Package ‘extracat’

November 21, 2015

Type Package
Title Categorical Data Analysis and Visualization
Version 1.7-4
Date 2015-11-21
Author Alexander Pilhoefer <alexander.pilhoefer@gmail.com>
Maintainer Alexander Pilhoefer <alexander.pilhoefer@gmail.com>
Description Categorical Data Analysis and Visualization.
Depends R (>= 3.1.0)
Suggests ca, amap, MASS, proto
Imports grid, colorspace, hexbin, scales, ggplot2, plyr, TSP, methods,
       data.table, reshape2
LazyLoad yes
LazyData yes
License GPL (>= 2)
NeedsCompilation yes
Repository CRAN
Date/Publication 2015-11-21 16:49:01

R topics documented:

agaricus ................................................................. 3
ahist ................................................................. 4
approx.dcor ......................................................... 5
arsim ................................................................. 7
barysort .............................................................. 9
BCC ............................................................... 10
BCI ............................................................... 11
Burt ............................................................. 12
carcustomers ...................................................... 13
CBCI ............................................................ 15
cfcl ............................................................ 16
R topics documented:

cfluctile .................................................. 17
cmat ......................................................... 19
cohens ....................................................... 20
combel ....................................................... 21
CPScleuster ................................................ 21
cutbw ......................................................... 23
dcorMVdata ............................................... 24
dcorMVtable ................................................ 25
dendro ......................................................... 26
dmc .......................................................... 27
eco .......................................................... 28
extracat ..................................................... 36
facetshade .................................................. 38
fluctile ....................................................... 42
GeneEx ....................................................... 45
getbw ......................................................... 47
getcolors .................................................... 49
getIs ........................................................ 50
getIs2 ......................................................... 51
gpath ........................................................ 52
gsac .......................................................... 53
heattile ...................................................... 56
hexpie ......................................................... 58
idat .......................................................... 61
imat .......................................................... 62
innerval ....................................................... 62
itab .......................................................... 63
JBCI .......................................................... 64
kendalls ..................................................... 65
ME ............................................................. 66
MJnew ......................................................... 67
olives ........................................................ 77
optile ........................................................ 78
optME ........................................................ 86
plants ......................................................... 87
qBCI .......................................................... 90
quickfechner ............................................... 92
regmax ......................................................... 93
rmb .......................................................... 94
rmbmat ....................................................... 100
scpcp ......................................................... 103
setcover ..................................................... 105
sortandcut .................................................. 107
steptile ....................................................... 108
subtable ...................................................... 110
subtree ....................................................... 111
tfluctile ..................................................... 112
utableSet ................................................... 113
Description

Characteristics of more than 8000 mushrooms.

Usage

data("agaricus")

Format

A data frame with 8124 observations on the following 23 variables.

- classes: a factor with levels edible poisonous
- cap_shape: a factor with levels bell conical convex flat knobbed sunken
- cap_surface: a factor with levels fibrous grooves scaly smooth
- cap_color: a factor with levels brown buff cinnamon gray green pink purple red white yellow
- bruises: a factor with levels bruises no
- odor: a factor with levels almond anise creosote fishy foul musty none pungent spicy
- gill_attachment: a factor with levels attached free
- gill_spacing: a factor with levels close crowded
- gill_size: a factor with levels broad narrow
- gill_color: a factor with levels black brown buff chocolate gray green orange pink purple red white yellow
- stalk_shape: a factor with levels enlarging tapering
- stalk_root: a factor with levels bulbous club equal rooted
- stalk_surface_above_ring: a factor with levels fibrous scaly silky smooth
- stalk_surface_below_ring: a factor with levels fibrous scaly silky smooth
- stalk_color_above_ring: a factor with levels brown buff cinnamon gray orange pink red white yellow
- stalk_color_below_ring: a factor with levels brown buff cinnamon gray orange pink red white yellow
- veil_type: a factor with levels partial
- veil_color: a factor with levels brown orange white yellow
```
ring_number  a factor with levels none one two
ring_type    a factor with levels evanescent flaring large none pendant
spore_print_color a factor with levels black brown buff chocolate green orange purple white yellow
population   a factor with levels abundant clustered numerous scattered several solitary
habitat       a factor with levels grasses leaves meadows paths urban waste woods
```

**Source**

UCL Machine Learning Repository

**Examples**

```
data(agaricus)
## maybe str(agaricus); plot(agaricus) ...
```

---

**ahist**

*Histogram using active bins*

**Description**

A standard histogram using `getbw` to compute the binwidth and breakpoints.

**Usage**

```
ahist(x, k = NULL, m = NULL, fun = "qplot", col = "grey", ival = NULL)
```

**Arguments**

- `x` A numeric vector.
- `k` The desired number of active bins. A bin is active if it contains at least `min_n` observations. The default is \( k = \lceil 1 + 2\times\text{ceiling}(\log(N)/\log(2)) \rceil \).
- `m` The minimum number of observations necessary for a bin to count as an active bin. Defaults to \( m = \max(\log(N/10)/\log(10),1) \).
- `fun` Either "qplot" or "hist".
- `col` The color for the bars.
- `ival` If this is set to a numeric value in \( (0,1) \) then x is trimmed according to `interval(x, p = ival)`.

**Value**

The `ggplot` object.

**Note**

This is purely experimental at this time.
approx.dcor

Author(s)

Alexander Pilhoefer

See Also

getbw, cuthw

Examples

```r
ahist(rnorm(100))
ahist(rnorm(1000))
ahist(rnorm(10000))

ahist(rexp(100))
ahist(rexp(1000))
ahist(rexp(10000))
```

```r
# Not run:
ahist(rcauchy(1000))
ahist(rcauchy(1000), ival = 0.95)

x <- c(rnorm(400), rnorm(200, mean=6))
ahist(x)

x <- c(rnorm(400), rnorm(200, mean=16))
ahist(x)

x <- c(rnorm(400), rnorm(200, mean=32))
ahist(x)

# End(Not run)
```

approx.dcor  Distance Correlation Approximation

Description

Computes the distance correlation for two variables using an approximation based on binning and `wdcor.table`. The approximation underestimates the true value by a small error depending on the number of bins. (In simulations with the default of 50 bins the average error was about 0.001.)

Usage

```r
approx.dcor(x, y, n = 50, ep = 1, bin = "eq")
```
Arguments

x A numeric vector.
y A numeric vector.
n The number of bins per variable.
ep The euclidean distances are taken to the power of ep.
bin Either "eq" or "q" for equidistant breakpoints or quantile breakpoints.

Value

The correlation value which is between 0 and 1.

Author(s)

Alexander Pilhofer

References


Examples

## Not run:

# The straightforward way of approximating the distance correlation fails:
# for instance the computation of dcor for a random sample with 4000 observations
# takes pretty long but drawing samples of 500, 1000 or even 2000 observations
# leads to relatively big errors.
# The approximation via approx.dcor is very fast and for
# n = 50 or n=100 the results are very close to the truth

require(energy)
x <- rnorm(4000, mean=10, sd=3)
y <- rnorm(1, sd=0.01)*(x-10)^3 + rnorm(1, sd=0.1)*(x-10)^2
  + rnorm(1)*(x-10)+rnorm(4000, sd=4)

system.time(dd <- dcor(x, y))

system.time(dd0 <- wdcor(x, y)[[3]])
dd - dd0

system.time(da100 <- approx.dcor(x, y, 100))[[3]]
da100-dd0

# For a smaller sample size we can try out how good the approximation really is:

test <- replicate(100,
  N <- 1000
  x <- rnorm(N, mean=10, sd=3)
y <- rnorm(1, sd=0.01)*(x-10)^3 + rnorm(1, sd=0.1)*(x-10)^2
y <- y + rnorm(1)*(x-10)+rnorm(N, sd=4)
arsim

block-structured arrays

Description

Generates an array or matrix that includes k fully separated block-clusters.

Usage

```r
arsim(n, dim, k, noise = 0, shuffle = TRUE, v = 0.1, minc = 1,
exp.prop = NULL, min.prop = 1/dim/4, noise.type = "s",
dimnames=list(LETTERS,1:max(dim)))
```

Arguments

- **n**: The number of observations in the array.
- **dim**: The dimension of the array.
- **k**: The number of clusters. 1 for no clusters.
- **noise**: The proportion of noise among the observations. There are two choices for noise.type.
- **shuffle**: Whether or not to shuffle the original category orders randomly.
arsim

v  A variability parameter for the assignment of the observations to the block clusters. Small values lead

minc  The minimum number of categories each cluster must have in each variable. E.g. minc = 2 means, that each block cluster covers at least 2 categories in each dimension.

exp.prop  Optional: expected proportions of the observations which fall into the block clusters.

min.prop  Minimum proportion of observations in each cluster.

noise.type  Either "s" or "I". The noise type "s" means that n*noise observations are drawn at random from the block-diagonal matrix. Then for these observations the category labels are permuted at random. "I" adds noise in form of a random sample from the independence matrix with the same marginal totals as the block matrix.

dimnames  A list of 2: The first entry defines the variable labels (default: A,B,C,...) and the second entry defines the category labels (default 1:k).

Details

Not a very sophisticated way of generating random arrays but it serves for tests and illustrations of the other functions.

Value

A simulated data array.

Examples

A <- arsim(1000, c(12,12), 3, shuffle = FALSE)
fluctile(A)

A <- arsim(1000, c(12,12), 3, shuffle = FALSE, dimnames = list(NULL,letters))
dimnames(A)

A <- arsim(4000, c(11,7,5), 3, shuffle = TRUE, dimnames = list(0:2,letters))
dimnames(A)

## Not run:
A2 <- arsim(1000, c(12,12,12), 3, shuffle = FALSE)
fluctile3d(A2, shape = "oct")

## End(Not run)
### barysort

#### row and column moment reordering

**Description**

An iterative row and column reordering procedure based on the barycenter heuristic.

**Usage**

```r
barysort(x, vs = 1)
```

**Arguments**

- **x**: The data matrix.
- **vs**: version. no effect.

**Value**

The reordered matrix.

**Examples**

```r
# a good and quick solution:
a <- arsim(2000,c(24,24),6, noise=0.4)
fluctile(a2<-barysort(a))
BCI(a2)

# which is near
a3 <- optile(a, iter=100)
BCI(a3)

## Not run:
a <- arsim(64000,c(256,256),16, noise=0.4)
s1 <- system.time( bci1 <- BCI(a1<-optile(a, fun = "barysort", foreign=".Call", iter = 1)) )[[3]]  
s2 <- system.time( bci2 <- BCI(a2<-optile(a, iter=1)) )[[3]]  
s3 <- system.time( bci3 <- BCI(a3<-optile(a, fun="IBCC",iter=1)) )[[3]]

## End(Not run)
```
The Bertin Classification Criterion

Description
Computes the Bertin Classification Criterion for a contingency table of any dimensions.

Usage
BCC(x)

Arguments
x A data matrix, table or array.

Details
The BCC counts the number of observation pairs which differ in all variables but are not fully concordant, (i.e. neither of the two observations of each pair is larger than the other in all variables).

Value
The criterion value.

Author(s)
Alexander Pilhoefer

See Also
kendalls

Examples
M <- arsim(1000, c(12,12), 3)
BCC(M)

M2 <- optile(M, iter = 100)
BCC(M2)
Description

Computes the Bertin Classification Index for a contingency table of any dimensions.

Usage

\[ \text{BCI}(x) \]

Arguments

- \( x \) A data matrix, table or array.

Details

The BCI is the Bertin Classification Criterion (BCC) normalized by the BCC value under independence.

Value

The criterion value.

Author(s)

Alexander Pilhoefer

See Also

kendalls

Examples

```r
# for an unoptimized matrix we take the minimum of BCI(M) and BCI(M[,12:1])
M <- arsim(10000, c(12,12), 3)
min(BCI(M), BCI(M[,12:1]))

# an strongly related alternative (for two-way data)
kendalls(M)

M2 <- optile(M, iter = 100)
BCI(M2)
kendalls(M2)

M3 <- arsim(100000, c(12,13,15), 4, noise=0.2, shuffle=FALSE)
BCI(M3)
```
Description

The Burt matrix is a quadratic matrix where each row and column corresponds to a category in one of the variables. The entries of the matrix are the frequencies of the corresponding combination of categories.

Usage

\texttt{Burt(x)}

Arguments

\texttt{x} \hspace{1cm} A dataframe with factor variables or a contingency table.

Value

A matrix.

Author(s)

Alexander Pilhoefer

See Also

idat, imat

Examples

\texttt{require(MASS)}
\texttt{Burt(housing)}
\texttt{th <- xtabs(Freq\textasciitilde Sat\textasciitilde Infl\textasciitilde Type, data = housing)}
\texttt{Burt(th)}
Description

This dataset is taken from the website of the Department of Statistics, University of Munich. The data are based upon a poll from a German car-company. In 1983 questionnaires were sent to 2000 customers, who had purchased a new car approximately three months earlier. The point of interest was the degree of satisfaction, reasons for the particular choice, consumer profile, etc. Participation was of course voluntary. Only 1182 persons answered the questions and after removing forms with "missing values" only 793 questionnaires remained. Each form contained 46 questions, which resulted in a dataset of 46 covariates with 793 observations each. Due to the abundance of ordinal and categorical covariates the dataset is particularly suited for generalized linear models.

Usage

data(carcustomers)

Format

A data frame with 774 observations on the following 47 variables.

- model: a factor with levels A B C D
- gear: a factor with levels 4-gear 5-gear (overdrive) 5-gear (sport) Automatic
- lease: a factor with levels bought leased
- usage: a factor with levels business private private and business
- premod: a factor with levels Audi BMW 3er BMW 5er BMW 7er Ford Mercedes Benz Opel other origin Volkswagen other: a factor with levels No Yes, both Yes, other manufact Yes, same manufact.
- testdrv: influence on buying decision: testdrive
- promotion: influence on buying decision: promotion
- exp: influence on buying decision: experience
- recom: influence on buying decision: recommendation
- clear: influence on buying decision: clearness
- eco: influence on buying decision: economical aspects
- drvchar: influence on buying decision: driving character
- service: influence on buying decision: service
- interior: influence on buying decision: interior
- quality: influence on buying decision: overall quality
- tech: influence on buying decision: technical aspects
- evo: influence on buying decision: evolution
- comfort: influence on buying decision: comfort
reliable influence on buying decision: reliability
handling influence on buying decision: handling
prestige influence on buying decision: prestige
concept influence on buying decision: overall concept
character influence on buying decision: character
power influence on buying decision: engine power
value decrease influence on buying decision: value decrease
styling influence on buying decision: styling
safety influence on buying decision: safety
sport influence on buying decision: sportive
fuel consumption influence on buying decision: fuel consumption
space influence on buying decision: available space
sat overall satisfaction with the car: 1(very satisfied) to 5(not satisfied)
adv1 satisfaction with concept and styling: a factor with levels does not suit neither nor suits
adv2 satisfaction with body/bare essentials: a factor with levels does not suit neither nor suits
adv3 satisfaction with chassis/drive/gearshift: a factor with levels does not suit neither nor suits
adv4 satisfaction with engine/power: a factor with levels does not suit neither nor suits
adv5 satisfaction with electronics: a factor with levels does not suit neither nor suits
adv6 satisfaction with financial aspects: a factor with levels does not suit neither nor suits
adv7 satisfaction with equipment: a factor with levels does not suit neither nor suits
spoco balance variables: a factor with levels comfort could be better handling could be better well balanced
favor usual driving style: a factor with levels economical extreme normal powerful
speed usual speed (Autobahn): a factor with levels >110 mph 60-80 mph 81-90g mph 96-110 mph
fuel consumption satisfaction with fuel consumption: a factor with levels appropriate definitely too high
just okay pleasingly low
sex customer’s gender: a factor with levels Female Male
prof customer’s profession: a factor with levels Employee/Workman Free lanced Self employed
family customers’s family type: a factor with levels >3 persons 1-2 persons
Freq the weighting variable

Source

http://www.stat.uni-muenchen.de/service/datenarchiv/auto/auto_e.html

Examples

data(Autos)
## maybe str(Autos) ; plot(Autos) ...
The Conditional Independence Bertin Classification Index

Description

Computes the Conditional Independence Bertin Classification Index which uses conditional independence as a reference for normalization. High values indicate that the BCC is not far from the expectation if we know the two marginal 2D BBC values.

Usage

\[
\text{CBCI}(x, r = 1, \text{joint.order} = \text{FALSE})
\]

Arguments

- **x**: The 3D table with non-negative entries.
- **r**: The index of the conditioning variable, e.g. \(r = 1\) uses the table with variables 2 and 3 conditionally independent given 1 for normalization.
- **joint.order**: Whether or not to use a joint ordering for all variables. Otherwise the pairwise values are computed using separate reorderings.

Details

The BCI of a 3D table but instead of the total independence case the conditional independence case is used for normalization.

Value

Numeric value in \([0,1]\).

Author(s)

Alexander Pilhoefer

See Also

BCI, JBCI, WBCI

Examples

```r
## Not run:
A <- optile(arSim(10000, c(1,12,13), 4, 0.1))
BCI(A)

CBCI(A,1,TRUE)
CBCI(A,1,FALSE)
```
cfcl  

**Extract clusters from cfluctile**

### Description

Extract clusters from cfluctile

### Usage

```r
cfcl(x, y = NULL, ll)
```

### Arguments

- **x**  
  vector or dataframe.

- **y**  
  if `x` is a vector, `y` needs to be specified.

- **ll**  
  The list with the names of the levels which are combined.

### Value

A 2-column dataframe with the cluster factors.

### See Also

- `cfluctile`

### Examples

```r
a <- arsim(2000, c(12,17), 5, noise=0.2, shuffle = FALSE)
cfa <- cfluctile(a)
da <- as.data.frame(a)
clusters <- cfcl(da, ll = cfa)

dev.new()
fluctile(xtabs(da$Freq~clusters[,1] + clusters[,2]))
table(combc1(clusters))
```
cfluctile

Pseudo-Diagonal Partitioning for two-way tables

Description

Identifies a diagonal of block-clusters in a two-way table using a top-down-partitioning algorithm then plots the table and adds the clusters as rectangles.

Usage

cfluctile(x, tau0 = NULL, method = "Kendall", nsplit = NULL,
maxsplit = NULL, trafo = I, gap.prop = 0.2, floor = 0,
rev.y = FALSE, add = FALSE, shape = "r", just = "c",
dir = "b", plot = TRUE, rect.opt = list(), border =
NULL, label = TRUE, lab.opt = list(), tile.col =
hsv(0.1, 0.1, 0.1, alpha = 0.6),
tile.border = NA, bg.col = "lightgrey", ...)

Arguments

x A 2-way table or matrix.
tau0 The minimum acceptable value of Kendall’s tau, Cohen’s Kappa or WBCI. De-faults to the criterion of the input matrix x.
method Either "Kendall" for Kendall’s Tau, "Cohen" for Cohen’s Kappa, "WBCI" for the Weighted Bertin Classification Criterion and "s" for the minimum residual method.
nsplit The number of splits to make. tau0 is ignored.
maxsplit The maximum number of splits.
trafo A transformation of the table entries for the plot, but not for the computation of the splits. E.g. trafo = function(z) log(1+z).
gap.prop proportion of the gaps between rows/columns.
floor floor censored zooming: all cases will be plotted but only those with a frequency of at least floor will be considered for the clustering.
rev.y revert the y axis.
add Whether to make a new plot or to add to an existing one.
shape The shape of the objects. See fluctile.
just See fluctile.
dir See fluctile.
plot Whether or not to create a plot via fluctile.
rect.opt A list with optional parameters for the rectangles. Possible parameters are:

col The rectangle color.
lwd The line width. Default is "red".
lty The line type. Default is 1 (solid).
fill The color to fill the rectangles. Defaults is NULL. A sensible choice is for instance alpha(col, 0.1).
The white margins around the plot which are also used for the labels. Must be a vector of length 1, 2 or 4 with values in \([0, 1]\). Default is `border = 0.05`.

Whether or not to draw labels.

Label options, see `fluctile`.

Color(s) for the tiles, see `fluctile`.

Border color for the tiles. Can also be a matrix.

Color for the background of the cells, see `fluctile`.

... dots

Details

This function calls `fluctile` to create a 2-way fluctuation diagram and then adds cluster rectangles to it. The cluster rectangles are computed in the following way:

The algorithm cuts the data matrix once horizontally and once vertically and computes a criterion for the 2x2 table consisting of the sums of the four parts that resulted from the cuts. This is done for all possible horizontal and vertical cuts and the best combination is chosen. Then the same procedure is applied to the bottom right submatrix and the top left submatrix. The algorithm stops if no cut yields a better criterion value than `tau0`.

Value

`invisible(TRUE)`

Note

This was part of the Google Summer of Code 2011.

Author(s)

Alexander Pilhoefer
Department for Computer Oriented Statistics and Data Analysis
University of Augsburg
Germany

See Also

`optile`, `sortandcut`, `tfluctile`

Examples

```r
M <- arsim(10000, c(30, 40), 8, noise = 0.4)
cfluctile(M2 <- optile(M, iter=20))
cfluctile(M3 <- sortandcut(M))
cfluctile(M3, nsplit = 4)
```
Describes pairwise association matrix


cmat pairwise association matrix

Description

Computes pairwise BCI values via qBCI.

Usage

```r
cmat(x, sort = TRUE, crit = BCI, k = 5, iter = 20,
p = NULL, jitter = TRUE, freqvar = NULL, diag = NULL,
fun = "BCC", foreign = NULL)
```

Arguments

- `x`: A `data.frame` with factor variables or numeric variables which will be transformed to ordinal interval variables via `cut`. The breakpoints are quantiles of the variables such that for each pair of numeric variables the expected number of observations in each combination of intervals is at least `k`.
- `sort`: Whether or not to sort the pairwise tables via `optile`.
- `crit`: The criterion function, e.g. `kendalls`, `bci`, `wbci` or `wdcor`.
- `k`: The minimum expected number for each cell after quantile binning. See also `qBCI`.
- `iter`: An `optile` parameter.
- `p`: The quantile distance. See `qBCI`.
- `jitter`: Whether or not to use jittering in order to avoid ties. This is equivalent to a random assignment of ranks to observations with the same value.
- `freqvar`: Optional weights, e.g. a frequency variable.
- `diag`: An optional value for the diagonal. Avoids unnecessary function calls for the diagonal elements. E.g. `diag = 0` for `crit = BCI` or `diag = 1` for `crit = kendalls` makes sense.
- `fun`: See `optile`.
- `foreign`: See `optile`.

Details

Uses pairwise complete cases only!
Cohen’s Kappa for rectangular matrices

Description

Cohen’s Kappa for quadratic and non-quadratic matrices using L1-weights.

Usage

cohen(x)

Arguments

x A matrix with non-negative entries.

Value

Cohen’s Kappa

See Also

kendalls, BCI, WBCI

Examples

a <- arsim(2000,c(12,12),6)
cohen(a)
cohen(optile(a))
combcl

Combine categorical variables from cfluctile and cfcl

Description

Combines variables obtained via `cfcl` and `cfluctile` to a single factor variable with one level per block-cluster and one level for the rest.

Usage

```
combcl(x)
```

Arguments

- `x` A matrix, table or data.frame. All variables should have the same number of categories.

Value

A factor variable with 1 level per diagonal element and 1 level for the rest.

Examples

```
a <- arsim(2000, c(12,17),5, noise=0.2, shuffle = FALSE)
cfa <- cfluctile(a)
da <- as.data.frame(a)
clusters <- cfcl( da, ll = cfa)
dev.new()
fluctile(xtabs(da$Freq-clusters[,1] + clusters[,2]))
table(combcl(clusters))
```

CPScluster


Description

Different hierarchical clusterings and k-means clusterings as well as a model-based clustering have been applied to several financial variables for a random sample of ten thousand observations.

Usage

```
data(CPScluster)
```
Format

A data frame with 10000 observations on the following 39 variables.

Age  a numeric vector
Sex  a factor with levels female male
Race a factor with levels Black White
Ethnic a factor
Marital.Status a factor
Kind.of.Family a factor
Classical a factor with levels All other Classical Husband-Wife family
Family.Type a factor
Number.of.Persons.in.Family a numeric vector
Number.of.Kids a numeric vector
Education.of.Head a factor
Labor.Status a factor
Class.of.Worker a factor
Working.Hours a numeric vector
Income.of.Head a numeric vector
Family.Income a numeric vector
Taxable.Income a numeric vector
Federal.tax a numeric vector
Family.sequence.number a numeric vector
State a factor
Division a factor
Region a factor with levels Midwest North East South West
hc4 a numeric vector
hc6 a numeric vector
hc8 a numeric vector
hc12 a numeric vector
hcs4 a numeric vector
hcs6 a numeric vector
hcs8 a numeric vector
hcs12 a numeric vector
hcw4 a numeric vector
hcw6 a numeric vector
hcw8 a numeric vector
hcw12 a numeric vector
km4 a numeric vector
km6 a numeric vector
km8 a numeric vector
km12 a numeric vector
mc12 a numeric vector
**cutbw**

**Examples**

```r
data(CPScluster)
## maybe str(CPScluster) ; plot(CPScluster) ...
```

<table>
<thead>
<tr>
<th>cutbw</th>
<th><em>Active binning</em></th>
</tr>
</thead>
</table>

**Description**

Uses `cut` with breakpoints derived by `getbw`.

**Usage**

```r
cutbw(x, k = NULL, min_n = NULL, warn = FALSE)
```

**Arguments**

- **x**: A numeric variable.
- **k**: The desired number of active bins. A bin is active if it contains at least `min_n` observations. The default is `k <- 1 + 2*ceiling(log(N)/log(2))`.
- **min_n**: The minimum number of observations necessary for a bin to count as an active bin. Defaults to `min_n = max(log(N/10)/log(10),1)`.
- **warn**: Whether or not to print a warning if the desired number of bins is not possible.

**Value**

An ordinal factor variable.

**Note**

Experimental.

**Author(s)**

Alexander Pilhoefer

**See Also**

`getbw`, `ahist`

**Examples**

```r
y <- cutbw(c(rnorm(200), rnorm(100, mean=8)), k = 30, min_n = 1)
barplot(table(y))
```
Multivariate Distance Correlation for two sets of variables

Description

Computes the distances within two sets of variables and the corresponding distance correlation.

Usage

dcorMVdata(x, ind = 1, method = "euclidean", approx = FALSE)

Arguments

x The data.frame which should only contain non-factor variables. For factor variables use xtabs in combination with dcorMVtable.
ind The indices for the first set of variables. The second set consists of all remaining variables.
method The method for dist.
approx FALSE for no approximation via binning or an integer value for the number of bins.

Value

The distance correlation between 0 and 1 for the distances from the two sets of variables.

Note

This code has not been tested thoroughly and may still contain errors.

Author(s)

Alexander Pilhoefer

See Also

dcorMVtable, wdcor, approx.dcor

Examples

```r
## Not run:
so <- scale(olives[,3:8])
dcorMVdata(so,ind=1)

pl <- princomp(so)
sol1 <- cbind(so,pl$scores[,1])
sol2 <- cbind(so,pl$scores[,2])
sol2 <- cbind(sol2,pl$scores[,1:2])
```
**dcorMVtable**

```
dcorMVdata(so1, ind=7)
dcorMVdata(so2, ind=7)
dcorMVdata(so12, ind=7:8)
# how about principal curves?

## End(Not run)
```

---

**dcorMVtable**  
*Multivariate Distance Correlation for two sets of variables*

**Description**

Computes the distances within two sets of variables and the corresponding distance correlation.

**Usage**

```
dcorMVtable(x, ind = 1, method = "euclidean")
```

**Arguments**

- **x**: A contingency table of class `table`.
- **ind**: The indices for the first set of variables. The second set consists of all remaining variables.
- **method**: The method for `dist`.

**Value**

The distance correlation between 0 and 1 for the distances from the two sets of variables.

**Note**

This code has not been tested thoroughly and may still contain errors.

**Author(s)**

Alexander Pilhoefer

**See Also**

`dcorMVdata, wdcor, approx.dcor`
Examples

```r
## Not run:
A2 <- arsim(2000,c(8,9),5,0.1)
A2 <- optile(A2, iter=100)
BCI(A2)
wdcor(A2)

p1 <- runif(11)+0.1
p1 <- p1/sum(p1)
A2b <- apply(A2,1:2,function(z) rmultinom(1,z,p1))

# now the first variable is roughly independent from the other two:
dcorMVtable(as.table(A2b),ind = 1)

# here the third variable is NOT independent from the others:
A3 <- arsim(2000,c(8,9,11),5,0.1)
A3 <- optile(A3, iter=100)
BCI(A3)
dcorMVtable(A3,ind = 3)
```

## End(Not run)

---

**dendro**

**Waterfall Dendrogram**

**Description**

Draws an alternative to the dendrogram using cpcp coordinates. Colors according to a specific number of clusters make the interpretation easier. Also splits which follow each other within a margin of min.gap (proportion of maximum height) can be displayed by boxes.

**Usage**

```r
dendro(x, k = 30, color.id = k - 2, label = FALSE, opts = list(),
min.gap = 0.01, spline = FALSE, ...)
```

**Arguments**

- `x` A hierarchical clustering object.
- `k` The maximum number of clusters to plot. Possible are values up to N-1 but should usually be smaller.
- `color.id` The number of clusters for the coloring.
- `label` Whether or not to draw observation labels. Makes sense for small datasets.
- `opts` Graphics and colour parameters such as lwd, ps or alpha.
min.gap  Joins which are closer than min.gap from each other will be packed and displayed as a box.
spline  Whether or not to use spline curves instead of straight line connections between the points.
...  dots

Value

TRUE

Examples

```r
## Not run:
library(amap)
hc <- hcluster(USArrests)
# the full plot:
dendro(hc, k = 24, min.gap = 0.00)

# aggregation splits within 0.02 maximum height
dendro(hc, k = 24, min.gap = 0.02)

# the same graphic with spline curves instead of straight lines.
dendro(hc, k = 24, min.gap = 0.02, spline = TRUE)

# olive oil data
sx <- scale(olives[, -c(1, 2, 11)])
hc <- hcluster(sx)
plot(hc)
dendro(hc, 120, color.id = 6, min.gap=0.005)
dendro(hc, 120, color.id = 6, min.gap=0.1)

dendro(hc, 120, color.id = 6, min.gap=0.1, spline = TRUE)

## End(Not run)
```

Description

five insurance variables from the dmc 2009 dataset, which have a ordinal structure which has been lost somehow. Can we find it again?

Usage

```r
data(dmc)
```
Format

A data frame with 693 observations on the following 6 variables.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>eiw_scr</td>
<td>a factor with levels 5 6 4 3 2 1</td>
</tr>
<tr>
<td>eih_scr</td>
<td>a factor with levels 6 3 5 1 4 2</td>
</tr>
<tr>
<td>ifi_scr</td>
<td>a factor with levels 4 3 5 2 1 6</td>
</tr>
<tr>
<td>tec_scr</td>
<td>a factor with levels 5 1 3 2 4 6</td>
</tr>
<tr>
<td>klv_scr</td>
<td>a factor with levels 2 5 6 1 3 4</td>
</tr>
<tr>
<td>Freq</td>
<td>a numeric vector</td>
</tr>
</tbody>
</table>

Details

The Data Mining Cup (dmc) is a competition for students.

Examples

```r
data(dmc)
```

---

Description

ADAC Ecotest data with clusterings.

Usage

```r
data("eco")
```

Format

A data frame with 753 observations on the following 21 variables.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>a factor with levels Hersteller 1 10 100 101 102 103 104 105 106 107 108 109 11 110 111</td>
</tr>
<tr>
<td>V2</td>
<td>112 113 114 115 116 117 118 119 12 120 121 122 123 124 125 126 127 128 129 13 130 131</td>
</tr>
<tr>
<td>V3</td>
<td>132 133 134 135 136 137 138 139 14 140 141 142 143 144 145 146 147 148 149 15 150 151</td>
</tr>
<tr>
<td>V4</td>
<td>152 153 154 155 156 157 158 159 16 160 161 162 163 164 165 166 167 168 169 17 170 171</td>
</tr>
<tr>
<td>V5</td>
<td>172 173 174 175 176 177 178 179 18 180 181 182 183 184 185 186 187 188 189 19 190 191</td>
</tr>
<tr>
<td>V6</td>
<td>192 193 194 195 196 197 198 199 2 20 200 201 202 203 204 205 206 207 208 209 21 210</td>
</tr>
<tr>
<td>V7</td>
<td>211 212 213 214 215 216 217 218 219 22 220 221 222 223 224 225 226 227 228 229 23 230</td>
</tr>
<tr>
<td>V8</td>
<td>231 232 233 234 235 236 237 238 239 24 240 241 242 243 244 245 246 247 248 249 25 250</td>
</tr>
<tr>
<td>V9</td>
<td>251 252 253 254 255 256 257 258 259 26 260 261 262 263 264 265 266 267 268 269 27 270</td>
</tr>
<tr>
<td>V11</td>
<td>291 292 293 294 295 296 297 298 299 3 30 300 301 302 303 304 305 306 307 308 309 31</td>
</tr>
<tr>
<td>V12</td>
<td>310 311 312 313 314 315 316 317 318 319 32 320 321 322 323 324 325 326 327 328 329 33</td>
</tr>
<tr>
<td>V13</td>
<td>330 331 332 333 334 335 336 337 338 339 34 340 341 342 343 344 345 346 347 348 349 35</td>
</tr>
</tbody>
</table>
V2 a factor with levels Alfa Romeo Audi BMW BMW Alpina Brilliance Cadillac Chevrolet (EU) Chrysler Citroen Dacia Daewoo Daihatsu Dodge Fiat Ford Honda Hyundai Jaguar Jiangling Kia KIA Lada Lancia Land Rover Lexus Mazda Mercedes MG Mini Mitsubishi Modell Nissan Opel Peugeot Porsche Renault Rover Saab Seat Skoda smart SsangYong Subaru Suzuki Toyota Volvo VW

V3 a factor with levels Alfa Romeo 147 1.9 JTD 16V M-Jet Distinctive Alfa Romeo 159 1.9 JTDM 16V Distinctive Alfa Romeo 159 Sportwagon 1.9 JTDM 16V Distinctive (DPF) Alfa Romeo 166 2.4 JTD 20V Multijet Distinctive Alfa Romeo Brera 2.2 JTS 16V Skyview Alfa Romeo Brera 2.4 JTDM 20V Q-Tronic (DPF) Alfa Romeo GT 1.9 JTD 16V Multijet Progression Alfa Romeo GT 1.9 JTDM 16V Q2 Progression (DPF) Alfa Romeo GT 2.0 16V JTS 16V Distinctive Alfa Romeo Spider 2.2 JTS 16V Exclusive Alfa Romeo Spider 2.4 JTDM 20V Exclusive (DPF) Audi A2 1.4 TDI Audi A3 1.6 Attraction Audi A3 1.6 FSI Ambiente Audi A3 1.9 TDI Ambition (DPF) Audi A3 1.9 TDI Attraction Audi A3 1.9 TDI e Ambition (DPF) Audi A3 2.0 FSI Ambition Audi A3 2.0 TDI Attraction Audi A3 Cabriolet 1.8 TFSI Ambition Audi A3 Sportback 1.8 TFSI Ambition Audi A3 Sportback 2.0 TDI Attraction Audi A3 Sportback 2.0 TFSI Ambition S tronic (DSG) Audi A4 1.8 TFSI Ambition Audi A4 2.0 TDI Ambition (DPF) Audi A4 2.0 TDI (DPF) Audi A4 2.0 TDI Ambition (DPF) Audi A4 2.0 TFSI Audi A4 2.7 TDI Ambition multitrionic (DPF) Audi A4 3.0 TDI Ambition quattro (DPF) Audi A4 3.2 FSI Ambition quattro tiptronic Audi A4 Avant 1.8 TFSI Ambition Audi A4 Avant 2.0 TDI Audi A4 Avant 3.0 Audi A4 Avant 3.0 multitrionic Audi A5 3.0 TDI quattro (DPF) Audi A5 3.2 FSI multitrionic Audi A6 2.0 TDI (DPF) Audi A6 2.4 Audi A8 3.0 TDI quattro tiptronic (DPF) Audi A8 4.2 TDI quattro tiptronic (DPF) Audi Q7 3.0 TDI quattro tiptronic (DPF) Audi R8 Audi S3 Audi TT Roadster 1.8 T tiptronic Audi TT Roadster 2.0 TFSI BMW 116i BMW 118d BMW 118d (DPF) BMW 120d BMW 120d (DPF) BMW 120d Cabriolet (DPF) BMW 120i BMW 123d (DPF) BMW 123d Coup<e9> (DPF) BMW 125i Cabriolet BMW 130i BMW 135i Coup<e9> BMW 318d (DPF) BMW 318d touring (DPF) BMW 318i BMW 318ti compact Edition Lifestyle BMW 320Cd Cabriolet BMW 320Ci Cabriolet
BMW 320d (DPF) BMW 320d touring (DPF) BMW 325i Cabriolet Steptronic
BMW 325d (DPF) BMW 330Cd Coupé<e9> BMW 330d Cabriolet Steptronic (DPF)
BMW 330d Steptronic (DPF) BMW 330i BMW 330i Cabriolet BMW 330i SMG BMW 330i Steptronic
BMW 330xi touring Steptronic BMW 335d Coupé<e9> Steptronic (DPF) BMW 520d (DPF)
BMW 520i Steptronic BMW 525d (DPF) BMW 530d BMW 530d touring Steptronic (DPF)
BMW 530i BMW 535d Steptronic (DPF) BMW 630Ci Coupé<e9> Steptronic BMW 645Ci Cabriolet
BMW 730d Steptronic (DPF) BMW 745d Steptronic (DPF) BMW Alpina D3 (DPF)
BMW M3 Cabriolet M DKG BMW M3 Coupé<e9> BMW M5 touring SMG BMW X3 2.0d Steptronic (DPF)
BMW X3 2.5si Steptronic BMW X3 3.0d Automatik BMW X3 3.0sd Steptronic (DPF)
BMW X5 3.0d Steptronic BMW X5 3.0d Steptronic (DPF) BMW X5 4.4i Steptronic
BMW X6 xDrive35d Sport-Automatic (DPF) BMW Z4 Roadster 2.2i BMW Z4 Roadster 3.0i Steptronic
Brilliance BS6 2.0 Deluxe Cadillac BLS Wagon 1.9 TiD Sport Automatik (DPF)
Cadillac CTS 2.6 V6 Elegance Automatik Chevrolet (EU) Aveo 1.2 LS Chevrolet (EU) Captiva 2.0 D LT S
Chevrolet (EU) Captiva 3.2 LT 4WD Automatik (7-Sitzer) Chevrolet (EU) Epica 2.5 LT Automatik
Chevrolet (EU) HHR 2.4 LT Chevrolet (EU) Kalos 1.4 16V SX Chevrolet (EU) Matiz 0.8 LPG S (Autogasbetrieb)
Chevrolet (EU) Matiz 0.8 LPG S (Benzinbetrieb) Chevrolet (EU) Matiz 1.0 SX
Chevrolet (EU) Nubira 1.6 SX Chevrolet (EU) Nubira Wagon 1.8 LPG CDX (Autogasbetrieb) (LPG)
Chevrolet (EU) Nubira Wagon 1.8 LPG CDX (Benzinbetrieb) Chevrolet (EU) Nubira Wagon 2.0 D CDX (DPF)
Chevrolet (EU) Rezzo 2.0 LPG CDX (Autogasbetrieb) Chevrolet (EU) Rezzo 2.0 LPG CDX (Benzinbetrieb)
Chrysler 300C 5.7 V8 Automatik Chrysler Crossfire 3.2 V6 Automatik Chrysler Crossfire Roadster 3.2
Chrysler PT Cruiser Cabrio 2.4 Limited Chrysler Sebring 2.0 CRD Limited
Chrysler Sebring Cabrio 2.7 Limited Automatik Chrysler Sebring Cabrio 2.7 LX Automatik
Citroen Berlingo Kombi 1.4 Bivalent Multispace Plus (Benzinbetrieb) Citroen Berlingo Kombi 1.4 BIV
Citroen C-Crosser 2.2 HDi FAP Exclusive Citroen C1 1.0 Style Citroen C1 HDi 55 Style
Citroen C2 1.4 16V StopStart Sensodrive Citroen C2 1.4 VTR Sensodrive
Citroen C3 1.4 16V Stop Start Sensodrive Citroen C3 HDi 110 FAP Exclusive
Citroen C3 HDi 90 Exclusive Citroen C3 Pluriel 1.6 16V Exclusive Sensodrive
Citroen C4 1.6 16V Confort Citroen C4 Coupé<e9> 1.6 16V VTR Citroen C4 Coupé<e9> HDi 135 FAP VTR Plus
Citroen C4 HDi 110 FAP Confort Citroen C4 Picasso 1.8 16V Tendance Citroen C4 Picasso HDi 110 FAP Tendance
Citroen C5 2.0 16V Confort Citroen C5 HDi 110 FAP Style Citroen C5 HDi 135 FAP Confort Automatik
Citroen C5 HDi 135 FAP Exclusive Citroen C5 Kombi 1.8 16V Tendance Citroen C5 Kombi 2.0 16V Tendance Automatik
Citroen C5 Kombi HDi 110 FAP Tendance Citroen C5 Tourer HDi 110 FAP Confort
Citroen C6 HDi 170 Biturbo FAP Pallas Citroen C6 V6 HDI 205 Biturbo FAP Exclusive Automatik
Citroen C8 2.0 16V Tendance Citroen Grand C4 Picasso 1.8 16V Tendance
Citroen Grand C4 Picasso HDi 135 FAP Exclusive EGS6 Citroen Jumpy Kombi HDi 135 FAP Club lang (8-Sitze)
Citroen Xsara Kombi 1.4 HDi SX Citroen Xsara Picasso 2.0 16V Exclusive Automatik
Corvette C5 Cabrio Dacia Logan 1.4 Ambiance Dacia Logan 1.5 dCi Laur<e9>ate
Dacia Logan 1.6 Laur<e9>ate Dacia Logan MCV 1.5 dCi Laur<e9>ate Daewoo Evanda 2.0 CDX
Daewoo Lacetti 1.8 CDX Daewoo Matiz 1.0 SE Daewoo Nubira 1.8 CDX Daewoo Nubira Wagon 1.6 SX
daihatsu Copen Daihatsu Cuore 1.0 Top Daihatsu Materia 1.5 Daihatsu Sirion 1.3
daihatsu Trevis 1.0 Dodge Caliber 2.0 CRD SXT (DPF offen) Dodge Nitro 2.8 CRD SXT 4WD Automatik (DPF)
Fiat 500 1.3 JTD Multijet 16V Lounge (DPF) Fiat 500 1.4 16V Lounge Fiat 500 1.4 16V Sport
Fiat Bravo 1.4 T-Jet 16V Dynamic Fiat Bravo 1.9 JTD Multijet 8V Emotion (DPF)
Fiat Croma 1.9 JTD Multijet 16V Emotion Automatik (DPF) Fiat Doblo f2 Kombi 1.9 JTD Multijet 8V Dynamic
Fiat Grande Punto 1.2 8V Dynamic Fiat Grande Punto 1.3 JTD Multijet 16V Dynamic
Fiat Idea 1.3 JTD 70 Multijet 16V Dynamic Fiat Idea 1.9 JTD 100 Multijet 8V Emotion
Fiat Linea 1.4 T-Jet 16V Emotion Fiat Multipla 1.9 JTD Dynamic Fiat Multipla 1.9 JTD Multijet 8V Dynamic
Fiat Panda 1.1 8V Active Fiat Panda 1.2 8V Emotion Fiat Panda 1.2 8V Natural Power Panda (Benzinbetrieb)
Fiat Panda 1.2 8V Natural Power Panda Panda (Erdgasbetrieb)
Fiat Panda 1.3 JTD Multijet 16V Cross
Fiat Panda 1.3 JTD Multijet 16V Emotion
Fiat Panda 1.4 16V 100 HP
Fiat Punto 1.3 JTD 70 Multijet Dynamic
Fiat Scudo Kombi 140 Multijet DPF Panorama Executive lang
Fiat Stilo 1.2 16V Active
Fiat Stilo 1.9 JTD Multijet 16V Dynamic
Ford C-MAX 2.0 1.4 TDCi DPF Titanium
Ford Fiesta 1.4 TDCi Ghia Ford Fiesta 1.6 Trend
Ford Fiesta ST
Ford Focus 1.4 Trend
Ford Focus 1.6 Futura Ford Focus 1.6 TDCi DPF Ghia
Ford Focus 1.6 Ti-VCT Trend Ford Focus 1.8 FFV Ambiente (Ethanol-Betrieb)
Ford Focus 1.8 Style Ford Focus 2.0 Sport Ford Focus 2.0 TDCi DPF Titanium
Ford Focus C-MAX 1.8 Trend Ford Focus C-MAX 2.0 TDCi DPF Ghia Ford Focus C-MAX 2.0 TDCi DPF Trend
Ford Focus C-MAX 2.0 TDCi Trend Ford Focus Coupé <e9> - Cabriolet 2.0 TDCi DPF Titanium
Ford Focus Coupé <e9> - Cabriolet 2.0 Titanium Ford Focus Turnier 1.6 Ghia
Ford Focus Turnier 1.8 TDCi Futura Ford Focus Turnier 2.0 TDCi DPF Titanium
Ford Galaxy 2.0 Trend Ford Kuga 2.0 TDCi DPF Titanium 4x4 Ford Mondeo 1.6 Trend
Ford Mondeo 1.8 Futura Ford Mondeo 1.8 SCI Ambiente Ford Mondeo 2.0 TDCi DPF Ghia X
Ford Mondeo Turnier 2.0 TDCi DPF Titanium Ford Mondeo Turnier 2.0 TDCi Trend (5-Gang)
Ford Mondeo Turnier 2.0 Trend Ford Mondeo Turnier 2.5 Turbo Titanium X
Ford Ranger 2.5 TDCi Doppelkabine XLT Limited Ford S-MAX 2.5 Trend Ford Streetka 1.6 8V Elegance
Ford Tourneo Connect Kombi 1.8 TDCi kurz Honda Accord 2.0i Executive
Honda Accord 2.4 Executive Automatik Honda Accord Tourer 2.2 i-CTDi Sport
Honda Civic 1.3 DSI IMA Honda Civic 1.3 i-DSI i-VTEC IMA Honda Civic 1.4i LS
Honda Civic 1.6i ES Honda Civic 2.2 i-CTDi Type S (DPF) Honda CR-V 2.0i ES Automatik
Honda FR-V 1.8 Executive Automatik Honda FR-V 2.0i Executive Honda FR-V 2.2 i-CTDi Executive (DPF)
Honda FR-V 2.2i-CTDi Comfort Honda Jazz 1.4 ES Sport Honda Legend 3.5 V6 Automatik
Honda S2000 Honda Stream 2.0i Sport Hyundai Atos 1.1 Comfort Version Hyundai Coupé <e9> 2.0 GLS
Hyundai Elantra 2.0 GLS Hyundai Getz 1.1 GL Hyundai Getz 1.4 GLS Hyundai Getz 1.5 CRDi GLS
Hyundai Grandeur 2.2 CRDi Automatik (DPF) Hyundai Grandeur 3.3 V6 Automatik
Hyundai i10 1.1 Classic Hyundai i10 1.1 CRDi Style Hyundai i30 1.4 Comfort
Hyundai i30cw 1.6 Comfort Hyundai i30cw 1.6 CRDi Comfort (DPF) Hyundai Matrix 1.5 CRDi VGT GLS
Hyundai Sonata 2.4 GLS Automatik Hyundai Terracan 2.9 CRDi GLS Automatik
Hyundai Trajet 2.0 GLS Hyundai Tucson 2.0 CRDi GLS 4WD Hyundai XG 350 Automatik
Jaguar XJ6 3.0 V6 Automatik Jaguar XK R Coupe <e9> Automatik Jiangling Landwind 2.4 SC2 4WD
KIA Carens 2.0 CW EXKIA Carnival 2.9 CRDi EX (DPF offen) KIA Carnival II 2.9 CRDi EX
KIA cee'd 1.6 CRDi 115 EX (DPF) KIA cee'd 1.6 EXKIA cee'd 1.6 EXKIA cee'd 0.8d Sporty Wagon 1.6 CRDi 115 LX (DPF)
KIA Cerato 1.6 EXKIA Cerato 2.0 CRDi EXKIA Cerato 2.0 EXKIA Magentis 2.0 CRDi EX (DPF)
KIA Opelis 3.8 Automatik KIA Picanto 1.1 Cool KIA Picanto 1.1 EXKIA pro_ee=cde=d 2.0 CRDi TX (DPF)
KIA Rio 1.4 EX KIA Rio 1.5 CRDi EX Top KIA Rio 1.5 LS KIA Rio 1.6 EX Top
KIA Shuma ii 1.8 Active KIA Sportage 3.5 EX Automatik KlasseLada 1118 Kalina 1.6 8V
Lancia Musa 1.9 jtd Multijet 8v Oro Lancia Thesis 2.4 jtd Multijet 20v Emblema
Lancia Ypsilon 1.3 jtd Multijet 16v Argento D.F.N-System Lancia Ypsilon 1.3 jtd Multijet 16v Platinum
Lancia Ypsilon 1.4 16v OrO Land Rover Freelander Td4 SE (DPF) LEXUS GS 430 Luxury Automatik
Lexus GS 460 Luxury Automatik Lexus IS 220d Luxury (DPF) Lexus IS 250 Luxury Automatik
Lexus LS 600h L Ambience Wellness Automatik
Lexus RX 300 Executive Automatik Lexus RX 350 Executive Automatik Lexus RX 400h Luxury Automatik
Lexus SC 430 Automatik Mazda 2 1.3 Impression Mazda 2 1.4 Active Mazda 2 1.4 CD Active
Mazda 2 1.4 CD Independence Mazda 2 1.4 Exclusive Mazda 2 1.5 Impression
Mazda 2 1.6 Top Mazda 3 2.0 Top Mazda 3 Sport 1.4 Comfort Mazda 3 Sport 1.6 Exclusive
Mazda 3 Sport 1.6 Exclusive ActiveVatism Mazda 3 Sport 2.0 CD DPF Active
Mazda 5 1.8 Exclusive Mazda 5 2.0 CD DPF Top Mazda 5 2.0 Top Mazda 6 1.8 Comfort
Mazda 6 1.8 Exclusive Mazda 6 2.0 CD DPF Active Mazda 6 Sport 2.0 Exclusive Mazda 6 Sport 2.3 Top Mazda 6 Sport Kombi 2.0 CD Top (DPF) Mazda 6 Sport Kombi 2.0 Exclusive Mazda BT-50 XL-Cab Toplands Mazda CX-7 2.3 DISI Expression Mazda MPV 2.0 TD Comfort6 Mazda MX-5 1.9 Mazda MX-5 Roadster-Coupe 2.0 Expression Mazda RX-8 High-Power Revolution Mazda RX-8 STD-Power Revolution Mercedes A 160 CDI Classic Mercedes A 160 LR Elegance Mercedes A 170 Classic Mercedes A 180 CDI Avantgarde (DPF) Mercedes A 200 Classic Autotronic Mercedes B 150 Mercedes B 170 Mercedes B 200 CDI Autotronic (DPF) Mercedes B 200 Turbo Autotronic Mercedes C 180 Kompressor Elegance Mercedes C 180 Kompressor T-Modell Elegance Mercedes C 200 CDI T-Modell Classic Mercedes C 200 Kompressor Classic Mercedes C 200 Kompressor T-Modell Avantgarde Automatik Mercedes C 220 CDI Elegance (DPF) Mercedes C 220 T-Modell Avantgarde (DPF) Mercedes C 220 CDI T-Modell Elegance (DPF) Mercedes C 230 T-Modell Avantgarde Mercedes C 280 Avantgarde Mercedes C 320 CDI Elegance 7G-Tronic Mercedes CL 500 7G-Tronic Mercedes CL 500 Automatik Mercedes CLC 220 CDI (DPF) Mercedes CLK 220 CDI Coup<e9> Elegance (DPF) Mercedes CLK 270 CDI Coup<e9> Elegance Automatik Mercedes CLS 320 CDI 7G-Tronic (DPF) Mercedes CLS 63 AMG 7G-Tronic Mercedes E 200 Kompressor Classic Mercedes E 200 Kompressor T-Modell Elegance Mercedes E 220 CDI Classic (DPF) Mercedes E 220 CDI Elegance Automatik Mercedes E 220 CDI Elegance Automatik (DPF) Mercedes E 220 CDI T-Modell Elegance Mercedes E 280 CDI Elegance (DPF) Mercedes E 300 Bluetec Elegance 7G-Tronic Mercedes E 320 CDI Avantgarde Automatik Mercedes E 320 CDI Elegance 7G-Tronic (DPF) Mercedes E 320 CDI Elegance Automatik (DPF) Mercedes E 350 CGI Elegance 7G-Tronic Mercedes GL 320 CDI 4Matic (7G-Tronic) (DPF) Mercedes ML 270 CDI Automatik Mercedes S 320 CDI 7G-Tronic (DPF) Mercedes S 320 CDI Automatik (DPF) Mercedes S 350 7G-Tronic Mercedes S 350 Automatik Mercedes S 420 CDI 7G-Tronic (DPF) Mercedes SL 350 Sportmotor 7G-Tronic Mercedes SL 500 7G-Tronic Mercedes SLK 200 Kompressor Mercedes SLK 280 Mercedes SLK 350 Mercedes SLK 350 7G-Tronic Mercedes Sprinter Kombi 315 CDI kurz 3 Mercedes Viano 2.0 CDI Trend kompakt (DPF) Mercedes Viano 2.2 CDI Fun kompakt MG TF 135 Mini Cooper Mini Cooper Cabrio Mini Cooper D (DPF) Mini Cooper D Clubman (DPF) Mini Cooper S Mini Cooper S Clubman Mini One Cabrio Mini One Seven Mitsubishi Colt 1.5 DI-D Invite Mitsubishi Colt 1.5 DI-D Invite Automatik Mitsubishi Colt 1.5 Instyle Mitsubishi Colt CZ3 1.3 Invite Mitsubishi Grandis 2.0 DI-D Intense Mitsubishi Grandis 2.4 Intense Mitsubishi L200 2.5 DI-D Double Cab Intense Super Select 4WD Mitsubishi Lancer 1.6 Comfort Mitsubishi Lancer 1.8 MPI Invite Mitsubishi Lancer 2.0 DI-D Instyle (DPF) Mitsubishi Lancer Kombi 2.0 Sport Mitsubishi Outlander 2.0 DI-D Instyle (DPF offen Mitsubishi Outlander 2.2 DI-D Instyle (DPF)) Mitsubishi Pajero 3.2 DI-D Intense Automatik (DPF) Mitsubishi Pajero Classic 2.5 TD Nissan 350Z Coup<e9> Premium Pack Nissan 350Z Roadster Premium Pack Nissan Almera 1.5 acenta Nissan Almera Tino 1.8 acenta plus Nissan Almera Tino 2.2 dCi tekna Nissan Micra 1.2 Acenta Nissan Micra 1.2 visia Nissan Micra 1.2 Visia Nissan Micra 1.4 Acenta Sport Nissan Micra 1.5 dCi Tekna Nissan Micra C+C 1.6 Premium Nissan Navara Double Cab 2.5 dCi LE (DPF) Nissan Primera Traveller 1.9 dCi acenta Nissan Primera Traveller 2.0 tekna CVT-Automatik Nissan Qashqai 1.5 dCi DPF acenta 4x2 Nissan Qashqai 1.6 acenta 4x2 Nissan Qashqai 2.0 tekna 4x4 Nissan Tiida 1.6 tekna Nissan Tiida 1.8 tekna Nissan X-Trail 2.0 dCi LE 4x4 (DPF) Nissan X-Trail 2.5 Elegance 4x4 Nissan X-Trail 2.5 LE 4x4 CVT-Automatik Opel Agila 1.2 Edition Opel Agila 1.3 CDTI Enjoy Opel Antara 2.0 CDTI Edition (DPF) Opel Astra 1.3 CDTI (DPF) Opel Astra 1.4 Twinport Enjoy Opel Astra 1.6 Opel Astra 1.6 Twinport Enjoy Opel Astra 1.7 CDTI Edition (6-Gang) Opel Astra 1.8 Sport Opel Astra Caravan 1.7 CDTI Cosmo (DPF) Opel Astra Caravan 1.9 CDTI Elegance Opel Astra GTC 1.6 Turbo Cosmo Opel Astra GTC 1.9 CDTI Cosmo (DPF) Opel Combo Combi 1.7 CDTI Sport Opel Combo Tour 1.3 CDTI Opel Corsa 1.0 Twinport Catch Me Now Opel Corsa 1.0 Twinport Edition
Opel Corsa 1.2 Twinport Edition Opel Corsa 1.3 CDTI Cosmo Opel Corsa 1.7 DTI
Opel GT 2.0 TurboOpel Meriva 1.7 CDTI CosmoOpel Meriva 1.8 EnjoyOpel Signum 2.2 direct Cosmo
Opel Tigra TwinTop 1.3 CDTI Enjoy Opel Tigra TwinTop 1.4 Twinport Enjoy
Opel Vectra 1.9 CDTI DPF Elegance Opel Vectra Caravan 1.9 CDTI DPF Edition
Opel Vectra Caravan 2.2 direct Elegance Opel Vivaro Life 2.5 CDTIOpel Zafira 1.6 CNG Edition (Erdgasbetrieb)
Opel Zafira 1.7 CDTI Edition (DPF) Opel Zafira 1.9 CDTI Cosmo (DPF) Opel Zafira 1.9 CDTI Edition Aut.
Opel Zafira 2.2 DTI Edition Peugeot 1007 110 Sport 2-Tronic Peugeot 1007 75 Filou
Peugeot 1007 HDI FAP 110 Sport Peugeot 206 60 Grand Filou Peugeot 206 90 ESP Pr<e9>mium
Peugeot 206 CC 135 Roland Garros Peugeot 206 CC HDI FAP 110 Tendance
Peugeot 206 HDI <e9>co 70 Grand Filou Peugeot 207 110 Sport Peugeot 207 CC 150 THP Sport
Peugeot 207 CC HDI FAP 110 Sport Peugeot 207 HDI FAP 110 Platinum Peugeot 207 RC Cup
Peugeot 207 SW 75 Tendance Peugeot 307 110 Grand Filou Cool Peugeot 307 110 Tendance
Peugeot 307 Break HDI 90 Esplanade Peugeot 307 Break HDI FAP 135 Tendance
Peugeot 307 Break HDI FAP 90 Tendance Peugeot 307 CC 135 Peugeot 307 CC HDI FAP 135 Sport
Peugeot 307 HDI FAP 110 Platinum Peugeot 307 HDI FAP 135 Pr<e9>mium Peugeot 307 SW HDI FAP 110 Pr<e9>mium
Peugeot 308 120 VTi Sport Peugeot 308 HDI FAP 110 Sport Peugeot 308 SW 120 VTi Sport Plus
Peugeot 308 SW HDI FAP 135 Sport Plus Peugeot 4007 HDI FAP 155 Platinum (7-Sitzer)
Peugeot 406 Break HDI FAP 135 Pr<e9>mium Peugeot 406 Break HDI 140 Sport
Peugeot 406 Coup<e9> HDI FAP 135 Platinum Peugeot 407 115 Esplanade Peugeot 407 Coup<e9> V6 HDI FAP 140 Sport
Peugeot 407 HDI FAP 135 Tendance Peugeot 407 SW HDI FAP 135 Platinum
Peugeot 407 SW V6 HDI FAP 205 Bi-Turbo Platinum Automatik Peugeot 607 V6 HDI FAP 205 Bi-Turbo Platinum
Peugeot 807 HDI FAP 130 Tendance Peugeot 807 HDI FAP 135 Platinum Peugeot Partner Tepee HDI FAP 110
Porsche Boxster S 3.4 Porsche Cayman S 3.4 Renault Clio 1.2 16V Authentique
Renault Clio 1.2 16V Confort Authentique Renault Clio 1.2 16V Privile<e8>ge Quickshift
Renault Clio 1.5 dCi Confort Dynamique Renault Clio 1.5 dCi Dynamique
Renault Clio 1.5 dCi ESP Edition Dynamique Renault Clio 1.5 dCi Initi ale
Renault Clio 1.6 16V Edition Dynamique Renault Espace 2.0 dCi FAP Privile<e8>ge
Renault Grand Espace 3.0 dCi Initi ale Automatik Renault Grand Modus 1.5 dCi ESP FAP Dynamique
Renault Grand Sc<e9>nic 2.0 16V Emotion Renault Kangoo 1.6 16V Edition Campus
Renault Kangoo 1.6 16V Privile<e8>ge Renault Laguna 2.0 16V Expression
Renault Laguna 2.0 16V IDE Privile<e8>ge Renault Laguna 2.0 dCi FAP Dynamique
Renault Laguna Grandtour 2.0 16V Dynamique Renault Laguna Grandtour 2.0 dCi FAP GT
Renault Laguna Grandtour 2.0 dCi FAP Initi ale Renault M<e9>gane 1.5 dCi FAP Dynamique
Renault M<e9>gane Luxe Privil<e9>ege Renault M<e9>gane 1.6 16V Luxe Privil<e9>ege Renault M<e9>gane 1.9 dCi FAP Dynamique Renault M<e9>gane 2.0 16V Confort Privil<e9>ege
Renault Modus 1.2 16V Advantage Renault Modus 1.6 16V Dynamique Renault Sc<e9>nic 1.6 16V Dynamique
Renault Sc<e9>nic 1.6 16V Luxe Privil<e8>ge Renault Sc<e9>nic 1.6 dCi Luxe Dynamique
Renault Trafic Passenger 2.0 dCi 115 Privil<e8>ege Renault Twingo 1.2 16V Dynamique
Renault Twingo 1.2 16V TCE GT Renault Twingo 1.2 Authentique Renault Twingo 1.5 dCi Dynamique
Renault Vel Satis 3.0 dCi Privil<e8>ge Renault Vel Satis 3.5 V6 Initi ale Automatik
Rover 75 2.0 CDTi Celeste S Saab 9-3 1.8i Vector Saab 9-3 Cabriolet 1.9 TDi Aero (DPF)
Saab 9-3 Cabriolet 2.0t Vector Saab 9-3 SportCombi 1.9 TiD Vector (DPF)
Saab 9-3 SportCombi 2.0t Biopower Vector (Ethanol-Betrieb) Saab 9-3 SportCombi 2.0t Vector
Saab 9-5 SportCombi 1.9 TiD Vector (DPF) Saab 9-5 SportCombi 2.0t Biopower Vector (Ethanol-Betrieb)
Seat Altea 1.6 Reference Seat Altea 1.9 TDI Styline Seat Altea XL 1.8T FSI Styline
Seat Altea XL 2.0 TDI Styline (DPF) Seat Ibiza 1.2 12V Signo Seat Ibiza 1.4 TDi Signo
Seat Ibiza 1.8 20V T Cupra Seat Le<f3>n 2.0 TDI Styline (DPF) Seat Le<f3>n Cupra R
Seat Toledo 1.9 TDI Signo Seat Toledo 2.0 FSI Styline Seat Toledo 2.0 TDI Sport Edition DSG
Skoda Fabia 1.2 HTP 12V Comfort Skoda Fabia 1.4 16V Elegance Skoda Fabia 1.4 TDI Elegance (DPF)
Skoda Fabia 1.9 TDI RS Skoda Fabia Combi 1.4 TDI Ambiente (DPF) Skoda Fabia Combi 1.9 TDI Extra
Skoda Octavia 1.6 FSI Ambiente Skoda Octavia 1.9 TDI Ambiente Skoda Octavia Combi 1.8 Turbo LK
Skoda Octavia Combi 1.9 TDI Elegance Skoda Octavia Combi 2.0 FSI Elegance
Skoda Octavia Combi 2.0 TDI Elegance DSG (DPF) Skoda Octavia Combi RS TDI (DPF)
Skoda Roomster Scout 1.6 16V Skoda Superb 1.9 TDI Elegance Skoda Superb 2.0 TDI LK (DPF)
Skoda Superb 2.8 V6 Elegance Tiptronic smart forfour 1.5 cdci pulse smart forfour 1.5 passion softouch
smart forfour Brabus smart forfour cabrio 0.8 cdci passion (DPF offen)
smart forfour coup<e9> 0.8 cdci pure (DPF offen) smart forfour coup<e9> 1.0 mhd pulse
smart forfour coup<e9> 1.0 passion smart forfour coup<e9> 1.0 pure smart forfour coup<e9> 1.0 turbo passion
smart forfour coup<e9> passion SsangYong Actyon Sports 200 Xdi 4x4 Subaru B9 Tribeca 3.0R Exclusive
Subaru Forester 2.0X Exclusive Subaru Impreza 2.0R Comfort Subaru Justy 1.0 Active
Subaru Legacy 3.0R Comfort Navigation Automatik Subaru Legacy Kombi 2.0D Comfort (DPF)
Subaru Outback 2.5 ecomatic Comfort Automatik (Autogasbetrieb) (LPG) Subaru Outback 2.5 ecomatic
Suzuki Grand Vitara 1.9 DDiS Comfort+ (DPF) Suzuki Ignis 1.3 DDiS Comfort
Suzuki Ignis 1.5 Comfort Suzuki Jimny 1.5 DDiS Comfort Suzuki Jimny Cabrio 1.3 Classic
Suzuki Liana 1.4 DDiS Comfort Suzuki Liana 1.6 Comfort Suzuki Splash 1.0 Comfort
Suzuki Splash 1.2 Comfort Suzuki Swift 1300 Comfort Suzuki Swift 1300 DDiS Comfort (DPF)
Suzuki Swift 1600 Sport Suzuki Wagon R+ 1.3 DDiS Comfort Toyota Auris 1.6 Executive
Toyota Auris 2.0 D-4D Executive (DPF) Toyota Auris 2.2 D-4D Toyota Avensis 2.0 D-CAT Executive
Toyota Avensis 2.0 Executive Toyota Avensis Combi 2.0 D-4D Executive Toyota Avensis Combi 2.2 D-CAT Executive
Toyota Avensis Verso 2.0 Executive Toyota Aygo 1.0 City MMT Toyota Aygo 1.0 Club MMT
Toyota Celica 1.8 TS Toyota Corolla 1.4 D-4D Sol Toyota Corolla 1.6 Executive
Toyota Corolla 1.6 Sol Automatik Toyota Corolla 2.0 D-4D Sol Toyota Corolla Combi 1.6 Sol
Toyota Corolla Verso 1.8 Executive Toyota Corolla Verso 1.8 Executive (7-Sitzer)
Toyota Corolla Verso 1.8 lineae sol Toyota Corolla Verso 2.2 D-4D Executive (7-Sitzer) (DPF)
Toyota Corolla Verso 2.2 D-4D Executive Toyota Hilux 3.0 D-4D Double Cab Executive 4x4
Toyota Land Cruiser 3.0 D-4D Executive Automatik Toyota MR2 Roadster SMT
Toyota Previa 2.0 D-4D Sol Toyota Prius 1.5 Hybrid Executive Toyota Prius 1.5 Hybrid Sol
Toyota RAV4 2.0 Sol 4x4 Toyota RAV4 2.2 D-CAT Executive 4x4 Toyota Yaris 1.0 Sol MMT
Toyota Yaris 1.4 D-4D Executive Toyota Yaris 1.4 D-4D lineae sol Toyota Yaris TS
Volvo C30 1.6D Momentum (DPF) Volvo C30 1.8F Summum (Ethanol-Betrieb)
Volvo C30 D5 Summum Geartronic (DPF) Volvo S40 2.0D Summum (DPF) Volvo S40 2.4i Momentum
Volvo S60 D5 Comfort Volvo V50 1.6D Momentum (DPF) Volvo V50 2.0D Momentum
Volvo V50 2.0D Summum (DPF) Volvo V70 2.0D Summum (DPF) Volvo V70 2.4 Bi-Fuel CNG Momentum (Benzinbetrieb)
Volvo V70 2.4 Bi-Fuel CNG Momentum (Erdgasbetrieb) Volvo V70 T5 Summum
Volvo XC90 D5 Executive Geartronic (7-Sitzer) (DPF) VW Caddy Life 1.6
VW Caddy Life 1.9 TDI VW Caddy Life 1.9 TDI (DPF) VW CrossGolf 1.4 TSI
VW CrossGolf 1.4 TSI VW Eos 2.0 TDI (DPF) VW Fox 1.2 VW Fox 1.4 TDI
VW Golf 1.4 VW Golf 1.4 16V Trendline VW Golf 1.4 TSI Comfortline VW Golf 1.4 TSI Comfortline DSG (7-Gang)
VW Golf 1.4 TSI GT Sport VW Golf 1.6 FSI Comfortline VW Golf 1.9 TDI Comfortline
VW Golf 1.9 TDI Comfortline (DPF) VW Golf 1.9 TDI Sportline 4Motion VW Golf 2.0 FSI Comfortline
VW Golf BlueMotion Trendline (DPF) VW Golf GTI DSG VW Golf Plus 1.4 TSI Comfortline
VW Golf Plus 1.9 TDI Sportline VW Golf Variant 1.4 TSI Comfortline VW Jetta 2.0 TDI Comfortline
VW Lupo 1.4 VW New Beetle 1.9 TDI VW Passat 2.0 FSI Sportline VW Passat 2.0 TDI Highline DSG (DPF)
VW Passat 2.0 TDI Sportline (DPF) VW Passat BlueMotion (DPF) VW Passat CC 1.8 TSI
VW Passat Variant 2.0 5V Trendline VW Passat Variant 2.0 TDI Highline DSG (DPF)
VW Passat Variant 2.0 TDI Trendline (DPF) VW Passat Variant 2.0 Turbo FSI Highline Tiptronic
VW Passat Variant BlueMotion (DPF) VW Phaeton V6 Tiptronic (5-Sitzer)
VW Polo 1.2 VW Polo 1.4 FSI Highline VW Polo 1.4 TDI Trendline (DPF)
VW Polo 1.6 Trendline VW Polo 1.9 TDI Sportline VW Polo BlueMotion (DPF)
VW Polo Fun 1.4 TDI VW Polo Polo Limousine 1.4 Comfortline VW Sharan 2.0 TDI Trendline (DPF)
VW Sharan 2.8 V6 Highline Tiptronic VW T5 Caravelle 1.9 TDI Trendline kurz (DPF)
VW Tiguan 2.0 TDI Sport Style (DPF) VW Touareg V6 FSI Tiptronic VW Touareg V6 TDI (DPF)
VW Touareg V8 Tiptronic VW Touran 1.4 TSI Highline DSG VW Touran 1.9 TDI Trendline
VW Touran 1.9 TDI Trendline DSG (DPF) VW Touran 2.0 FSI Trendline VW Touran EcoFuel Trendline (Erdgasbetrieb)

V4 a factor with levels aktuell Kleinstwagen Kleinwagen Microwagen Mittelklasse Obere Mittelklasse Oberklasse Untere Mittelklasse
V5 a factor with levels Ja Nein Steuerklasse
V6 a factor with levels Euro3 Euro4 Motor
V7 a factor with levels Diesel Gas Hybrid kW Otto Wankel

V8 a factor with levels 100 103 104 105 106 107 108 110 112 114 115 118 119 120 121 122 124
125 126 127 128 129 130 131 132 133 135 136 140 141 142 143 145 147 149 150 153 155
160 162 165 169 170 171 173 175 176 177 180 190 191 194 195 196 200 203 206 208 210
215 217 225 228 232 235 240 242 245 250 253 255 280 285 306 309 327 337 373 38 40
43 44 45 46 47 48 49 50 51 52 54 55 56 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75
76 77 78 79 80 81 82 83 84 85 88 89 90 91 92 93 94 95 96 98 99 CCM

V9 a factor with levels 1086 1108 1120 1124 1149 1198 1206 1229 1240 1242 1248 1298 1308
1328 1332 1339 1348 1349 1360 1364 1368 1386 1388 1390 1396 1398 1399 1422 1461 1468
1490 1493 1495 1497 1498 1499 1500 1502 1504 1506 1508 1510 1512 1514 1516 1518 1519
1686 1689 1699 1749 1753 1769 1781 1793 1794 1796 1798 1799 1840 1870 1896 1910 1951
2184 2188 2198 2207 2212 2261 2351 2354 2359 2378 2387 2393 2399 2400 2401 2429 2435
2457 2463 2477 2488 2492 2494 2496 2497 2499 2500 2521 2597 2685 2698 2720 2736 2771
2777 2792 2902 2958 2967 2976 2979 2982 2987 2993 2995 2996 3000 3189 3195 3197 3199
3200 3222 3311 3342 3387 3456 3471 3497 3498 3597 3724 3778 3996 4134 4163 4172
4196 4293 4398 4423 4608 4966 4969 4999 5461 5654 5666 6208 659 698 796 799 989 995
996 998 999 Verbrauch

V10 a factor with levels 10.02 10.03 10.05 10.08 10.09 10.12 10.14 10.18 10.19 10.29 10.33
10.34 10.41 10.43 10.47 10.50 10.55 10.58 10.62 10.63 10.68 10.69 10.7 10.78 10.8
10.84 10.89 10.94 10.98 11.13 11.22 11.23 11.24 11.28 11.32 11.33 11.37 11.39 11.4
13.43 13.85 14.04 14.13 14.54 0.43 0.4 4.14 0.5 0.6 4.3 0.7 4.7 0.8 0.8
4.82 4.83 4.86 4.89 4.93 0.45 0.46 0.47 0.48 0.49 0.5 0.52 0.54 0.56 0.57 0.58 0.59 0.62 0.64 0.66
0.67 0.68 0.69 0.7 0.71 0.73 0.74 0.75 0.76 0.77 0.78 0.79 0.8 0.81 0.84 0.85
0.86 0.87 0.88 0.9 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 0.6 0.62 0.64 0.65
0.66 0.67 0.68 0.69 0.7 0.71 0.72 0.73 0.74 0.75 0.76 0.77 0.78 0.79 0.8 0.81 0.82 0.83 0.84
0.85 0.86 0.87 0.89 0.9 0.91 0.92 0.94 0.95 0.96 0.97 0.98 0.99 7.0 0.7 0.81 0.83 0.84
7.85 7.86 7.87 7.88 7.89 7.9 7.1 11.7 11.7 14.7 15.7 16.7 19.7 19.7 21.7 23.7 24.7 25.7 27
7.29 7.3 7.31 7.32 7.33 7.34 7.35 7.36 7.38 7.39 7.41 7.42 7.43 7.44 7.46 7.47 7.48
extracat

Categorical Data Analysis and Visualization

Description

This package offers a variety of functions that can be used for categorical data analysis or at least have to do with categorical data.

Among the most interesting features are

Source

www.adac.de

Examples

data(eco2plus)
## maybe str(eco2plus) ; plot(eco2plus) ...

extracat
RMB plots visualize contingency tables. The function offers different visualizations of conditional distributions and their corresponding weights (frequencies) including multiple barcharts, spineplots and piecharts. There are different ways of displaying the residuals from statistical models.

A static version of CPCP plots with labeling and color options.

The OPTILE interface was developed for the Google Summer of Code 2011. It offers a variety of reordering techniques for contingency tables. The reordering of the categories not only improves visualizations.

A matrix of RMB plots not unlike a Scatterplot matrix (SPLOM) is produced. It uses binning for continuous data.

After a hexagonal binning of x and y a third categorical target variable is displayed via piecharts (or embedded hexagons) within the hexagons. This avoids problems with overplotting.

A very fast implementation of the fechnerian scaling technique, which computes a fechnerian distance matrix from a (dis.)similarity matrix. The technique is equivalent to a shortest path algorithm.

Offers two- or multidimensional fluctuation diagrams and multiple barcharts.

Simulates categorical data arrays of any dimension. The number of observations, the number of block clusters and the noise level and type can be chosen.

Details

Package: extracat
Type: Package
Version: 1.6-4
Date: 2013-12-11
License: -
LazyLoad: yes

Author(s)

Alexander Pilhoefer
Department for Computer Oriented Statistics and Data Analysis
University of Augsburg
Germany

Maintainer: Alexander Pilhoefer <alexander.pilhoefer@math.uni-augsburg.de>

References

**facetshade**

**Description**

This function makes it possible to create ggplots using `facet_grid` with a plot of the complete data in the background of each facet. There are two options: If `geom` is specified then the background data is put into a separate layer. The original data is stored in the main object. The other option is to not specify a `geom`. In this case the modified data is stored in the main body. See examples.

**Usage**

```r
facetshade(data, mapping, f, geom, geom.mapping, bg.all = TRUE, keep.orig = FALSE, ...)
```

**Arguments**

- `data` (The dataframe used for the background plots in the first layer.
- `mapping` (The aesthetic mapping constructed via `aes`.
- `f` (The formula specifying the grid for `facet_grid` or a facet/wrap.
- `geom` (The geom used for the shade.
- `geom.mapping` (Aesthetics for the shade.
- `bg.all` (Whether or not to use all data points for each background plot. If `FALSE` then the data for the background is the complement of the data in the facet.
- `keep.orig` (Logical. Whether to keep the original faceting variables defined by `f`. Those are renamed by adding `orig.` as a prefix. For example `f = .~variable` will work fine with `group = orig.variable`. See FinCal example.
- `...` (Further arguments for the background layer or the main `ggplot` object.

**Value**

A `ggplot` object.

**Author(s)**

Alexander Pilhoefer

**See Also**

`facet_grid`
Examples

```r
# produces a modified data.frame mdata and returns:
# ggplot(data = mdata, mapping, ... ) + facet_grid(f)

require(scales)
require(ggplot2)

# facetshade object:
fs1 <- facetshade( data = olives, aes(x = palmitoleic, y = oleic),
                  f = .~Region )

# only the background-data
fs1 + geom_point( colour = alpha(1, 0.2) )

# the actual data added in a second layer:
fs1 + geom_point( colour = alpha(1, 0.2) ) +
   geom_point( data = olives )

# now again with colours:
fs1 + geom_point( colour = alpha(1, 0.2) ) +
   geom_point( data = olives, aes(colour = Area) )

# a different geom for the background-plot:
fs1 + geom_density2d(colour=alpha(1,0.1)) +
   geom_point( data = olives, aes(colour = Area) )

## Not run:
## OPTION 2: specify geom in facetshade call:
fs1b <- facetshade( data = olives, aes(x = palmitoleic, y = oleic),
                    f = .~Region , geom = geom_point)
fs1b + geom_point(aes(colour = Area))

## End(Not run)

# compare with complement:
fs2 <- facetshade( data = olives, aes(x = palmitoleic, y = oleic),
                  f = .~Region , bg.all = FALSE)

fs2 + geom_density2d(colour=alpha(1,0.1)) +
   geom_point( data = olives, aes(colour = Area) )

## Not run:
## OPTION 2: specify geom in facetshade call:
fs2b <- facetshade( data = olives, aes(x = palmitoleic, y = oleic),
                    f = .~Region , geom = geom_density2d, bg.all = FALSE)
fs2b + geom_point(aes(colour = Area))

## End(Not run)

# a second dataset:
## Not run:
```
```r
data(EURO4PlayerSkillsSep11, package="SportsAnalytics")
e4 <- subset(EURO4PlayerSkillsSep11, Attack > 0 & Defence > 0)

fs3 <- facetshade( data = e4, aes(x = Attack, y = Defence),
f = .~Position, compare.all = TRUE)

fs3 + geom_point( colour = alpha(1, 0.1) ) +
geom_point( data = e4, aes(colour = Position), alpha=0.3)

fs3 + geom_bin2d( colour = alpha(1, 0.1) ) +
geom_point( data = e4, aes(colour = Position), alpha=0.3)

# now with two facet variables
fs4 <- facetshade( data = e4, aes(x = Attack, y = Defence),
f = Position~Side, compare.all = TRUE)

fs4 + geom_point( colour = alpha(1, 0.1) ) +
geom_point( data = e4, aes(colour = Position))

## End(Not run)
## Not run:
library(FinCal)
sh13 <- get.ohlcs.google(symbols=c("AAPL", "GOOG", "IBM", "MSFT"),
start="2013-01-01", end="2013-12-31")

# OPTION 1 ------------
require(reshape2)
SH13 <- data.frame(date = as.Date(sh13$AAPL$date),
  sapply(sh13, "\[\]" , "close",USE.NAMES=TRUE))
names(SH13) <- c("date",names(sh13))
SH13[,-1] <- apply(SH13[,-1], 2, function(x) 100*x/x[1])
SH13am <- melt(SH13, id="date")

# OPTION 2 ------------
#SH13am <- do.call(rbind,
# mappty(function(z,y){
# data.frame(
# date = as.Date(z$date),
# value = 100*z$close/z$close[1],
# variable = y)
# } , z = sh13, y = names(sh13), SIMPLIFY = FALSE))
#----------------------

# original plot from GDAR:
ggplot(SH13am, aes(date, y=value, colour=variable,group=variable)) +
geom_line() + xlab("") + ylab("") +
theme(legend.position="bottom") +
theme(legend.title=element_blank())
```
# facetshade:
# compare to "average" of others:
facethade(SH13am, aes(x=date, y=value), f = ~variable, bg.all = FALSE) +
gem_smooth(aes(x=date, y=value), method="loess", span = 1/20) +
gem_line(data=SH13am, aes(colour=variable), show_guide=FALSE) +
lab(""") + ylab("")

# compare to all others
facethade(SH13am, aes(x=date, y=value),
    f = ~variable, bg.all = FALSE, keep.orig = TRUE) +
gem_line(aes(x=date, y=value, group=orig.variable), colour = alpha(1,0.3)) +
gem_line(data=SH13am, aes(colour=variable), show_guide=FALSE, size = 1.2) +
xlab(""") + ylab("")

# --- parallel coordinates --- #
sc <- scale(olives[,3:10])

# OPT: order by var
ord <- order(apply(sc,2,sd))
sc <- sc[,ord]

require(scales)
# OPT: align at median
sc <- apply(sc,2,function(z) rescale_mid(z, mid = median(z,na.rm=TRUE)))

require(reshape2)
require(ggplot2)

msc <- melt(sc)
msc$Area <- olives$Area

f1 <- facethade(msc, aes(x=Var2,y=value,group=Var1), f=~Area, bg.all = FALSE)
f1+geom_line(alpha=0.05)+
gem_line(data=msc, aes(colour=Area),alpha=0.2)+
faceth_wrap(f=~Area,nrow=3)

## End(Not run)
## Not run:
# TESTCODE: instead of creating a new object
# a shade layer is added to an existing ggplot
# NOTE: function CHANGES the object!

# highlighting + alpha
pp0 <- ggplot()+geom_point(data = olives,
aes(x = palmitoleic, y = palmitic), colour = 2) + faceth_wrap(~Area, ncol = 3)
extracat:::facethade2(pp0, alpha = 0.1, colour = 1)
# colours for both, alpha for shade
pl <- ggplot()+geom_point(data = olives, 
aes(x = palmitoleic, y = oleic, colour = Area)) + facet_grid(.~Region)
extracat:::facetshade2(pl, alpha = 0.1)

# different geom and colour for shade
pp2 <- ggplot()+geom_point(data = olives, 
aes(x = palmitoleic, y = oleic, colour = Area)) + facet_grid(.~Region)
extracat:::facetshade2(pp2, geom = geom_density2d, 
mapping = aes(colour = NULL), colour = ?)

# smooth over points shade with matching colours
pp3 <- ggplot()+geom_smooth(data = olives, 
aes(x = palmitoleic, y = oleic, colour = Region)) + facet_grid(.~Region)
extracat:::facetshade2(pp3, geom = geom_point, 
mapping = aes(colour = orig.Region), keep.orig = TRUE)

## End(Not run)

---

**fluctile**

**fluctuation diagrams**

---

**Description**

Create a fluctuation diagram from a multidimensional table.

**Usage**

fluctile(tab, dir = "b", just = "c", hsplit = FALSE, shape ="r", gap.prop = 0.1, 
border = NULL, label = TRUE, lab.opt = list(), add = FALSE, maxv = NULL, 
tile.col = hsv(0.1,0.1,0.1,alpha=0.6), bg.col = ifelse(add,NA,"lightgrey"), 
tile.border = NA, vp = NULL, ... )

**Arguments**

- **tab**: The table which is to be plotted.
- **dir**: The bar/rectangle direction: "v" and "h" stand for vertical or horizontal bars. "b" stands for "both" and leads to standard fluctuation diagrams with quadratic rectangles. Use "n" for a same-binsize-plot.
- **just**: A shortcut version of the argument used in grid for the anchorpoint of the rectangles: "rb" is equivalent to c("right", "bottom"), "t" is equivalent to "ct" or c("centre", "top") and so on. See examples.
fluctile

hsplit A logical for alternating columns and rows or a vector of logicals with TRUE for each variable on the x-axis.

shape Instead of rectangles ("r") it is possible to use circles ("c"), diamonds ("d") or octagons ("o"). The arguments dir and just work for rectangular shapes only.

gap.prop proportion of the gaps between the rows/columns within each block.

border The proportion of the space used for the labels.

label Whether or not to plot labels.

lab.opt A list with options for the labels. Currently lab.cex and abbrev work.

add Whether to create a new plot or add it to an existing one.

maxv The maximum value for the scale. Default is equivalent to maxv = max(x).

tile.col The color of the tiles.

tile.border The color for the tile border.

bg.col The background color in each cell.

vp An optional viewport to plot in. vp = c(i, j) can be used as a shortcut to viewport(layout.pos.row = i, layout.pos.col = j)

dots... Value

The viewport tree behind the graphic.

Note

This was part of the Google Summer of Code 2011.

Author(s)

Alexander Pilhoefer
Department for Computer Oriented Statistics and Data Analysis
University of Augsburg
Germany

See Also

mosaicplot

Examples

M <- arsim(1000, c(12,12), 3)
fluctile(M)

M2 <- optile(M)

# the standard fluctuation diagram with centralized rectangles
fluctile(M2)

# the standard fluctuation diagram with centralized octagons
fluctile(M2, shape = "o")

# another option such as it is used in iplots or MONDRIAN
# is to plot the rectangles in the bottom left corner
fluctile(M2, just = "lb")

# a multiple barchart
fluctile(M2, just = "b", dir = "h")

# or with vertical bars
fluctile(M2, just = "l", dir = "v")

# a same-binsize-plot
fluctile(M2, dir = "n")

require(MASS)
fluctile(xtabs(Freq~Type+Infl+Cont+Sat,data=housing),dir="h",just="b",
lab.opt=list(lab.cex=1))

A <- arsim(2000, c(6,6,4,4), 3, shuffle = FALSE, noise = 0.05)
fluctile(A)

## Not run:
# airport footprints: Unique Carrier vs. Destination
require(grid)
iata <- c("ATL", "BOS", "CLT", "DEN", "DFW", "DTW",
"EWR", "IAH", "LAS", "LAX", "MCO", "MSP", "ORD", "PHX", "SFO", "SLC")

mat.layout <- grid.layout(nrow = 4, ncol = 4, widths = 1/4, heights=1/4)
grid.newpage()
vp.mat <- viewport(layout = mat.layout)
pushViewport(vp.mat)

for(i in seq_along(iata)){
ap <- assign(iata[i],read.table(
paste("http://rosuda.org/lehre/SS09-f/datasets/air07s_",
"iata[i],".txt",sep=""),sep="\t",quote="",header=T))

 tt <- with(ap, table(UniCarri,Dest))

 jj <- ceiling(i/4)
 ii <- i - (jj-1)*4

fluctile(optile(tt,iter=100),wp=c(ii,jj),
lab.opt=list(rot=0,lab.cex=0.5),
border=c(0.1,.02,.02,.15),gap.prop=0.2)

pushViewport(viewport(layout.pos.row = ii, layout.pos.col = jj))
grid.text(iata[i],0.5,0.8,gp=gpar(col=2))
popViewport()
}
Description

Gene expression dataset.

Usage

data(GeneEx)

Format

A data frame with 7705 observations (genes) on the following 52 variables (samples).

CLID  clinical ID
NAME  a numeric vector
shee177 a numeric vector
shfa047 a numeric vector
shfs151 a numeric vector
shfa044 a numeric vector
shee146 a numeric vector
shee129 a numeric vector
shee118 a numeric vector
shee109 a numeric vector
shee100 a numeric vector
shet058 a numeric vector
shet057 a numeric vector
shet033 a numeric vector
shee045 a numeric vector
shco045 a numeric vector
shco044 a numeric vector
shco039 a numeric vector
shco038 a numeric vector
<table>
<thead>
<tr>
<th>GeneEx</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>shco031</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shco030</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shco029</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfe113</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfe085</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfe084</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfe081</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfe052</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shcz090</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shcz089</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shcz087</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shcz086</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shcz085</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shcz064</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shcz061</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shdj104</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shdj102</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shdj094</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shdj089</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shdj067</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfs240</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfs229</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfs227</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfs226</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfa109</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfa107</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shee216</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shet172</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shee198</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfa067</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfa066</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shee188</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>shfs159</td>
<td>a numeric vector</td>
</tr>
</tbody>
</table>

**Details**

See source.
getbw

Source

http://seurat.r-forge.r-project.org/

Examples

data(GeneEx)
## maybe str(GeneEx) ; plot(GeneEx) ...

getbw(x, k = NULL, min_n = NULL, warn = FALSE)

Description

Computes binwidth and breakpoints for a numeric or integer variable. The binwidth is a multiple of the minimal non-zero distance between two neighboring observations. The binwidth is chosen such that the number of active bins is as close as possible to a desired number \( k \). An active bin is a bin which contains at least \( \text{min}_n \) observations.

Usage

getbw(x, k = NULL, min_n = NULL, warn = FALSE)

Arguments

- **x**: A numeric variable.
- **k**: The desired number of active bins. A bin is active if it contains at least \( \text{min}_n \) observations. The default is \( k \leftarrow 1 + 2 \times \text{ceiling}(\log(N)/\log(2)) \).
- **min_n**: The minimum number of observations necessary for a bin to count as an active bin. Defaults to \( \text{min}_n = \text{max}(\log(N/10)/\log(10), 1) \).
- **warn**: I don’t want to know about problems.

Value

A vector of breakpoints with attributes "bw", "k" and "outlier". The first one is the binwidth, the second one is the number of active bins which is as close as possible to the specified parameter \( k \). "outlier" is a logical vector indicating which bins are not active.

Note

Experimental.

Author(s)

Alexander Pilhoefer
See Also

nclass.scott, cutbw, ahist

Examples

```r
require(scales)

hist(x <- rexp(200,1/10), breaks=gb<-getbw(x,24, min_n = 5, warn=TRUE),
    col = alpha(attr(gb,"outlier")+1,0.3))

hist(x <- rexp(2000,1/10), breaks=gb<-getbw(x,24, min_n = 5,warn=TRUE),
    col = alpha(attr(gb,"outlier")+1,0.3))

x <- rlnorm(1000,log(10),log(4))
x <- c(x, rnorm(500,400,30))

hist(x, breaks=gb<-getbw(x,24, min_n = 5,warn=TRUE),
    col = alpha(attr(gb,"outlier")+1,0.3))

# Not run:
bws1 <-replicate(1000,{
x <- rexp(200,1/10)
  gb <- getbw(x,20)
  attr(gb,"bw")
})
hist(bws1,breaks=getbw(bws1,30))

bws2 <-replicate(1000,{
x <- rnorm(200)
x <- x/rnorm(200)
  gb <- getbw(x,20)
  attr(gb,"bw")
})
hist(bws2,breaks=getbw(bws2,30))
```
getcolors

Create a color vector

Description

Creates a color vector using different palettes, e.g. from the colorspace package.

Usage

getcolors(N, palette, col.opt = list(), revert = FALSE)

Arguments

N Number of colors.
palette Palette shortcut:

"rgb", "hsv" RGB rainbow colors. See rainbow.
"hcl" HCL rainbow colors. See rainbow_hcl.
"s", "seq", "sqt", "sqn", "sequential" See sequential_hcl
"d", "div", "diverging", "diverge" See diverge_hcl
"h", "heat", "heatcolors" See heat_hcl
"t", "ter", "terrain" See terrain_hcl
getIs

\begin{itemize}
\item col.opt Options for the palette. See help pages for the specific palettes.
\item revert Logical. The color vector is returned in reverse order.
\end{itemize}

\textbf{Value}

A color vector.

\textbf{References}


\textbf{Examples}

\begin{verbatim}
## Not run:
require(MASS)
mat.layout <- grid.layout(nrow = 2, ncol = 2)
vp.mat <- viewport(layout = mat.layout)
pushViewport(vp.mat)
rbm(formula = ~Type+Infl+Cont+Sat, data = housing, col = "rgb", vp = c(1,1))
rbm(formula = ~Type+Infl+Cont+Sat, data = housing, col = "q17", vp = c(1,2))
rbm(formula = ~Type+Infl+Cont+Sat, data = housing, col = "hcl", vp = c(2,1))
rbm(formula = ~Type+Infl+Cont+Sat, data = housing, col = "seq", vp = c(2,2))
popViewport()

## End(Not run)
\end{verbatim}

\textbf{Description}

Basically this is an auxiliary function used by \texttt{heattile}: It extracts the cluster indices from a biclust object and optimizes the order of the rows and columns in the data matrix with respect to these clusters. Uses the Measure of Effectiveness as an optimization criterion. See \texttt{ME} and \texttt{optME}.

\textbf{Usage}

\begin{verbatim}
getIs(biclust, dim, nstart = 20, solver = "nn", adjust.dist = TRUE)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
\item \texttt{biclust} The biclust object.
\item \texttt{dim} The dimension of the matrix.
\item \texttt{nstart} Number of starting points for the TSP solver in \texttt{optME}.
\item \texttt{solver} The TSP solver to use with \texttt{optME}: See \texttt{solve_TSP}.
\item \texttt{adjust.dist} If \texttt{TRUE} the ME values used as a distance matrix for the TSP are slightly adjusted by adding a the hamming distance divided by a constant. This keeps identical cases together and also preserves the orders within such groups.
\end{itemize}
### Details
The algorithm first computes an indicator matrix for each cluster and then combines these matrices to a 3D table. Then for the first and the second dimension the category orders are optimized with respect to ME. The optimization is done via optME which uses a TSP solver.

### Value
The cluster indices with respect to the optimized row and column orders in form of a list. The optimized orders for the data matrix are attached as an attribute `attr(x, "orders")`.

### Author(s)
Alexander Pilhoefer

### See Also
heattile, getIs2

---

### getIs2

**ME reordering for biclust objects.**

### Description
Basically this is an auxiliary function used by heattile: It extracts the cluster indices from a biclust object and optimizes the order of the rows and columns in the data matrix with respect to these clusters. Uses the Measure of Effectiveness as an optimization criterion. See ME and optME.

### Usage
```r
getIs2(bic, dim, nstart = 20, solver = "nn", cpr = FALSE, cpc = TRUE, adjust.dist = FALSE)
```

### Arguments
- **bic**: The biclust object.
- **dim**: The dimension of the matrix.
- **nstart**: Number of starting points for the TSP solver in optME.
- **solver**: The TSP solver to use with optME: See solve_TSP.
- **cpr**: Whether or not to combine identical rows.
- **cpc**: Whether or not to combine identical columns.
- **adjust.dist**: If TRUE the ME values used as a distance matrix for the TSP are slightly adjusted by adding a the hamming distance divided by a constant. This keeps identical cases together (which is only necessary if cpr = FALSE or cpc = FALSE) and also preserves the orders within such groups.
Details

The algorithm first computes an indicator matrix for each cluster and then combines these matrices to a 3D table. Then for the first and the second dimension the category orders are optimized with respect to ME. The optimization is done via optME which uses a TSP solver.

The difference to getIs is that rows and columns which are identical with respect to the biclusters are combined before the optimization. This keeps identical categories together and also speeds up the algorithm considerably (depending on the TSP solver).

The TSP solver solver = "nearest_insertion" for instance is inefficient in this case since it has to add (identical) cases one by one.

Value

The cluster indices with respect to the optimized row and column orders in form of a list. The optimized orders for the data matrix are attached as an attribute attr(x, "orders").

Author(s)

Alexander Pilhoefer

See Also

heattile, getIs2

---

getpath Path extraction from quickfechner objects

Description

Uses the path.matrix to obtain the shortest paths of the quickfechner object.

Usage

gWithPath(fm, pm = NULL, from = 1, to = nrow(fm))

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fm</td>
<td>A Fechner matrix computed via quickfechner</td>
</tr>
<tr>
<td>pm</td>
<td>The path.matrix. Only necessary if the corresponding attribute in fm is missing.</td>
</tr>
<tr>
<td>from</td>
<td>Start index.</td>
</tr>
<tr>
<td>to</td>
<td>Final index.</td>
</tr>
</tbody>
</table>

Details

The path.matrix is defined as follows: The entry of the i-th row and j-th column is the index of the last node visited before j on the shortest path from i to j.
Value

A vector of indices defining the shortest path from i to j in the original matrix passed to the quick-fechner function.

Author(s)

Alexander Pilhoefer

Examples

#not a distance matrix, but a similarity matrix in some sense
cx <- 1-abs(cor(olives[-c(1,2,11)]))

cx2 <- quickfechner(cx)
getpath(cx2,from=1,to=5)

gsac

Description

The generalized sort-and-cut algorithm. Reordering via optile and partitioning via cfluctile are iteratively combined to a clustering algorithm.

Usage

gsac(x, nc = Inf, maxiter = 40, zero = TRUE, r0 = 0, force.cs = FALSE, force.rs = FALSE, resort = "complete", method = "Kendall", tau0 = 0.5, stack = "max", clean = TRUE, clean.Is = TRUE, cutoff = -20, ... )

Arguments

x
The data matrix.

nc
The desired maximum number of clusters. Useful to restrict the algorithm further.

maxiter
Maximum number of iterations. Depending on the partitioning method/threshold and the size of the matrix the number of steps can sometimes become very large.

zero
Each partition identifies a sparse part of the table. If zero = TRUE this part is set to 0 or r0 times the expectation from the marginals.

r0
A parameter controlling how sparse areas are handled. Usually left at zero, which means that sparse areas are zeroed.

force.cs
Logical. If TRUE clusters may not share columns.

force.rs
Logical. If TRUE clusters may not share rows.
resort  The reordering variation: "none" for no subtable reordering, "c" for restricted common reordering (the subtables share the orders), "s" for independent/unrestricted reordering.

method  Method used for the partitioning via cfluctile.

tau0  Threshold used for the partitioning via cfluctile.

stack  The rule which element (cluster) from the list to use next. "last" is the FILO and "first" uses FIFO. "max" and "min" mean that the largest / the smallest elements will be handled first.

clean  Whether or not to clean the results using setcover.

clean.Is  Whether or not to check the current list of non-finished clusters for redundancies.

cutoff  Clusters are pruned by removing rows and columns with an average residual below this value.

...  Further arguments passed to optile.

Details

The clusters are returned as a 2D list. The first element lists the clusters by their row indices vectors, the second element lists the corresponding column indices. This can be used with getIs and heattile.

Value

A 2D list: row and column indices of the clusters.

Author(s)

Alexander Pilhoefer

See Also

sortandcut, cfluctile

Examples

```r
## Not run:
ss <- sample(1:nrow(plants),500)
M <- t(as.matrix(plants[ss,-1]))

gs <- gsac(M, fun="IBCC", foreign=".Call")

heattile(M, Is = getIs2(gs, dim(M)), fluct = TRUE, hm.palette = 1)

## End(Not run)

# simulated example:
A <- arsim(3000,c(8,5),1)
B <- arsim(2000,c(7,6),1)
C <- arsim(4000,c(9,9),1)
```
M <- matrix(0, 16, 16)
M[1:8, 1:5] <- A
M[4:10, 6:11] <- B
M[8:16, 8:16] <- C

M <- as.table(optile(M, iter=20))
t0 <- 0.6

# no subtable reordering
test1 <- gsac(M, resort="none", method= "BCI", tau0=t0)
require(scales)
heattile(M, ls=test1, hm.palette=alpha(1, 0.8), shape="r",
        fluct = TRUE, label = c(TRUE,TRUE), bg.col=NA, lab.opt = list(rot=c(0,90)))

## Not run:
# unrestricted subtable reordering
test2 <- gsac(M, resort="s", method= "BCI", tau0=t0)
# common reordering
test3 <- gsac(M, resort="c", method= "BCI", tau0=t0)
# clusters do not share rows, columns, both

test4 <- gsac(M, resort="s", force.cs=TRUE, method = "BCI", tau0=t0)
test5 <- gsac(M, resort="s", force.rs=TRUE, method = "BCI", tau0=t0)
test6 <- gsac(M, resort="s", force.rs=TRUE, force.cs=TRUE, tau0=t0)

## End(Not run)

## Not run:
heattile(M, Is=test2, hm.palette=alpha(1, 0.8), shape="r",
        fluct = TRUE, label = c(TRUE,TRUE), bg.col=NA, lab.opt = list(rot=c(0,90)))

heattile(M, Is=test3, hm.palette=alpha(1, 0.8), shape="r",
        fluct = TRUE, label = c(TRUE,TRUE), bg.col=NA, lab.opt = list(rot=c(0,90)))

heattile(M, Is=test4, hm.palette=alpha(1, 0.8), shape="r",
        fluct = TRUE, label = c(TRUE,TRUE), bg.col=NA, lab.opt = list(rot=c(0,90)))

heattile(M, Is=test5, hm.palette=alpha(1, 0.8), shape="r",
        fluct = TRUE, label = c(TRUE,TRUE), bg.col=NA, lab.opt = list(rot=c(0,90)))

heattile(M, Is=test6, hm.palette=alpha(1, 0.8), shape="r",
        fluct = TRUE, label = c(TRUE,TRUE), bg.col=NA, lab.opt = list(rot=c(0,90)))

## End(Not run)
heattile  

Heatmap with Biclusters

Description

Draws a heatmap using fluctile as the workhorse and offers the possibility to add rectangles which visualize the biclusters.

Usage

heattile(x, biclust = NULL, Is = NULL, shape = "r", fluct = FALSE, gap.prop = 0, border = c(0.05, 0.03, 0.03, 0.05), label = c(TRUE, FALSE), lab.opt = list(abbrev = 24, lab.cex = 1, rot = 0), bg.col = "lightgrey", sym = FALSE, breaks = 20 + 10*sym, clust.col = NULL, clust.palette = "rgb", hm.palette = "div", clust.col.opt = list(), hm.col.opt = list(revert = TRUE))

Arguments

x  
A two-was data matrix.

biclust  
A biclustering object. The matrix is displayed in its original order.

Is  
Instead of biclust one can define the indices of the clusters as a list where each element represents a cluster and is itself a list of length 2 containing the row indices and the column indices respectively. getIs or getIs2 return such lists and the row and column orders are changed with respect to the clusters.

shape  
Shape of the tiles, see fluctile.

fluct  
Plots polygons whose sizes are proportional to their corresponding values, see fluctile. If FALSE (default) a colored same-binsize plot is produced.

gap.prop  
gaps between the tiles, see fluctile.

border  
plot margins, see fluctile.

label  
Whether or not to draw labels, see fluctile.

lab.opt  
Label options, see fluctile.

bg.col  
A background color, see fluctile.

sym  
Whether or not the colors should be on a symmetric scale around zero.

breaks  
The matrix entries are cut into intervals via fluctile. see fluctile. If FALSE (default) a colored same-binsize plot is produced.

clust.col  
A color vector for the cluster rectangles.

clust.palette  
If no colors are specified a palette is used to obtain them: Usually a quantitative palette is a reasonable choice, e.g. "rbg" for rainbow and "hcl" for rainbow_hcl. "seq" and "div" stand for sequential_hcl and diverge_hcl.

hm.palette  
The color vector for the heatmap or a color palette. Usually "seq" and "div" which stand for sequential_hcl and diverge_hcl respectively will make sense. "rgb" for rainbow and "hcl" for rainbow_hcl are useful if the matrix entries are categorical. "terrain" and "heat" are also available.

clust.col.opt  
Options for the cluster color palette. See col.opt for rmb.

hm.col.opt  
Options for the heatmap color palette. See col.opt for rmb.
## Value

TRUE

## Author(s)

Alexander Pilhoefer

## See Also

fluctile

## Examples

```r
## Not run:

ss <- sample(1:nrow(plants), 500)
M <- t(as.matrix(plants[ss, -1]))
M <- optME(M)
heattile(M, hm.palette = "seq")

require(biclust)

GE <- t(na.omit(GeneEx[,3:52]))

# draw a sample of 1000 genes
ss <- sample(1:ncol(GE),1000)

EY <- GE[,ss]
SEY <- scale(EY)

# compute sensible initial row and column orders:
require(seriation)
sl <- seriate(dist(SEY), method="GW")
s2 <- seriate(dist(t(SEY)), method="GW")

o1 <- get_order(sl,1)
o2 <- get_order(s2,1)

SEY <- SEY[o1,o2]

# A plaid model with row effects
b1 <- biclust(SEY, method=BCPlaid(), row.release=0.4, 
col.release=0.4, fit.model = y ~ m + a )

# index sets from b1
Is2 <- getIs(b1,dim(SEY), nstart = 1)
```
hexpie

**Hexagonal Binning and Piecharts**

**Description**

This function bins two continuous variables into a hexagonal grid and represents a third variable (which is usually a factor) via piecharts or nested hexagons within the bins. The main idea is to avoid overplotting and unfortunate effects that emerge from mixing up colors, e.g. with alpha-blending.

**Usage**

```r
hexpie(x, y = NULL, z = NULL, n = 24, shape = "hex", p.rule = "radial",
        decr.by.rank = NULL, freq.trans = I, alpha.freq = FALSE, col = "hcl",
        col.opt = list(), show.hex = TRUE, random = NULL, xlim = range(x),
        ylim = range(y), label.opt = list(), vp = NULL)
```

**Arguments**

- **x**
  The variable for the horizontal axis. Should be integer or numeric.

- **y**
  The variable for the vertical axis. Should be integer or numeric.

- **z**
  The target variable for the colors which is handled as a factor.

- **n**
  The number of bins into which x is divided. See `hexbin`.

- **shape**
  There are two possibilities: "hex", "hexagonal", and "h" lead to hexagonal representations and "pie", "piechart", "circular" and "c" lead to circular representations.

- **p.rule**
  This controls the rules for the representation of the relative frequencies of the target categories. For shape = "hex" this should be one of "rad", "radius", "radial" meaning that the probabilities are represented by the radii. For shape = "circular" it is also possible to create piecharts via "angular", "angles" or "ang".

- **decr.by.rank**
  Whether or not to sort the categories within each hexagon individually by their frequencies in decreasing order. Defaults to NULL for no reordering but may be either TRUE (decreasing order) or FALSE (increasing order).

- **freq.trans**
  A function which is used to rescale the total counts of the cells. `sqrt` is a common choice.
alpha.freq The frequencies may additionally be reflected in terms of the alpha values of the colors.

col The choice of a color palette. See `rmb` for further explanations.

col.opt Additional color options to replace the defaults. See `rmb` for further explanations.

show.hex Whether or not to draw the hexagons. Setting `col.opt = list(line.col.hex = NA)` leaves the lines out and draws the background only.

random If this is not NULL in each bin a random sample of \( n = \text{random} \) observations will be drawn (with replacement) from the corresponding data points. The resulting frequencies are then used to draw the piechart or hexagon. The main idea is to use `random = 1` with larger numbers of bins such as \( n = 120 \) and `show.hex = FALSE`.

xlim A vector of length 2 defining the x-limits e.g. computed via `ininterval`.

ylim A vector of length 2 defining the y-limits e.g. computed via `ininterval`.

label.opt Additional labeling options to replace the defaults. Not yet implemented.

vp A viewport to plot in, e.g. for conditional plots.

Value

invisible(TRUE)

Author(s)

Alexander Pilhoefer

See Also

`stat_binhex`, `hexbin`

Examples

data(olives)
x <- olives$oleic
y <- olives$linoleic
z <- olives$Region

# the default
hexpie(x,y,z)

## Not run:
# zooming in (transformation of the total number of obs in each bin)
hexpie(x,y,z, freq.trans=sqrt)

# circular shapes
hexpie(x,y,z, freq.trans=sqrt, shape="pie")

# classical piecharts
hexpie(x,y,z, freq.trans=sqrt, shape="pie", p.rule="angles")
# the total numbers of obs are reflected via alpha-blending,
# the grid is not shown and RGB colors are used
hexpie(x,y,z, freq.trans=sqrt, shape="hex", p.rule ="radial",
alpha.freq=TRUE, col ="rgb",show.hex=F)

hexpie(x,y,z, freq.trans=NULL, shape="hex", p.rule ="radial",
alpha.freq=TRUE, col ="rgb",show.hex=T)

require(ggplot2)
data(diamonds)
x2 <- diamonds$carat
y2 <- diamonds$price
z2 <- diamonds$color

# a standard plot with colors via ggplot2
qplot(x2,y2,colour=z2)

# the hexpie version
hexpie(x2,y2,z2,n=36)

# due to the few bins with the majority of observations
# it is sensible to zoom in
hexpie(x2,y2,z2,n=36,freq.trans=function(s) log(1+s))

# the same, but this time the central color is the most frequent one
hexpie(x2,y2,z2,n=36,freq.trans=function(s) log(1+s), decr.by.rank = TRUE)

# this way the difference is more obvious
# (although the color palette is better suited for ordinal target variables)

mat.layout <- grid.layout(nrow = 1 , ncol = 2 , widths = c(1/2,1/2), heights=1)
grid.newpage()
vp.mat <- viewport(layout = mat.layout)
pushViewport(vp.mat)

vp1 <- viewport(layout.pos.row = 1, layout.pos.col = 1)
pushViewport(vp1)

hexpie(x2,y2,z2,n=18,freq.trans=NULL,
decr.by.rank=NULL,col="div", vp = vp1)

vp2 <- viewport(layout.pos.row = 1, layout.pos.col = 2)
pushViewport(vp2)

hexpie(x2,y2,z2,n=18,freq.trans=NULL,
decr.by.rank=T,col="div", vp = vp1)
popViewport()

# random samples from the data (within bins) with many bins
# (takes some time)
require(scales)
grid.newpage()
**idat**

*indicator dataframe*

---

**Description**

Converts all factor variables in a dataframe to a set of binary variables.

**Usage**

```r
idat(x, allcat = FALSE, keep = "Freq")
```

**Arguments**

- **x**: dataframe
- **allcat**: Whether or not to keep all categories or leave the last one out.
- **keep**: Variables which are kept unchanged such as a frequency variable.

**Value**

dataframe

**See Also**

Burt, imat

**Examples**

```r
require(MASS)
idat(housing)
```
**imat**

*indicator variables*

**Description**

converts a single categorical variable into indicator variables

**Usage**

imat(x, allcat = TRUE)

**Arguments**

- x: A factor variable.
- allcat: Whether or not to keep all categories or leave the last one out.

**Value**

matrix

**Examples**

```r
require(MASS)
imat(housing$Type)
```

---

**innerval**

*Interval boundaries*

**Description**

This function computes the boundaries of an interval which is symmetric around the median and includes a given percentage of the data. If that's impossible due to ties the interval is chosen to minimize the squared difference between the desired percentage and the actual percentage of the observations included.

**Usage**

innerval(x, p = 0.95, data.points = TRUE)

**Arguments**

- x: A data vector.
- p: The percentage of observations inside the interval.
- data.points: Whether to return the most extreme data points within the interval or the interval boundaries.
Value

A vector with the lower and upper boundaries of the interval.

Author(s)

Alexander Pilhoefer

See Also

quantile

Examples

```r
x1 <- rnorm(200)
innerval(x1)
quantile(x1, c(0.025, 0.975))

x2 <- rexp(200)
innerval(x2, data.points = FALSE)
innerval(x2)
quantile(x2, c(0.025, 0.975))
```

---

**itab**

*Independence Table*

Description

Computes the independence table for a data table with non-negative entries. The entries of the independence table are defined by the multiplication of all corresponding marginal relative frequencies.

Usage

```r
itab(x)
```

Arguments

- `x` A data table of any dimension but with non-negative entries.

Value

A data table of the same dimension as the input table.

Author(s)

Alexander Pilhoefer
**Examples**

```r
A <- optile(arsim(4000, c(13, 17), 4, 0.1), iter=10)
fluctile(A)
fluctile(itab(A))
D <- A-itab(A)
G <- (D)^2 / itab(A)
fluctile(G, tile.col = c(2, 4)[1 + (D>0)])
```

---

**JBCI**  
*The Joint Bertin Classification Index*

**Description**
Computes the Joint Bertin Classification Criterion which uses joint independence as a reference for normalization.

**Usage**

```r
JBCI(x, r = 1)
```

**Arguments**

- `x`  
The 3D table with non-negative entries.
- `r`  
The index of the variable which is tested for joint independence of the other two.

**Details**

The `BCI` of a 3D table but instead of the total independence case the joint independence case is used for normalization. With an optimal reordering we have `JBCI(x) >= BCI(x)`.

**Value**

Numeric value in [0,1].

**Author(s)**

Alexander Pilhoefer

**See Also**

`BCI`, `CBCI`, `WBCI`
**Examples**

```r
## Not run:
A <- optile( arsim(144*5*20,c(12,12),6,0.1) , iter = 1000)

p1 <- 0.1 + runif(5)
p1 <- p1/sum(p1)

A2 <- apply(A,1:2,function(z) rmultinom(1,z,p1))
A2 <- optile(A2, iter = 1000,return.type="table")

BCI(A)
BCI(A2)

DA2 <- subtable(A2,1:3)
names(DA2) <- c("X","Y","Z","Freq")

rmb(~Y+Z+X,data=DA2)

JBCI(A2,3)

## End(Not run)
```

---

**kendalls**

*Kendalls Tau for a matrix*

---

**Description**

Computes Kendalls Tau for a two-way table or matrix.

**Usage**

```r
ekendalls(x)
```

**Arguments**

- `x` A two-way table or matrix.

**Details**

Kendalls tau is a rank-correlation coefficient.

**Value**

numeric between -1 and +1.

**Author(s)**

Alexander Pilhoefer
Examples

```r
M <- arsim(300, c(8, 8), 3)
kendalls(M)
kendalls(optile(M))
```

<table>
<thead>
<tr>
<th>ME</th>
<th>Measure of Effectiveness</th>
</tr>
</thead>
</table>

Description

Computes the measure of effectiveness for a table, a matrix or an array.

Usage

```r
ME(x)
```

Arguments

- `x`: A matrix, table or array.

Value

The ME value.

See Also

`optME`

Examples

```r
a <- arsim(2000, c(8, 9, 10), 3, 0.2)
ME(a)
a2 <- optME(a)
ME(a2)
```
## Description

US Convictions

## Usage

data(MJnew)

## Format

A data frame with 508 observations on the following 107 variables.


12/19/2003 12/20/2006 12/21/2001

Investigation. Type.s. a factor with levels Agent Provocateur, Informant Agent Provocateur, Informant, Sting Agent Provocateur, Sting Informant Informant, Sting Sting

Charges a factor with levels Air piracy, Air safety violations, Explosives, Murder/attempted murder of US officials, Threats against the president, Property damage, Trials of officials, Chemical weapons, Conscription of civilians
Air piracy, Air safety violations, Funding terrorists, Material support for terrorists, Threats against the president, Property damage, Trials of officials, Chemical weapons, Conscription of civilians
Air safety violations, Air safety violations, Attempted murder, Firearms violations, Weapons of mass destruction
Air safety violations, Bombings, Explosives, Terrorist attacks on mass transport
Air safety violations, Conspiracy to kill Americans, Firearms violations, Terrorist attacks on mass transport
Air safety violations, Immigration violations, Making false statements
Arms control violations, Criminal conspiracy, Firearms violations, Material support for terrorists, Threats against the president, Property damage, Trials of officials, Chemical weapons, Conscription of civilians
Arms control violations, Criminal conspiracy, RICO/racketeering, Arms control violations, Drug violations, Terrorism, Money laundering, Smuggling, Asset forfeiture, Fraud/financial crimes
Bombings, Explosives, Terrorist attacks on mass transport
Bribery, Contraband tobacco, Criminal conspiracy, Exploitation, Police corruption, Terrorism, Money laundering, Smuggling, Asset forfeiture, Fraud/financial crimes
Bomber, Explosives, Criminal conspiracy, Exploitation, Police corruption, Terrorism, Money laundering, Smuggling, Asset forfeiture, Fraud/financial crimes
Military sabotage, Murder/attempted murder of officials, Terrorism, Money laundering, Smuggling, Asset forfeiture, Fraud/financial crimes
Conspiracy to kill Americans, Explosives, Firearms violations, Funding terrorists, Material support for terrorists, Threats against the president, Property damage, Trials of officials, Chemical weapons, Conscription of civilians
Conspiracy to kill Americans, Murder/attempted murder of US officials, Weapons of mass destruction
Conspiracy to kill Americans, Killing Americans abroad, Material support for terrorists, Obtaining of material support
Conspiracy to murder, Firearms violations, Murder/attempted murder of US officials
Conspiracy to murder, Firearms violations, Murder/attempted murder of US officials, Weapons of mass destruction
Conspiracy to murder, Firearms violations, Seditious conspiracy
Conspiracy to murder, Kidnap or Maim overseas, Criminal conspiracy, Drug violations, Firearms violations, Immigration violations, Material support for terrorists, Threats against the president, Property damage, Trials of officials, Chemical weapons, Conscription of civilians
Conspiracy to murder, Kidnap or Maim overseas, Criminal conspiracy, Immigration violations, Material support for terrorists, Threats against the president, Property damage, Trials of officials, Chemical weapons, Conscription of civilians
Conspiracy to murder, Kidnap or Maim overseas, Criminal conspiracy, Material support for terrorists, Threats against the president, Property damage, Trials of officials, Chemical weapons, Conscription of civilians
Conspiracy to murder, Kidnap or Maim overseas, Criminal conspiracy, Material support for terrorists, Threats against the president, Property damage, Trials of officials, Chemical weapons, Conscription of civilians
Conspiracy to murder, kidnap, or maim overseas, Criminal conspiracy, Weapons of mass destruction
Conspiracy to murder, kidnap, or maim overseas, Criminal solicitation, Firearms violations, Immigration violations
Conspiracy to murder, kidnap, or maim overseas, Explosives, Making false statements, Material support for terrorists
Conspiracy to murder, kidnap, or maim overseas, Explosives, Material support for terrorists, Threats
Conspiracy to murder, kidnap, or maim overseas, Firearms violations, Material support for terrorism
Conspiracy to murder, kidnap, or maim overseas, Firearms violations, Material support for terrorism
Conspiracy to murder, kidnap, or maim overseas, Immigration violations, Making false statements, Material support for terrorism
Conspiracy to murder, kidnap, or maim overseas, Making false statements, Material support for terrorism
Conspiracy to murder, kidnap, or maim overseas, Material support for terrorists
Conspiracy to murder, kidnap, or maim overseas, Material support for terrorists, Terrorist conspiracy
Conspiracy to murder, kidnap, or maim overseas, Material support for terrorists, Weapons of mass destruction
Conspiracy to murder, Murder/attempted murder of US officials
Conspiracy to murder, Murder/attempted murder of US officials
Contraband tobacco, Criminal conspiracy, Drug violations, Obtaining missiles to destroy airplanes
Contraband tobacco, Criminal conspiracy, Fraud/financial crimes, Immigration violations, RICO/racketeering
Contraband tobacco, Criminal conspiracy, Stolen goods/counterfeits
Contraband tobacco, Fraud/financial crimes, Immigration violations, Material support for terrorism
Contraband tobacco, Fraud/financial crimes, Material support for terrorists, RICO/racketeering, Terrorism fundraising
Contraband tobacco, Material support for terrorists, RICO/racketeering, Stolen goods/counterfeits
Contraband tobacco, RICO/racketeering
Criminal conspiracy, Criminal conspiracy, Criminal conspiracy, Criminal conspiracy, Criminal conspiracy, Criminal conspiracy
Criminal conspiracy, Drug violations, Firearms violations, Material support for terrorists
Criminal conspiracy, Drug violations, Fraud/financial crimes, Immigration violations
Criminal conspiracy, Drug violations, Immigration violations, Making false statements, RICO/racketeering
Criminal conspiracy, Explosives, Immigration violations, Material support for terrorists, Plotting explosions
Criminal conspiracy, Explosives, Material support for terrorists, Receiving explosives
Criminal conspiracy, Export fraud, Funding terrorists
Criminal conspiracy, Firearms violations, Funding terrorists
Criminal conspiracy, Firearms violations, Making false statements, Plotting explosions
Criminal conspiracy, Firearms violations, Making false statements, Plotting explosions
Criminal conspiracy, Firearms violations, Material support for terrorists, Plotting expedition against friendly nation
Criminal conspiracy, Firearms violations, Plotting expedition against friendly nation
Criminal conspiracy, Fraud/financial crimes, Criminal conspiracy
Criminal conspiracy, Fraud/financial crimes, Immigration violations, Making false statements
Criminal conspiracy, Fraud/financial crimes, Immigration violations, Making false statements, RICO/racketeering
Criminal conspiracy, Fraud/financial crimes, Immigration violations, Making false statements, RICO/racketeering
Criminal conspiracy, Fraud/financial crimes, Making false statements
Criminal conspiracy, Fraud/financial crimes, Making false statements, RICO/racketeering
Criminal conspiracy, Fraud/financial crimes, Material support for terrorists, RICO/racketeering
Criminal conspiracy, Fraud/financial crimes, Obstruction of justice, RICO/racketeering
Criminal conspiracy, Fraud/financial crimes, RICO/racketeering
Criminal conspiracy, Hostage taking
Criminal conspiracy, Immigration violations, Making false statements
Criminal conspiracy, Immigration violations, Making false statements, Criminal conspiracy
Criminal conspiracy, Immigration violations, Material support for terrorists
Criminal conspiracy, Immigration violations, Material support for terrorists, RICO/racketeering
Criminal conspiracy, Immigration violations, Stolen goods/counterfeits
Criminal conspiracy, Making false statements, Material support for terrorists
Criminal conspiracy, Making false statements, Obstruction of justice
Criminal conspiracy, Making false statements, RICO/racketeering, Stolen goods/counterfeits
Criminal conspiracy, Making false statements, Tax violations
Criminal conspiracy, Material support for terrorists, RICO/racketeering, Tax violations
Criminal conspiracy, Material support for terrorists, Weapons of mass destruction
Criminal conspiracy, Obstruction of justice, RICO/racketeering
Criminal conspiracy, RICO/racketeering, Stolen goods/counterfeits
Criminal conspiracy, Stolen goods/counterfeits
Criminal solicitation, Explosives, Firearms violations, Material support for terrorists
Drug violations, Drug violations, Firearms violations, Terrorist conspiracy
Drug violations, Funding terrorists
Drug violations, Funding terrorists, Material support for terrorists, Obtaining missiles to destroy airplanes
Drug violations, Immigration violations
Drug violations, Material support for terrorists
Explosives, Explosives, Firearms violations, Material support for terrorists
Explosives violations, Stolen goods/counterfeits
Explosives violations, Terrorist conspiracy
Explosives, Funding terrorists
Explosives, Making false statements
Explosives, Material support for terrorists
Explosives, Seditious conspiracy
Explosives, Weapons of mass destruction
Explosives violations, Funding terrorists
Explosives violations, Making false statements
Explosives violations, Stolen goods/counterfeits
Explosives violations, Terrorism conspiracy
Explosives violations, Terrorist conspiracy
Explosives, Fraud/financial crimes
Explosives, Making false statements
Explosives, Making false statements
Explosives, Material support for terrorists
Explosives, Material support for terrorists
Explosives, Making false statements
Explosives, Fraud/financial crimes
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explosives, Terrorism conspiracy
Explo
<table>
<thead>
<tr>
<th>Convictions</th>
<th>Outcome</th>
<th>Accessory after the fact</th>
<th>Air piracy</th>
<th>Air safety violations</th>
<th>Explosives</th>
<th>Murder, attempted</th>
<th>Material support for terrorists</th>
<th>Threats against</th>
<th>Terrorism</th>
<th>Trade restrictions</th>
<th>Weapons of mass destruction</th>
</tr>
</thead>
</table>
Criminal solicitation, Extortion, Material support for terrorists
Drug violations, Immigration violations Drug violations, Material support for terrorists
Explosives, Firearms violations Explosives, Firearms violations, Material support for terrorists
Explosives, Funding terrorists Explosives, Material support for terrorists
Explosives, Material support for terrorists Seditious conspiracy Firearms violations
Firearms violations, Funding terrorists Firearms violations, Material support for terrorists
Firearms violations, Murder/attempted murder of US officials, Terrorist acts
Firearms violations, Seditious conspiracy Fraud/financial crimes Fraud/financial crimes, Funding terrorists
Fraud/financial crimes, Immigration violations Fraud/financial crimes, Immigration violations, Making false statements
Fraud/financial crimes, RICO/racketeering Funding terrorists Funding terrorists, Immigration violations
Funding terrorists, Immigration violations, Tax violations Funding terrorists, Material support for terrorists
Hostage taking Immigration violations Immigration violations, Making false statements
Immigration violations, Making false statements, Obtaining sensitive defense information
Killing Americans abroad Making false statements Making false statements, Material support for terrorists
Making false statements, Material support for terrorists RICO/racketeering
Making false statements, Obstruction of justice, Perjury Material support for terrorists
Material support for terrorists, Obtaining missiles to destroy airplanes
Material support for terrorists, Obtaining sensitive defense information
Material support for terrorists, RICO/racketeering Material support for terrorists, Terrorist attacks
Military sabotage Murder/attempted murder of US officials, Weapons of mass destruction
Obscenity Obstruction of justice Obtaining missiles to destroy airplanes, Weapons of mass destruction
Perjury Receiving terrorist training RICO/racketeering RICO/racketeering, Terrorist fundraising
Seditious conspiracy Stolen goods/counterfeits Tax violations Violating trade restrictions
Weapons of mass destruction

State a factor with levels Alabama Alaska Arizona Arkansas California Colorado Connecticut
Delaware District of Columbia Florida Georgia Illinois Indiana Iowa Kentucky
Louisiana Maine Maryland Massachusetts Michigan Minnesota Mississippi Missouri
Montana New Jersey New York North Carolina Ohio Oklahoma Oregon Pennsylvania
South Carolina Tennessee Texas Virginia Washington Wisconsin

Alleged.Affiliation a factor with levels Abu Sayyaf al-Aqsa Martyrs' Brigades \code{al-Barakat} \code{al-Badr}
Jemaah Islamiyah Khalistan Commando Force (KCF) Lashkar-e-Taiba Mujahideen-e-Khalq
Palestinian Islamic Jihad Revolutionary Armed Forces of Colombia (FARC)
Taliban Tamil Tigers Terrorist training camps in Afghanistan and Pakistan
United Self-Defense Forces of Colombia (AUC)

month a numeric vector
day a numeric vector

Bcriminal.conspiracy a numeric vector
Bfirearms.violations a numeric vector
Bimmigration.violations a numeric vector
Bmaking.false.statements a numeric vector
Bplotting.expedition.against.friendly.nation a numeric vector
Bcontraband.tobacco a numeric vector
Bstolen.goods.counterfeits a numeric vector
BFunding.terrorists a numeric vector
BMaterial.support.for.terrorists a numeric vector
BConspiracy.to.murder..kidnap..or.maim.overseas a numeric vector
BFraud.financial.crimes a numeric vector
BCriminal.solicitation a numeric vector
BConspiracy.to.murder a numeric vector
BMurder.attempted.murder.of.US.officials a numeric vector
BWeapons.of.mass.destruction a numeric vector
BAir.safety.violations a numeric vector
BAttempted.murder a numeric vector
BRICO.racketeering a numeric vector
BDrug.violations a numeric vector
BExplosives a numeric vector
BSedititious.conspiracy a numeric vector
BAir.piracy a numeric vector
BThreats.against.the.president a numeric vector
BTax.violations a numeric vector
BObtaining.sensitive.defense.information a numeric vector
BPerjury a numeric vector
B Terrorist.conspiracy a numeric vector
B Terrorist.attacks.on.mass.transport a numeric vector
BObstruction.of.justice a numeric vector
BConspiracy.to.kill.Americans a numeric vector
BObtaining.missiles.to.destroy.airplanes a numeric vector
BMilitary.sabotage a numeric vector
B Terrorist.fundraising a numeric vector
BKilling.Americans.abroad a numeric vector
BArms.control.violations a numeric vector
BBombings a numeric vector
BReceiving.terrorist.training a numeric vector
BAsset.forfeiture a numeric vector
BExport.fraud a numeric vector
BBribery a numeric vector
BMurder a numeric vector
BMurder.of.foreign.officials a numeric vector
BChemical.weapons a numeric vector
BObscenity a numeric vector
BSmuggling a numeric vector
BHosage.taking a numeric vector
BViolating.trade.restrictions a numeric vector
BExtortion a numeric vector
CFirearms.violations a numeric vector
CFunding.terrorists a numeric vector
CMaking.false.statements a numeric vector
CContraband.tobacco a numeric vector
CMaterial.support.for.terrorists a numeric vector
CStolen.goods.counterfeits a numeric vector
CCriminal.conspiracy a numeric vector
CFraud.financial.crimes a numeric vector
CRICO.racketeering a numeric vector
CDrug.violations a numeric vector
CExplosives a numeric vector
CAir.piracy a numeric vector
CAir.safety.violations a numeric vector
CThreats.against.the.president a numeric vector
CTax.violations a numeric vector
CObtaining.sensitive.defense.information a numeric vector
CImmigration.violations a numeric vector
CConspiracy.to.murder..kidnap..or.maim.overseas a numeric vector
CTerrorist.attacks.on.mass.transport a numeric vector
CWeapons.of.mass.destruction a numeric vector
CConspiracy.to.kill.Americans a numeric vector
CConspiracy.to.murder a numeric vector
CMurder.attempted.murder.of.US.Officials a numeric vector
CObtaining.missiles.to.destroy.airplanes a numeric vector
CMilitary.sabotage a numeric vector
CCriminal.solicitation a numeric vector
CSedititious.conspiracy a numeric vector
CPerjury a numeric vector
CTerrorist.fundraising a numeric vector
CObsrtuction.of.justice a numeric vector
CExtortion a numeric vector
CBombings a numeric vector
CKilling.Americans.abroad a numeric vector
olives

CAsset.forfeiture a numeric vector
CArms.control.violations a numeric vector
CMurder a numeric vector
CMurder.of.foreign.officials a numeric vector
CAccessory.after.the.fact a numeric vector
CObscenity a numeric vector
CAccessory.after.the.fact a numeric vector
CAccessories.after.the.fact a numeric vector
CReceiving.terrorist.training a numeric vector
CTerrorist.conspiracy a numeric vector
CHostage.taking a numeric vector
CTerrorist.conspiracy a numeric vector

Examples

data(MJnew)
## maybe str(MJnew) ; plot(MJnew) ...

---

olives  olive oil fatty acids

Description

Various fatty acid measurements.

Usage

data(olives)

Format

A data frame with 572 observations on the following 11 variables.

Area a factor with levels Calabria Coast-Sardinia East-Liguria Inland-Sardinia North-Apulia Sicily South-Apulia Umbria West-Liguria
Region a factor with levels North Sardinia South
palmitic a numeric vector
palmitoleic a numeric vector
stearic a numeric vector
oleic a numeric vector
linoleic a numeric vector
linolenic a numeric vector
arachidic a numeric vector
eicosenoic a numeric vector
Test.Training a factor with levels Test Training

Examples

data(olives)

optile

Reordering Categorical Data

Description

This function will take a categorical data object (data.frame, table, ftable, matrix, array) and optimize its category orders. Most of the implemented techniques aim for a (pseudo-) diagonalization of the data matrix or table. This improves graphical representations (e.g. by minimizing crossings in scpcp plots) and can also be useful to compute clusters (e.g. via cfluctile).

The function offers an interface which will by default return the same type of object that has been passed to the function such that it is possible to write myplot( optile(x) ) for an optimized version of myplot(x). It is possible to use custom reordering functions (as long as they meet the requirements, see details).

Usage

optile(x, fun = "BCC", foreign = NULL,
args = list(), perm.cat = TRUE, method = NULL, iter = 1,
freqvar = NULL, return.data = TRUE,
return.type = "data.frame", vs = 0, tree = NULL, sym = FALSE, ...)

## S3 method for class 'list'
optile(x, fun = "BCC", foreign = NULL,
args = list(), perm.cat = TRUE, method = NULL, iter = 1,
freqvar = NULL, return.data = TRUE,
return.type = "table", vs = 0, tree = NULL,
sym = FALSE, k = NULL, h = NULL, ...)

Arguments

x The categorical data of one of the following classes: data.frame, table, ftable, matrix, array
fun The optimization function. Currently available are: BCC, WBCC, CA, csvd, rmca, symtile, barysort and IBCC. For more information see details.
optile

where to find the optimization function `fun`. `NULL` for an R function or for instance ".Call" and ".C" for an external function. E.g. `barysort` needs `foreign = ".Call"`.

args

Further arguments which will be passed to `fun`.

perm.cat

A logical vector indicating which variables are reordered and which will remain untouched. For example `perm.cat = c(FALSE, TRUE)` means that only the second variable is reordered. Has no effect if `fun = "casort"`.

method

Either `NULL`, "joint" or "stepwise". `method = NULL` means that the whole data table is passed to `fun`. `method = "joint"` uses the Burt matrix instead of the whole table which only reflects two-way associations not unlike a covariance matrix. `method = "stepwise"` will repeatedly call `fun` for 2, 3, 4, and so on variables.

iter

Some optimizations depend on the initial category orders (e.g. "BCC" and "IBCC"). If `iter > 1` the optimization is repeated for `iter` random initial category orders and the best result is returned. In this case `fun` must return comparable values.

freqvar

The name of the frequency variable, if any.

return.data

Whether to return the data or just the new orders.

return.type

The class of the object which will be returned. Defaults to the input type.

vs

An optional version number. "WBCC" is currently equivalent to "BCC" and `vs = 1`.

tree

A list whose entries are either tree objects (e.g. from `hclust`) or the string "hc". If the i-th entry is a tree object, the i-th variable is the result from cutting the tree into `dim(x)[i]` clusters via `subtree`. "hc" will compute a hierarchical clustering for the rows and columns with arguments specified in `args`.

sym

If `fun` is BCC or IBCC it is possible to run a symmetric version of the algorithm by setting `sym = TRUE`.

k

A vector of integers specifying the numbers of clusters into which the tree objects shall be cut. See `subtree`.

h

Instead of a number of clusters `k` the height at which the dendrogram shall be cut can be specified. See `subtree`.

... dots

Details

The `optile` interface makes it possible to resort the categories in different representations of categorical data.

The most important points to know are

- The function by default returns the same type of object as was passed in the function call.
- It is possible to specify custom optimization functions via `fun` and `foreign`.
- The function is able to handle tree objects which specify a hierarchical tree graph on the categories.
- The function can pass either multidimensional tables, the corresponding Burt matrix (`method = "joint"`) or a hierarchical series of tables (`method = "stepwise"`) to the optimization functions.
How to add a custom optimization function:

It is possible to use custom functions for the optile interface as long as they meet the following requirements:

The function should have the form

```r
fun( data, dims, perm.cat, ...) or
foreign( "fun", data, dims, perm.cat, ...)
```

where `fun` is the name of the function and `foreign` is `".Call", ".C", ...

The function returns a vector of the new category orders (minus 1) and the resulting criterion, e.g.

```r
c( 0,2,4,1,3, 4,3,2,1,0,5,6, 0.7612 )
```

dims is a vector with the number of categories for each variable and `perm.cat` is a 0/1 vector which indicates whether or not to change the category order of a variable.

There are three possible types for the data argument of `fun` which can be set via method:

The argument `method` can be one of `null`, "stepwise" or "joint". The default method = `null` indicates that `fun` accepts a multidimensional table as for instance can be produced via `xtabs`.

If `method = "joint"` a Burt matrix is computed and passed to `fun` (c.f. Burt). For instance "fun=casort" uses this data representation.

If `method = "stepwise"` or `method = "sw"` passes `fun`, `data`, `foreign` as well as any `args` to a function called `steptile` which initially builds a 2-way table of the first pair of variables, passes it to `fun` and stores the computed category orders. Afterwards the other variables are added one by one. i.e. in a step for the k-th variable the function passes a k-way table to `fun` and a new category order for this variable is computed given the (already fixed) category orders of the variables 1 to k-1. This version is well suited for hierarchical visualizations like classical mosaicplots. A slightly different implementation which is not embedded in the optile framework but uses optile as its workhorse is `steptile`.

CURRENTLY AVAILABLE REORDERING FUNCTIONS:

"BCC" and "WBCC": minimize the Bertin Classification Criterion and the Weighted Bertin Classification Criterion. BCC is the number of observation pairs which are not fully concordant among all relevant observation pairs (pairs which differ in all variables). A pair of observations a and b is fully concordant if all entries in a are smaller than those in b or vice versa. Full concordance results in a so-called pseudo-diagonal. WBCC uses the Hamming distance between the observations as weights for the contradictions to such a diagonal form and also takes pairs within the same row or column into account.

"casort": computes a correspondence analysis (SVD) and sorts by the first coordinate vector of each dimension. For more than two dimensions Multiple CA based on the Burt matrix is used.

"rmca": Adopts the idea of CA for k > 2 dimensions without dropping information: For each dimension d = 1..k with categories d1...dr compute the scaled average k-1 dimensional profile sdd and perform an SVD of (sd1...sdr)-sdd. Like in correspondence analysis the first coordinate vector is used for the reordering.

"csvd": For each variable d in 1..k (iteratively) compute the cumulative sums over the multidimensional table for each variable except d. Transform this multidimensional table to an r x s matrix with r being the number of categories of variable d and s being the product of these numbers for all other variables. Resort the categories of variable d by the first coordinate vector of an SVD of that matrix. Repeat this procedure for all variables in turn until a stopping criterion is met. Idea: for any variable h /= d we have h1 < h2 < ... < hx due to the cumulative sums. Hence the current
order of the categories will (tend to) be the same as in the coordinates of the svd which means that the svd computes coordinates for variable d with respect to the current category orders of the other variables. The algorithm uses casort for an initial solution to start from.

"distcor": Two-way tables or matrices can also be optimized by means of the distance correlation. See wdcor.

"IBCC": Iteratively sorts the categories of one variable at a time. Therefore it computes the average over the remaining dimensions and scales the profiles of each category as well as the average profile. It then computes the classification criterion between each category profile and the average profile which results in one value per category. The categories are then sorted by this criterion. The procedure is very quick and yields good results in most cases. It strongly depends on the initial category orders as do the BCC or WBCC algorithms. This function is written in C which means that foreign = "Call" must be set. Alternatively it can be used to presort the data via the shortcut presort = TRUE but this is deprecated and not recommended.

"distcor": Two-way tables or matrices can also be optimized by means of the distance correlation. See wdcor.

"barysort": Uses the barycenter heuristic to minimize the number of crossings heuristically. The heuristic is fast and yields good results but only works for two dimensions. For multiple dimensions use either "IBCC" or steptile. In optile the barysort is implemented in C and therefore requires foreign = ".Call").

Value

The function returns the reordered data. The return type is by default the same as the input type but can be redefined via return.type.

Note

Some parts of the code have been developed for the Google Summer of Code 2011.

Author(s)

Alexander Pilhöfer
Department for Computer Oriented Statistics and Data Analysis
University of Augsburg
Germany

Examples

# simple simulated example
A <- arsim(2000, c(11, 13), 3, 0.3)

fluctile(A)
fluctile(optile(A))
fluctile(optile(A, iter = 100))
fluctile(optile(A, fun = "CA"))
fluctile(optile(A, fun = "barysort", foreign = ".Call"))

# simulated mv example
A2 <- arsim(20000, c(6,7,8,9), 3, 0.1)
sccpp(A2, sel="data[,1]"

scpcp(A3 <- optile(A2, iter=20), sel="data[,1]"

dev.new()
fluctile(A3)

## Not run:

----------- EXAMPLE I -----------

# ----- Cluster results from the Current Population Survey ----- #
data(CPScluster)
cpsX = subtable(CPScluster, c(5, 26, 34, 38, 39), allfactor=TRUE)

# joint and stepwise optimization of BCC
ss <- optile(cpsX, presort=TRUE, return.data=TRUE, method="joint")
ss2 <- optile(cpsX, presort=TRUE, return.data=TRUE, method="sw")

# original cpcp plot

cpcp(cpsX)

# cpcp for joint algorithm

cpcp(ss)

# cpcp and fluctuation for the stepwise algorithm
# (should be good for pcp plots and hierarchical plots)
fluctile(xtabs(Freq~., data=ss2[,4]))
cpcp(ss2)

# The multivariate algorithm
ss3 <- optile(cpsX, presort=TRUE, return.data=TRUE, method=NULL)
cpcp(ss3)

# cpcp for casort algorithm
ssca <- optile(cpsX, presort=FALSE, fun = "casort", return.data=TRUE, method="joint")
cpcp(ssca)

# cpcp for rmca algorithm results. works better for the dmc data
ssR <- optile(cpsX, presort=FALSE, fun = "rmca", return.data=TRUE, method=NULL)
cpcp(ssR)

# cpcp for csvd algorithm
ssC <- optile(cpsX, presort=FALSE, fun = "csvd", return.data=TRUE, method=NULL)
fluctile(xtabs(Freq~., data=ssC[,4]))
cpcp(ssC)

# cpcp for presort algorithm with 20 iterations
ssP <- optile(cpsX, presort=FALSE, fun = "IBCC",
return.data=TRUE, method=NULL, foreign = ".Call", iter=20)
cpcp(ssP)
### EXAMPLE II

```r
library(MMST)
data(wine)

swine <- scale(wine[,1:13])
kmd <- data.frame(wine$class, replicate(9, kmeans(swine, centers = 6)$cluster) )
kmd <- subtable(kmd, 1:10, allfactor = TRUE)
cpcp(kmd)

# there is a good joint order and hence the joint result is better than the stepwise
kmd2 <- optile(kmd, method = "sw")
kmd3 <- optile(kmd, method = "joint")
cpcp(kmd2)
cpcp(kmd3)
```

### EXAMPLE III

```r
library(biclust)
data(BicatYeast)

Dby <- dist(BicatYeast)
hc1 <- hclust(Dby, method = "ward")
hc2 <- hclust(Dby, method = "average")
hc3 <- hclust(Dby, method = "complete")

hcc1 <- cutree(hc1, k = 6)
hcc2 <- cutree(hc2, k = 6)
hcc3 <- cutree(hc3, k = 6)

km1 <- kmeans(BicatYeast, centers = 6, nstart = 100, iter.max = 30)$cluster

library(mclust)
mc1 <- Mclust(BicatYeast, G = 6)$class

clusterings <- data.frame(hcc1,hcc2,hcc3,km1,mc1)
clusterings <- subtable(clusterings, 1:5, allfactor = TRUE)

clusterings2 <- optile(clusterings, method = "joint")
clusterings3 <- optile(clusterings, fun = "casort")
cpcp(clusterings2)

# a fluctuation diagram of all but the avg. clustering
fluctile(xtabs(Freq~.,data=clusterings2[-,2]))
```
# compute agreement via Fleiss kappa in irr:
require(irr)
rawdata <- untableSet(clusterings2)
for(i in 1:5) levels(rawdata[,i]) <- 1:6
(kappam.fleiss(rawdata))
(kappam.fleiss(rawdata[,2]))

## Let's have a look at kmeans with 2:12 clusters
library(biclust)
data(BicatYeast)

cs <- NULL
for(i in 2:12) cs <- cbind(cs, kmeans(BicatYeast, centers=i,nstart=100)$cluster)

names(cs) <- paste("V",2:12,sep="")
ocs <- optile(cs,method="joint")
cpcp(ocs,sort.individual=TRUE)
# and set alpha-blending, show.dots = TRUE

# and with hierarchical clusterings
library(amap)
hc <- hcluster(BicatYeast)
for(i in 2:20) cs2 <- cbind(cs2, subtree(hc,k=i)$data)

names(cs2) <- paste("V",2:20,sep="")
cpcp(cs2,sort.individual=TRUE)
# and set alpha-blending to about 0.6, show.dots = TRUE, then
ss <- iset()
ibar(ss$6)
# and VIEW >> Set color (rainbow)
# Ideally the axes would be at a distance according to the heights of the cuts.
# e.g. for the first 12 clusters (after that there are some cuts at about the same height)

# the complete dendrogram doesn't look too attractive:
plot(hc)

# and plotting the top cuts only omits the information
# on how many cases are in each node or leaf
xcoords <- rev(tail(hc$height,11))
xcoords <- xcoords/max(hc$height)
ycoords <- data.matrix(ss[,20:30])
ycoords <- apply(ycoords,2,function(s){
y <- s - min(s)
y <- y/max(y)
return(y)
})
ycoords <- cbind(ycoords, as.integer(as.matrix(ss[,5])))
colv <- rainbow_hcl(6)
dev.new()
```r
par(mfrow=c(1,2))
plot(1,pch="", x1lim=c(0,1), ylim=c(min(xcoords)-.007,1))

apply(ycoords,1,function(s){
  points(x=s[-12], y=xcoords,
  points(x=s[-12], y=xcoords,pch=19, col = colv[s[12]]))
  lines(x=s[-12], y=xcoords, col = colv[s[12]])
})
hc$height <- hc$height/max(hc$height)
plclust(subtree(hc,12),hang=0.02)

# Example IV
library(biclust)
data(EisenYeast)
SEY <- scale(EisenYeast)

Dby2 <- dist(SEY)

hc1 <- hclust(Dby2, method = "ward")
hc2 <- hclust(Dby2, method = "complete")

hcc1 <- cutree(hc1, k = 16)
km1 <- kmeans(scale(EisenYeast), centers = 16, nstart = 20, iter.max = 30)$cluster
optile(table(hcc1, km1))

# Example V
library(biclust)
data(BicatYeast)
require(irr)

st <- Sys.time()
fk <- NULL
for(k in 3:8){
  test <- subtable(replicate(100,kmeans(BicatYeast, centers = k)$cluster),1:100)
  test <- optile(test, fun = "casort")
  test <- optile(test, method="joint")
  test <- untableSet(test)
  for(i in 1:100) levels(test[,i]) <- 1:k
  fk <- c(fk,kappam.fleiss(test)$value)
}
Sys.time()-st
```
plot(x = 3:8, y = fkc, type = "l", lwd = 2)

############# EXAMPLE VI #############
# hierarchical clustering #

# A list with hierarchical clustering objects:
require(ape)

hc1 <- hclust(t(plants[, -1]), method = "manhattan", link = "ward")
hc2 <- hclust(t(plants[, -1]), method = "manhattan", link = "complete")

hclist <- list(hc1, hc2)
tfluctile(optile(hclist, k = c(8, 8)))

# or a table with corresponding tree objects:

tt <- table(subtree(hc1, 12)$data, subtree(hc2, 8)$data)
tfluctile(optile(tt, tree = list(hc1, hc2)))

# only one tree object, the other variable is free:

tt <- table(subtree(hc1, 8)$data, kk <- kmeans(t(plants[, -1]), centers = 8)$cluster)
tfluctile(optile(tt, tree = list(hc1, NA)))

## End(Not run)

---

**optME**

**Optimizing ME**

**Description**

Computes optimal category orders for each dimension separately. Uses a TSP solver to achieve the best ME value.

**Usage**

```r
optME(x, dims = NULL, nstart = 1,
      solver = "nearest_insertion",
      return.table = TRUE, adjust.dist = FALSE)
```
**plants**

**Arguments**

- **x**: A matrix, table or array.
- **dims**: Which dimensions to reorder. The dimensions are reordered independently.
- **nstart**: The number of different starting points for the TSP solver. If `nstart` is greater or equal to the number of cities in a dimension, the solver uses each city once.
- **solver**: Should be one of "nn", "nearest_insertion", "cheapest_insertion", "farthest_insertion". See `solve_TSP`.
- **return.table**: Whether or not to return the optimized table. If FALSE only the new category orders are returned. If TRUE the new orders are attached to the table as an attribute "orders".
- **adjust.dist**: If TRUE a small proportion of the euclidean distances between the category profiles (e.g. rows) is added to the ME distance value. The idea is to keep identical profiles together which is otherwise not guaranteed, since the ME values can be identical even if the profiles aren’t.

**Details**

Each dimension is optimized separately via a TSP solver.

**Value**

The passed object as a table with optimized category orders.

**See Also**

- ME

**Examples**

```r
a <- arsim(2000,c(8,9,10),3,0.2)
ME(a)
a2<-optME(a)
ME(a2)
```

---

**plants**

**Plants**

**Description**

Binary state variables indicating which of more than 30000 plants grow in that state.

**Usage**

```r
data(plants)
```
Format

A data frame with 34781 observations on the following 70 variables.

V1 name
ab  a numeric vector
ak  a numeric vector
ar  a numeric vector
az  a numeric vector
ca  a numeric vector
cb  a numeric vector
cd  a numeric vector
cf  a numeric vector
ga  a numeric vector
hi  a numeric vector
id  a numeric vector
il  a numeric vector
in  a numeric vector
ia  a numeric vector
ks  a numeric vector
ky  a numeric vector
la  a numeric vector
me  a numeric vector
md  a numeric vector
ma  a numeric vector
mi  a numeric vector
mn  a numeric vector
ms  a numeric vector
mo  a numeric vector
mt  a numeric vector
ne  a numeric vector
nv  a numeric vector
nh  a numeric vector
nj  a numeric vector
nm  a numeric vector
ny  a numeric vector
nc  a numeric vector
plants

nd a numeric vector
oh a numeric vector
ok a numeric vector
or a numeric vector
pa a numeric vector
pr a numeric vector
ri a numeric vector
sc a numeric vector
sd a numeric vector
tn a numeric vector
tx a numeric vector
ut a numeric vector
vt a numeric vector
va a numeric vector
vi a numeric vector
wa a numeric vector
wv a numeric vector
wi a numeric vector
wy a numeric vector
al a numeric vector
bc a numeric vector
mb a numeric vector
nb a numeric vector
lb a numeric vector
nf a numeric vector
nt a numeric vector
ns a numeric vector
nu a numeric vector
on a numeric vector
pe a numeric vector
qc a numeric vector
sk a numeric vector
yt a numeric vector
dengl a numeric vector
fraspm a numeric vector

Source
http://archive.ics.uci.edu/ml/datasets/Plants
**qBCI**

*Quantile BCI*

**Description**

Bins numeric variables according to their quantiles and computes the Bertin Classification Index BCI. The `data.frame` method computes the multivariate qBCI and not the pairwise values (c.f. `cmat`).

**Usage**

```r
qBCI(x, ...)  
## Default S3 method:  
qBCI(x, y, p = NULL, k = 5, iter=20, ...)  
## S3 method for class 'data.frame'  
qBCI(x,p = NULL, k = 5, sort = TRUE, iter=20, ...)
```

**Arguments**

- `x` A numeric vector (in this case `y` needs to be specified) or a `data.frame` with numeric or factor variables.
- `y` A numeric vector.
- `p` A percentage to use for the quantiles sequence. See details.
- `k` A minimum expected number of observations in each cell after the binning.
- `sort` Whether or not to compute the BCI for the optimized tables or not. If not, `kendalls` is usually a better alternative.
- `iter` An optile parameter.
- `...` `dots`

**Details**

The breakpoints for the binning are the data quantiles according to equidistant probabilities `seq(0,1,p)` where `p` is minimal under the condition that each cell has an expected number of observations of at least `k`.

**Value**

A value between 0 and 1.

**Author(s)**

Alexander Pilhoefer

**See Also**

`BCI`, `kendalls`, `wdcor`, `cmat`
Examples

```r
## Not run:
qBCI(rnorm(100), runif(100))

# non-functional relationship:
x1 <- runif(500, 0, 10)
x2 <- runif(500, 0, 10)
y1 <- x1 + rnorm(500, sd = 1)
y2 <- 10 - x2 + rnorm(500, sd = 1)
x <- c(x1, x2)
y <- c(y1, y2)
plot(x, y, pch = 19)
wdcor(x, y)
1 - qBCI(x, y)

y1 <- x1 + rnorm(500, sd = 0.1)
y2 <- 10 - x2 + rnorm(500, sd = 0.1)
x <- c(x1, x2)
y <- c(y1, y2)
plot(x, y, pch = 19)
wdcor(x, y)
1 - qBCI(x, y)

# or a quadratic curve:
test <- sapply(seq(0, 4, 0.2), function(s){
x <- runif(200, -1, 1)
y <- 5 + 12 * x^2 + rnorm(200, sd = s)
return(c(cor(x, y),
wdcor(x, y),
1 - qBCI(x, y)))})

plot(test[3,], type = "l", ylim = c(-0.2, 1))
lines(test[1,], col = 2, lwd = 2)
lines(test[2,], col = 3, lwd = 2)

## End(Not run)
```
quickfechner  fechnerian scaling

Description

This function computes a fechnerian distance matrix from either a similarity matrix or a dissimilarity matrix. In addition to the basic procedure which looks for the shortest paths between the objects in the dissimilarity matrix a second approach is offered which connects similarities in a multiplicative manner.

Usage

quickfechner(x, x.type = "diss", scale = "-", path.op = "+", sym.op = "*",
rescale = FALSE, exclude.zero = FALSE, check = TRUE)

Arguments

x      A similarity or dissimilarity matrix.

x.type  The type of the matrix ('sim' or 'diss').

scale  Either divide the similarities by the diagonal entries ('div', '/', '*', 'exp', 'expected', 'mult', 'multiplicative' or subtract the diagonal entries in the dissimilarity matrix ('-', '+', 'add', 'additive')

path.op  Whether to use the similarities to find multiplicative paths ('*', 'exp', 'expected', 'mult', 'multiplicative' or to use the dissimilarities and find additive paths ('+', 'add', 'additive', 'max', 'maximum').

Note that similarity matrices are simply converted to dissimilarity matrices by subtraction from 1. Other transitions such as 2M/(1+M) are not yet implemented and have to be done by hand.

sym.op  This sets the function which is used to ensure symmetry. "min" uses the minimum value, "+", "sum" or "mean" use the sum. "none", NA or FALSE stand for no operation and hence the resulting matrix will not necessarily be symmetric.

rescale  Whether or not the original diagonal will be used for a correction of the results.

exclude.zero  If TRUE zero-entries are not considered in the updating algorithm.

check  Whether or not to check for regular minimality or maximality.

Details

The algorithm first computes a dissimilarity matrix with a zero-diagonal. Then it iteratively tries to find shorter paths between the items.

Value

The Fechnerian distance matrix.

Author(s)

Alexander Pilhoefer
**Examples**

```r
data(olives)
# not a distance matrix, but a similarity matrix in some sense
cx <- 1-abs(cor(olives[-c(1,2,11)]))

cx2 <- quickfechner(cx)
rownames(cx2) <- names(olives)[-c(1,2,11)]
plot(hclust(as.dist(cx2)))

dm <- matrix(runif(100),10,10)
dm <- dm+t(dm)
diag(dm) <- 0
dm2 <- quickfechner(dm)

dmS <- 1-dm/max(dm)
dmS2 <- quickfechner(dmS, x.type="sim", path.op = "*")

## Not run:
# check triangular inequality:
extracat::trinq(dm)
extracat::trinq(dm2)
extracat::trinq(dmS2)

## End(Not run)
```

---

**Description**

Checks whether or not a matrix fulfills the regular maximality or minimality condition.

**Usage**

```r
regmax(x)
regmin(x)
```

**Arguments**

- `x` A symmetric data matrix.

**Value**

boolean

**Author(s)**

Alexander Pilhoefer
Examples

```r
x <- replicate(20, rnorm(20))
cx <- abs(cor(x))
regmax(x)
regmin(x)

diag(cx) = runif(20)
regmax(x)
regmin(x)
```

---

**rmb**

Multiple Barchart for relative frequencies and generalized Spineplots

**Description**

The `rmb` function basically produces a Multiple Barchart for the relative frequencies of some target categories within each combination of the explanatory variables. The weights of those combinations (that is the absolute frequencies) are represented in the total width of the corresponding barchart. The result is a graphic which allows to read the conditional target distributions exactly from the graphic without losing the information about the importance (in the sense of the number of observations) of the different combinations.

Additionally the `rmb` function allows to draw spineplots instead of the barcharts within each explanatory combination. On that score it can be seen as a generalization of Spineplots.

**Usage**

```r
## S3 method for class 'formula'
function(formula, data, col.vars = NULL, spine = FALSE, circular = FALSE, eqwidth = FALSE, cat.ord = NULL, cut = NULL, innerval = 1, freq.trans = NULL, num.mode = FALSE, max.scale = 1, use.na = FALSE, expected = NULL, residuals = NULL, model.opt = list(), gap.prop = 0.2, gap.mult = 1.5, col = "hcl", col.opt = list(), label = TRUE, label.opt = list(), vp = NULL, ...)```

```r
## S3 method for class 'ftable'
function(x, col.vars = NULL, spine = FALSE, circular = FALSE, eqwidth = FALSE, cat.ord = NULL, freq.trans = NULL, max.scale = 1, use.na = FALSE, expected = NULL, residuals = NULL, model.opt = list(), gap.prop = 0.2, gap.mult = 1.5, col = "hcl", col.opt = list(), label = TRUE, label.opt = list(), vp = NULL, ...)
```

**Arguments**

- `x`  Either a table or a model of class "glm" and family "poisson" or "binomial".

A table must be either of class `table` or of class `ftable`. The latter also implicitly defines the order in which the variables will be added to the plot. The arguments `formula` and `data` will be omitted. Please note that the model based version is still beta and will be improved in a future release.
The formula specifying the variables in their given order with the last variable being the target variable. The left hand side defines a weighting variable. If the weights are frequencies in a variable called "Freq" this is detected automatically if no other variable is defined.

data
The dataset as a data.frame or ftable.

col-vars
Logical vector with split directions where TRUE stands for horizontal splitting. The last (target) variable is always arranged on the x-axis.

spine
If TRUE a spineplot will be drawn instead of each barchart. This is recommended for binary target variables.

circular
If TRUE a piechart will be drawn instead of each barchart. spine is set to FALSE.

eqwidth
If TRUE the bar length of the multiple barchart in the background no longer restricts the width of the barcharts/spineplots for the relative frequencies of the target variable.

cat.ord
A vector specifying the categories of the target variable which will be visualized in the specified order. The default is to use all categories.

cut
Numeric variables will be cut into this number of intervals. May also be a vector with specifications for each variable.

innerval
The function innerval is used to reduce numeric variables to an interval which is symmetric around the median contains the specified proportion of observations (or as close to this as possible).

freq.trans
This parameter allows to transform the absolute frequencies used for the underlying multiple barchart. Possible values are "log", "sqrt" or list("sqrt",k). The latter stands for the k-th root transformation.

num.mode
In the numeric mode the gaps are removed and axes typical for numeric variables are drawn. Ignored for factor variables.

max.scale
The maximum value of the probability (y-axis) scale for each combination. Unsurprisingly the default is 1. The axis will be drawn if yaxis is TRUE.

use.na
If TRUE missing values will be changed to a level "N/A" and else (which is the default) the function na.omit will be called to reduce the dataset to complete cases only.

expected
There are three possibilities how to specify this parameter:

1. A list of integer vectors denoting the interaction terms in the poisson or proportional odds model, e.g. list( c(1,2,3), c(1,4) ) for all interactions between variables 1,2 and 3 as well as between 1 and 4.
2. A logical indicating whether or not to use a model (logit independence model).
3. A vector with expected values, e.g. from a model. If residuals remains undefined the response residuals will be plotted.

If undefined or set to FALSE only the observed values will be plotted.

residuals
If expected is a vector with expected values it is also possible to specify residuals. This is used internally by rmb.glm.
A list with optional parameters for model specifications. Possible parameters are:

model.opt
use.expected.values: A logical specifying whether or not to use the frequencies predicted by the model instead of the observed frequencies.

mod.type: Either "poisson" or "polr". See `glm` and `polr`.

resid.type: "pearson", "deviance", working, partial or "response". For polr models only the latter is available.

resid.display: One of "values", "color" or "both". "values" will result in bars or wedges for both expected and observed frequencies. Hence the raw residuals are shown in the graph. "color" will set the col argument aside and use colors on a red-blue-scale to represent (pearson) residuals. "both" does both.

max.rat: If a model is specified and resid.display = "both" the x-scales will not be reduced to less than 1/max.rat: The x-scales are reduced whenever an observed frequency exceeds the maximal scale.

gap.prop: The maximum proportion of the total plot width which is used for the gaps.

gap.mult: The incremental multiplier for the gaps of different dimensions. The gaps corresponding to any one variable are gap.mult times larger than those corresponding to the next variable on the same axis.

col: Either a vector defining the colors of the bars or a name specifying a palette: "hsv" and "rgb" for hsv-based rainbow colors, "hcl" for hcl-based rainbow colors (default), "div" or "diverge" for hcl-based diverging colors and finally "seq" or "sequential" for hcl-based sequential colors. Additional arguments can be specified via the col.opt argument according to the underlying functions in the colorspace package, e.g. `rainbow_hcl`. For the hsv-based colors see `rainbow`. Specifying a color or palette has no effect if an expected model is defined.

col.opt: Further options for the color palettes. See e.g. `rainbow_hcl` or `rainbow`. Other parameters are:

    col2 for the color of the background/weight bars,
    line.col for the color of all lines (bars, rectangles),
    bg for the background color of the whole graphic,
    bgs for the background color of each tile

label: Either a logical specifying whether or not to draw labels or a numeric vector defining which variables shall be labelled.

label.opt: A list with optional parameters for label specifications. Possible parameters are:

    yaxis: If TRUE a vertical axis will be drawn at both sides of the plot.
    This is recommended when changing the max.scale argument.
    boxes: Should the labels be surrounded by boxes?
    lab.tv: Should the target variable be included in the labeling?
    varnames: Should the variable names be shown as labels?
    abbrev: An single integer value or a vector of integer values specifying the number of characters to which the labels will automatically be abbreviated.
lab.cex  The fontsize multiplier.

vp             An optional viewport to plot in. vp = c(i, j) can be used as a shortcut to
             viewport(layout.pos.row = i, layout.pos.col = j)
... further arguments. Usually not necessary.

Details

A similar way to regard the graphic is the following: A Multiple Barchart of the explanatory vari-
ables is drawn with bars in horizontal direction. Then within each of the resulting bars a barchart
of the conditional distribution of the target variable is drawn with bars in vertical direction.

Value

invisible(TRUE)

Author(s)

Alexander Pilhoefer
Department for Computer Oriented Statistics and Data Analysis
University of Augsburg
Germany

References

Alexander Pilhoefer, Antony Unwin (2013). New Approaches in Visualization of Categorical Data:

See Also

mosaicplot

Examples

require(MASS)
  # simple example
  rmb(formula = ~Type+Infl+Cont+Sat, data = housing, gap.mult = 2,
      col.vars = c(FALSE,TRUE,TRUE,FALSE), label.opt = list(abbrev = 3))

  # with sqrt-transformation and horizontal splits only
  rmb(formula = ~Type+Infl+Cont+Sat, data = housing, gap.mult = 2,
      col.vars = c(TRUE,TRUE,TRUE,TRUE), freq.trans = "sqrt",
      label.opt = list(abbrev = 3))

  # a generalized spineplot with the first category highlighted
  rmb(formula = ~Type+Infl+Cont+Sat, data = housing, spine = TRUE,
      cat.ord = 1, mult = 2, col.vars = c(1,3,4),
      freq.trans = list("sqrt",3), label.opt = list(abbrev = 2))
## Not run:

# a generalized spineplot with all categories highlighted
# in a changed order
rmb(formula = -Type+Infl+Cont+Sat, data = housing, spine = TRUE,
cat.ord = c(3,1,2), gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
freq.trans = "sqrt", label.opt = list(abbrev = 3))

# the barchart version only for categories 1 and 3
rmb(formula = -Type+Infl+Cont+Sat, data = housing,
cat.ord = c(1,3), gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
freq.trans = "sqrt", label.opt = list(abbrev = c(4,1,1,1)))

# with equal widths
rmb(formula = -Type+Infl+Cont+Sat, data = housing, eqwidth = TRUE,
gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
label.opt = list(abbrev = 2, lab.tv = TRUE))

# ----- models and residuals ----- #

# using the logistic model: Sat by Type only

# residual shadings and expected values
rmb(formula = -Type+Infl+Cont+Sat, data = housing,
gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
label.opt = list(abbrev = 3), expected = list(c(1,2,3),c(1,4)),
model.opt = list(use.expected.values = TRUE, resid.display = "color") )

# residual values without shadings
rmb(formula = -Type+Infl+Cont+Sat, data = housing,
gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
label.opt = list(abbrev = 3), expected = list(c(1,2,3),c(1,4)),
model.opt = list( resid.display = "values") )

# residual shadings and expected values
rmb(formula = -Type+Infl+Cont+Sat, data = housing,
gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
label.opt = list(abbrev = 3), expected = list(c(1,2,3),c(1,4)),
model.opt = list(use.expected.values = TRUE, resid.display = "color") )

# barcharts with residual shadings and values
rmb(formula = -Type+Infl+Cont+Sat, data = housing,
gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
label.opt = list(abbrev = 3), expected = list(c(1,2,3),c(1,4)) )

# spineplots with residual shadings and values
rmb(formula = -Type+Infl+Cont+Sat, data = housing, spine = TRUE,
gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
label.opt = list(abbrev = 3), expected = list(c(1,2,3),c(1,4)) )

# piecharts with residual shadings and values
rmb(formula = -Type+Infl+Cont+Sat, data = housing, circular = TRUE,
gap.mult = 2, col.vars = c(TRUE,TRUE,TRUE,TRUE),
label.opt = list(abbrev = 3), expected = list(c(1,2,3),c(1,4)) )
# ---- using an ftable to create the plot ---- #
tt = xtabs(Freq~Type+Cont+Infl+Sat, data = housing)
ft = ftable(tt, col.vars= c(1:4))
rmb(tt)
rmb(ft)

# ---- using a glm model ---- #
mod1 <- glm(Freq ~ Type+Infl+Cont + Type*Sat, data = housing, family = poisson)
rmb(mod1, circular = TRUE,
   gap.mult = 2, col.vars = c(TRUE,FALSE,TRUE,TRUE),
   label.opt = list(abbrev = 3), model.opt = list(use.expected.values = TRUE) )

# ---- the numeric mode and cuts ---- #
data(olives)
# only three cuts to show how it works
rmb(~palmitoleic+stearic+Region, data = olives, cut = c(3,3,0))

require(ggplot2)
data(diamonds)
diamonds$lprice <- log(diamonds$price)
# a minority of extreme observations mess the display up:
rmb(~depth+table+lprice, data = diamonds, eqwidth = TRUE, spine = TRUE,
cut = c(36,36,5), col = "seq", num.mode = TRUE)

# we can zoom in via innerval:
rmb(~depth+table+lprice, data = diamonds, circular = TRUE,
cut = c(36,36,5), col = "div", innerval = 0.95,
num.mode = TRUE, freq.trans ="log")

   # price, carat and color
diamonds$lprice <- log(diamonds$price)
diamonds$lcarat <- log(diamonds$Carat)
rmb(~lcarat+lprice+color, data = diamonds,
cut = c(24,24,0), col = "rgb", num.mode = TRUE,
freq.trans="sqrt", eqwidth=TRUE, max.scale=0.5)

## End(Not run)

---

**rmbmat**

**Pairwise RMB-Plots**

**Description**

This function generates a matrix with RMB-plots of all pairs of variables with a specified target variable. Both categorical and numerical variables are accepted and the latter will be binned. This makes the graphic useful for a mixture of variable types and the binning avoids overplotting and color mash as it occurs in (colored) scatterplots of large datasets.
**Usage**

```r
rmbmat(x, tv, cut = 20, freqvar = NULL, plot.tv = FALSE, num.mode = TRUE,
  mode = "circular", eqwidth = FALSE, freq.trans = "sqrt", innerval = 1,
  allocation = 1, max.scale = 1, use.na = FALSE, expected = FALSE,
  model.opt = list(), gap.prop = 0.2, gap.mult = 1.5, col = "hcl", col.opt = list(),
  label = FALSE, label.opt = list(), diag.opt = list(), lower.opt = list(),
  upper.opt = list(), rc.opt = list(), factor.opt = list(), ...)
```

**Arguments**

- **x**
  Anything that can be converted to a `data.frame` via `as.data.frame`.

- **tv**
  The index of the target variable. The target variable will not be plotted unless `plot.tv` is `TRUE`.

- **cut**
  The number of intervals into which numeric variables will be cut.

- **freqvar**
  An optional frequency variable. "Freq" is handled automatically.

- **plot.tv**
  Whether or not to include the target variable(s) in the plot.

- **num.mode**
  Whether or not to use the numeric mode (no gaps and a numeric axis) for numeric variables.

- **mode**
  One of "circular", "pie", "piechart", "p" or "c" for piecharts, "spine" or "s" for spineplots, "bars", "bar" or "b" for barcharts.
  NOT YET IMPLEMENTED: "rect" or "r" for nested rectangles. "nested.circles" are abbreviated by "nc" or "ncircles".

- **eqwidth**
  See `rmb`.

- **freq.trans**
  See `rmb`.

- **innerval**
  See `rmb`.

- **allocation**
  The widths and heights for the plots are proportional to `allocation(nlevels(x))`.

- **max.scale**
  See `rmb`.

- **use.na**
  See `rmb`.

- **expected**
  See `rmb`.

- **model.opt**
  See `rmb`.

- **gap.prop**
  See `rmb`.

- **gap.mult**
  See `rmb`.

- **col**
  See `rmb`.

- **col.opt**
  See `rmb`.

- **label**
  See `rmb`.

- **label.opt**
  See `rmb` and details.

- **diag.opt**
  A list with `rmb` parameters. These overwrite the general parameters for all plots on the diagonal.

- **lower.opt**
  The same as `diag.opt` but for the lower triangular matrix. Additionally it is possible to define a second target variable, e.g. `lower.opt = list(tv2 = 3, ...)`.
upper.opt  The same as diag.opt but for the upper triangular matrix. Additionally it is possible to define a second target variable, e.g. upper.opt = list(tv2 = 3, ...).

rc.opt  A list with which it is possible to define parameters for single matrix cells (plots), columns or rows. This will overwrite all other parameters for the specified plots. It works like this:

rc.opt = list( r2c12 = list(spine = FALSE), r1 = list(col="rgb"), c4 = list(col="seq") )

where the plot in row 2 and column 14 is a spineplot, the first row uses RGB colors and the fourth column a sequential color palette. Later arguments overwrite the preceding ones. For instance in the example the plot in row 1 and column 4 will use the sequential color palette.

factor.opt  The same as diag.opt, lower.opt, upper.opt but for all pairs of two categorical variables. This overwrites the other option lists.

...  Further parameters.

Details

Creates a matrix of all pairwise rmb-plots using all possible rmb parameters except cat.ord, expected = list() and residuals. The parameters are applied to all plots and afterwards possibly overwritten by one of the parameter lists.

Value

An environment with the parameter lists and matrices. This can be used to update (parts of) the plot without a complete new construction. The update.rmbmat function is under development.

Author(s)

Alexander Pilhoefer

See Also

rmb, pairs

Examples

data(olives)

## Not run:

# mode = "c" piecharts are currently slow
rmbmat(olives, tv=2, mode = "s")

rmbmat(olives[,1:5], tv=2, col ="div", plot.tv = TRUE, lower.opt = list(tv2 = 1, col ="rgb") )

rmbmat(olives[,c(1:5,11)], tv=2, upper.opt=list(mode="s", eqwidth = TRUE), rc.opt = list( c5 = list(eqwidth=FALSE,mode="s"),

explanation
scpcp

Static Categorical Parallel Coordinates Plot

Description

This function creates a static CPCP plot using base R graphics. The function offers color brush / highlighting and several options for the labels and colors. Efficiency is improved by replacing sets of parallel lines by polygons.

A ggplot version is under construction. A deprecated interactive version based on iplots (without labeling) is still available as extracat:::cpcp.

For reordering of category orders in CPCP plots see steptile.

Usage

scpcp(data, freqvar = "Freq", max.N = 1e6, gap = 0.2, sort.individual = TRUE, level.width = 0.2, polygon = TRUE, base.colour = alpha("black", 0.7), label = TRUE, lab.opt = list(rot = 0, col = 1, bg = TRUE, abbr = FALSE, abbr.var=12, hide.sel=TRUE, var.labels = TRUE), sel = NULL, sel.hide = TRUE, sel.palette = NULL, col.opt = list(), plot = TRUE, return.coords = !plot)

Arguments

data
freqvar
max.N
gap
sort.individual
level.width
polygon
base.colour
label
lab.opt

The data.frame which can contain a variable called "Freq".
Optional specification of a frequency variable.
The plot handles each case as a separate polyline, similar to conventional pcps. This option limits the number of observations.
The size of the gaps between the categories as a total proportion.
Whether or not the cases (lines) are additionally rearranged according to the neighboring variable. This minimizes crossings.
The width of the rectangles representing the variables/categories.
Whether or not to replace parallel lines by polygons. This improves both efficiency and undesirable color effects.
The standard color used for the cases which are not highlighted via sel.
Whether or not to draw category labels.
A list with options for the labels. See e.g. rmb

A selection defining colors. This can either be an integer vector, a factor or an expression which returns such a vector. For instance `sel = "data[,4]"` colors by the fourth variable, `sel = "sample(1:6,nrow(data),T)"` leads to random colors and `sel = "Sex=='Male' & Survived=='Yes'"` selects survivors among the men (in the titanic data, see examples). The objects (observations/lines) are additionally sorted by their color which brings colors together and makes it possible to see proportions of the selections.

Whether or not to hide the `sel` variable or plot it as the first variable.

The color palette for the selection. See `getcolors`.

A list of options for the color palette. See `getcolors`.

Whether or not to plot.

Whether or not to return the coordinates per observation. `plot = FALSE` together with `return.coords = FALSE` is therefore rather stupid. The polygon coordinates are currently not returned.

Either a logical value or the coordinates defining the polylines per observation.

A grid-based version is still under construction. In polygon mode single cases are still shown as a line rather than a ribbon of width 1/n.

Karin Maria Gehweiler and Alexander Pilhoefer.


`steptile`

```
data(Titanic)
titanic <- as.data.frame(Titanic)
sccpc(titanic)

# sccpc(titanic, level.width=0)
```
# setcover(titanic, gap=0)

#default with highlighting
setcover(titanic, sel="data[,4]")

# random colors like for instance from a clustering
setcover(titanic, sel="sample(1:6,nrow(data),T)")

# another one with some formal changes
require(scales)
setcover(data=titanic, sel="Sex=='Male' & Survived=='Yes'", sel.palette = "w",
    col.opt=list(alpha=0.7,border=alpha(1,0.3)), gap = 0.5, level.width= 0.3)

## Not run:

# mushroom data from the UCI machine learning repository
data(agaricus)
MR <- agaricus

levels(MR$stalk_root) <- c(levels(MR$stalk_root),"N/A")
MR$stalk_root[which(is.na(MR$stalk_root))] <- "N/A"
op <- optile(MR[,1:12], method="joint")

setcover(op, sel = "odor",sel.paleta="w",
    col.opt = list(border = alpha(1,0.1)), lab.opt=list(rot=45))

# ADAC ecotest data with four clusterings (k-means, mclust, hc Ward, hc complete)
data(eco)

# illustrate reordering success using coloring
scpcp(eco[,13:16], sel = "data[,1]", sel.palette="d")

scpcp(optile(eco[,13:16]), sel = "data[,1]", sel.palette="d",
    col.opt = list(border=alpha(1,0.1)))

# car classes (lower to upper class)
eco$Klasse <- factor(eco$Klasse, levels = levels(eco$Klasse)[c(3,1,2,7,4,5,6)])

scpcp(eco[,17:20], sel = eco$Klasse, sel.palette="s", col.opt = list(h=140))

# the color variable included
scpcp(eco[,c(3,17:20)], sel = eco$Klasse, sel.palette="s",
    col.opt = list(h=140),lab.opt = list(abbr=5))

## End(Not run)
Description

This function takes an indicator matrix with rows representing objects and columns representing sets and computes a minimal redundancy free set using the greedy setcover optimization algorithm. The aim is to find a minimal set of clusters which covers all objects (or a minimum proportion \( \text{rat} \)). Alternatively the number of clusters \( k \) can be specified. Then the problem becomes a maximum coverage problem. Both versions also permit weights such as frequencies (weighted setcover/maximum coverage).

Usage

\[
\text{setcover}(x, k = \text{NULL}, \text{rat} = 1, s = \text{NULL}, w = \text{NULL}, \text{check} = \text{TRUE})
\]

Arguments

- **x**: The indicator matrix.
- **k**: An optional number of clusters.
- **rat**: The minimum proportion of objects that is to be covered by the cluster set. If weights are specified in \( w \) then those are respected.
- **s**: If weights are specified but not all objects are covered by one of the sets it can be necessary to specify the total weight in order to compute a sensible ratio.
- **w**: Optional weights per object.
- **check**: Whether or not to check for redundancies.

Value

The indices of the clusters in the minimal redundancy-free set. The result is not always the globally optimal solution since the algorithm is greedy.

Note

This is written supporting the GSAC algorithm.

Author(s)

Alexander Pilhoefer

See Also

gsac

Examples

```r
# compute 100 clusterings with 24 clusters each:
scale <- scale(olives[,3:10])
kmean100 <- as.data.frame(replicate(100, kmeans(scale, centers = 24)$cluster))

# convert to indicator matrix
I100 <- idat(kmean100)
```
sortandcut

# select from all clusters a minimum set:
scover <- setcover(as.matrix(I100))

cdata <- subtable(
  as.data.frame(cbind(olives[,1:2],
  I100[,scover])),1:(length(scover)+2))
scpcp(cdata,sel="Area")

---

# Description

An implementation of the sort-and-cut algorithm which is a mixture of the top-down-partitioning
algorithm used by cfluctile and the reordering techniques available through optile. See details.

# Usage

sortandcut(x,iter=20, tau0 = NULL, fun = "BCC", method = "WBCI")

# Arguments

- **x**: A matrix or 2D table.
- **iter**: The number of random initial orderings for optile.
- **tau0**: The minimum criterion value for a new cut. See cfluctile.
- **fun**: The reordering function used by optile. Currently "BCC", "barysort" and "preclass" are available.
- **method**: The method argument for cfluctile which defines the criterion used to find an
  optimal partition.

# Details

The algorithm sorts a matrix using optile and cuts the reordered matrix once using cfluctile with
nsplit = 1. Then the same procedure is applied to the resulting submatrices at the top left and the
bottom right. The partitioning stops when the best cut leads to a criterion below tau0.

# Value

The reordered matrix. The row and column order vectors are attached as an attributes attr(x, "orders").

# See Also

cfluctile
Examples

```
M <- arsim(12000, c(30, 40), 7, noise = 0.3)
c1 <- cfuctile(M <- optile(M, iter = 20))
c2 <- cfuctile(M2 <- sortandcut(M))
```

steptile  

**Description**

Starts with the first \( k+1 \) variables and applies `optile` to the corresponding subtable. Then one additional variable at a time is reordered using the subtable defined by this variable and the last \( k \) variables. Only the current variable is reordered and the others are fixed since they have been reordered in the previous steps.

**Usage**

```r
steptile(x, k = 1, cpcp = FALSE, ...)
```

**Arguments**

- `x`  
  The `data.frame` (which is better for high-dimensional data) or data table.
- `k`  
  The number of preceding variables used for the reordering. E.g. if \( k = 3 \) then variable 6 is reordered using the variables 3, 4, 5, 6.
- `cpcp`  
  If TRUE a special version of the algorithm which minimizes crossings in CPCP plots (e.g. `scpcp`) is used. This modification works with aggregations of the last \( k \) variables and is much faster than the standard procedure if \( k > 1 \).
- `...`  
  Arguments passed to `optile`.

**Details**

The `optile` function also offers stepwise reordering via the argument `method = "sw"` but always starts with the first pair of variables and then considers the complete past: for the reordering of variable \( i \) all variables \( 1 \ldots (i-1) \) are considered. The stepwise algorithms are applicable to high-dimensional problems with a large number of variables where the multivariate techniques fail. Even if \( k \) is high (i.e. the subtables are also high-dimensional) the procedure is very fast since it can use the following trick: instead of applying `optile` to the multidimensional table it is applied to a 2D-table with one dimension defined by the variable that is reordered and the other dimension defined by the (ordered) combinations of all other variables. This way only combinations which appear at least once in the dataset matter and all empty entries (the majority in high-dimensional tables) can be left aside. The maximum possible size of such a table is therefore \( N \times \max(n_i) \) when \( N \) is the number of observations and \( n_i \) is the number of categories in dimension \( i \).

**Value**

The reordered data either as a table or `data.frame` depending on the input type.
Author(s)

Alexander Pilhoefer

See Also

optile

Examples

## Not run:

# scaled numeric variables from the olives data
# and 20 k-means solutions
so <- scale(olives[,3:10])
rr <- replicate(20,
  kmeans(so,8)$cluster
)

# par(mfrow=c(3,1))

# initial cluster orders
x <- as.data.frame(cbind(olives[,1:2],rr))
require(scales)
scpcp(x, sel = "data[,1]",
  sel.palette="rgb", col.opt=list(alpha=0.5))

# reordering using steptile.
# optile does not work for the complete table since it has 9*3*2^60 > 3E19 entries
# colors by the first unordered example:

x2 <- steptile(x, k = 4)
scpcp(x2, sel = "match(data[,1],levels(.GlobalEnv$x[,1]))",
  sel.palette="rgb", col.opt=list(alpha=0.5))

# additionally reordering the variables ... cmat takes about 20-30 seconds
CM <- cmat(x[,3:22])
require(seriation)
sM <- get_order(seriate(1-CM))
x3 <- steptile(x2[, c(1,2,2+sM,23)], k = 4)
scpcp(x3, sel = "match(data[,1],levels(.GlobalEnv$x[,1]))",
  sel.palette="rgb", col.opt=list(alpha=0.5))

## End(Not run)
subtable  
data.frame reduction

Description

Reduces a data.frame into a frequency table with prespecified entries. Uses a modified version of the count function which also accepts weights. Zero-entries can be included or excluded and the variables can be coerced into factors if necessary.

Usage

subtable(data, cols, freqvar = "Freq", keep.zero = FALSE, allfactor = FALSE, return.type = class(data))

Arguments

data  The data.frame to reduce.
cols  An ordered integer vector containing the indices of the columns to keep.
freqvar  Optional name of a frequency variable in V.
keep.zero  A logical indicating whether to include zero-cases in the output.
allfactor  A logical indicating whether to convert all variables into factor variables. Integer variables will be applied a fitting (non-lexicographic) level order.
return.type  The function is able to convert the output to a data.frame or table. The default is to use the same type as the input had.

Details

This function uses count as a workhorse and offers additional arguments keep.zero and allfactor. Both raw datasets, datasets with a frequency variable and tables can be handled.

Value

A data.frame including a "Freq" variable.

Author(s)

Alexander Pilhoefer
Department for Computer Oriented Statistics and Data Analysis
University of Augsburg
Germany
Examples

```r
require(MASS)
housing <- summary(housing)
headers(housing)

head(housing)

R <- arima(housing, order=c(3, 1, 1))
simtable(A, c(1,4))
```

subtree

<table>
<thead>
<tr>
<th>subtree</th>
<th>subtrees</th>
</tr>
</thead>
</table>

Description

Takes a subtree of a dendrogram object such as generated by `hclust` according to a prespecified number of clusters or a prespecified height.

Usage

```r
subtree(tree, k = NULL, h = NULL)
```

Arguments

- **tree**: The tree object which contains the attributes `merge` and `height` in the same way as an `hclust` object.
- **k**: The number of clusters at which to cut.
- **h**: The height at which to cut.

Details

Does the same as `cutree` with two differences: Firstly it gives back an entire tree object, i.e. an object with attributes `merge`, `height`, `labels` and `order`, as well as `data`, which contains the cluster ids. Secondly the cluster ids are chosen by the heights at which the clusters were built.

Value

An `hclust` object.

Author(s)

Alexander Pilhoefer

See Also

cutree
Examples

hc <- hclust(dist(USArrests), "ave")

hcs <- subtree(hc, k = 7)
hcs2 <- subtree(hc, h = 30)

attributes(hcs)
fluctile(table(hcs$data, cutree(hc, k = 7)))

par(mfrow = c(1, 3))
plot(hc)
plot(hcs)
plot(hcs2)

tfluctile

Fluctuation diagram with additional dendrograms

Description

Plots a fluctuation diagram via fluctile and adds dendrograms for the rows and columns to it.

Usage

tfluctile(x, tree = NULL, dims = c(1, 2), tw = 0.2, border = NULL,
shape = "r", dir = "b", just = "c",
tile.col = hsv(0.1, 0.1, 0.1, alpha = 0.6), bg.col = "lightgrey", vp = NULL,
lab.opt = list(), ...)

Arguments

x The two-way table or matrix with the data.
tree A list with tree objects. This may be NULL or will be disregarded if x has an attribute attr(x, "tree") which should also be a list. The latter way is the standard for objects returned by optile.list or optile.
dims If x has more than two dimensions this vector of length 2 indicates which variables to plot.
tw The proportion of the total space to the left and at the top which is used for the dendrogram.
border How much space is left white around the dendrogram.
shape Instead of rectangles ("r") it is possible to use circles ("c"), diamonds ("d") or octagons ("o"). The arguments dir and just work for rectangular shapes only.
dir The bar/rectangle direction: "v" and "h" stand for vertical or horizontal bars. "b" stands for "both" and leads to standard fluctuation diagrams with quadratic rectangles. Use "n" for a same-binsize-plot
just

A shortcut version of the argument used in grid for the anchorpoint of the rectangles: "rb" is equivalent to c("right", "bottom"). "t" is equivalent to "ct" or c("centre", "top") and so on. See examples.

tile.col

The tile color.

bg.col

A background color for the cells.

vp

A viewport to which the plot should be added or NULL.

lab.opt

A list with options for the labels. Currently lab.cex and abbrev work. Also lwd and line.col are the line width and the line color for the dendrogram.

... further args

Value

invisible(TRUE)

Author(s)

Alexander Pilhoefer

See Also

fluctile, cfluctile

Examples

## Not run:
library(amap)
hc1 <- hclust(t(plants[,,-1]), method="manhattan", link = "ward")
hc2 <- hclust(t(plants[,,-1]), method="manhattan", link = "complete")

hclist <- list(hc1, hc2)
tfluctile( tt<-optile(hclist, k= c(8,8) ) )

s1 <- subtree(hc1, k = 12)
s2 <- subtree(hc2, k = 10)

tfluctile( table(s1$data, s2$data), tree = list(s1,s2))

## End(Not run)

Description

Converts a frequency table into a raw data.frame.
Usage

\texttt{untableSet(data, freqvar = "Freq")}

Arguments

data 
\hspace{1cm} The data.frame including a frequency variable "Freq".

freqvar 
\hspace{1cm} The name of the frequency/weights variable which is used to expand \( V \).

Value

A data.frame.

Author(s)

Alexander Pilhoefer  
Department for Computer Oriented Statistics and Data Analysis  
University of Augsburg  
Germany

References

Alexander Pilhoefer  \emph{New approaches in visualization of categorical data: R-package extracat}  
Journal of Statistical Software, submitted March 2010

Examples

\texttt{require(MASS)}

\texttt{hs2 = untableSet(housing)}
\texttt{summary(hs2)}

---

\textbf{USR} \hspace{1cm} \emph{MovieLens USER data}

Description

The user data from the MovieLens 1M data.

Usage

\texttt{data(USR)}
**Format**

A data frame with 6040 observations on the following 25 variables.

- UserID  a numeric vector
- Gender  a factor with levels F M
- Age  a numeric vector
- Occupation  a numeric vector
- Zip code  a factor
- occupation  a factor
- zip1  a numeric vector
- zip2  a numeric vector
- zip12  a numeric vector
- UserVotes  a numeric vector
- meanUserRat  a numeric vector
- medianUserRat  a numeric vector
- sdUserRat  a numeric vector
- mInts  a numeric vector
- AvgRat  a numeric vector
- meanDiffRat  a numeric vector
- meanAbsDiffRat  a numeric vector
- Pct.Action  a numeric vector
- Pct.Adventure  a numeric vector
- Pct.Animation  a numeric vector
- Pct.Children.s  a numeric vector
- Pct.Fantasy  a numeric vector
- Pct.Horror  a numeric vector
- Pct.Sci.Fi  a numeric vector
- Pct.Comedy  a numeric vector

**Source**

MovieLens.org

**Examples**

```r
data(USR)
## maybe str(USR) ; plot(USR) ...```
visid

Visualizing Indicator Matrices and Missing Values

Description

This function aggregates a binary dataframe or matrix using `subtable` and visualizes the combinations along with the marginal distributions using `fluctile`. Options include reordering of rows and columns, filtering the most important rows and columns as well as ceiling censored zooming for the marginals.

Missing values can be visualized using `is.na` and datasets with categorical variables can be transformed via `idat`. `visna(x)` is a shortcut for `visid(is.na(x) + 0)`. `visdf(x, freqvar)` is a shortcut for `visid(idat(x, allcat = TRUE, keep = freqvar))`.

Usage

```r
visid(x, freqvar = "Freq", tp = FALSE, fr = 1, fc = 1, sort = "n", 
      sort.method = "count", col = "w", 
      mar.col = c(alpha("black", 0.7), alpha("darkred", 0.8), "red", "green"), 
      s = Inf, pmax = 1, opts = list(), plot = TRUE, return.data = !plot, ...)
visna(x, freqvar = "Freq", tp = FALSE, fr = 1, fc = 1, sort = "n", 
      sort.method = "count", col = "w", 
      mar.col = c(alpha("black", 0.7), alpha("darkred", 0.8), "red", "green"), 
      s = Inf, pmax = 1, opts = list(), plot = TRUE, return.data = !plot, ...)
visdf(x, freqvar = "Freq", tp = FALSE, fr = 1, fc = 1, sort = "n", 
      sort.method = "count", col = "w", 
      mar.col = c(alpha("black", 0.7), alpha("darkred", 0.8), "red", "green"), 
      s = Inf, pmax = 1, opts = list(), plot = TRUE, return.data = !plot, ...)
```

Arguments

- **x**  A binary dataframe or matrix. `is.na` and `idat` can be used to bring in missing values and categorical variables.

- **freqvar**  An optional frequency variable. If this is not found the data is aggregated using `subtable`.

- **tp**  Logical. Whether or not to transpose the indicator matrix for the visualization.

- **fr**  This controls the row filtering: Only the `fr` most frequent rows are kept. Values below 1 are interpreted as proportions and only the minimum number of rows covering at least `fr` percent of the observations are kept.

- **fc**  See `fr`.

- **sort**  One of "n" (no reordering), "r" (row reordering), "c" (column reordering) or "b" (row and column reordering).

- **sort.method**  The default is reordering by frequency (rows) and average (columns). Other options include "ME" for reordering by the measure of effectiveness (see `optME`) and "optile" which enables all reorderings offered by `optile`. 
visid

col  The color palette. For the basic indicator case only a single color is meaningful. Transformations via idat feature coloring by variable.

mar.col  Colors for the marginals: rows, columns, row markers, column markers (see s and pmax).

s  Ceiling censored zooming for the rows. The maximum of the scale is s times the second largest value.

pmax  The maximum for the average/percentage scale used for the column marginal plot.

opts  A list of options. Currently:

gap.prop  gaps proportion used in fluctile).

The default is to use no gaps if more than 40 rows or columns are involved.

mar  Vector controlling the size of the marginal plots.

border  Borders around the plot.

shape  Shape of the symbols. Default is rectangles.

bg.col  Background color center/right/bottom

abbrev  Label abbreviation.

plot  Whether or not to draw the plot.

return.data  Whether or not to return the data after filtering and reordering took place.

...  Further arguments passed to either optME or optile. Can for instance be used to choose the reordering method.

Value

The filtered and reordered data or invisible(TRUE) (default).

Author(s)

Alexander Pilhoefer

See Also

fluctile, optile, idat

Examples

## Not run:
require(reshape2)
require(scales)

MJ <- read.table(
"http://www.rosuda.org/~pilhoefer/MJnew.txt",
header=T,sep="\t",quote="")
MJS <- MJ[,13:105]

visid(MJS)

# sort by count/percentage
visid(MJS, sort="b")

# sort via ME
visid(MJS, sort = "b", sort.method="ME")

# only rows, only columns
visid(MJS, sort = "r", sort.method="ME")
visid(MJS, sort = "c", sort.method="ME")

# sort via optile
visid(MJS, sort = "b", sort.method="optile")

visid(MJS, sort = "b", sort.method="optile", iter=10)
visid(MJS, sort = "b", sort.method="optile", fun="ca")

# 24 rows
visid(MJS, sort = "r", sort.method="optile", fr=24)

# 24 rows, >= 40
visid(MJS, sort = "r", sort.method="optile", fr=24, fc = 0.4)

# zoom y marginal
visid(MJS, sort = "r", sort.method="optile", fr=24, s=1)

# zoom x marginal
visid(MJS, sort = "r", sort.method="optile", fr=24, pmax=0.1, s =0.5)

## End(Not run)

# NA-example: GeneEx
visna(GeneEx, sort = "b", sort.method="optile", fr=50, pmax=0.05, s = 2)

require(MASS)
visdf(housing)
visdf(housing,opts=list(var.col="w"))
visdf(housing,opts=list(var.col="w"), sort="r")
Description

The weighted Bertin Classification Criterion using weights according to the Hamming distance is normalized by means of the independence case.

Usage

\texttt{WBCI(x)}

Arguments

\texttt{x} \quad \text{A data matrix.}

Value

The criterion value.

Author(s)

Alexander Pilhoefer

See Also

\texttt{kendalls}

Examples

\begin{verbatim}
M <- arsim(1000, c(12,12), 3)
BCI(M)
WBCI(M)

M2 <- optile(M, iter = 100)
BCI(M2)
WBCI(M2)

M3 <- optile(M, fun = "WBCC", iter = 100)
BCI(M3)
WBCI(M3)
\end{verbatim}
Description

An efficient implementation of the distance correlation for two variables with the additional option
to weight the observations. The main application for the weights is to use frequencies according to
ordinal variables which can be represented by a contingency table (wdcor.table). Another idea
is to make the distance correlation more robust by assigning small weights to observations which
are far from the rest of the data.

For large datasets the distance correlation is often said to be too inefficient to be of any great use.
The function approx.dcor offers a pretty good approximation of the distance correlation via binning
and wdcor.table).

 wdcor.data.frame computes a distance correlation matrix. Factor variables are transformed to
 integer via data.matrix.

Usage

wdcor(x,...)

## Default S3 method:
wdcor(x,y,w = NULL,ep = 1, approx = FALSE, n = 50,na.rm = TRUE, ...)
## S3 method for class 'table'
wdcor(x,ep = 1,...)
## S3 method for class 'data.frame'
wdcor(x, w = NULL, ep = 1, approx = FALSE, n = 50, ...)

Arguments

x A data crosstable or a numeric vector.
y A numeric vector.
w Weights, typically frequencies. The default weights all cases the same, which
leads to the standard distance correlation.
ep The euclidean (absolute) distances can be taken to the power of ep.
approx Whether or not to use approx.dcor instead of wdcor. This is automatically cho-
sen for vectors with more than 20000 entries.
n The number of bins used by approx.dcor.
a.rm Whether or not to remove missing values.
... dots.

Value

The correlation value which is between 0 and 1.
Note

Automatically uses an approximation for vectors larger than 20000 entries!

Author(s)

Alexander Pilhoefer

References


See Also

approx.dcor

Examples

# repeat and change N for different results and computation times.
N <- 2000
x1 <- rnorm(N, mean=10, sd=3)
x2 <- runif(N, 0, 40)
x3 <- rnorm(N, mean=30, sd=4)
x <- sample(c(x1, x2, x3), N)

y <- rnorm(1, sd=0.0001)*(x-mean(x))^4 + rnorm(1, sd=0.01)*(x-mean(x))^3
y <- y + rnorm(1, sd=0.1)*(x-mean(x))^2
y <- y + rnorm(N, sd=runif(N, 3, 10))
y <- y + runif(N, 0, 20)*sin(abs(scale(x))*2*pi)

require(scales)
plot(x, y, pch=19, col=alpha("black", 0.2))

system.time(dd <- wdcor(x, y))

y2 <- runif(2000)

system.time(dde <- wdcor(x, y2))
dd
dde

## Not run:
y <- diamonds$price
x <- diamonds$carat

length(x) # 53940

# auto approximation via approx.dcor
wdcor(x, y)

# the weighted distance correlation is also applicable to
# discrete data:
A <- arsim(2000, c(12, 12), 4, 0.1)
wdcor(A)
wdcor(optile(A))
wdcor(optile(A, fun = "distcor"))

# kernel density weights:
k <- kde2d(x, y, n=50)

xy <- expand.grid(k$d$x, k$d$y)
wdcor(xy[, 1], xy[, 2], w = k$d$z)
# this is the approximate distance correlation for the 2D density estimate

## End(Not run)

# a pairwise matrix:
D <- wdcor(olives[, 3:10])
fluctile(D^2, shape="c")
Index

*Topic textasciitildekwd1
  steptile, 108
*Topic textasciitildekwd2
  steptile, 108
*Topic categorical
  extracat, 36
*Topic datasets
  agaricus, 3
carcustomers, 13
CPScluster, 21
dmc, 27
eco, 28
GeneEx, 45
MJnew, 67
olives, 77
plants, 87
USR, 114
*Topic frequency table
  untabelleSet, 113
*Topic interactive
  extracat, 36
*Topic mosaicplots
  extracat, 36
*Topic multiple barcharts
  extracat, 36
*Topic parallel coordinates
  extracat, 36
  
aes, 38
agaricus, 3
ahist, 4, 23, 48
approx.dcor, 5, 24, 25, 120, 121
arsim, 7, 37
barysort, 9
BCC, 10
BCI, 11, 15, 20, 64, 90
Burt, 12, 61, 79, 80
carcustomers, 13

cBCI, 15, 64
cfcl, 16, 21
cfluctile, 16, 17, 21, 53, 54, 78, 107, 113
cmat, 19, 90
cohen, 20
combscl, 21
count, 110
CPScluster, 21
cut, 19, 23
cutbw, 5, 23, 48
cutree, 111
dcorMVdata, 24, 25
dcorMVtable, 24, 25
dendro, 26
dist, 24, 25
diverge_hcl, 49, 56
dmc, 27
eco, 28
extracat, 36
facet_grid, 38
facetshade, 38
fluctile, 17, 18, 37, 42, 56, 57, 112, 113, 116, 117
formula, 95
ftable, 95
GeneEx, 45
getbw, 4, 5, 23, 47
getcolors, 49, 104
getis, 50, 52, 54, 56
getIs2, 51, 51, 52, 56
getpath, 52
ggplot, 38
glm, 97
gsac, 53, 106
hclust, 79, 111
heat_hcl, 49
heattile, 50–52, 54, 56
hexbin, 58, 59
hexpie, 37, 58

idat, 12, 61, 116, 117
imat, 12, 61, 62
innerval, 59, 62, 95
is.na, 116
itab, 63

JBCI, 15, 64

kendalls, 10, 11, 20, 65, 90, 119

ME, 50–52, 66, 87
Mjnew, 67
mosaicplot, 43, 98

na.omit, 95
nclass.scott, 48

olives, 77
optile, 18, 19, 37, 53, 54, 78, 90, 107–109, 116, 117
optME, 50–52, 66, 86, 116, 117

pairs, 102
plants, 87
polr, 97

qBCI, 19, 20, 90
quantile, 63
quickfechner, 37, 52, 92

rainbow, 49, 56, 97
rainbow_hcl, 49, 56, 97
regmax, 93
regmin (regmax), 93
rmb, 37, 56, 59, 94, 101–103
rmbmat, 37, 100

scpcp, 37, 78, 103, 108
sequential_hcl, 49, 56
setcover, 54, 105
solve_TSP, 50, 51, 87
sortandcut, 18, 54, 107
stat_binhex, 59
steptile, 80, 81, 103, 104, 108
subtable, 110, 116
subtree, 79, 111
terrain_hcl, 49
tfluctile, 18, 112
untabset, 113
USR, 114
visdf (visid), 116
visid, 116
visna (visid), 116

WBCI, 15, 20, 64, 119
wddcor, 20, 24, 25, 81, 90, 120, 120
wddcor_table, 5

xtabs, 24