Package ‘gclus’

October 13, 2022

Version 1.3.2

Author Catherine Hurley

Date 2019-01-07

Maintainer Catherine Hurley <catherine.hurley@mu.ie>

Title Clustering Graphics

Description Orders panels in scatterplot matrices and parallel coordinate displays by some merit index. Package contains various indices of merit, ordering functions, and enhanced versions of pairs and parcoord which color panels according to their merit level.

Depends R (>= 2.10), cluster

License GPL (>= 2)

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2019-01-07 20:00:09 UTC

R topics documented:

`ac` *Clustering coefficients from package cluster.*

Description

Computes clustering coefficients from `cluster`, where `x` and `y` give the object coordinates.

Usage

```
ac(x, y, ...)
sil(x, y, groups, ...)
```

Arguments

`x`	is a numeric vector.
`y`	is a numeric vector.
`groups`	is a vector of group memberships, used by `sil` only.
`...`	are passed to `agnes` in `ac` and to `dist` in `sil`.

Details

`ac` - Computes clustering coefficient from `agnes{cluster}`.

`sil` - Computes the silhouette coefficient from from package `cluster`.

Value

The clustering coefficient is returned.

Author(s)

Catherine B. Hurley

References

Kaufman, L. and Rousseeuw, P.J. (1990). Finding Groups in Data: An Introduction to Cluster Analysis . Wiley, New York.

See Also

agnes, silhouette, dist.

Examples

```
x <- runif(20)
y <- runif(20)
g <- rep(c(”a”,”b”),10)

ac(x,y)
sil(x,y,g)
```

bank *Swiss bank notes data*

Description

Data from "Multivariate Statistics A practical approach", by Bernhard Flury and Hans Riedwyl, Chapman and Hall, 1988, Tables 1.1 and 1.2 pp. 5-8. Six measurements made on 100 genuine Swiss banknotes and 100 counterfeit ones.

Usage

```
data(bank)
```

Format

This data frame contains the following columns:

Status: 0 = genuine, 1 = counterfeit

Length: Length of bill, mm

Left: Width of left edge, mm

Right: Width of right edge, mm

Bottom: Bottom margin width, mm

Top: Top margin width, mm

Diagonal: Length of image diagonal, mm

Source

Flury, B. and Riedwyl, H. (1988), *Multivariate Statistics A Practical Approach*, London: Chapman and Hall.

body *Exploring Relationships in Body Dimensions*

Description

This dataset contains 21 body dimension measurements as well as age, weight, height, and gender on 507 individuals. The 247 men and 260 women were primarily individuals in their twenties and thirties, with a scattering of older men and women, all exercising several hours a week.

Measurements were initially taken by Grete Heinz and Louis J. Peterson - at San Jose State University and at the U.S. Naval Postgraduate School in Monterey, California. Later, measurements were taken at dozens of California health and fitness clubs by technicians under the supervision of one of these authors.

Usage

```
data(body)
```

Format

This data frame contains the following columns:

Biacrom: Biacromial diameter (cm)

Biiliac: Biiliac diameter, or "pelvic breadth" (cm)

Bitro: Bitrochanteric diameter (cm)

ChestDp: Chest depth between spine and sternum at nipple level, mid-expiration (cm)

ChestD: Chest diameter at nipple level, mid-expiration (cm)

ElbowD: Elbow diameter, sum of two elbows (cm)

WristD: Wrist diameter, sum of two wrists (cm)

KneeD: Knee diameter, sum of two knees (cm)

AnkleD: Ankle diameter, sum of two ankles (cm)

ShoulderG: Shoulder girth over deltoid muscles (cm)

ChestG: Chest girth, nipple line in males and just above breast tissue in females, mid-expiration (cm)

WaistG: Waist girth, narrowest part of torso below the rib cage, average of contracted and relaxed position (cm)

AbdG: Navel (or "Abdominal") girth at umbilicus and iliac crest, iliac crest as a landmark (cm)

HipG: Hip girth at level of bitrochanteric diameter (cm)

ThighG: Thigh girth below gluteal fold, average of right and left girths (cm)

BicepG: Bicep girth, flexed, average of right and left girths (cm)

ForearmG: Forearm girth, extended, palm up, average of right and left girths (cm)

KneeG: Knee girth over patella, slightly flexed position, average of right and left girths (cm)

CalfG: Calf maximum girth, average of right and left girths (cm)

AnkleG: Ankle minimum girth, average of right and left girths (cm)

WristG: Wrist minimum girth, average of right and left girths (cm)

Age: in years

Weight: in kg

Height: in cm

Gender: 1 - male, 0 - female

Source

Heinz, G., Peterson, L.J., Johnson, R.W. and Kerk, C.J. (2003), "Exploring Relationships in Body Dimensions", *Journal of Statistics Education* , 11.

References

The data file is taken from `http://jse.amstat.org/datasets/body.dat.txt` This information file is based on `http://jse.amstat.org/datasets/body.txt`

`colpairs` *Applies a function to all pairs of columns*

Description

Given an nxp matrix `m` and a function `f`, returns the pxp matrix got by applying `f` to all pairs of columns of `m` .

Usage

```
colpairs(m, f, diag = 0, na.omit = FALSE, ...)
```

Arguments

`m`	a matrix
`f`	a function of two vectors, which returns a single result.
`diag`	if supplied, this value is placed on the diagonal of the result.
`na.omit`	If `TRUE`, rows with missing values are omitted for each pair of columns.
`...`	argments are passed to `f`.

Value

a matrix matrix got by applying `f` to all pairs of columns of `m` .

Author(s)

Catherine B. Hurley

See Also

gave, partition.crit, order.single,order.endlink

Examples

```
data(state)
state.m <- colpairs(state.x77,
function(x,y)  cor.test(x,y,"two.sided","kendall")$estimate, diag=1)
state.col <- dmat.color(state.m)
# This is equivalent to state.m <- cor(state.x77,method="kendall")


layout(matrix(1:2,nrow=1,ncol=2))
cparcoord(state.x77, panel.color= state.col)
# Get rid of the panels with lots of line crossings (yellow) by reorderings
cparcoord(state.x77, order.endlink(state.m), state.col)
layout(matrix(1,1))


# m is a homogeneity measure of each pairwise variable plot
m <- -colpairs(scale(state.x77), gave)

o<- order.single(m)
pcols = dmat.color(m)
# Color panels by level of m and reorder variables so that
# pairs with high m are near the diagonal.
cpairs(state.x77,order=o, panel.colors=pcols)

# In this case panels showing either of Area or Population
# exhibit the most clumpiness because these variables
# are skewed.
```

cpairs *Enhanced scatterplot matrix*

Description

This function draws a scatterplot matrix of data. Variables may be reordered and panels colored in the display.

Usage

```
cpairs(data, order = NULL, panel.colors = NULL, border.color = "grey70",
show.points = TRUE, ...)
```

Arguments

`data`	a numeric matrix
`order`	the order of variables. Default is the order in data.
`panel.colors`	a matrix of panel colors. If supplied, dimensions should match those of the pairs plot. Diagonal entries are ignored.
`border.color`	used for panel border.
`show.points`	If FALSE, no points are drawn.
`...`	graphical parameters passed to `pairs.default`.

Author(s)

Catherine B. Hurley

References

Hurley, Catherine B. “Clustering Visualisations of Multidimensional Data”, to appear in JCGS.

See Also

`pairs`, `cparcoord`, `dmat.color`,`colpairs`, `order.single`.

Examples

```
data(USJudgeRatings)
judge.cor <- cor(USJudgeRatings)
judge.color <- dmat.color(judge.cor)
# Colors variables by their correlation.
cpairs(USJudgeRatings,panel.colors=judge.color,pch=”.”,gap=.5)
judge.o <- order.single(judge.cor)
# Reorder variables so that those with highest correlation
# are close to the  diagonal.
cpairs(USJudgeRatings,judge.o,judge.color,pch=”.”,gap=.5)

# Specify your own color scheme
judge.color <- dmat.color(judge.cor, breaks=c(-1,0,.5,.9,1), colors =
cm.colors(4))

data(bank)
# m is a homogeneity measure of each pairwise variable plot
m <- -colpairs(scale(bank[,-1]), partition.crit,gfun=gave,groups=bank[,1])

# Color panels by level of m and reorder variables so that
# pairs with high m are near the diagonal. Panels shown
# in pink have the highest amount of group homogeneity, as measured by
# gave.
cpairs(bank[,-1],order=order.single(m), panel.colors=dmat.color(m),
gap=.3,col=c(”purple”,”black”)[bank[,”Status”]+1],
pch=c(5,3)[bank[,”Status”]+1])
```

cparcoord *Enhanced parallel coordinate plot*

Description

This function draws a parallel coordinate plot of data. Variables may be reordered and panels colored in the display. It is a modified version of `parcoord {MASS}`.

Usage

```
cparcoord(data, order = NULL, panel.colors = NULL, col = 1, lty = 1,
horizontal = FALSE, mar = NULL, ...)
```

Arguments

`data`	a numeric matrix
`order`	the order of variables. Default is the order in data.
`panel.colors`	either a vector or a matrix of panel colors. If a vector is supplied, the ith color is used for the ith panel. If a matrix, dimensions should match those of the variables. Diagonal entries are ignored.
`col`	a vector of colours, recycled as necessary for each observation.
`lty`	a vector of line types, recycled as necessary for each observation.
`horizontal`	If TRUE, orientation is horizontal.
`mar`	margin parameters, passed to `par`.
`...`	graphics parameters which are passed to matplot.

Details

If `panel.colors` is a matrix and `order` is supplied, `panel.colors` is reordered.

Author(s)

Catherine B. Hurley

References

Hurley, Catherine B. “Clustering Visualisations of Multidimensional Data”, Journal of Computational and Graphical Statistics, vol. 13, (4), pp 788-806, 2004.

See Also

`cpairs`, `parcoord`, `dmat.color`, `colpairs`, `order.endlink`.

Examples

```
data(state)
state.m <- colpairs(state.x77,
function(x,y)  cor.test(x,y,"two.sided","kendall")$estimate, diag=1)
# OR, Works only in R1.8,  state.m <-cor(state.x77,method="kendall")


state.col <- dmat.color(state.m)

cparcoord(state.x77, panel.color= state.col)
# Get rid of the panels with lots of line crossings (yellow) by reordering:
cparcoord(state.x77, order.endlink(state.m), state.col)

# To get rid of the panels with lots of long line segments:
#  use a different panel merit measure- pclen:

mins <- apply(state.x77,2,min)
ranges <- apply(state.x77,2,max) - mins
state.m <- -colpairs(scale(state.x77,mins,ranges), pclen)
cparcoord(state.x77, order.endlink(state.m), dmat.color(state.m))
```

---

diameter *Cluster heterogeneity of 2-d data*

Description

Computes measures of cluster heterogeneity of 2-d data, where x and y give the object coordinates.

Usage

```
diameter(x, y, ...)
star(x, y, ...)
km2(x,y)
gtot(x,y, ...)
gave(x,y, ...)
```

Arguments

x	is a numeric vector.
y	is a numeric vector.
...	are passed to dist.

Details

diameter computes the cluster diameter- the maximum distance between objects.

star computes the cluster star distance- the smallest total distance from one object to another.

km2 computes the kmeans distance.

gtot computes the sum of all inter-object distances.

gave computes the per-object average of all inter-object distances.

Value

The cluster measure is returned.

Author(s)

Catherine B. Hurley

References

See Gordon, A. D. (1999).“Classification”. Second Edition. London: Chapman and Hall / CRC

See Also

colpairs, cpairs, order.single

Examples

```
x <- runif(20)
y <- runif(20)
diameter(x,y)
```

---

dmat.color *Colors a symmetric matrix*

Description

Accepts a dissimilarity matrix or dist m, and returns a matrix of colors. Values in m are cut into categories using breaks (ranked distances if byrank is TRUE) and categories are assigned the values in colors.

Usage

```
dmat.color(m, colors = default.dmat.color, byrank = NULL, breaks = length(colors))
```

Arguments

m	a dissimilarity matrix or the result of dist
colors	a vector of colors. The default is default.dmat.color.
byrank	boolean, default TRUE is unless breaks has length > 1.
breaks	the number of break points.

Details

breaks are passed to the functioncut. If byrank is TRUE, values in m are ranked before they are categorized. If byrank is TRUE and breaks is an integer, then there are breaks equal-sized categories.

Value

Returns a matrix of colors. The matrix is symmetric, with NAs on the diagonal.

Author(s)

Catherine B. Hurley

See Also

cut, cpairs, cparcoord

Examples

```
data(longley)
longley.cor <- cor(longley)
# A matrix with equal (or nearly equal) number of entries of each color.
longley.color <- dmat.color(longley.cor)

# Plot the colors
plotcolors(longley.color,dlabels=rownames(longley.color))

# Try different color schemes

# A matrix where each color represents an equal-length interval.
longley.color <- dmat.color(longley.cor, byrank=FALSE)
# Specify colors and breaks

longley.color <- dmat.color(longley.cor, breaks=c(-1,0,.5,.8,1),
cm.colors(4))


# Could also reorder variables prior to plotting:

longley.o <- order.single(longley.cor)
longley.color <- longley.color[longley.o,longley.o]

# The colors can be used in a scatterplot matrix or parallel
# coordinate display:

cpairs(longley, panel.color= longley.color)
cparcoord(longley, panel.color= longley.color)
```

`order.clusters` *Orders clustered objects using hierarchical clustering*

Description

Reorders objects so that similar (or high-merit) object pairs are adjacent. The clusters argument specifies (possibly ordered) groups, and objects within a group are kept together.

Usage

```
order.clusters(merit,clusters,within.order = order.single,
    between.order= order.single,...)
```

Arguments

`merit`	is either a symmetric matrix of merit or similarity score, or a `dist`.
`clusters`	specifies a partial grouping. It should either be a list whose ith element contains the indices of the objects in the ith cluster, or a vector of integers whose ith element gives the cluster membership of the ith object. Either representation may be used to specify grouping, the first is preferrable to specify adjacencies.
`within.order`	is a function used to order the objects within each cluster.
`between.order`	is a function used to order the clusters.
`...`	arguments are passed to `within.order`.

Details

`within.order` may be NULL, in which case objects within a cluster are assumed to be in order. Otherwise, `within.order` should be one of the ordering functions `order.single`,`order.endlink` or `order.hclust`.

`between.order` may be NULL, in which case cluster order is preserved. Otherwise, `betweem.order` should be one of the ordering functions that uses a partial ordering, `order.single` or `order.endlink`.

Value

A permutation of the objects represented by `merit` is returned.

Author(s)

Catherine B. Hurley

See Also

order.single,order.endlink,order.hclust.

Examples

```
data(state)
state.d <- dist(state.x77)


# Order the states, keeping states in a division together.
state.o <- order.clusters(-state.d, as.numeric(state.division))
cmat <- dmat.color(as.matrix(state.d), rev(cm.colors(5)))


op <- par(mar=c(1,6,1,1))
rlabels <- state.name[state.o]
plotcolors(cmat[state.o,state.o], rlabels=rlabels)
par(op)


# Alternatively, use kmeans to place the  states into 6 clusters
state.km <- kmeans(state.d,6)$cluster

# An ordering obtained from the kmeans clustering...
state.o <- unlist(memship2clus(state.km))


layout(matrix(1:2,nrow=1,ncol=2),widths=c(0.1,1))
op <- par(mar=c(1,1,1,.2))
state.colors <- cbind(state.km,state.km)
plotcolors(state.colors[state.o,])

par(mar=c(1,6,1,1))
rlabels <- state.name[state.o]
plotcolors(cmat[state.o,state.o], rlabels=rlabels)

par(op)
layout(matrix(1,1))



# In the ordering above, the ordering of clusters and the
# ordering of objects within the clusters is arbitrary.
# order.clusters gives an improved order but preserves the kmeans clusters.

state.o <- order.clusters(-state.d, state.km)

# and replot
layout(matrix(1:2,nrow=1,ncol=2),widths=c(0.1,1))
op <- par(mar=c(1,1,1,.2))
state.colors <- cbind(state.km,state.km)
plotcolors(state.colors[state.o,])

par(mar=c(1,6,1,1))
rlabels <- state.name[state.o]
plotcolors(cmat[state.o,state.o], rlabels=rlabels)
```

```
par(op)
layout(matrix(1,1))
```

order.single *Orders objects using hierarchical clustering*

Description

Reorders objects so that similar (or high-merit) object pairs are adjacent. A permutation vector is returned.

Usage

```
order.single(merit,clusters=NULL)
order.endlink(merit,clusters=NULL)
order.hclust(merit, reorder=TRUE,...)
```

Arguments

`merit`	is either a symmetric matrix of merit or similarity score, or a `dist`.
`clusters`	if non-null, specifies a partial ordering. It should be a list whose ith element contains the indices the objects in the ith ordered cluster.
`reorder`	if TRUE, reorders the default ordering from `hclust`.
`...`	arguments are passed to `hclust`.

Details

`order.single` performs a variation on single-link cluster analysis, devised by Gruvaeus and Wainer (1972). When two ordered clusters are merged, the new cluster is formed by placing the most similar endpoints of the joining clusters adjacent to each other. When applied to variables, the resulting order is useful for scatterplot matrices.

`order.endlink` is another variation on single-link cluster analysis, where the similarity between two ordered clusters is defined as the minimum distance between their endpoints. When two ordered clusters are merged, the new cluster is formed by placing the most similar endpoints of the joining clusters adjacent to each other. When applied to variables, the resulting order is useful for parallel coordinate displays.

`order.hclust` returns the order of objects from `hclust` if `reorder` is `FALSE`. Otherwise, it reorders the objects using `hclust.reorder` so that when two ordered clusters are merged, the new cluster is formed by placing the most similar endpoints of the joining clusters adjacent to each other. `order.hclust(m,method="single")` is equivalent to `order.single` when `clusters` is `NULL`. The default method of `hclust` is "complete", see `hclust` for other possibilities.

Value

A permutation of the objects represented by `merit` is returned.

Author(s)

Catherine B. Hurley

References

Hurley, Catherine B. "Clustering Visualisations of Multidimensional Data", Journal of Computational and Graphical Statistics, vol. 13, (4), pp 788-806, 2004.

Gruvaeus, G. and Wainer, H. (1972), "Two Additions to Hierarchical Cluster Analysis", British Journal of Mathematical and Statistical Psychology, 25, 200-206.

See Also

`cpairs`, `cparcoord`,`plotcolors`, `reorder.hclust`,`order.clusters`, `hclust`.

Examples

```
data(state)
state.cor <- cor(state.x77)
order.single(state.cor)
order.endlink(state.cor)
order.hclust(state.cor,method="average")

# Use for plotting...

cpairs(state.x77, panel.colors=dmat.color(state.cor), order.single(state.cor),pch=".",gap=.4)
cparcoord(state.x77, order.endlink(state.cor),panel.colors=dmat.color(state.cor))


# Order the states instead of the variables...

state.d <- dist(state.x77)
state.o <- order.single(-state.d)

op <- par(mar=c(1,6,1,1))
cmat <- dmat.color(as.matrix(state.d), rev(cm.colors(5)))
plotcolors(cmat[state.o,state.o], rlabels=state.name[state.o])
par(op)
```

`ozone` *Ozone data from Breiman and Friedman, 1985*

Description

This is the Ozone data discussed in Breiman and Friedman (JASA, 1985, p. 580). These data are for 330 days in 1976. All measurements are in the area of Upland, CA, east of Los Angeles.

Usage

```
data(ozone)
```

Format

This data frame contains the following columns:

Ozone: Ozone conc., ppm, at Sandbug AFB.

Temp: Temperature F. (max?).

InvHt: Inversion base height, feet

Pres: Daggett pressure gradient (mm Hg)

Vis: Visibility (miles)

Hgt: Vandenburg 500 millibar height (m)

Hum: Humidity, percent

InvTmp: Inversion base temperature, degrees F.

Wind: Wind speed, mph

Source

Breiman, L and Friedman, J. (1985), "Estimating Optimal Transformations for Multiple Regression and Correlation", *Journal of the American Statistical Association*, 80, 580-598.

partition.crit *Combines the results of appplying an index to each group of observations*

Description

Applies the function gfun to each group of x and y values and combines the results using the function cfun

Usage

```
partition.crit(x, y, groups, gfun = gave, cfun = sum, ...)
```

Arguments

x	is a numeric vector.
y	is a numeric vector.
groups	is a vector of group memberships.
gfun	is applied to the x and y data in each group.
cfun	combines the values returned by gfun.
...	arguements are passed to gfun.

Details

The function `gfun` is applied to each group of `x` and `y` values. The function `cfun` is applied to the vector or matrix of `gfun` results.

Value

The result of applying `cfun`.

Author(s)

Catherine B. Hurley

References

See Gordon, A. D. (1999). *Classification*. Second Edition. London: Chapman and Hall / CRC

See Also

`gave`, `colpairs`, `order.single`

Examples

```
x <- runif(20)
y <- runif(20)
g <- rep(c("a","b"),10)

partition.crit(x,y,g)


data(bank)
# m is a homogeneity measure of each pairwise variable plot
m <- -colpairs(scale(bank[,-1]), partition.crit,gfun=gave,groups=bank[,1])

# Color panels by level of m and reorder variables so that
# pairs with high m are near the diagonal. Panels shown
# in pink have the highest amount of group homogeneity, as measured by
# gave.
cpairs(bank[,-1],order=order.single(m), panel.colors=dmat.color(m),
gap=.3,col=c("purple","black")[bank[,"Status"]+1],
pch=c(5,3)[bank[,"Status"]+1])

# Try  a different measure
m <- -colpairs(scale(bank[,-1]), partition.crit,gfun=diameter,groups=bank[,1])

cpairs(bank[,-1],order=order.single(m), panel.colors=dmat.color(m),
gap=.3,col=c("purple","black")[bank[,"Status"]+1],
pch=c(5,3)[bank[,"Status"]+1])


# Result is the same, in this case.
```

pclen *Profile smoothness measures*

Description

Computes measures of profile smoothness of 2-d data, where x and y give the object coordinates.

Usage

```
pclen(x, y)
pcglen(x, y)
```

Arguments

x	is a numeric vector.
y	is a numeric vector.

Details

pclen computes the total line length in a parallel coordinate plot of x and y.

pcglen computes the average (per object) line length in a parallel coordinate plot where all pairs of objects are connected.

Usually, the data is standardized prior to using these functions.

Value

The panel measure is returned.

Author(s)

Catherine B. Hurley

References

Hurley, Catherine B. “Clustering Visualisations of Multidimensional Data”, Journal of Computational and Graphical Statistics, vol. 13, (4), pp 788-806, 2004.

See Also

cparcoord, colpairs, order.endlink.

Examples

```
x <- runif(20)
y <- runif(20)
pclen(x,y)


data(state)
mins <- apply(state.x77,2,min)
ranges <- apply(state.x77,2,max) - mins
state.m <- -colpairs(scale(state.x77,mins,ranges), pclen)
state.col <- dmat.color(state.m)
cparcoord(state.x77, panel.color= state.col)
# Get rid of the panels with long line segments (yellow) by reordering:
cparcoord(state.x77, order.endlink(state.m), state.col)
```

`plotcolors` *Plots a matrix of colors*

Description

`plotcolors` plots a matrix of colors as an image or as points.

`imageinfo` is a utility that given a matrix of colors, returns a structure useful for the `image` function.

Usage

```
plotcolors(cmat, na.color = "white", dlabels = NULL, rlabels = FALSE, clabels = FALSE,
ptype = "image", border.color = "grey70", pch = 15, cex = 3, label.cex = 0.6, ...)

imageinfo(cmat)
```

Arguments

`cmat`	a matrix of numbers, nas are allowed.
`na.color`	used for NAs in `cmat`.
`dlabels`	vector of labels for the diagonals.
`rlabels`	vector of labels for the rows.
`clabels`	vector of labels for the columns.
`ptype`	should be "image" or "points"
`border.color`	color of border drawn around the plot.
`pch`	point type used when ptype="points".
`cex`	point cex used when ptype="points".
`label.cex`	cex parameter used for labels.
`...`	graphical parameters

Value

imageinfo returns a list with components:

x a vector of x coordinates.

y a vector of y coordinates.

z a matrix containing values to be plotted.

col the colors to be used.

Author(s)

Catherine B. Hurley

See Also

plot, image

Examples

```
plotcolors(matrix(1:20,nrow=4,ncol=5))

plotcolors(matrix(1:20,nrow=4,ncol=5),ptype="points",cex=6)

plotcolors(matrix(1:20,nrow=4,ncol=5),rlabels = c("a","b","c","d"))


data(longley)
longley.cor <- cor(longley)
# A matrix with equal (or nearly equal) number of entries of each color.
longley.color <- dmat.color(longley.cor)


plotcolors(longley.color, dlabels=rownames(longley.color))

# Could also reorder variables prior to plotting:
longley.o <- order.single(longley.cor)
longley.color <- longley.color[longley.o,longley.o]

op <- par(mar=c(1,6,6,1))
plotcolors(longley.color,rlabels=rownames(longley.color),clabels=rownames(longley.color) )
par(op)
```

reorder.hclust *Reorders object order of hclust, keeping objects within a cluster contiguous to each other.*

Description

Reorders objects so that nearby object pairs are adjacent.

Usage

```
## S3 method for class 'hclust'
reorder(x,dis,...)
```

Arguments

`x`	is the result of `hclust`.
`dis`	is a distance matrix or `dist`.
`...`	additional arguments.

Details

In hierarchical cluster displays, a decision is needed at each merge to specify which subtree should go on the left and which on the right. This algorithm uses the order suggested by Gruvaeus and Wainer (1972). At a merge of clusters A and B, the new cluster is one of (A,B), (A',B), (A,B'),(A',B'), where A' denotes A in reverse order. The new cluster is chosen to minimize the distance between the object in A placed adjacent to an object from B.

Value

A permutation of the objects represented by `dis` is returned.

Author(s)

Catherine B. Hurley

References

Hurley, Catherine B. "Clustering Visualisations of Multidimensional Data", Journal of Computational and Graphical Statistics, vol. 13, (4), pp 788-806, 2004.

Gruvaeus, G. and Wainer, H. (1972), "Two Additions to Hierarchical Cluster Analysis", British Journal of Mathematical and Statistical Psychology, 25, 200-206.

See Also

`hclust`, `order.hclust` .

Examples

```
data(eurodist)
dis <- as.dist(eurodist)
hc <- hclust(dis, "ave")


layout(matrix(1:2,nrow=2,ncol=1))
op <- par(mar=c(1,1,1,1))
plot(hc)
hc1 <- reorder.hclust(hc, dis)
plot(hc1)
par(op)
layout(matrix(1,1))

# Both dedrograms correspond to the same tree structure,
# but the second one shows that
# Paris is closer to Cherbourg than Munich, and
# Rome is closer to Gibralter than to Barcelona.


# We can also compare both orderings with an
# image plot of the colors.
# The second ordering seems to place nearby cities
# closer to each other.


layout(matrix(1:2,nrow=2,ncol=1))
op <- par(mar=c(1,6,1,1))
cmat <- dmat.color(eurodist, rev(cm.colors(5)))
plotcolors(cmat[hc$order,hc$order], rlabels=labels(eurodist)[hc$order])

plotcolors(cmat[hc1$order,hc1$order], rlabels=labels(eurodist)[hc1$order])

layout(matrix(1,1))
par(op)
```

---

vec2distm *Various utility functions*

Description

vec2distm converts a vector to a distance matrix.

vec2dist converts a vector to a dist structure.

lower2upper.tri.inds is the same as lower.to.upper.tri.inds from package cluster. It computes an index vector for extracting or reordering a lower triangular matrix that is stored as a contiguous vectors.

`diag.off` returns a vector of off-diagonal elements of a matrix. `off` specifies the distance above the main (0) diagonal.

`clus2memship` converts a list whose ith element contains the indices of objects in the ith cluster into a vector whose ith element gives the cluster number of the ith object.

`memship2clus` converts a vector whose ith element gives the cluster number of the ith object into a list whose ith element contains the indices of objects in the ith cluster.

Usage

```
vec2distm(vec)
vec2dist(vec)
lower2upper.tri.inds(n)
diag.off(m,off=1)
clus2memship(clusters)
memship2clus(memship)
```

Arguments

`vec` is a vector.

`n` is an integer > 1.

`m` is a matrix.

`clusters` is a list whose ith element contains the indices of the objects belonging to the ith cluster.

`off` is an integer specifying the distance above the main (0) diagonal.

`memship` is a vector whose ith element gives the cluster number of the ith object.

Author(s)

Catherine B. Hurley

See Also

`dist`, `diag`.

Examples

```
vec <- 1:15
vec2distm(vec)
vec2dist(vec)
diag.off(vec2distm(vec))
lower2upper.tri.inds(5)
clus2memship(list(c(1,3,5),c(2,6),4))
memship2clus(c(1,3,4,2,1,4,2,3,2,3))
```

`wine` *Wine recognition data*

Description

Data from the machine learning repository. A chemical analysis of 178 Italian wines from three different cultivars yielded 13 measurements. This dataset is often used to test and compare the performance of various classification algorithms.

Usage

```
data(wine)
```

Format

This data frame contains the following columns:

Class: There are 3 classes

Alcohol: Alcohol

Malic: Malic acid

Ash: Ash

Alcalinity: Alcalinity of ash

Magnesium: Magnesium

Phenols: Total phenols

Flavanoids: Flavanoids

Nonflavanoid: Nonflavanoid phenols

Proanthocyanins: Proanthocyanins

Intensity: Color intensity

Hue: Hue

OD280: OD280/OD315 of diluted wines

Proline: Proline

Source

Forina, M. et al, PARVUS - An Extendible Package for Data Exploration, Classification and Correlation. Institute of Pharmaceutical and Food Analysis and Technologies, Via Brigata Salerno, 16147 Genoa, Italy.

References

Blake, C.L. and Merz, C.J. (1998), UCI Repository of machine learning databases, \ http://www.ics.uci.edu/~mlearn/MLRepository.html. Irvine, CA: University of California, Department of Information and Computer Science.

The database does not list the variable names. These were located at http://www.radwin.org/michael/projects/learning/about-wine.html.

Index