more, the number of overlappings could increse as reaching 5 or greater what makes almost impossible to the experimenter to really distinguish the real groups to which the means should belong. The Scott & Knott method does not have this problem, what is often cited as a very good quality of this procedure.

The Scott & Knott method make use of a clever algorithm of cluster analysis, where, starting from the the whole group of observed mean effects, it divides, and keep dividing the sub-groups in such a way that the intersection of any two groups formed in that manner is empty.

Using their own words ‘we study the consequences of using a well-known method of cluster analysis to partition the sample treatment means in a balanced design and show how a corresponding likelihood ratio test gives a method of judging the significance of difference among groups abtained’.

Many studies, using the method of Monte Carlo, suggest that the Scott Knott method performs very well compared to other methods due to fact that it has high power and type I error rate almost always in accordance with the nominal levels. The **ScottKnott** package performs this algorithm starting either from `vectors`, `matrices` or `data.frames` joined as `default`, a `aov` or `aovlist` resulting object of previous analysis of variance. The results are given in the usual way as well as in graphical way using thermometers with diferent group colors.

In a few words, the test of Scott & Knott is a clustering algorithm used as an one of the alternatives where multiple comparizon procedures are applied with a very important and almost unique characteristic: it does not present overlapping in the results.

Author(s)

Enio Jelihovschi (<`eniojelihovs@gmail.com`>)
José Cláudio Faria (<`joseclaudio.faria@gmail.com`>)
Ivan Bezerra Allaman (<`ivanalaman@gmail.com`>)

References

Bony S., Pichon N., Ravel C., Durix A., Balfourier F., Guillaumin J.J. 2001. The Relationship be-tween Mycotoxin Synthesis and Isolate Morphology in Fungal Endophytes of Lolium perenne. *New Phytologist*, **1521**, 125-137.

Borges L.C., FERREIRA D.F. 2003. Poder e taxas de erro tipo I dos testes Scott-Knott, Tukey e Student-Newman-Keuls sob distribuicoes normal e nao normais dos residuos. Power and type I errors rate of Scott-Knott, Tukey and Student-Newman-Keuls tests under normal and no-normal distributions of the residues. *Rev. Mat. Estat.*, Sao Paulo, **211**: 67-83.

Calinski T., Corsten L.C.A. 1985. Clustering Means in ANOVA by Simultaneous Testing. *Biometrics*, **411**, 39-48.

Da Silva E.C, Ferreira D.F, Bearzoti E. 1999. Evaluation of power and type I error rates of Scott-Knotts test by the method of Monte Carlo. *Cienc. agrotec.*, Lavras, **23**, 687-696.

Dilson A.B, David S.D., Kazimierz J., William W.K. 2002. Half-sib progeny evaluation and selection of potatoes resistant to the US8 genotype of Phytophthora infestans from crosses between resistant and susceptible parents. *Euphytica*, **125**, 129-138.

Gates C.E., Bilbro J.D. 1978. Illustration of a Cluster Analysis Method for Mean Separation. *Agron J*, **70**, 462-465.

Wilkinson, G.N, Rogers, C.E. 1973. Journal of the Royal Statistical Society. Series C (Applied Statistics), Vol. 22, No. 3, pp. 392-399.

Jyotsna S., Zettler L.W., van Sambeek J.W., Ellersieck M.R., Starbuck C.J. 2003. Symbiotic Seed Germination and Mycorrhizae of Federally Threatened Platanthera PraeclaraOrchidaceae. *American Midland Naturalist*, **1491**, 104-120.

Ramalho M.A.P., Ferreira DF, Oliveira AC 2000. *Experimentação em Genética e Melhoramento de Plantas*. Editora UFLA.

Scott R.J., Knott M. 1974. A cluster analysis method for grouping mans in the analysis of variance. *Biometrics*, **30**, 507-512.

Examples

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

## The parameters can be: vectors, design matrix and the response variable,
## data.frame or aov
data(CRD2)

## From: design matrix (dm) and response variable (y)
sk1 <- with(CRD2,
            SK(x=dm,
               y=y,
               model='y ~ x',
               which='x'))
summary(sk1)
plot(sk1,
     col=rainbow(max(sk1$groups)),
     mm.lty=3,
     id.las=2,
     rl=FALSE,
     title='Factor levels')

## From: data.frame (dfm)
sk2 <- with(CRD2,
            SK(x=dfm,
               model='y ~ x',
               which='x',
               dispersion='se'))
summary(sk2)
plot(sk2,
     col=rainbow(max(sk2$groups)),
     id.las=2,
     rl=FALSE)

## From: aov
av <- with(CRD2,
           aov(y ~ x,
               data=dfm))
summary(av)

sk3 <- SK(x=av,
          which='x')
summary(sk3)
plot(sk3,
     col=rainbow(max(sk3$groups)),
     rl=FALSE,
     id.las=2,
     title=NULL)

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

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

data(RCBD)

## Design matrix (dm) and response variable (y)
sk1 <- with(RCBD,
            SK(x=dm,
               y=y,
               model='y ~ blk + tra',
               which='tra'))
summary(sk1)
plot(sk1)

## From: data.frame (dfm), which='tra'
sk2 <- with(RCBD,
            SK(x=dfm,
               model='y ~ blk + tra',
               which='tra'))
summary(sk2)
plot(sk2,
     mm.lty=3,
     title='Factor levels')

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

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

data(LSD)

## From: design matrix (dm) and response variable (y)
sk1 <- with(LSD,
            SK(x=dm,
               y=y,
               model='y ~ rows + cols + tra',
               which='tra'))
summary(sk1)
plot(sk1)

## From: data.frame
sk2 <- with(LSD,
            SK(x=dfm,
               model='y ~ rows + cols + tra',
               which='tra'))
summary(sk2)
plot(sk2,
     title='Factor levels')

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

sk3 <- SK(av,
          which='tra')
summary(sk3)
plot(sk3,
     title='Factor levels')

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

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

## Note: The factors are in uppercase and its levels in lowercase!

data(FE)

## From: design matrix (dm) and response variable (y)
## Main factor: N
sk1 <- with(FE,
            SK(x=dm,
               y=y,
               model='y ~ blk + N*P*K',
               which='N'))
summary(sk1)
plot(sk1,
     title='Main effect: N')

## Nested: p1/N
## Studing N inside of level one of P
nsk1 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + N*P*K',
                     which='P:N',
                     fl1=1))
summary(nsk1)
plot(nsk1,
     title='Effect: p1/N')

## Nested: k1/P
nsk2 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + N*P*K',
                     which='K:P',
                     fl1=1))
summary(nsk2)
plot(nsk2,
     title='Effect: k1/P')

## Nested: k2/p2/N
nsk3 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + N*P*K',
                     which='K:P:N',
                     fl1=2,
                     fl2=2))
summary(nsk3)
plot(nsk3,
     title='Effect: k2/p2/N')
```

```
## Nested: k1/n1/P
## Studing P inside of level one of K and level one of N
nsk4 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + N*P*K',
                     which='K:N:P',
                     fl1=1,
                     fl2=1))
summary(nsk4)
plot(nsk4,
     title='Effect: k1/n1/P')

## Nested: p1/n1/K
nsk5 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + N*P*K',
                     which='P:N:K',
                     fl1=1,
                     fl2=1))
summary(nsk5)
plot(nsk5, title='Effect: p1/n1/K')

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

## Note: The factors are in uppercase and its levels in lowercase!

data(SPE)

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

## From: design matrix (dm) and response variable (y)
## Main factor: P
sk1 <- with(SPE,
            SK(x=dm,
               y=y,
               model='y ~ blk + P*SP + Error(blk/P)',
               which='P',
               error='blk:P'))
summary(sk1)
plot(sk1)

## Main factor: SP
sk2 <- with(SPE,
            SK(x=dm,
               y=y,
               model='y ~ blk + P*SP + Error(blk/P)',
               which='SP',
               error='Within'))
summary(sk2)
plot(sk2,
     xlab='Groups',
     ylab='Main effect: sp',
     title='Main effect: SP')

## Nested: p1/SP
skn1 <- with(SPE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + P*SP + Error(blk/P)',
                     which='P:SP',
                     error='Within',
                     fl1=1 ))
summary(skn1)
plot(skn1,
     title='Effect: p1/SP')

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

## Note: The factors are in uppercase and its levels in lowercase!

data(SSPE)

## From: design matrix (dm) and response variable (y)
## Main factor: P
sk1 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
               which='P',
               error='blk:P'))
summary(sk1)

# Main factor: SP
sk2 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
               which='SP',
               error='blk:P:SP'))
summary(sk2)

# Main factor: SSP
sk3 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
               which='SSP',
               error='Within'))
summary(sk3)

## Nested: p1/SP
skn1 <- with(SSPE,
             SK.nest(dm,
                     y,
                     model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
                     which='P:SP',
                     error='blk:P:SP',
                     fl1=1))
summary(skn1)

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

## Nested: p/sp/SSP (at various levels of sp and p)
skn6 <- SK.nest(av,
                which='P:SP:SSP',
                error='Within',
                fl1=1,
                fl2=1)
summary(skn6)

skn7 <- SK.nest(av,
                which='P:SP:SSP',
                error='Within',
                fl1=2,
                fl2=1)
summary(skn7)
```

CRD1 *Completely Randomized Design (CRD)*

Description

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

Usage

```
data(CRD1)
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 **ScottKnott** package related to Completely Randomized Design ('`CRD`').

Usage

```
data(CRD2)
CRD2
```

Details

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

FE *Factorial Experiment (FE)*

Description

A `list` illustrating the resources of **ScottKnott** package related to Factorial Experiment ('`FE`').

Usage

```
data(FE)
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 **ScottKnott** package related to Latin Squares Design ('`LSD`').

Usage

```
data(LSD)
LSD
```

Details

A simulated data to model a Latin Squares Design ('`LSD`') with 5 factor levels 5 rows and 5 columns.

m.inf *Calculates the Means and Dispersion*

Description

Calculates the means and dispersion for one, two and three factor(s) by `model` and `model.frame` functions.

Usage

```
m.inf.1a(x,
         which,
         dispersion=c('mm', 's', 'se'))
m.inf.1b(x,
         which,
         dispersion=c('mm', 's', 'se'))
m.inf.2a(x,
         which1,
         which2,
         dispersion=c('mm', 's', 'se'))
m.inf.2b(x,
         which1,
         which2,
         dispersion=c('mm', 's', 'se'))
m.inf.3a(x,
         which1,
         which2,
         which3,
         dispersion=c('mm', 's', 'se'))
```

```
m.inf.3b(x,
         which1,
         which2,
         which3,
         dispersion=c('mm', 's', 'se'))
```

Arguments

x A SK object.

which The name of the treatment to be used in the comparison. For all the value is determined internally by the package.

which1 The name of the treatment to be used in the comparison. For all the value is determined internally by the package.

which2 The name of the treatment to be used in the comparison. For all the value is determined internally by the package.

which3 The name of the treatment to be used in the comparison. For all the value is determined internally by the package.

dispersion The dispersion to be considered to the means.
The possible vaues are: ''mm'' = 'minimum and maximum', ''s'' = 'standart deviation', ''se'' = 'standart deviation of the mean'.

Note

This function is mainly for internal use in the **ScottKnott** package.

Author(s)

Enio Jelihovschi (<eniojelihovs@gmail.com>)
José Cláudio Faria (<joseclaudio.faria@gmail.com>)
Ivan Bezerra Allaman (<ivanalaman@gmail.com>)

MaxValue *Algorithm for Pre-order Path in Binary Decision Tree*

Description

Builds groups of means, according to the method of Scott & Knott.

Usage

```
MaxValue(g,
         means,
         mMSE,
         dfr,
         sig.level,
         k,
         group,
         ngroup,
         markg,
         g1=g,
         sqsum=rep(0, g1))
```

Arguments

`g` A vector of length 1 giving the upper limit of the possible groups.

`means` A vector of the group of treatments means in decresing order.

`mMSE` A vector of length 1 giving the MSE divided by the number of replications.

`dfr` A vector of length 1 giving the degrees of freedom of MSE.

`sig.level` A vector of length 1 giving the level of significance of the test.

`k` A vector of length 1 giving the lower limit of the possible groups.

`group` A vector of the same length as '`means`' marking the groups generated.

`ngroup` A vector of length 1 giving the number of groups generated.

`markg` A vector of the same length as '`means`' marking the upper limit of the last group generated before the process goes on recursively.

`g1` A vector of length 1 which keeps, during the whole process, the value of the initial '`g`'.

`sqsum` A vector of length 1 giving the sum of the square between groups.

Details

The function MaxValue builds groups of means, according to the method of Scott & Knott.

Basically it is an algorithm for pre-order path in binary decision tree.

Every node of this tree, represents a different group of means and, when the algorithm reaches this node it takes the decision to either split the group in two, or form a group of means.

If the decision is to divide then this node generates two children and the algorithm follows for the node on the left, if, on the other hand, the decision is to form a group, then it returns to the parent node of that node and follows to the right node.

In this way it follows until the last group is formed, the one containing the highest (or the least) mean. In case that the highest (or the least) mean becomes itself a group of one element, the algorithm continues to the former group. In the end, each node without children represents a group of means.

Value

An '`vector`' with the groups of means.

Note

This function is mainly for internal use in the **ScottKnott** package.

Author(s)

Enio Jelihovschi (<eniojelihovs@gmail.com>)
José Cláudio Faria (<joseclaudio.faria@gmail.com>)
Ivan Bezerra Allaman (<ivanalaman@gmail.com>)

References

Ramalho M.A.P., Ferreira D.F., Oliveira A.C. 2000. *Experimentação em Genética e Melhoramento de Plantas*. Editora UFLA.

Scott R.J., Knott M. 1974. A cluster analysis method for grouping mans in the analysis of variance. *Biometrics*, **30**, 507-512.

See Also

SK

Examples

```
##
## Examples: Completely Randomized Design (CRD)
##

data(CRD1)

av      <- with(CRD1,
                aov(y ~ x,
                    data=dfm))               # Doing an ANOVA
mm      <- model.tables(av,
                        "means")             # summary tables for model fits
tabs    <- mm$tables[-1]                     # all model means
which   <- names(av$model)[2]
tabs    <- tabs[which]                       # specified group means
nn      <- mm$n[names(tabs)]                 # repetions number of specified groups
MSE     <- sum(resid(av)^2)/av$df.residual
tab     <- tabs[[which]]                     # tab=means
means   <- as.vector(tab)
mnumber <- length(means)                     # number of means
nms     <- names(tab)
r       <- nn[[which]]                       # groups and its number of replicates
ord     <- order(means,
                 decreasing=TRUE)
mMSE    <- MSE/r
dfr     <- av$df.residual                    # residual degrees of freedom
means   <- means[ord]                        # decreasing ordered means
g       <- mnumber

## Grouping the means: sig.level=.05
groups  <- MaxValue(g,
                    means,
                    mMSE,
                    dfr,
                    sig.level=.05,
                    1,
                    rep(0, g),
                    0,
                    rep(0, g))

## The result: sig.level=.05
groups


## Grouping the means: sig.level=.10
groups  <- MaxValue(g,
                    means,
                    mMSE,
                    dfr,
                    sig.level=.10,
                    1,
                    rep(0, g),
                    0,
                    rep(0, g))

## The result: sig.level=.10
groups
```

plot.SK *Plot SK and SK.nest Objects*

Description

S3 method to plot SK and SK.nest objects.

Usage

```
## S3 method for class 'SK'
plot(x,
     pch=19,
     col=NULL,
     xlab=NULL,
     ylab=NULL,
     xlim=NULL,
     ylim=NULL,
     id.lab=NULL,
     id.las=1,
     id.col=TRUE,
     rl=TRUE,
     rl.lty=3,
     rl.col='gray',
     mm=TRUE,
     mm.lty=1,
     title="Means grouped by color(s)", ...)
```

Arguments

`x` A `SK` object.

`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.

`id.lab` Factor level names at x-axis.

`id.las` Factor level names written either horizontally or vertically.

`id.col` A logical value. If `TRUE` (the default), the `col` parameter will be used for the ‘`x`’ axis.

`rl` Horizontal line connecting the circle to the y-axis.

`rl.lty` Line type of ‘`rl`’.

`rl.col` Line color of ‘`rl`’.

`mm` Vertical line through the circle (mean value) linking the minimum to the maximum of the factor level values corresponding to that mean value.

`mm.lty` Line type of ‘`mm`’.

`title` A title for the plot.

`...` Optional plotting parameters.

Details

The `plot.SK` function is a S3 method to plot `SK` and `SK.nest` objetcs. It generates a serie of points (the means) and a vertical line showing the minimum e maximum of the values corresponding to each group mean. The groups are diferentiated by colors.

Author(s)

Enio Jelihovschi (<`eniojelihovs@gmail.com`>)
José Cláudio Faria (<`joseclaudio.faria@gmail.com`>)
Ivan Bezerra Allaman (<`ivanalaman@gmail.com`>)

References

Murrell, P. 2005. R Graphics. Chapman & Hall/CRC Press.

See Also

`plot`

Examples

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

library(ScottKnott)
data(CRD2)

## From: vectors x and y
sk1 <- with(CRD2,
            SK(x=x,
               y=y,
               model='y ~ x',
               which='x'))

plot(sk1,
     id.las=2,
     rl=FALSE,
     title='Factor levels')

## From: design matrix (dm) and response variable (y)
sk2 <- with(CRD2,
            SK(x=dm,
               y=y,
               model='y ~ x',
               which='x'))
plot(sk2,
     col=rainbow(max(sk2$groups)),
     mm.lty=3,
     id.las=2,
     rl=FALSE,
     title='Factor levels')

## From: data.frame (dfm)
sk3 <- with(CRD2,
            SK(x=dfm,
               model='y ~ x',
               which='x'))

plot(sk3,
     col=rainbow(max(sk3$groups)),
     id.las=2,
     id.col=FALSE,
     rl=FALSE)

## From: aov
av <- with(CRD2,
           aov(y ~ x,
               data=dfm))
summary(av)
```

```
sk4 <- with(CRD2,
            SK(x=av,
               which='x'))

plot(sk4,
     col=rainbow(max(sk4$groups)),
     rl=FALSE,
     id.las=2,
     id.col=FALSE,
     title=NULL)
```

RCBD *Randomized Complete Block Design (RCBD)*

Description

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

Usage

```
data(RCBD)
RCBD
```

Details

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

SK *The ScottKnott Clustering Algoritm for Single Experiments*

Description

These are methods for objects of class `vector`, `matrix` or `data.frame` joined as default, `aov` and `aovlist` for single experiments.

Usage

```
  ## Default S3 method:
SK(x,
   y=NULL,
   model,
   which,
   id.trim=3,
   error,
   sig.level=.05,
   dispersion=c('mm', 's', 'se'), ...)
  ## S3 method for class 'aov'
SK(x,
   which=NULL,
   id.trim=3,
   sig.level=.05,
   dispersion=c('mm', 's', 'se'), ...)
  ## S3 method for class 'aovlist'
SK(x,
   which,
   id.trim=3,
   error,
   sig.level=.05,
   dispersion=c('mm', 's', 'se'), ...)
```

Arguments

`x` A design matrix, `data.frame` or an `aov` object.

`y` A vector of response variable. It is necessary to inform this parameter only if '`x`' represent the design matrix.

`which` The name of the treatment to be used in the comparison. The name must be inside quoting marks.

`model` If '`x`' is a `data.frame` object, the model to be used in the aov must be specified.

`id.trim` The number of character to trim the id label.

`error` The error to be considered.

`sig.level` Level of Significance used in the `SK` algorithm to create the groups of means. The default value is 0.05.

`dispersion` The dispersion to be considered to the means.
The possible vaues are: ``'mm''` = '`minimum and maximum`', ``'s''` = '`standart deviation`', ``'se''` = '`standart deviation of the mean`'.

`...` Potential further arguments (require by generic).

Details

The function `SK` returns an object of class `SK` respectivally 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.

Value

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

`av` A `list` storing the result of `aov`.

`groups` A `vector` of length equal the number of factor levels marking the groups generated.

`nms` A `vector` of the labels of the factor levels.

`ord` A `vector` which keeps the position of the means of the factor levels in decreasing order.

`m.inf` A `matrix` which keeps the means and the dispersion of the factor levels in decreasing order.

`sig.level` A `vector` of length 1 giving the level of significance of the test.

Author(s)

Enio Jelihovschi (<`eniojelihovs@gmail.com`>)
José Cláudio Faria (<`joseclaudio.faria@gmail.com`>)
Ivan Bezerra Allaman (<`ivanalaman@gmail.com`>)

References

Ramalho M.A.P., Ferreira D.F., Oliveira A.C. 2000. *Experimentação em Genética e Melhoramento de Plantas*. Editora UFLA.

Scott R.J., Knott M. 1974. A cluster analysis method for grouping mans in the analysis of variance. *Biometrics*, **30**, 507-512.

Examples

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

## The parameters can be: vectors, design matrix and the response variable,
## data.frame or aov
data(CRD2)

## From: design matrix (dm) and response variable (y)
sk1 <- with(CRD2,
            SK(x=dm,
               y=y,
               model='y ~ x',
               which='x'))
summary(sk1)
plot(sk1,
     col=rainbow(max(sk1$groups)),
     mm.lty=3,
     id.las=2,
     rl=FALSE,
     title='factor levels')

## From: data.frame (dfm)
sk2 <- with(CRD2,
            SK(x=dfm,
               model='y ~ x',
              which='x',
              dispersion='s'))
summary(sk2)
plot(sk2,
     col=rainbow(max(sk2$groups)),
     id.las=2,
     rl=FALSE)

## From: aov
av <- with(CRD2,
           aov(y ~ x,
               data=dfm))
summary(av)

sk3 <- with(CRD2,
            SK(x=av,
               which='x',
               dispersion='se'))
summary(sk3)
plot(sk3,
     col=rainbow(max(sk3$groups)),
     rl=FALSE,
     id.las=2,
     title=NULL)

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

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

data(RCBD)

## Design matrix (dm) and response variable (y)
sk1 <- with(RCBD,
            SK(x=dm,
               y=y,
               model='y ~ blk + tra',
               which='tra'))
summary(sk1)
plot(sk1)

## From: data.frame (dfm), which='tra'
sk2 <- with(RCBD,
            SK(x=dfm,
               model='y ~ blk + tra',
               which='tra'))
summary(sk2)
plot(sk2,
     mm.lty=3,
     title='Factor levels')
```

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

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

data(LSD)

## From: design matrix (dm) and response variable (y)
sk1 <- with(LSD,
            SK(x=dm,
               y=y,
               model='y ~ rows + cols + tra',
               which='tra'))
summary(sk1)
plot(sk1)

## From: data.frame
sk2 <- with(LSD,
            SK(x=dfm,
               model='y ~ rows + cols + tra',
               which='tra'))
summary(sk2)
plot(sk2,
     title='Factor levels')

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

sk3 <- SK(av,
          which='tra')
summary(sk3)
plot(sk3,
     title='Factor levels')

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

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

## Note: The factors are in uppercase and its levels in lowercase!

data(FE)
## From: design matrix (dm) and response variable (y)
## Main factor: N
sk1 <- with(FE,
            SK(x=dm,
               y=y,
               model='y ~ blk + N*P*K',
               which='N'))
summary(sk1)
plot(sk1,
     title='Main effect: N')

## Nested: p1/N
nsk1 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + N*P*K',
                     which='P:N',
                     fl1=1))
summary(nsk1)
plot(nsk1,
     title='Effect: p1/N')
```

SK.nest *The ScottKnott Clustering Algoritm for Factorial, Split-plot and Split-SPlit plot Experiments*

Description

These are methods for objects of class `vector`, `matrix` or `data.frame` joined as default, `aov` and `aovlist` for factorial, split-plot and split-split-plot experiments.

Usage

```
  ## Default S3 method:
SK.nest(x,
        y=NULL,
        model,
        which,
        id.trim=3,
        error,
        fl1,
        fl2=0,
        sig.level=.05,
        dispersion=c('mm', 's', 'se'), ...)
  ## S3 method for class 'aov'
SK.nest(x,
        which,
        id.trim=3,
        fl1,
        fl2=0,
        sig.level=.05,
        dispersion=c('mm', 's', 'se'), ...)
  ## S3 method for class 'aovlist'
SK.nest(x,
        which,
        id.trim=3,
        error,
        fl1,
        fl2=0,
        sig.level=.05,
        dispersion=c('mm', 's', 'se'), ...)
```

Arguments

`x` A design matrix, `data.frame` or an `aov` object.

`y` A vector of response variable. It is necessary to inform this parameter only if `x` represent the design matrix.

`which` The name of the treatment to be used in the comparison. The name must be inside quoting marks.

`model` If `x` is a `data.frame` object, the model to be used in the aov must be specified.

`id.trim` The number of character to trim the id label.

`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, only in case of split-plots experiments.

`sig.level` Level of Significance used in the SK algorithm to create the groups of means. The default value is 0.05.

`dispersion` The dispersion to be considered to the means.
The possible vaues are: ‘`'mm'`’ = ‘`minimum and maximum`’, ‘`'s'`’ = ‘`standart deviation`’, ‘`'se'`’ = ‘`standart deviation of the mean`’.

`...` Potential further arguments (require by generic).

Details

The function `SK.nest` returns an object of class `SK.nest` 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.

Value

The function `SK.nest` returns a list of the class `SK.nest` with the slots:

`av` A `list` storing the result of `aov`.

`groups` A vector of length equal the number of factor levels marking the groups generated.

`nms` A vector of the labels of the factor levels.

ord A vector which keeps the position of the means of the factor levels in decreasing order.

m.inf A matrix which keeps the means and the dispersion of the factor levels in decreasing order.

sig.level A vector of length 1 giving the level of significance of the test.

r A vector of length 1 giving the number of replicates.

which The name of the factor whose levels were tested.

tab An array keeping the names of the factors and factor levels and also the mean value of the repetitions for every combination of factor levels.

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.

Author(s)

Enio Jelihovschi (<eniojelihovs@gmail.com>)
José Cláudio Faria (<joseclaudio.faria@gmail.com>)
Ivan Bezerra Allaman (<ivanalaman@gmail.com>)

References

Ramalho M.A.P., Ferreira D.F., Oliveira A.C. 2000. *Experimentação em Genética e Melhoramento de Plantas*. Editora UFLA.

Scott R.J., Knott M. 1974. A cluster analysis method for grouping mans in the analysis of variance. *Biometrics*, **30**, 507-512.

Examples

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

## Note: The factors are in uppercase and its levels in lowercase!

data(SSPE)
## From: design matrix (dm) and response variable (y)
## Main factor: P
sk1 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
               which='P',
               error='blk:P'))
summary(sk1)

# Main factor: SP
sk2 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
               which='SP',
               error='blk:P:SP',
               dispersion='s'))
summary(sk2)

# Main factor: SSP
sk3 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
               which='SSP',
               error='Within',
               dispersion='se'))
summary(sk3)

## Nested: p1/SP
skn1 <- with(SSPE,
             SK.nest(dm,
                     y,
                     model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
                     which='P:SP',
                     error='blk:P:SP',
                     fl1=1))
summary(skn1)

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

## Nested: p/sp/SSP
## Studing SSP inside of level one of P and level one of SP
skn2 <- SK.nest(av,
                which='P:SP:SSP',
                error='Within',
                fl1=1,
                fl2=1)
summary(skn2)

## Studing SSP inside of level one of P and level two of SP
skn3 <- SK.nest(av,
                which='P:SP:SSP',
                error='Within',
                fl1=1,
                fl2=2)
summary(skn3)
```

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)
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 '`tr`' 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, D.F. and Oliveira, A.C. (2000). Experimentação em Genética e Melhoramento de Plantas. Editora UFLA, Lavras, Brazil, page 167.

Examples

```
library(ScottKnott)

data(sorghum)

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

sk <- SK(av,
         which='x',
         sig.level=0.05)

summary(sk)

plot(sk)
```

SPE *Split-plot Experiment (SPE)*

Description

A `list` to ilustrate the resources of **ScottKnott** package related to Split-plot Experiment (‘`SPE`’).

Usage

```
data(SPE)
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)
SPET
```

Source

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

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

Description

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

Usage

```
data(SSPE)
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 SK and SK.nest Objects*

Description

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

Usage

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

Arguments

`object`	A given object of the class `SK` or `SK.nest`.
`...`	Potential further arguments (require by generic).

Author(s)

Enio Jelihovschi (<eniojelihovs@gmail.com>)
José Cláudio Faria (<joseclaudio.faria@gmail.com>)
Ivan Bezerra Allaman (<ivanalaman@gmail.com>)

References

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

See Also

ScottKnott

Examples

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

## The parameters can be: vectors, design matrix and the response variable,
## data.frame or aov
data(CRD2)

## From: design matrix (dm) and response variable (y)
sk1 <- with(CRD2,
            SK(x=dm,
               y=y,
               model='y ~ x',
               which='x'))
summary(sk1)

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

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

data(RCBD)

## Design matrix (dm) and response variable (y)
sk1 <- with(RCBD,
            SK(x=dm,
               y=y,
               model='y ~ blk + tra',
               which='tra'))
summary(sk1)

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

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

```
data(LSD)

## From: design matrix (dm) and response variable (y)
sk1 <- with(LSD,
            SK(x=dm,
               y=y,
               model='y ~ rows + cols + tra',
               which='tra'))
summary(sk1)

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

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

## Note: The factors are in uppercase and its levels in lowercase!

data(FE)

## From: design matrix (dm) and response variable (y)
## Main factor: N
sk1 <- with(FE,
            SK(x=dm,
               y=y,
               model='y ~ blk + N*P*K',
               which='N'))
summary(sk1)

## Nested: p1/N
nsk1 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + N*P*K',
                     which='P:N',
                     fl1=1))
summary(nsk1)

## Nested: k2/p2/N
nsk2 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + N*P*K',
                     which='K:P:N',
                     fl1=2,
                     fl2=2))
summary(nsk2)

## Nested: k1/n1/P
nsk3 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + P*N*K',
                     which='K:N:P',
                     fl1=1,
                     fl2=1))
summary(nsk3)

## Nested: p1/n1/K
nsk4 <- with(FE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + K*N*P',
                     which='P:N:K',
                     fl1=1,
                     fl2=1))
summary(nsk4)

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

## Note: The factors are in uppercase and its levels in lowercase!

data(SPE)

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

## From: design matrix (dm) and response variable (y)
## Main factor: P
sk1 <- with(SPE,
            SK(x=dm,
               y=y,
               model='y ~ blk + SP*P + Error(blk/P)',
               which='P',
               error='blk:P'))
summary(sk1)

## Nested: p1/SP
skn1 <- with(SPE,
             SK.nest(x=dm,
                     y=y,
                     model='y ~ blk + SP*P + Error(blk/P)',
                     which='P:SP',
                     error='Within',
                     fl1=1))
summary(skn1)

data(SSPE)

## From: design matrix (dm) and response variable (y)
## Main factor: P
```

```
sk1 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
               which='P',
               error='blk:P'))
summary(sk1)

# Main factor: SP
sk2 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
               which='SP',
               error='blk:P:SP'))
summary(sk2)

# Main factor: SSP
sk3 <- with(SSPE,
            SK(dm,
               y,
               model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
              which='SSP',
              error='Within'))
summary(sk3)

## Nested: p1/sp
skn1 <- with(SSPE,
             SK.nest(dm,
                     y,
                     model='y ~ blk + P*SP*SSP + Error(blk/P/SP)',
                     which='P:SP',
                     error='blk:P:SP',
                     fl1=1))
summary(skn1)

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

## Nested: p/sp/SSP (at various levels of SP and P)
skn2 <- SK.nest(av,
                which='P:SP:SSP',
                error='Within',
                fl1=1,
                fl2=1)
summary(skn2)

skn3 <- SK.nest(av,
                which='P:SP:SSP',
                error='Within',
                    fl1=1,
                    fl2=2)
summary(skn3)
```

Index