Package 'GMD'

December 27, 2016

Type Package

Version 0.3.3

Date 2014-08-26 15:46:45 EDT

Title Generalized Minimum Distance of distributions

Description GMD is a package for non-parametric distance measurement between two discrete frequency distributions.

License GPL (>= 2)

LazyLoad TRUE

NeedsCompilation yes

URL http://CRAN.R-project.org/package=GMD

Repository CRAN

Depends R (>= 3.1.0)

Imports stats, grDevices, gplots

Suggests datasets, MASS, cluster

Collate 'GMD-package.R' 'imports.R' 'util.R' 'GMD-internal.R' 'GMD-data.R' 'ghist.R' 'gdist.R' 'css.R' 'heatmap3.R' 'gmdp.R' 'gmdm.R'

Maintainer Xiaobei Zhao <xiaobei@binf.ku.dk>

Author Xiaobei Zhao [aut, cre, cph], Albin Sandelin [aut]

Date/Publication 2016-12-27 15:28:19

R topics documented:

`GMD-package` *The Package for Generalized Minimum Distance (GMD) Computation*

Description

Computate Generalized Minimum Distance (GMD) between discrete distributions and clustering tools

Details

Package:	GMD
Type:	Package
License:	GPL (>= 2)

This package contains:
1) modules and functions for GMD computation, with GMD algorithm implemented in C to interface with R.
2) related clustering and visualization tools for distributions.

An overview of functions

Function	Description
`ghist`	Generalized Histogram Computation and Visualization
`gdist`	Generalized Distance Matrix Computation
`css`	Computing Clustering Sum-of-Squares and evaluating the clustering by the *"elbow"* method
`heatmap.3`	Enhanced Heatmap Representation with Dendrogram and Partition
`gmdp`	Computation of GMD on a pair of histograms
`gmdm`	Computation of GMD Matrix on a set of histograms

To install from online repositories (e.g. CRAN) install.packages(pkgs="GMD", repos="http://cran.r-project.org")

Sometimes the offical repository might not be up to date, then ## you may install it from a downloaded source file; please replace ## '<current-version>' with actual version numbers: Note that as ## new versions are release, the '<current-version>' changes. install.packages(pkgs="GMD_<current-version>.tar.gz", repos=NULL)

Load the package and get a complete list of functions, use library(GMD) ls("package:GMD")

help documantation of the package help(GMD) # this page

Value

NULL

Author(s)

Xiaobei Zhao and Albin Sandelin

Maintainer: Xiaobei Zhao <xiaobei (at) binf.ku.dk>

References

Zhao et al (2011), "Systematic Clustering of Transcription Start Site Landscapes", *PLoS ONE* **6**(8): e23409.
http://dx.plos.org/10.1371/journal.pone.0023409

See `citation("GMD")` for BibTeX entries for LaTeX users.

See Also

gmdp, gmdm, cage, chipseq, ghist, gdist, css, elbow, heatmap.3

Examples

```
## Not run:
require(GMD)        # load GMD
help(GMD)           # a help document of GMD
data(package="GMD") # a list of datasets available in GMD
ls("package:GMD")   # a list of functions available in GMD
help(package="GMD") # help documentation on GMD
citation("GMD")     # citation for publications
demo("GMD-demo")    # run the demo

## view GMD Description
packageDescription("GMD")

## view GMD vignette
vignette("GMD-vignette",package="GMD")

## End(Not run)
```

bedgraph.to.depth *Convert a BedGraph fiel to a vector of depth-like signals*

Description

Convert a BedGraph fiel to a vector of depth-like signals

Usage

```
bedgraph.to.depth(inFpath, chr, start, end, reverse = FALSE)
```

Arguments

inFpath	character, the path of a bedgraph file
chr	character, the chromosome
start	numeric, the start position (0-base)
end	numeric, the end position
reverse	logical, whether a reverse strand

Value

numeric

Author(s)

Xiaobei Zhao

cage *CAGE Data: Transcription Start Site Distributions (TSSDs) by CAGE tags*

Description

Transcription Start Site Distributions (TSSDs) by CAGE tags.

Usage

```
cage
cagel
```

Details

cage is a list of 20 named TSSDs. cagel is a longer version of cage, with 50 named TSSDs.

References

Zhao et al (2011), "Systematic Clustering of Transcription Start Site Landscapes", *PLoS ONE* **6**(8): e23409.

http://dx.plos.org/10.1371/journal.pone.0023409

See Also

gmdp and gmdm, with examples using cage. chipseq for histone marks by ChIP-seq reads.

Examples

```
help(cage)
data(cage)
class(cage)
length(cage)
names(cage)
## Not run: cage

data(cagel)
names(cagel)
## Not run: cagel
```

---

chipseq *ChIP-seq data: ChIP-seq Enrichment around TSSs*

Description

The Distributions of Histone Modification Enrichment (and Others) by ChIP-seq reads that are binned, aligned and averaged around +/-5000nt of Transcription Start Sites (TSSs) of scattered-type TSSDs (see References).

Usage

```
chipseq_mES
chipseq_hCD4T
```

Details

chipseq_mES is a list of 6 named ChIP-seq read distributions from mouse ES cells.
chipseq_hCD4T is a list of 40 named ChIP-seq read distributions from human CD4+ T cells.

References

Zhao et al (2011), "Systematic Clustering of Transcription Start Site Landscapes", *PLoS ONE* **6**(8): e23409.

http://dx.plos.org/10.1371/journal.pone.0023409

See Also

gmdp and gmdm, with examples using chipseq. cage for data of Transcription Start Sites (TSSs) by CAGE tags.

Examples

```
require(GMD)
help(chipseq)
data(chipseq_mES)
class(chipseq_mES)
length(chipseq_mES)
names(chipseq_mES)
## Not run: chipseq_mES

data(chipseq_hCD4T)
names(chipseq_hCD4T)
```

---

css *Clustering Sum-of-Squares for clustering evaluation*

Description

Evaluation on the varaince of a clustering model using squared Euclidean distances, based on distance matrix and cluster membership.

Usage

```
css(dist.obj,clusters)

## Computing Sum-of-Squares given Hierarchical Clustering
## S3 method for class 'hclust'
css(dist.obj, hclust.obj=NULL,hclust.FUN=hclust,
hclust.FUN.MoreArgs=list(method="ward"),k=NULL)
```

Arguments

dist.obj	a 'dist' object as produced by dist or gdist.
clusters	a vector with cluster memberships.
k	numeric, the upper bound of the number of clusters to compute. DEFAULT: *20* or the number of observations (if less than *20*).
hclust.obj	a 'hclust' object, generated by hclust
hclust.FUN	a function, to generate a hierarchical clustering. Ignored with hclust.obj specified. DEFAULT: hclust
hclust.FUN.MoreArgs	a list, containing arguments that are passed to hclust.FUN.

Details

Clustering Sum-of-Squares for clustering evaluation.

Value

`css` returns a 'css' object, which is a list containing the following components

k	number of clusters
wss	k within-cluster sum-of-squares
totwss	total within-cluster sum-of-square
totbss	total between-cluster sum-of-square
tss	total sum of squares of the data

, and with an attribute 'meta' that contains the input components

dist.obj	(the input) distance matrix
clusters	(the input) cluster membership

`css.hclust` returns a 'css.multi' object, which is a data.frame containing the following columns

k	number of clusters
ev	explained variance given k
totbss	total between-cluster sum-of-square
tss	total sum of squares of the data

, and with an attribute 'meta' that contains

cmethod	the clustering method
dist.obj	(the input) distance matrix
k	(the input) number of clusters
clusters	the 'hclust' object that is either by input or computed by default

See Also

`elbow` for "elbow" plot using 'css.multi' object

`elbow` *The "Elbow" Method for Clustering Evaluation*

Description

Determining the number of clusters in a data set by the "elbow" rule.

Usage

```
## find a good k given thresholds of EV and its increment.
elbow(x,inc.thres,ev.thres,precision=3,print.warning=TRUE)

## a wrapper of `elbow' testing multiple thresholds.
elbow.batch(x,inc.thres=c(0.01,0.05,0.1),
ev.thres=c(0.95,0.9,0.8,0.75,0.67,0.5,0.33),precision=3)

## S3 method for class 'elbow'
plot(x,elbow.obj=NULL,main,xlab="k",
ylab="Explained_Variance",type="b",pch=20,col.abline="red",
lty.abline=3,if.plot.new=TRUE,print.info=TRUE,
mar=c(4,5,3,3),omi=c(0.75,0,0,0),...)
```

Arguments

x	a 'css.multi' object, generated by css.hclust
inc.thres	numeric with value(s) from 0 to 1, the threshold of the increment of EV. A single value is used in elbow while a vector of values in elbow.batch.
ev.thres	numeric with value(s) from 0 to 1, the threshold of EV. A single value is used in elbow while a vector of values in elbow.batch.
precision	integer, the number of digits to round for numerical comparison.
print.warning	logical, whether to print warning messages.
elbow.obj	a 'elbow' object, generated by elbow or elbow.batch
main	an overall title for the plot.
ylab	a title for the y axis.
xlab	a title for the x axis.
type	what type of plot should be drawn. See help("plot", package="graphics").
pch	Either an integer specifying a symbol or a single character to be used as the default in plotting points (see par).
col.abline	color for straight lines through the current plot (see option col in par).
lty.abline	line type for straight lines through the current plot (see option lty in par).
if.plot.new	logical, whether to start a new plot device or not.
print.info	logical, whether to print the information of 'elbow.obj'.
mar	A numerical vector of the form 'c(bottom, left, top, right)' which gives the number of lines of margin to be specified on the four sides of the plot (see option mar in par). The default is 'c(4, 5, 3, 3) + 0.1'.
omi	A vector of the form 'c(bottom, left, top, right)' giving the size of the outer margins in inches (see option omi in par).
...	arguments to be passed to method plot.elbow, such as graphical parameters (see par).

Details

Determining the number of clusters in a data set by the "elbow" rule and thresholds in the explained variance (EV) and its increment.

Value

Both `elbow` and `elbow.btach` return a 'elbow' object (if a "good" `k` exists), which is a list containing the following components

k	number of clusters
ev	explained variance given `k`
inc.thres	the threshold of the increment in EV
ev.thres	the threshold of the EV

, and with an attribute 'meta' that contains

description	A description about the "good" `k`

See Also

`css` and `css.hclust` for computing Clustering Sum-of-Squares.

Examples

```
## load library
require("GMD")

## simulate data around 12 points in Euclidean space
pointv <- data.frame(x=c(1,2,2,4,4,5,5,6,7,8,9,9),
y=c(1,2,8,2,4,4,5,9,9,8,1,9))
set.seed(2012)
mydata <- c()
for (i in 1:nrow(pointv)){
  mydata <- rbind(mydata,cbind(rnorm(10,pointv[i,1],0.1),
rnorm(10,pointv[i,2],0.1)))
}
mydata <- data.frame(mydata); colnames(mydata) <- c("x","y")
plot(mydata,type="p",pch=21, main="Simulated data")

## determine a "good" k using elbow
dist.obj <- dist(mydata[,1:2])
hclust.obj <- hclust(dist.obj)
css.obj <- css.hclust(dist.obj,hclust.obj)
elbow.obj <- elbow.batch(css.obj)
print(elbow.obj)

## make partition given the "good" k
k <- elbow.obj$k; cutree.obj <- cutree(hclust.obj,k=k)
```

```
mydata$cluster <- cutree.obj

## draw a elbow plot and label the data
dev.new(width=12, height=6)
par(mfcol=c(1,2),mar=c(4,5,3,3),omi=c(0.75,0,0,0))
plot(mydata$x,mydata$y,pch=as.character(mydata$cluster),
col=mydata$cluster,cex=0.75,main="Clusters of simulated data")
plot(css.obj,elbow.obj,if.plot.new=FALSE)

## clustering with more relaxed thresholds (, resulting a smaller "good" k)
elbow.obj2 <- elbow.batch(css.obj,ev.thres=0.90,inc.thres=0.05)
mydata$cluster2 <- cutree(hclust.obj,k=elbow.obj2$k)

dev.new(width=12, height=6)
par(mfcol=c(1,2), mar=c(4,5,3,3),omi=c(0.75,0,0,0))
plot(mydata$x,mydata$y,pch=as.character(mydata$cluster2),
col=mydata$cluster2,cex=0.75,main="Clusters of simulated data")
plot(css.obj,elbow.obj2,if.plot.new=FALSE)
```

`equalize.list` *Make members of a list equal size*

Description

Make member bins of a hist object equal size

Usage

```
equalize.list(x)
```

Arguments

`x`	a list of numeric vectors

Details

Make members of a list equal size

`gdist` *Generalized Distance Matrix Computation*

Description

`gdist` computes and returns the distance matrix computed by using user-defined distance measure.

Usage

```
gdist(x,method="euclidean",MoreArgs=NULL,diag=FALSE,upper=FALSE)

is.dist(d)
```

Arguments

x	a numeric matrix, data frame or 'dist' object.
method	the distance measure to be used. This can either be one of the methods used in dist (see help("dist", package="stats")) or "correlation", "correlation.of.observations" and "correlation.of.variables". In addition, user-defined distance measure are also allowed, which returns a *dist* object and should at least have attributes *"Size"* and *"Labels"*.
MoreArgs	a list of other arguments to be passed to gdist.
diag	logical value indicating whether the diagonal of the distance matrix should be printed by print.dist.
upper	logical value indicating whether the upper triangle of the distance matrix should be printed by print.dist.
d	an R object.

Details

is.dist tests if its argument is a 'dist' object.

The distance (or dissimilarity) function (FUN) can be any distance measure applied to x. For instance, "euclidean", "maximum", "manhattan","canberra", "binary", "minkowski", "correlation.of.variables", "correlation.of.observations" or gmdm. "correlation.of.variables" computes the correlation distance of the variables (the columns); all the other compute the distances between the observations (the rows) of a data matrix.

Value

gdist returns an object of 'dist'.
is.dist returns a logical value whether an object is 'dist'.

Examples

```
## load library
require("GMD")
require(cluster)

## compute distance using Euclidean metric (default)
data(ruspini)
x <- gdist(ruspini)

## see a dendrogram result by hierarchical clustering
dev.new(width=12, height=6)
plot(hclust(x),
```

```
     main="Cluster Dendrogram of Ruspini data",
     xlab="Observations")

## convert to a distance matrix
m <- as.matrix(x)

## convert from a distance matrix
d <- as.dist(m)
stopifnot(d == x)

## Use correlations between variables "as distance"
data(USJudgeRatings)
dd <- gdist(x=USJudgeRatings,method="correlation.of.variables")
dev.new(width=12, height=6)
plot(hclust(dd),
     main="Cluster Dendrogram of USJudgeRatings data",
     xlab="Variables")
```

`get.sep` *Get row or column lines of separation for heatmap.3*

Description

Get row or column lines of separation for `heatmap.3` according to clusters

Usage

```
get.sep(clusters, type = c("row", "column", "both"))
```

Arguments

`clusters`	a numerical vector, indicating the cluster labels of observations.
`type`	string, one of the following: `c("row","column","both")`

Details

Get row or column lines of separation for `heatmap.3` according to clusters

ghist *Generalized Histogram Computation*

Description

Generalized Histogram Computation with classes to contain a single histogram or multiple histograms

Usage

```
ghist(data,n,breaks=if (!.invalid(n)) NULL else "Sturges",
bins=NULL,digits=1)

gbreaks(data, n)

is.ghist(x)

as.ghist(x,bins)

is.mhist(x)

as.mhist(x,bins)

mhist2matrix(h)
```

Arguments

data	a vector of values for which the histogram is desired.
n	a single number giving the number of bins for the histogram.
breaks	a vector giving the breakpoints between histogram bins, or a character string naming an algorithm to compute the number of bins, or a function to compute the number of bins (see `help("dist", package="graphics")`).
bins	character vecter, the bin labels.
digits	integer, the number of digits to round for breaks.
x	an R object.
h	an object of class `mhist`

Details

`ghist` generates a single histogram.

`gbreaks` generate bin boundaries for a histogram.

`is.ghist` returns `TRUE` if `x` is an object of codeghist and `FALSE` otherwise.

`as.ghist` is a generic function. The method for numeric vectors will return a `ghist` object.

`is.mhist` returns `TRUE` if `x` is an object of codemhist and `FALSE` otherwise.

as.mhist is a generic function. The method is for numeric list, matrices or data frames and will return a mhist object.

mhist2matrix convert a mhist object into a numeric matrix, filling observations by row.

See Also

plot.mhist mhist.summary plot.mhist.summary

Examples

```
## load library
require("GMD")

## create two normally-distributed samples
## with unequal means and unequal variances
set.seed(2012)
v1 <- rnorm(1000,mean=-5, sd=10)
v2 <- rnorm(1000,mean=10, sd=5)

## create common bins
n <- 20 # desired number of bins
breaks <- gbreaks(c(v1,v2),n) # bin boundaries
x <-
  list(ghist(v1,breaks=breaks,digits=0),
       ghist(v2,breaks=breaks,digits=0))
mhist.obj <- as.mhist(x)
```

---

gmdm *Generalized Minimum Distance Matrix*

Description

Computing Generalized Minimum Distance Matrix

Usage

```
gmdm(data,labels,pseudocount=0,sliding=TRUE,resolution=1)

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

## convert a `gmdm' object into a `dist' object
gmdm2dist(m, diag=FALSE, upper=FALSE)

## compute GMDM and convert into a `dist' object
gmdm_dist(data, diag=FALSE, upper=FALSE, ...)
```

Arguments

data	a list of numeric vectors, a numeric matrix or data.frame
x	a gmdm object.
m	a gmdm object.
labels	a character vector of the same length of x, giving the names of the numeric vectors.
pseudocount	a numeric value to be allocated for each position to reduce bias; by default pseudocount = 0.
sliding	logical, indicating whether sliding is allowed or not for an optimal solution; by default sliding = TRUE.
resolution	relative resolution, numeric (>=1), changing the size of the bin by multiplying the value. A larger value (lower resolution) is more computational efficiet but missing details.
diag	logical value indicating whether the diagonal of the distance matrix should be printed by print.dist.
upper	logical value indicating whether the upper triangle of the distance matrix should be printed by print.dist.
...	arguments to be passed to method

Details

Computing Generalized Minimum Distance Matrix

Value

gmdm returns an object of class gmdm, a list with components

labels: a string vector, giving the names of distributions

data.ori: a list of numeric vectors, giving the original input

data: a list of numeric vectors, giving the normalized version of the original input

dm: a numeric numeric, the pairwise distance matrix of *GM-Distances*

gap.pair: a numeric matrix, giving the gap pair of each alignment per row: i.e. relative shifts between distributions of the optimal hit

sliding: logical, indicating whether sliding is performed

pseudocount: a numeric value that is allocated at each position in addition to original values

References

See citation("GMD")

See Also

plot.gmdm, gmdp

gmdp *Generalized Minimum Distance between a pair of distributions*

Description

Generalized Minimum Distance between a pair of distributions

Usage

```
gmdp(v1, v2, labels=c("v1","v2"), pseudocount=0, sliding=TRUE,
resolution=1)

## S3 method for class 'gmdp'
print(x, mode=c("brief","detailed","full"),
digits=3, ...)

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

Arguments

v1	a numeric vector, giving positional counts as a discrete distribution.
v2	a numeric vector, giving positional counts as a discrete distribution.
labels	a string vector of length 2, giving the names of v1 and v2 respectively.
pseudocount	a numeric value to be allocated for each position to reduce bias; by default `pseudocount = 0`.
sliding	logical, indicating whether sliding is allowed or not for an optimal solution; by default `sliding = TRUE`.
resolution	relative resolution, numeric (>=1), changing the size of the bin by multiplying the value. A larger value (lower resolution) is more computational efficiet but missing details.
x	an object of class gmdp.
object	an object of class gmdp.
mode	a string of the following: `c("brief","detailed","full")`, indicating whether to print in *full* mode (*default*).
digits	integer, indicating the number of decimal places to be printed.
...	arguments to be passed to method.

Details

Generalized Minimum Distance between a pair of distributions

Value

gmdp returns an object of class gmdp, a numeric with an attribute of *meta* in a list with components:

labels: a string vector, giving the names of distributions

v1.ori: a numeric vector, the first input distribution

v2.ori: a numeric vector, the second input distribution

v1: a numeric vector, the normalized version of the first input distribution

v2: a numeric vector, the normalized version of the second input distribution

distance: numeric, the *GM-Distance* (*GMD*)

sliding: logical, indicating whether sliding is performed

pseudocount: a numeric value that is allocated at each position in addition to original values

gap.pair: a numeric matrix, giving one gap pair per row: i.e. relative shifts between distributions of one optimal hit

n.hit: numeric, the number of (equally good) optimal hits

References

See `citation("GMD")`

See Also

print.gmdp, summary.gmdp, plot.gmdp gmdm

Examples

```
require(GMD)
gmdp(c(4,1,1,0,0,0,3,1),c(2,1,1,0,0,0,3,3),sliding=FALSE)
x <- gmdp(c(4,1,1,0,0,0,3,1), c(1,1,2,1,1,0,0,0,3,3,5,5),
pseudocount=1, sliding=TRUE)
print(x)
print(x, "full")
```

---

heatmap.3 *Enhanced Heatmap Representation with Dendrogram and Partition*

Description

Enhanced heatmap representation with dendrograms and partition given the *elbow criterion* or a desired number of clusters.
1) a dendrogram added to the left side and to the top, according to cluster analysis;
2) partitions in highlighted rectangles, according to the "elbow" rule or a desired number of clusters.

Usage

```
heatmap.3(x, diss = inherits(x, "dist"), Rowv = TRUE, Colv = TRUE,
  dendrogram = c("both", "row", "column", "none"), dist.row, dist.col,
  dist.FUN = gdist, dist.FUN.MoreArgs = list(method = "euclidean"),
  hclust.row, hclust.col, hclust.FUN = hclust,
  hclust.FUN.MoreArgs = list(method = "ward"), scale = c("none", "row",
  "column"), na.rm = TRUE, cluster.by.row = FALSE, cluster.by.col = FALSE,
  kr = NA, kc = NA, row.clusters = NA, col.clusters = NA,
  revR = FALSE, revC = FALSE, add.expr, breaks, x.center,
  color.FUN = gplots::bluered, sepList = list(NULL, NULL),
  sep.color = c("gray45", "gray45"), sep.lty = 1, sep.lwd = 2, cellnote,
  cex.note = 1, notecol = "cyan", na.color = par("bg"),
  trace = c("none", "column", "row", "both"), tracecol = "cyan", hline,
  vline, linecol = tracecol, labRow = TRUE, labCol = TRUE,
  srtRow = NULL, srtCol = NULL, sideRow = 4, sideCol = 1,
  margin.for.labRow, margin.for.labCol, ColIndividualColors,
  RowIndividualColors, cexRow, cexCol, labRow.by.group = FALSE,
  labCol.by.group = FALSE, key = TRUE, key.title = "Color Key",
  key.xlab = "Value", key.ylab = "Count", keysize = 1.5, mapsize = 9,
  mapratio = 4/3, sidesize = 3, cex.key.main = 0.75,
  cex.key.xlab = 0.75, cex.key.ylab = 0.75, density.info = c("histogram",
  "density", "none"), denscol = tracecol, densadj = 0.25,
  main = "Heatmap", sub = "", xlab = "", ylab = "", cex.main = 2,
  cex.sub = 1.5, font.main = 2, font.sub = 3, adj.main = 0.5,
  mgp.main = c(1.5, 0.5, 0), mar.main = 3, mar.sub = 3, if.plot = TRUE,
  plot.row.partition = FALSE, plot.col.partition = FALSE,
  cex.partition = 1.25, color.partition.box = "gray45",
  color.partition.border = "#FFFFFF", plot.row.individuals = FALSE,
  plot.col.individuals = FALSE, plot.row.clusters = FALSE,
  plot.col.clusters = FALSE, plot.row.clustering = FALSE,
  plot.col.clustering = FALSE, plot.row.individuals.list = FALSE,
  plot.col.individuals.list = FALSE, plot.row.clusters.list = FALSE,
  plot.col.clusters.list = FALSE, plot.row.clustering.list = FALSE,
  plot.col.clustering.list = FALSE, row.data = FALSE, col.data = FALSE,
  if.plot.info = FALSE, text.box, cex.text = 1, ...)
```

Arguments

x	data matrix or data frame, or dissimilarity matrix or 'dist' object determined by the value of the 'diss' argument. ##diss logical flag: if TRUE (default for dist or dissimilarity objects), then x is assumed to be a dissimilarity matrix. If FALSE,then x is treated as a matrix of observations by variables.
diss	logical, whether the x is a dissimilarity matrix
Rowv	one of the following: TRUE, a 'dend' object, a vector or NULL/FALSE; determines if and how the *row* dendrogram should be reordered.
Colv	one of the following: "Rowv", TRUE, a 'dend' object, a vector or NULL/FALSE; determines if and how the *column* dendrogram should be reordered.

dendrogram	character string indicating whether to draw 'none', 'row', 'column' or 'both' dendrograms. Defaults to 'both'.
dist.row	a dist object for *row* observations.
dist.col	a dist object for *column* observations.
dist.FUN	function used to compute the distance (dissimilarity) between both rows and columns. Defaults to gdist.
dist.FUN.MoreArgs	a list of other arguments to be passed to gdist
hclust.row	a hclust object (as produced by hclust) for *row* observations.
hclust.col	a hclust object (as produced by hclust) for *column* observations.
hclust.FUN	function used to compute the hierarchical clustering when "Rowv" or "Colv" are not dendrograms. Defaults to hclust.
hclust.FUN.MoreArgs	a list of other arguments to be passed to hclust. Defaults to list(method="ward")
scale	character indicating if the values should be centered and scaled in either the row direction or the column direction, or none. The default is "none".
na.rm	logical, whether NA values will be removed when scaling.
cluster.by.row	logical, whether to cluster *row* observations and reorder the input accordingly.
cluster.by.col	logical, whether to cluster *column* observations and reorder the input accordingly.
kr	numeric, number of clusters in rows; suppressed when row.cluster is specified. DEFAULT: NULL.
kc	numeric, number of clusters in columns; suppressed when col.cluster is specified. DEFAULT: NULL.
row.clusters	a numerical vector, indicating the cluster labels of *row* observations.
col.clusters	a numerical vector, indicating the cluster labels of *column* observations.
revR	logical indicating if the row order should be 'rev'ersed for plotting.
revC	logical indicating if the column order should be 'rev'ersed for plotting, such that e.g., for the symmetric case, the symmetry axis is as usual.
add.expr	expression that will be evaluated after the call to image. Can be used to add components to the plot.
breaks	numeric, either a numeric vector indicating the splitting points for binning x into colors, or a integer number of break points to be used, in which case the break points will be spaced equally between range(x). DEFAULT: 16 when not specified.
x.center	numeric, a value of x for centering colors to
color.FUN	function or function name in characters, for colors in the heatmap
sepList	a list of length 2 giving the row and column lines of separation.
sep.color	color for lines of separation.
sep.lty	line type for lines of separation.

sep.lwd	line width for lines of separation.
cellnote	(optional) matrix of character strings which will be placed within each color cell, e.g. cell labels or p-value symbols.
cex.note	relative font size of cellnote.
notecol	color of cellnote.
na.color	Color to use for missing value (NA). Defaults to the plot background color.
trace	character string indicating whether a solid "trace" line should be drawn across "row"s or down "column"s, "both" or "none". The distance of the line from the center of each color-cell is proportional to the size of the measurement. Defaults to "none".
tracecol	character string giving the color for "trace" line. Defaults to "cyan";
hline	Vector of values within cells where a horizontal dotted line should be drawn. only plotted if 'trace' is 'row' or 'both'. Default to the median of the breaks.
vline	Vector of values within cells where a vertical dotted line should be drawn; only drawn if 'trace' 'column' or 'both'. vline default to the median of the breaks.
linecol	the color of hline and vline. Defaults to the value of 'tracecol'.
labRow	character vectors with row labels to use; defaults to rownames(x).
labCol	character vectors with column labels to use; defaults to colnames(x).
srtRow	numerical, specifying (in degrees) how row labels should be rotated. See help("par", package="graphi
srtCol	numerical, specifying (in degrees) how col labels should be rotated. See help("par", package="graphi
sideRow	2 or 4, which side row labels display.
sideCol	1 or 3, which side row labels display.
margin.for.labRow	a numerical value gives the margin to plot labRow.
margin.for.labCol	a numerical value gives the margin to plot labCol.
ColIndividualColors	(optional) character vector of length ncol(x) containing the color names for a horizontal side bar that may be used to annotate the columns of x.
RowIndividualColors	(optional) character vector of length nrow(x) containing the color names for a vertical side bar that may be used to annotate the rows of x.
cexRow	positive numbers, used as 'cex.axis' in for column axis labeling. The default currently only uses number of columns.
cexCol	positive numbers, used as 'cex.axis' in for the row axis labeling. The default currently only uses number of rows.
labRow.by.group	logical, whether group unique labels for rows.
labCol.by.group	logical, whether group unique labels for columns.
key	logical indicating whether a color-key should be shown.

key.title	character, title of the color-key ["Color Key"]
key.xlab	character, xlab of the color-key ["Value"]
key.ylab	character, ylab of the color-key ["Count"]
keysize	numeric value indicating the relative size of the key
mapsize	numeric value indicating the relative size of the heatmap.
mapratio	the width-to-height ratio of the heatmap.
sidesize	numeric value indicating the relative size of the sidebars.
cex.key.main	a numerical value giving the amount by which main-title of color-key should be magnified relative to the default.
cex.key.xlab	a numerical value giving the amount by which xlab of color-key should be magnified relative to the default.
cex.key.ylab	a numerical value giving the amount by which ylab of color-key should be magnified relative to the default.
density.info	character string indicating whether to superimpose a 'histogram', a 'density' plot, or no plot ('none') on the color-key.
denscol	character string giving the color for the density display specified by 'density.info', defaults to the same value as 'tracecol'.
densadj	Numeric scaling value for tuning the kernel width when a density plot is drawn on the color key. (See the 'adjust' parameter for the 'density' function for details.) Defaults to 0.25.
main	an overall title for the plot. See help("title", package="graphics").
sub	a subtitle for the plot, describing the distance and/or alignment gap (the "shift").
xlab	a title for the x axis. See help("title", package="graphics").
ylab	a title for the y axis. See help("title", package="graphics").
cex.main	a numerical value giving the amount by which main-title should be magnified relative to the default.
cex.sub	a numerical value giving the amount by which sub-title should be magnified relative to the default.
font.main	An integer which specifies which font to use for main-title.
font.sub	An integer which specifies which font to use for sub-title.
adj.main	The value of 'adj' determines the way in which main-title strings are justified.
mgp.main	the margin line (in 'mex' units) for the main-title.
mar.main	a numerical vector of the form c(bottom, left, top, right) which gives the number of lines of margin to be specified on the four sides of the main-title.
mar.sub	a numerical vector of the form c(bottom, left, top, right) which gives the number of lines of margin to be specified on the four sides of the sub-title.
if.plot	logical, whether to plot. Reordered matrix is returned without graphical output if FALSE.
plot.row.partition	logical, whether to plot *row* partition.

`plot.col.partition`
: logical, whether to plot *column* partition.

`cex.partition`
: a numerical value giving the amount by which `partition` should be magnified relative to the default.

`color.partition.box`
: color for the `partition` box.

`color.partition.border`
: color for the `partition` border.

`plot.row.individuals`
: logical, whether to make a plot of *row* observations.

`plot.col.individuals`
: logical, whether to make a plot of *column* observations.

`plot.row.clusters`
: logical, whether to make a summary plot of *row* clusters.

`plot.col.clusters`
: logical, whether to make a summary plot of *column* clusters.

`plot.row.clustering`
: logical, whether to make a summary plot of overall *row* clustering.

`plot.col.clustering`
: logical, whether to make a summary plot of overall *column* clustering.

`plot.row.individuals.list`
: a list of expressions that is used to `plot.row.individuals`

`plot.col.individuals.list`
: a list of expressions that is used to `plot.col.individuals`

`plot.row.clusters.list`
: a list of expressions that is used to `plot.row.clusters`

`plot.col.clusters.list`
: a list of expressions that is used to `plot.col.clusters`

`plot.row.clustering.list`
: a list of expressions that is used to `plot.row.clustering`

`plot.col.clustering.list`
: a list of expressions that is used to `plot.col.clustering`

`row.data`
: (optional) data used to `plot.row.individuals`, `plot.row.clusters` or `plot.row.clustering`

`col.data`
: (optional) data used to `plot.col.individuals`, `plot.col.clusters` or `plot.col.clustering`

`if.plot.info`
: logical, whether to plot `text.box`.

`text.box`
: character plotted when `if.plot.info` is TRUE.

`cex.text`
: a numerical value giving the amount by which `text.box` should be magnified relative to the default.

`...`
: arguments to be passed to method `heatmap.3`.
e `help("image", package="graphics")`.

Details

Enhanced heatmap representation with partition and summary statistics (optional). This is an enhanced version of 'heatmap.2' function in the Package **gplots**. The enhancement includes: 1) Improved performance with optional input of precomputed `dist` object and `hclust` object. 2) Highlight of specific cells using rectangles. For instance, the cells of clusters of interests. (Examples should be included in future.) 3) Add-on plots in addition to the heatmap, such as cluster-wise summary plots and overall clustering summary plots, to the right of or under the heatmap.

Value

A reordered matrix according to *row* or/and *col* dendrogram(s) and indices that used for reordering.

Examples

```
## ------------------------------------------------------------------------
## Example1: mtcars
## ------------------------------------------------------------------------
## load library
require("GMD")

## load data
data(mtcars)

## heatmap on raw data
x  <- as.matrix(mtcars)

dev.new(width=10,height=8)
heatmap.3(x)                              # default, with reordering and dendrogram
## Not run:
heatmap.3(x, Rowv=FALSE, Colv=FALSE)      # no reordering and no dendrogram
heatmap.3(x, dendrogram="none")           # reordering without dendrogram
heatmap.3(x, dendrogram="row")        # row dendrogram with row (and col) reordering
heatmap.3(x, dendrogram="row", Colv=FALSE) # row dendrogram with only row reordering
heatmap.3(x, dendrogram="col")            # col dendrogram
heatmap.3(x, dendrogram="col", Rowv=FALSE) # col dendrogram with only col reordering
heatmapOut <-
  heatmap.3(x, scale="column")            # sacled by column
names(heatmapOut)                         # view the list that is returned
heatmap.3(x, scale="column", x.center=0)   # colors centered around 0
heatmap.3(x, scale="column",trace="column")  # trun "trace" on

## End(Not run)

## coloring cars (row observations) by brand
brands <- sapply(rownames(x), function(e) strsplit(e," ")[[1]][1])
names(brands) <- c()
brands.index <- as.numeric(as.factor(brands))
RowIndividualColors <- rainbow(max(brands.index))[brands.index]
heatmap.3(x, scale="column", RowIndividualColors=RowIndividualColors)

## coloring attributes (column features) randomly (just for a test :)
```

```
heatmap.3(x, scale="column", ColIndividualColors=rainbow(ncol(x)))

## add a single plot for all row individuals
dev.new(width=12,height=8)
expr1 <- list(quote(plot(row.data[rowInd,"hp"],1:nrow(row.data),
xlab="hp",ylab="",yaxt="n",main="Gross horsepower")),
quote(axis(2,1:nrow(row.data),rownames(row.data)[rowInd],las=2)))
heatmap.3(x, scale="column", plot.row.individuals=TRUE, row.data=x,
          plot.row.individuals.list=list(expr1))


## ------------------------------------------------------------------------
## Example2: ruspini
## ------------------------------------------------------------------------
## load library
require("GMD")
require(cluster)

## load data
data(ruspini)

## heatmap on a `dist' object
x <- gdist(ruspini)
main <- "Heatmap of Ruspini data"
dev.new(width=10,height=10)
heatmap.3(x, main=main, mapratio=1) # with a title and a map in square!
## Not run:
heatmap.3(x, main=main, revC=TRUE)  # reverse column for a symmetric look
heatmap.3(x, main=main, kr=2, kc=2) # partition by predefined number of clusters

## End(Not run)
## show partition by elbow
css.multi.obj <- css.hclust(x,hclust(x))
elbow.obj <- elbow.batch(css.multi.obj,ev.thres=0.90,inc.thres=0.05)
heatmap.3(x, main=main, revC=TRUE, kr=elbow.obj$k, kc=elbow.obj$k)

## Not run:
## show elbow info as subtitle
heatmap.3(x, main=main, sub=sub("\n"," ",attr(elbow.obj,"description")),
cex.sub=1.25,revC=TRUE,kr=elbow.obj$k, kc=elbow.obj$k)

## End(Not run)
```

---

legend *Add Legends to Plots*

Description

This function can be used to add legends to plots. Note that a call to the function `locator(1)` can be used in place of the `x` and `y` arguments.

Usage

```
legend(x, y = NULL, legend, fill = NULL, col = par("col"),
  border = "black", lty, lwd, pch, angle = 45, density = NULL,
  bty = "o", bg = par("bg"), box.lwd = par("lwd"), box.lty = par("lty"),
  box.col = par("fg"), pt.bg = NA, cex = 1, pt.cex = cex,
  pt.lwd = lwd, xjust = 0, yjust = 1, x.intersp = 1, y.intersp = 1,
  adj = c(0, 0.5), text.width = NULL, text.col = par("col"),
  merge = do.lines && has.pch, trace = FALSE, plot = TRUE, ncol = 1,
  horiz = FALSE, title = NULL, inset = 0, xpd, title.col = text.col,
  title.adj = 0.5, seg.len = 2)
```

Arguments

`x`	the x coordinates to be used to position the legend.
`y`	the y coordinates to be used to position the legend. `x` and `y` can be specified by keyword or in any way which is accepted by `xy.coords`: See 'Details'.
`legend`	a character or expression vector. of length ≥ 1 to appear in the legend. Other objects will be coerced by `as.graphicsAnnot`.
`fill`	if specified, this argument will cause boxes filled with the specified colors (or shaded in the specified colors) to appear beside the legend text.
`col`	the color of points or lines appearing in the legend.
`border`	the border color for the boxes (used only if `fill` is specified).
`lty`	the line types for lines appearing in the legend.
`lwd`	the line widths for lines appearing in the legend. One of `lty` and `lwd` *must* be specified for line drawing.
`pch`	the plotting symbols appearing in the legend, either as vector of 1-character strings, or one (multi character) string. *Must* be specified for symbol drawing.
`angle`	angle of shading lines.
`density`	the density of shading lines, if numeric and positive. If `NULL` or negative or `NA` color filling is assumed.
`bty`	the type of box to be drawn around the legend. The allowed values are `"o"` (the default) and `"n"`.
`bg`	the background color for the legend box. (Note that this is only used if `bty != "n"`.)
`box.lwd`	the line type for the legend box.
`box.lty`	the line width for the legend box.
`box.col`	the color for the legend box.
`pt.bg`	the background color for the `points`, corresponding to its argument `bg`.
`cex`	character expansion factor **relative** to current `par("cex")`.
`pt.cex`	expansion factor(s) for the points.
`pt.lwd`	line width for the points, defaults to the one for lines, or if that is not set, to `par("lwd")`.

xjust	how the legend is to be justified relative to the legend x location. A value of 0 means left justified, 0.5 means centered and 1 means right justified.
yjust	the same as xjust for the legend y location.
x.intersp	character interspacing factor for horizontal (x) spacing.
y.intersp	the same for vertical (y) line distances.
adj	numeric of length 1 or 2; the string adjustment for legend text. Useful for y-adjustment when labels are plotmath expressions.
text.width	the width of the legend text in x ("user") coordinates. (Should be positive even for a reversed x axis.) Defaults to the proper value computed by strwidth(legend).
text.col	the color used for the legend text.
merge	logical; if TRUE, merge points and lines but not filled boxes. Defaults to TRUE if there are points and lines.
trace	logical; if TRUE, shows how legend does all its magical computations.
plot	logical. If FALSE, nothing is plotted but the sizes are returned.
ncol	the number of columns in which to set the legend items (default is 1, a vertical legend).
horiz	logical; if TRUE, set the legend horizontally rather than vertically (specifying horiz overrides the ncol specification).
title	a character string or length-one expression giving a title to be placed at the top of the legend. Other objects will be coerced by as.graphicsAnnot.
inset	inset distance(s) from the margins as a fraction of the plot region when legend is placed by keyword.
xpd	if supplied, a value of the graphical parameter 'xpd' to be used while the legend is being drawn.
title.col	color for title.
title.adj	horizontal adjustment for title: see the help for par("adj").
seg.len	the length of lines drawn to illustrate lty and/or lwd (in units of character widths).

Details

see legend in package:graphics for details; Note: Old versions of graphics:::legend do not have 'border' option.

`mhist.summary` *Bin-wise summary of histograms*

Description

Bin-wise summary of a mhist object of histograms

Usage

```
mhist.summary(h, ...)

## S3 method for class 'mhist.summary'
plot(x,bins,plot.ci=TRUE,col=NULL,
ci.color="orchid1",tcl=-0.25,omi=c(0.5,0.5,1.0,0.25),mar=c(3,3,3,1),
mgp=c(2,0.5,0),if.plot.new=TRUE,...)
```

Arguments

h	a "mhist" object as produced by as.mhist
x	a mhist.summary object as produced by mhist.summary
bins	character vecter, the bin labels; if non-specific, bins are numbered/labeled starting with one.
plot.ci	logical, indicating whether plot error bars that represent the 0.50 confidence interval (CI)
col	color of the histogram
ci.color	color of the error bars
tcl	the length of tick marks as a fraction of the height of a line of text. See option tcl inhelp("par", package="graphics").
omi	a vector of the form 'c(bottom, left, top, right)' giving the size of the outer margins in inches. See option omi inhelp("par", package="graphics").
mar	a numerical vector of the form c(bottom, left, top, right) which gives the number of lines of margin to be specified on the four sides of the plot. See option mar inhelp("par", package="graphics").
mgp	the margin line (in 'mex' units) for the axis title, axis labels and axis line.
if.plot.new	logical, whether starting a new device or not.
...	arguments to be passed to method plot.mhist.summary. See help("barplot2", package="gplots").

Details

Bin-wise summary of a mhist object of histograms

Value

mhist.summary returns a mhist.summary object

See Also

mhist plot.mhist plot.gmdp plot.gmdm

plot.gmdm *S3 method for class 'gmdm'*

Description

S3 method for class gmdm

Usage

```
## S3 method for class 'gmdm'
plot(x, labels, colors, type = NULL, main, ylab = "Fraction",
  xlab = "Position", label.length.max = 8, label.line.max = 3,
  cex.text = 1, cex.tickmark = 0.75, if.plot.new = TRUE, ...)
```

Arguments

x	an object of class gmdm.
labels	a string vector of the same length as x$data, giving the names of the numeric vectors in x$data.
colors	the colors of the discrete distributions; the default is *"Dark2" colors in ColorBrewer palettes* if not specified.
type	type of plot, as in help("plot", package="graphics").
main	an overall title for the plot. See help("title", package="graphics"); the default title is used if not specified.
ylab	a title for the y axis. See help("title", package="graphics").
xlab	a title for the x axis. See help("title", package="graphics").
label.length.max	numeric, giving the maximum string width allowed in diagonal labels.
label.line.max	numeric, giving the maximum number of lines allowed in diagonal labels.
cex.text	a numerical value giving the amount by which plot text should be magnified relative to the default.
cex.tickmark	a numerical value giving the amount by which tickmarks should be magnified relative to the default.
if.plot.new	logical, indicating whether to start a new plot device.
...	arguments to be passed to methods, see gmdp.

Details

S3 method for class gmdm

References

See help(GMD)

See Also

gmdm, gmdp

Examples

```
## ------------------------------------------------------------------------
## Example1: CAGE
## ------------------------------------------------------------------------
require("GMD") # load library
data(cage)     # load data

## construct a distance matrix and visualize it
short.labels <- gsub("(.+) \\(.+","\\1",names(cage)) # get short labels
x <- gmdm(cage[1:6],labels=short.labels[1:6])
plot(x)

## Not run:
## ------------------------------------------------------------------------
## Example2: ChIP-seq
## ------------------------------------------------------------------------
data(chipseq_mES)   # load data
data(chipseq_hCD4T) # load data

## pairwise distance and alignment based on GMD metric
plot(gmdm(chipseq_mES,sliding=FALSE))

## clustering on spatial distributions of histone modifications
x <- gmdm(chipseq_hCD4T,sliding=FALSE,resolution=10)
heatmap.3(x,revC=TRUE)

## End(Not run)
```

---

plot.gmdp *Plot function for class gmdp*

Description

Plot Function for Class gmdp

Usage

```
## S3 method for class 'gmdp'
plot(x, labels = NULL, colors = NULL, main,
  ylab = "Fraction", xlab = "Position", xlim = NULL, type = NULL,
  if.text.gmd = TRUE, if.text.gap = TRUE, ...)
```

Arguments

x	an object of class gmdp.
labels	a string vector of the same length of x$labels, giving the names of the numeric vectors in x.
colors	the colors of the discrete distributions. See help("plot.mhist", package="GMD").
main	an overall title for the plot.
ylab	a title for the y axis. See help("plot.mhist", package="GMD").
xlab	a title for the x axis. See help("plot.mhist", package="GMD").
xlim	numeric vectors of length 2, giving the x coordinates ranges.
type	type of plot, as in help("plot", package="graphics").
if.text.gmd	logical, indicating whether *GM-Distance* is reported in the subtitle.
if.text.gap	logical, indicating whether *gap* is reported in the subtitle.
...	arguments to be passed to methods. See help("plot.mhist", package="GMD").

Details

Plot Function for Class gmdp

References

See help(GMD)

See Also

gmdp

Examples

```
require("GMD") # load library
data(cage)     # load data

## measure pairwise distance
x <- gmdp(cage[["Pfkfb3 (T02R00AEC2D8)"]],cage[["Csf1 (T03R0672174D)"]])
print(x)                    # print a brief version by default
print(x, mode="full")  # print a full version by default

## show alignment
plot(x,labels=c("Pfkfb3","Csf1"),beside=FALSE)

## show another alignment
plot(gmdp(cage[["Hig1 (T09R0743763C)"]],cage[["Cd72 (T04R028B8BC9)"]]),
     labels=c("Hig1 (T09R0743763C)","Cd72 (T04R028B8BC9)"),
     beside=FALSE)
```

plot.mhist *S3 method for class 'mhist'.*

Description

S3 method for class mhist

Usage

```
## S3 method for class 'mhist'
plot(x, beside = TRUE, labels = NULL, colors = NULL,
  main = NULL, sub = NULL, ylab = NULL, xlab = NULL, xticks = NULL,
  xlabels = NULL, vlinePos = NULL, x.las = 1, xticks.type = c("pretty",
  "original"), xlim = NULL, ylim = NULL, type = NULL, font.type = 1,
  font.family = c("sans", "serif", "mono"), cex.main = 1.75,
  cex.sub = cex.main * 0.9, cex.lab = 1.25, cex.tickmark = 0.75,
  cex.legend = 1.5, tcl = -0.25, omi = c(0.5, 0.5, 1, 0.25), mar = c(4,
  1, 0, 1), mgp = c(0, 0.5, 0), bin.unit = 0.8, legend.lab = labels,
  legend.pos = c("topright", "top", "topleft"), ...)
```

Arguments

x	a numeric matrix or data frame, representing distributions by rows (bins by columns); or a list of numeric vectors as distributions.
beside	logical, whether plot histograms side-by-side.
labels	a string vector of labels for the histograms in x; should have the same number as of the histograms.
colors	the colors for the histograms; by default they are set to colors generated from palette Dark2. Colors will be recycled if the size is smaller than the number of the histograms.
main	an overall title for the plot. See help("title", package="graphics").
sub	a subtitle for the plot, describing the distance and/or alignment gap (the "shift").
ylab	a title for the y axis. See help("title", package="graphics").
xlab	a title for the x axis. See help("title", package="graphics").
xticks	a string vector indicating the tickmark labels at x-axis. Defult: NULL.
xlabels	character, labels at x-axis.
vlinePos	numeric, posiitons for vertical lines.
x.las	numeric in 0,1,2,3; the style of axis labels. See option las inhelp("par", package="graphics").
xticks.type	stinrg in "pretty","original", whether plot the xticks in a pretty way or as is.
xlim	range of x values, as in help("plot", package="graphics").
ylim	range of y values, as in help("plot", package="graphics").
type	type of plot, as in help("plot", package="graphics").

font.type	the name of a font type for drawing text. See font in par. DEFAULT: font.type = 1, corresponding to plain text.
font.family	the name of a font family for drawing text. See family in par; DEFAULT: font.family = "sans", corresponding to san serif typeface.
cex.main	a numerical value giving the amount by which main-title should be magnified relative to the default.
cex.sub	a numerical value giving the amount by which sub-title should be magnified relative to the default.
cex.lab	a numerical value giving the amount by which xlab and ylab should be magnified relative to the default.
cex.tickmark	a numerical value giving the amount by which tickmarks should be magnified relative to the default.
cex.legend	a numerical value giving the amount by which legends should be magnified relative to the default.
tcl	the length of tick marks as a fraction of the height of a line of text. See option tcl inhelp("par", package="graphics").
omi	a vector of the form 'c(bottom, left, top, right)' giving the size of the outer margins in inches. See option omi inhelp("par", package="graphics").
mar	a numerical vector of the form c(bottom, left, top, right) which gives the number of lines of margin to be specified on the four sides of the plot. See option mar inhelp("par", package="graphics").
mgp	the margin line (in 'mex' units) for the axis title, axis labels and axis line. See option mgp inhelp("par", package="graphics").
bin.unit	numeric, indicating the width of a group of bar(s) in unit of x axis.
legend.lab	legend labels, a string vector of the same length of distributions in x, using labels by default. No legend is displayed when it is NA.
legend.pos	string, a keyword to be used to position the legend. See help("legend", package="graphics").
...	arguments to be passed to method plot.mhist, such as graphical parameters (see par).

Details

Given a list, matrix or data.frame of histograms, plot multiple histograms side-by-side or as subplots.

References

See help(GMD)

See Also

mhist mhist.summary plot.mhist.summary plot.gmdp plot.gmdm

Examples

```
## load library
require("GMD")

## create two normally-distributed samples
## with unequal means and unequal variances
set.seed(2012)
v1 <- rnorm(1000,mean=-5, sd=10)
v2 <- rnorm(1000,mean=10, sd=5)

## create common bins
n <- 20 # desired number of bins
breaks <- gbreaks(c(v1,v2),n) # bin boundaries
x <-
  list(ghist(v1,breaks=breaks,digits=0),
       ghist(v2,breaks=breaks,digits=0))
mhist.obj <- as.mhist(x)

## plot histograms side-by-side
plot(mhist.obj,mar=c(1.5,1,1,0),
main="Histograms of simulated normal distributions")

## plot histograms as subplots,
## with corresponding bins aligned
plot(mhist.obj,beside=FALSE,mar=c(1.5,1,1,0),
     main="Histograms of simulated normal distributions")
```

---

ts2df *Convert time series to data frame*

Description

A copy of wq:::ts2df; see `ts2df` in package:wq for details

Usage

```
ts2df(x, mon1 = 1, addYr = FALSE, omit = FALSE)
```

Arguments

`x`	monthly time series vector
`mon1`	starting month number, i.e., first column of the data frame
`addYr`	rows are normally labelled with the year of the starting month, but `addYr = TRUE` will add 1 to this year number
`omit`	if `TRUE`, then rows with any `NA` will be removed.

Details

see `ts2df` in package:wq for details. Note: wq_0.3-4 asks for R (>= 2.12.0); but GMD supports R (>= 2.9.0).

Index