Package 'MosaiClusteR'

October 21, 2026

Type Package

Title An Umbrella Framework for Multi-Source and Multi-Omics Clustering

Version 0.1.4

Description An umbrella framework (``MoSaIC:" Multi-Omics Similarity Aggregation and Integrative Clustering in R) that unifies a large collection of multi-source / multi-omics clustering methodologies behind a single, consistent list-of-matrices interface. It spans five integration paradigms - direct, similarity-based, graph-based, voting-based consensus, and hierarchy-based - and bundles a complete downstream workflow for method comparison and evaluation. The package features the multi-source the ability to compare many algorithms on the same footing, a data-nugget based feature-weighting scheme as a robust, big-data-friendly alternative to variance weighting, and a downstream suite for cluster characterisation, visualisation and biological interpretation.

License CC BY-SA 4.0

Encoding UTF-8

LazyData false

Depends R (>= 4.0)

Imports stats, utils, methods, graphics, grDevices, cluster, fastcluster, matrixStats, FD, ade4, e1071, gtools, data.table, Rcpp, Rdpack

LinkingTo Rcpp

Suggests SNFtool, datanugget, WCluster, LUCIDus, igraph, lsa, analogue, FactoMineR, pls, circlize, gplots, dendextend, plotrix, limma, MLP, biomaRt, org.Hs.eg.db, Biobase, a4Core, a4Base, plyr, genefilter, ggplot2, gridExtra, uwot, Rtsne, MOFAdata, testthat (>= 3.0.0), knitr, rmarkdown

RdMacros Rdpack

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/lucp12891/MosaiClusteR

BugReports https://github.com/lucp12891/MosaiClusteR/issues

NeedsCompilation yes

Config/roxygen2/version 8.0.0

Author Bernard Isekah Osang'ir [aut, cre] (ORCID:
<https://orcid.org/0000-0002-5557-3602>),
Marijke Van Moerbeke [aut],
Ziv Shkedy [ctb],
Surya Gupta [ctb],
Jürgen Claesen [ctb]

Maintainer Bernard Isekah Osang'ir <b.osangir@gmail.com>

Repository CRAN

Date/Publication 2026-10-21 13:20:40 UTC

Contents

ABCdeep.SingleInMultiple 4
ABCdist.SingleInMultiple 6
ABCpp.SingleInMultiple 8
ADC 10
ADEC 11
BinFeaturesPlot_MultipleData 13
BinFeaturesPlot_SingleData 15
BoxPlotDistance 17
CEC 19
CharacteristicFeatures 21
characterize_clusters 22
ChooseCluster 23
Cluster 25
ClusterCols 26
ClusteringAggregation 26
clustering_concordance 29
ClusterPlot 29
cluster_agreement 31
cluster_biology_panel 32
cluster_k_sweep 32
cluster_selection_plot 34
ColorPalette 35
ColorsNames 35
CompareInteractive 36
ComparePlot 38
CompareSilCluster 39
CompareSvsM 41
compare_clusterings 42
compare_methods_plot 43
ConsensusClustering 44
ContFeaturesPlot 46

create_data_nuggets . . . 48
CVAA . . . 49
Cyclogram . . . 51
DetermineWeight_SilClust . . . 53
DetermineWeight_SimClust . . . 55
DiffGenes . . . 57
DiffGenesSelection . . . 58
Distance . . . 60
distanceheatmaps . . . 61
EHC . . . 61
embedding_plot . . . 64
enriched_features . . . 65
EnsembleClustering . . . 66
EvidenceAccumulation . . . 68
f.clustABC.MultiSource . . . 70
FeatSelection . . . 71
FeaturesOfCluster . . . 72
FindCluster . . . 73
FindElement . . . 74
FindGenes . . . 75
Geneset.intersect . . . 76
Geneset.intersectSelection . . . 77
HBGF . . . 78
HeatmapPlot . . . 80
HeatmapSelection . . . 81
HierarchicalEnsembleClustering . . . 83
intNMF . . . 85
LabelCols . . . 86
LabelPlot . . . 86
LinkBasedClustering . . . 87
LUCID . . . 89
mosaic_labels . . . 91
mosaic_sim . . . 91
mosaic_toy . . . 92
M_ABC . . . 93
M_ABCdeep . . . 95
M_ABCdist . . . 96
M_ABCdist.WC . . . 98
NEMO . . . 100
Normalization . . . 101
nugget_cluster . . . 102
nugget_feature_weights . . . 103
PathwayAnalysis . . . 104
Pathways . . . 105
PathwaysIter . . . 108
PathwaysSelection . . . 110
PlotPathways . . . 111
plot_cluster_profile . . . 113

plot_pathways . . . 113
PreparePathway . . . 115
ProfilePlot . . . 116
ReorderToReference . . . 117
SelectnrClusters . . . 119
SharedComps . . . 120
SharedGenesPathsFeat . . . 121
SharedSelection . . . 122
SharedSelectionLimma . . . 123
SharedSelectionMLP . . . 124
SimilarityHeatmap . . . 124
SimilarityMeasure . . . 126
simulate_weighting_regime . . . 127
SNF . . . 129
spectral_clustering . . . 131
TrackCluster . . . 132
WeightedClust . . . 133
Whclust . . . 135
Wkmeans . . . 136
WonM . . . 137

Index **139**

`ABCdeep.SingleInMultiple` *Single-source ABC with a deep-learning (autoencoder) Step 4 (ABCdeep)*

Description

A deep-learning variant of `ABCpp.SingleInMultiple` that keeps the ABC algorithm of (Amaratunga et al. 2008) intact and replaces **only Step 4**: instead of clustering the raw feature sub-matrix, a neural autoencoder embeds the objects into a latent space and the latent vectors are clustered into `NC` base clusters. The stacked label matrix is fused across sources by `f.clustABC.MultiSource`, exactly as in `M_ABC`.

Usage

```
ABCdeep.SingleInMultiple(
  data,
  weighting = "var",
  normalize = NULL,
  gr = c(),
  bag = TRUE,
  numsim = 1000,
  numvar = 100,
  NC = NULL,
  topk_frac = 0.2,
```

```
  latent_dim = "auto",
  hidden_units = 32L,
  ae_epochs = 80L,
  ae_lr = 0.001,
  cluster_in_latent = "ward",
  linkage = "ward.D2",
  nugget_type = "between",
  nugget_args = list(),
  min_samples = 40L,
  seed = NULL
)
```

Arguments

`data`	Numeric matrix, **objects (samples) in rows**, features in columns (the MosaiCluster convention).
`weighting`	Feature weighting used to rank features for the top-K focus: `"var"`, `"cv"`, `"nugget"` or equal (as in `ABCpp.SingleInMultiple`).
`normalize`	Optional normalisation applied before the internal min-max scaling (see `Normalization`); `NULL`/`FALSE` = none.
`gr`	Unused; kept for interface compatibility.
`bag`	Logical; bootstrap the objects each iteration (Step 2).
`numsim`	Number of iterations R.
`numvar`	Kept for backward compatibility; unused.
`NC`	Base-cluster count (`NULL` = $\lfloor\sqrt{N}\rfloor$).
`topk_frac`	Fraction of the highest-weighted features the autoencoder is trained/embedded on (default 0.2). Focusing on the informative features drops the noise that would dilute a full-feature embedding on sparse signals, while keeping enough features for dense signals; the count is floored at `min(G, 20)` and capped at `min(G, 4000)`.
`latent_dim`	Autoencoder bottleneck size: an integer, or `"auto"` (default) = `max(NC + 2, 8)`, capped at the hidden width and $K - 1$.
`hidden_units`	Hidden-layer width of the encoder/decoder.
`ae_epochs`	Training epochs for the one-time autoencoder fit.
`ae_lr`	Adam learning rate.
`cluster_in_latent`	Latent clustering (Step 4b): `"ward"` (default, hierarchical) or `"kmeans"`.
`linkage`	Agglomeration method when `cluster_in_latent = "ward"`.
`nugget_type, nugget_args`	Passed to the nugget weighting when `weighting = "nugget"`.
`min_samples`	Emit a warning when `nrow(data)` is below this (the autoencoder is underdetermined at very small N); default 40.
`seed`	Optional RNG seed (controls the autoencoder init and the bootstrap).

Value

A data frame, `numsim` rows by one column per object; entries are base-cluster labels (`0` = object not selected that iteration) – the same contract as `ABCpp.SingleInMultiple`.

Speed (trained once, forward pass per iteration)

The autoencoder is trained a single time on the source's top-`topk_frac` most-informative features; each of the `numsim` iterations then performs only a forward pass over the bootstrapped objects. The network is a self-contained, vectorised R autoencoder (He initialisation, ReLU, Adam, MSE) – there is no Python, TensorFlow or torch dependency, and **no PCA or other fallback**: a single code path.

References

Amaratunga D, Cabrera J, Kovtun V (2008). "Microarray Learning with ABC." *Biostatistics*, **9**(1), 128–136. doi:10.1093/biostatistics/kxm017. https://academic.oup.com/biostatistics/article-abstract/9/1/128/254198.

See Also

`M_ABCdeep`, `ABCpp.SingleInMultiple`, `f.clustABC.MultiSource`

Examples

```
data(mosaic_toy)
lab <- ABCdeep.SingleInMultiple(mosaic_toy$List[[1]], numsim = 40, NC = 3,
                                ae_epochs = 40, seed = 1)
dim(lab)
```

`ABCdist.SingleInMultiple`

Single-source ABC with distance accumulation

Description

A variant of `ABCpp.SingleInMultiple` that accumulates the mean scaled pairwise distance between co-selected objects across iterations, yielding a `[0, 1]` dissimilarity matrix that `M_ABCdist` fuses across sources.

Usage

```
ABCdist.SingleInMultiple(
  data,
  distmeasure = "euclidean",
  weighting = "var",
  normalize = NULL,
  gr = c(),
```

```
  bag = TRUE,
  numsim = 1000,
  numvar = 100,
  linkage = "ward.D",
  NC = NULL,
  mds = FALSE,
  nfeat = NULL,
  nugget_type = "between",
  nugget_args = list()
)
```

Arguments

`data`	Numeric matrix with **objects (samples) in rows** and features in columns (the MosaiClusteR convention; transposed internally to the features-by-objects layout of the original engine).
`distmeasure`	Distance measure passed to `Distance`.
`weighting`	`"var"`, `"cv"`, `"nugget"` or equal.
`normalize`	Optional normalisation (see `Normalization`).
`gr`	Unused; kept for interface compatibility.
`bag`	Logical; bootstrap the objects each iteration (Step 2).
`numsim`	Number of iterations R.
`numvar`	Kept for backward compatibility; superseded by `nfeat`.
`linkage`	Agglomeration method for `fastcluster::hclust`.
`NC`	Base-cluster count (`NULL` = $\lfloor\sqrt{N}\rfloor$).
`mds`	Unused placeholder.
`nfeat`	Optional feature-subsample size. `NULL` (default) uses the classic $g = \lfloor\sqrt{G}\rfloor$; a value ≥ 1 sets an absolute count (raise it for low-dimensional sources where most features matter).
`nugget_type`, `nugget_args`	Passed to the data-nugget weighting when `weighting = "nugget"`.

Value

A list with `D` ($N \times N$ dissimilarity), `NC` and `ng` (per-iteration feature count); if `mds = TRUE` also `mds_coords`.

References

(Amaratunga et al. 2008)

See Also

`M_ABCdist`, `ABCpp.SingleInMultiple`

Examples

```
data(mosaic_toy)
out <- ABCdist.SingleInMultiple(mosaic_toy$List[[1]], numsim = 40)
dim(out$D)
```

`ABCpp.SingleInMultiple` *Single-source Aggregating Bundles of Clusters (ABC), C++-accelerated engine*

Description

The canonical single-source ABC engine of (Amaratunga et al. 2008). Over `numsim` iterations it (optionally) bootstraps the objects, draws a weighted subsample of features, clusters the sub-matrix with `fastcluster::hclust`, and records each selected object's base-cluster label. `M_ABC` turns the stacked label matrix into a consensus dissimilarity using the C++ co-clustering kernel.

Usage

```
ABCpp.SingleInMultiple(
  data,
  distmeasure = "euclidean",
  weighting = "var",
  normalize = NULL,
  gr = c(),
  bag = TRUE,
  numsim = 1000,
  numvar = 100,
  linkage = "ward.D2",
  NC = NULL,
  mds = FALSE,
  nfeat = NULL,
  nugget_type = "between",
  nugget_args = list()
)
```

Arguments

`data`	Numeric matrix with **objects (samples) in rows** and features in columns (the MosaiClusteR convention; transposed internally to the features-by-objects layout of the original engine).
`distmeasure`	Distance measure passed to `Distance`.
`weighting`	`"var"`, `"cv"`, `"nugget"` or equal.
`normalize`	Optional normalisation (see `Normalization`).
`gr`	Unused; kept for interface compatibility.

`bag`	Logical; bootstrap the objects each iteration (Step 2).
`numsim`	Number of iterations R.
`numvar`	Kept for backward compatibility; superseded by `nfeat`.
`linkage`	Agglomeration method for `fastcluster::hclust`.
`NC`	Base-cluster count (`NULL` = $\lfloor\sqrt{N}\rfloor$).
`mds`	Unused placeholder.
`nfeat`	Optional feature-subsample size. `NULL` (default) uses the classic $g = \lfloor\sqrt{G}\rfloor$; a value ≥ 1 sets an absolute count (raise it for low-dimensional sources where most features matter).
`nugget_type, nugget_args`	Passed to the data-nugget weighting when `weighting = "nugget"`.

Value

A data frame, `numsim` rows by one column per object; entries are base-cluster labels (`0` = object not selected that iteration).

Feature weighting (incl. data nuggets)

Features are subsampled with probability proportional to a rank-transformed importance score:

- `"var"` – feature variance (the classic ABC choice);
- `"cv"` – coefficient of variation;
- `"nugget"` – *NEW*: the data-nugget between-group weighted variance (`create_data_nuggets`), a robust, Big-Data-friendly alternative to the raw variance;
- anything else – equal weights.

References

Amaratunga D, Cabrera J, Kovtun V (2008). "Microarray Learning with ABC." *Biostatistics*, **9**(1), 128–136. doi:10.1093/biostatistics/kxm017. https://academic.oup.com/biostatistics/article-abstract/9/1/128/254198.

See Also

`M_ABC`, `create_data_nuggets`

Examples

```
data(mosaic_toy)
lab <- ABCpp.SingleInMultiple(mosaic_toy$List[[1]], numsim = 40, NC = 3,
                              weighting = "nugget")
dim(lab)
```

ADC *Aggregated data clustering*

Description

Aggregated Data Clustering (ADC) is a direct clustering multi-source technique. The data matrices are merged into a single fused data matrix on which a dissimilarity matrix is computed. Hierarchical clustering is then performed on this dissimilarity matrix.

Usage

```
ADC(
  List,
  distmeasure = "tanimoto",
  normalize = FALSE,
  method = NULL,
  clust = "agnes",
  linkage = "flexible",
  alpha = 0.625
)
```

Arguments

List	A list of data matrices of the same type. It is assumed the rows are corresponding with the objects.
distmeasure	Choice of metric for the dissimilarity matrix (character). Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to "tanimoto".
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to FALSE. This is recommended if different distance types are used. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is NULL.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character). Defaults to "flexible".
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".

Value

The returned value is a list with the following three elements.

AllData	Fused data matrix of the data matrices
DistM	The resulting distance matrix
Clust	The resulting clustering

The value has class 'ADC'. The Clust element will be of interest for further applications.

References

Fodeh, S. J., Punch, W. & Tan, P.-N. (2013). On unifying multi-source information from probabilistically generated functional data. Proteins 81:1298-1310.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
res_ADC <- ADC(List=L, distmeasure="euclidean", normalize=FALSE, method=NULL,
               clust="agnes", linkage="flexible", alpha=0.625)

## End(Not run)
```

ADEC *Aggregated data ensemble clustering*

Description

Aggregated Data Ensemble Clustering (ADEC) is a direct clustering multi-source technique. ADEC is an iterative procedure which starts with the merging of the data sets. In each iteration, a random sample of the features is selected and/or a resulting dendrogram is divided into k clusters for a range of values of k.

Usage

```
ADEC(
  List,
  distmeasure = "tanimoto",
  normalize = FALSE,
  method = NULL,
  t = 10,
  r = NULL,
  nrclusters = NULL,
  clust = "agnes",
  linkage = "flexible",
  alpha = 0.625
)
```

Arguments

List	A list of data matrices of the same type. It is assumed the rows are corresponding with the objects.
distmeasure	Choice of metric for the dissimilarity matrix (character). Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to "tanimoto".

normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to FALSE. This is recommended if different distance types are used. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is NULL.
t	The number of iterations. Defaults to 10.
r	The number of features to take for the random sample. If NULL (default), all features are considered.
nrclusters	A sequence of numbers of clusters to cut the dendrogram in. If NULL (default), the function stops.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character). Defaults to "flexible".
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".

Details

If r is specified and nrclusters is a fixed number, only a random sampling of the features will be performed for the t iterations (ADECa). If r is NULL and the nrclusters is a sequence, the clustering is performed on all features and the dendrogam is divided into clusters for the values of nrclusters (ADECb). If both r is specified and nrclusters is a sequence, the combination is performed (ADECc). After every iteration, either be random sampling, multiple divisions of the dendrogram or both, an incidence matrix is set up. All incidence matrices are summed and represent the distance matrix on which a final clustering is performed.

Value

The returned value is a list with the following three elements.

AllData	Fused data matrix of the data matrices
DistM	The resulting co-association matrix
Clust	The resulting clustering

The value has class 'ADEC'. The Clust element will be of interest for further applications.

References

Fodeh, S. J., Punch, W. & Tan, P.-N. (2013). On unifying multi-source information from probabilistically generated functional data. Proteins 81:1298-1310.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
res_ADEC <- ADEC(List=L, distmeasure="euclidean", normalize=FALSE, method=NULL,
                 t=5, r=50, nrclusters=5, clust="agnes", linkage="flexible", alpha=0.625)

## End(Not run)
```

`BinFeaturesPlot_MultipleData` *Visualization of characteristic binary features of multiple data sets*

Description

A tool to visualize characteristic binary features of a set of objects in comparison with the remaining objects for multiple data sets. The result is a matrix with coloured cells. Columns represent objects and rows represent the specified features. A feature which is present is give a coloured cell while an absent feature is represented by a grey cell. The labels on the right indicate the names of the features while the labels on the bottom are the names of the objects.

Usage

```
BinFeaturesPlot_MultipleData(
  leadCpds,
  orderLab,
  features = list(),
  data = list(),
  validate = NULL,
  colorLab = NULL,
  nrclusters = NULL,
  cols = NULL,
  name = c("Data1", "Data2"),
  colors1 = c("gray90", "blue"),
  colors2 = c("gray90", "green"),
  margins = c(5.5, 3.5, 0.5, 5.5),
  cexB = 0.8,
  cexL = 0.8,
  cexR = 0.8,
  spaceNames = 0.2,
  plottype = "new",
  location = NULL
)
```

Arguments

`leadCpds`	A character vector with the names of the objects in a first group, i.e., the group for which the specified features are characteristic. Default is NULL.
`orderLab`	A character vector with the order of the objects. Default is NULL.
`features`	A list with as elements character vectors with the names of the features to be visualized for each data set. Default is NULL.
`data`	A list with the different data sets. Default is NULL.
`validate`	Optional. A list with validation data sets. If a feature has a validation reference, these are added in a red colour. Default is NULL.

colorLab	Optional. A clustering object if the objects are to be coloured accoring to their clustering order. Default is NULL.
nrclusters	Optional. The number of clusters to divide the dendrogram of ColorLab. Default is NULL.
cols	Optional. A character vector with the colours of the different clusters. Default is NULL.
name	A character string with the names of the data sets. Default is c("Data1" ,"Data2") for two data sets.
colors1	A character vector with the colours to indicate the presence (first element) or the absence of the features for the objects in LeadCpds. Default is c('gray90','blue').
colors2	A character vector with the colours to indicate the presence (first element) or the absence of the features for the objects in the remaining objects. Default is c('gray90','green').
margins	A vector with the margings of the plot. Default is c(5.5,3.5,0.5,5.5).
cexB	The font size of the labels on the bottom: the object labels. Default is 0.80.
cexL	The font size of the labels on the left: the data labels. Default is 0.80.
cexR	The font size of the labels on the right: the feature labels. Default is 0.80.
spaceNames	A percentage of the height of the figure to be reserved for the names of the objects. Default is 0.20.
plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document, i.e. no new device is opened and the plot appears in the current device or document. Default is "new".
location	Optional. If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Value

A plot visualizing the presence and absence of the characteristic binary features across multiple data sets.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

Comps=FindCluster(list(MCF7_F),nrclusters=10,select=c(1,8))

MCF7_Char=CharacteristicFeatures(List=NULL,Selection=Comps,binData=
list(fingerprintMat,targetMat),datanames=c("FP","TP"),nrclusters=NULL,
topC=NULL,sign=0.05,fusionsLog=TRUE,weightclust=TRUE,names=c("FP","TP"))
```

```
FeatFP=MCF7_Char$Selection$Characteristics$FP$TopFeat$Names[c(1:10)]
FeatTP=MCF7_Char$Selection$Characteristics$TP$TopFeat$Names[c(1:10)]

BinFeaturesPlot_MultipleData(leadCpds=Comps,orderLab=MCF7_Char$Selection$
objects$OrderedCpds,features=list(FeatFP,FeatTP),data=list(fingerprintMat,targetMat),
validate=NULL,colorLab=NULL,nrclusters=NULL,cols=NULL,name=c("FP","TP"),colors1=
c('gray90','blue'),colors2=c('gray90','green'),margins=c(5.5,3.5,0.5,5.5),cexB=0.80,
cexL=0.80,cexR=0.80,spaceNames=0.20,plottype="new",location=NULL)

## End(Not run)
```

`BinFeaturesPlot_SingleData` *Visualization of characteristic binary features of a single data set*

Description

A tool to visualize characteristic binary features of a set of objects in comparison with the remaining objects for a single data set. The result is a matrix with coloured cells. Columns represent objects and rows represent the specified features. A feature which is present is give a coloured cell while an absent feature is represented by a grey cell. The labels on the right indicate the names of the features while the labels on the bottom are the names of the objects.

Usage

```
BinFeaturesPlot_SingleData(
  leadCpds = c(),
  orderLab = c(),
  features = c(),
  data = NULL,
  colorLab = NULL,
  nrclusters = NULL,
  cols = NULL,
  name = c("Data"),
  colors1 = c("gray90", "blue"),
  colors2 = c("gray90", "green"),
  highlightFeat = NULL,
  margins = c(5.5, 3.5, 0.5, 5.5),
  cexB = 1.1,
  cexR = 1.1,
  cexN = 1.1,
  plottype = "new",
  location = NULL
)
```

Arguments

leadCpds	A character vector with the names of the objects in a first group, i.e., the group for which the specified features are characteristic. Default is NULL.
orderLab	A character vector with the order of the objects. Default is NULL.
features	A character vector with the names of the features to be visualized. Default is NULL.
data	The data matrix. Default is NULL.
colorLab	Optional. A clustering object if the objects are to be coloured accoring to their clustering order. Default is NULL.
nrclusters	Optional. The number of clusters to divide the dendrogram of ColorLab. Default is NULL.
cols	Optional. A character vector with the colours of the different clusters. Default is NULL.
name	A character string with the name of the data. Default is "Data".
colors1	A character vector with the colours to indicate the presence (first element) or the absence of the features for the objects in LeadCpds. Default is c('gray90','blue').
colors2	A character vector with the colours to indicate the presence (first element) or the absence of the features for the objects in the remaining objects. Default is c('gray90','green').
highlightFeat	Optional. A character vector with names of features to be highlighted. The names of the features are coloured purple. Default is NULL.
margins	A vector with the margings of the plot. Default is c(5.5,3.5,0.5,5.5).
cexB	Font size of the bottom (object) labels. Default is 1.1.
cexR	Font size of the right (feature) labels. Default is 1.1.
cexN	Font size of the data-name label. Default is 1.1.
plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document, i.e. no new device is opened and the plot appears in the current device or document. Default is "new".
location	Optional. If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Value

A plot visualizing the presence and absence of the characteristic binary features.

Examples

```
## Not run:
data(fingerprintMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
```

```
Comps=FindCluster(list(MCF7_F),nrclusters=10,select=c(1,8))

MCF7_Char=CharacteristicFeatures(List=list(fingerprintMat),Selection=Comps,
binData=list(fingerprintMat),datanames=c("FP"),nrclusters=NULL,topC=NULL,
sign=0.05,fusionsLog=TRUE,weightclust=TRUE,names=c("FP"))
Feat=MCF7_Char$Selection$Characteristics$FP$TopFeat$Names[c(1:10)]

BinFeaturesPlot_SingleData(leadCpds=Comps,orderLab=MCF7_Char$Selection$
objects$OrderedCpds,features=Feat,data=fingerprintMat,colorLab=NULL,
nrclusters=NULL,cols=NULL,name=c("FP"),colors1=c('gray90','blue'),colors2=
c('gray90','green'),highlightFeat=NULL,margins=c(5.5,3.5,0.5,5.5),
plottype="new",location=NULL)

## End(Not run)
```

`BoxPlotDistance` *Box plots of one distance matrix categorized against another distance matrix.*

Description

Given two distance matrices, the function categorizes one distance matrix and produces a box plot from the other distance matrix against the created categories. The option is available to choose one of the plots or to have both plots. The function also works on outputs from ADEC and CEC functions which do not have distance matrices but incidence matrices.

Usage

```
BoxPlotDistance(
  Data1,
  Data2,
  type = c("data", "dist", "clusters"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  lab1,
  lab2,
  limits1 = NULL,
  limits2 = NULL,
  plot = 1,
  StopRange = FALSE,
  plottype = "new",
  location = NULL
)
```

Arguments

Data1	The first data matrix, cluster outcome or distance matrix to be plotted.
Data2	The second data matrix, cluster outcome or distance matrix to be plotted.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clusters" the single source clustering is skipped. Type should be one of "data", "dist" or "clusters".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
lab1	The label to plot for Data1.
lab2	The label to plot for Data2.
limits1	The limits for the categories of Data1.
limits2	The limits for the categories of Data2.
plot	The type of plots: 1 - Plot the values of Data1 versus the categories of Data2. 2 - Plot the values of Data2 versus the categories of Data1. 3 - Plot both types 1 and 2.
StopRange	Logical. Indicates whether the distance matrices with values not between zero and one should be standardized to have so. If FALSE the range normalization is performed. See Normalization. If TRUE, the distance matrices are not changed. This is recommended if different types of data are used such that these are comparable. Default is FALSE.
plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document, i.e. no new device is opened and the plot appears in the current device or document. Default is "new".
location	If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Value

One/multiple box plots.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
```

```
MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)


BoxPlotDistance(MCF7_F,MCF7_T,type="cluster",lab1="FP",lab2="TP",limits1=c(0.3,0.7),
limits2=c(0.3,0.7),plot=1,StopRange=FALSE,plottype="new", location=NULL)

## End(Not run)
```

CEC *Complementary ensemble clustering*

Description

Complementary Ensemble Clustering (CEC) Complementary Ensemble Clustering (CEC, *Fodeh2013*) shows similarities with ADEC. However, instead of merging the data matrices, ensemble clustering is performedon each data matrix separately. The resulting incidence matrices for each data sets are combined in a weighted linear equation. The weighted incidence matrix is the input for the final clustering algorithm. Similarly as ADEC, there are versions depending of the specification of the number of features to sample and the number of clusters.

Usage

```
CEC(
  List,
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  t = 10,
  r = NULL,
  nrclusters = NULL,
  weight = NULL,
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  weightclust = 0.5
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in Normalization.

method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
t	The number of iterations. Defaults to 10.
r	A vector with the number of features to take for the random sample for each element in List. If NULL (default), all features are considered.
nrclusters	A list with a sequence of numbers of clusters to cut the dendrogram in for each element in List. If NULL (default), the function stops.
weight	The weights for the weighted linear combination.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible"
weightclust	A weight for which the result will be put aside of the other results. This was done for comparative reason and easy access.

Details

If r is specified and nrclusters is a fixed number, only a random sampling of the features will be performed for the t iterations (CECa). If r is NULL and the nrclusters is a sequence, the clustering is performedon all features and the dendrogam is divided into clusters for the values of nrclusters (CECb). If both r is specified and nrclusters is a sequence, the combination is performed (CECc). After every iteration, either be random sampling, multiple divisions of the dendrogram or both, an incidence matrix is set up. All incidence matrices are summed and represent the distance matrix on which a final clustering is performed.

Value

The returned value is a list of four elements:

DistM	The resulting incidence matrix
Results	The hierarchical clustering result for each element in WeightedDist
Clust	The result for the weight specified in Clustweight

The value has class 'CEC'.

References

Fodeh SJ, Punch W, Tan P (2013). "On Unifying Multi-source Information to Improve Subcellular Protein Localization Prediction." *Proteins: Structure, Function, and Bioinformatics*, **81**, 1298–1310.

`CharacteristicFeatures` *Determine the characteristic features of clusters*

Description

`CharacteristicFeatures` identifies the features that are characteristic for the objects of each cluster of one or more clustering results. Binary features are evaluated with Fisher's exact test, continuous features with the t-test. Multiplicity correction is included.

Usage

```
CharacteristicFeatures(
  List,
  Selection = NULL,
  binData = NULL,
  contData = NULL,
  datanames = NULL,
  nrclusters = NULL,
  sign = 0.05,
  topChar = NULL,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL
)
```

Arguments

`List`	A list of clustering outputs to be compared. The first element of the list is used as the reference in `ReorderToReference`.
`Selection`	If a selection of objects is provided, the function only investigates the features of this selection. Default is NULL.
`binData`	A list of the binary feature data matrices. Default is NULL.
`contData`	A list of continuous feature data matrices. Default is NULL.
`datanames`	A vector with the names of the data matrices. Default is NULL.
`nrclusters`	The number of clusters to cut the dendrogram in. Default is NULL.
`sign`	The significance level. Default is 0.05.
`topChar`	The number of top characteristics to return. If NULL, only the significant characteristics are saved. Default is NULL.
`fusionsLog`	Logical. Indicator for the fusion of clusters. Default is TRUE.
`weightclust`	Logical. For the outputs of CEC, WeightedClust or WeightedSimClust, if TRUE only the result of the Clust element is considered. Default is TRUE.
`names`	Optional. Names of the methods. Default is NULL.

Value

A list with for each method the found (top) characteristics of the feature data per cluster.

References

Perualila-Tan N. et al. (2016). Weighted similarity-based clustering of chemical structures and bioactivity data in early drug discovery.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)
MCF7_Char=CharacteristicFeatures(List=L,Selection=NULL,binData=list(fingerprintMat),
contData=NULL,datanames=c("FP"),nrclusters=7,topChar=20)

## End(Not run)
```

---

characterize_clusters *Characterise clusters with external (biological / clinical) variables*

Description

For a clustering, summarise how each external variable differs across the clusters. Continuous variables get n, mean, SD, median, IQR and range *per cluster*, an overall Kruskal-Wallis test, a rank η^2 effect size, and **gradient detection** (are the cluster means monotonically ordered?). Categorical variables get per-cluster counts and column percentages, a chi-squared / Fisher test and Cramer's V. The emphasis is on effect sizes and descriptive summaries rather than p-values alone.

Usage

```
characterize_clusters(labels, metadata, continuous = NULL, categorical = NULL)
```

Arguments

`labels`	Cluster label of each object (vector; `NA` allowed).
`metadata`	A data frame of external variables, one row per object, in the same order as `labels`.
`continuous, categorical`	Optional character vectors naming the columns to treat as continuous / categorical. If `NULL`, numeric columns are taken as continuous and the rest as categorical.

Value

A list with `continuous` (per-cluster summary, long format), `continuous_tests` (one row per variable: `KW_p`, `eta2`, `gradient` – the clusters ordered by mean – and a monotone-trend `trend_rho`/`trend_p`), `categorical` (counts and column percentages) and `categorical_tests` (`p`, `cramers_v`).

See Also

`plot_cluster_profile`, `cluster_biology_panel`

Examples

```
data(mosaic_toy)
fit <- M_ABC(mosaic_toy$List, numsim = 40, NC = 3)
cl  <- mosaic_labels(fit, 3)
meta <- data.frame(score = rnorm(length(cl)), grp = sample(c("A", "B"), length(cl), TRUE))
ch  <- characterize_clusters(cl, meta)
ch$continuous_tests
```

ChooseCluster *Interactive plot to determine DE Genes and DE features for a specific cluster*

Description

If desired, the function produces a dendrogram of a clustering result. One or multiple clusters can be indicated by a mouse click. From these clusters DE genes and characteristic features are determined. It is also possible to provide the objects of interest without producing the plot.

Usage

```
ChooseCluster(
  Interactive = TRUE,
  leadCpds = NULL,
  clusterResult = NULL,
  colorLab = NULL,
  binData = NULL,
  contData = NULL,
  datanames = c("FP"),
  geneExpr = NULL,
  topChar = 20,
  topG = 20,
  sign = 0.05,
  nrclusters = NULL,
  cols = NULL,
  n = 1
)
```

Arguments

Interactive	Logical. Whether an interactive plot should be made. Defaults to TRUE.
leadCpds	A list of the objects of the clusters of interest. Defaults to NULL.
clusterResult	The output of one of the aggregated cluster functions. Default is NULL.
colorLab	The clustering result the dendrogram should be colored after. Default is NULL.
binData	A list of the binary feature data matrices. Default is NULL.
contData	A list of continuous data sets of the objects. Default is NULL.
datanames	A vector with the names of the data matrices. Default is c("FP").
geneExpr	A gene expression matrix, may also be an ExpressionSet. Default is NULL.
topChar	The number of top characteristics to return. Default is 20.
topG	The number of top genes to return. Default is 20.
sign	The significance level. Default is 0.05.
nrclusters	Optional. The number of clusters to cut the dendrogram in. Default is NULL.
cols	The colors to use in the dendrogram. Default is NULL.
n	The number of clusters one wants to identify by a mouse click. Default is 1.

Details

This function may require the suggested packages 'a4Base', 'a4Core' and 'limma' when a gene expression matrix is supplied.

Value

A list with one element per cluster of interest with elements objects, Characteristics and (optionally) Genes.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(geneMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

MCF7_Interactive=ChooseCluster(Interactive=TRUE,leadCpds=NULL,clusterResult=MCF7_T,
colorLab=MCF7_F,binData=list(fingerprintMat),datanames=c("FP"),geneExpr=geneMat,
topChar = 20, topG = 20,nrclusters=7,n=1)

## End(Not run)
```

`Cluster` *Single-source base clustering*

Description

Cluster one data (or distance) matrix with agglomerative hierarchical clustering. This is the Layer-2 baseline primitive of the MoSaIC framework: run it on each tile on its own to see what each modality recovers before integrating. It also underpins `HierarchicalEnsembleClustering`.

Usage

```
Cluster(
  Data,
  type = c("data", "dist"),
  distmeasure = "euclidean",
  normalize = FALSE,
  method = NULL,
  clust = "agnes",
  linkage = "ward",
  alpha = 0.625,
  StopRange = TRUE,
  ...
)
```

Arguments

`Data`	A data matrix (objects in rows) or a precomputed distance matrix.
`type`	`"data"` to compute a distance first, or `"dist"` if `Data` is already a dissimilarity matrix.
`distmeasure`	Distance measure (see `Distance`), used when `type = "data"`.
`normalize, method`	Normalisation controls passed to `Distance`.
`clust`	Clustering backend; currently `"agnes"`.
`linkage`	Agglomeration method. Default `"ward"`.
`alpha`	Flexible-linkage parameter for `cluster::agnes`.
`StopRange`	Logical; if `FALSE` and the dissimilarities fall outside `[0, 1]` they are range-normalised for comparability.
`...`	Ignored; accepted for backward compatibility (e.g. `gap`, `maxK`).

Value

A list with `DistM` (the dissimilarity matrix) and `Clust` (an `agnes` object).

See Also

`Distance`, `M_ABC`

Examples

```
data(mosaic_toy)
base1 <- Cluster(t(mosaic_toy$List[[1]]), type = "data",
                 distmeasure = "euclidean")
mosaic_labels(base1, k = 3)[1:6]
```

`ClusterCols` *Helper that colours dendrogram leaves by cluster membership*

Description

Internal helper applied via `dendrapply` that colours the leaves and edges of a dendrogram according to the cluster a leaf belongs to, or to a specific selection of objects. Used by `ClusterPlot`.

Usage

```
ClusterCols(x, Data, nrclusters = NULL, cols = NULL, colorComps = NULL)
```

Arguments

`x`	A node of a dendrogram (passed by `dendrapply`).
`Data`	The clustering result (an `agnes`/`hclust` object) used to determine cluster membership.
`nrclusters`	Optional. The number of clusters to cut the dendrogram in. Default is NULL.
`cols`	The colours for the clusters if nrclusters is specified. Default is NULL.
`colorComps`	Optional. A character vector of objects to highlight in red. Default is NULL.

Value

The dendrogram node `x` with coloured `nodePar` and `edgePar` attributes.

`ClusteringAggregation` *Clustering aggregation*

Description

The `ClusteringAggregation` includes the ensemble clustering methods Balls, Agglomerative (Aggl.) and Furthest which are graph-based consensus methods.

Usage

```
ClusteringAggregation(
  List,
  type = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  nrclusters = c(7, 7),
  gap = FALSE,
  maxK = 15,
  agglMethod = c("Balls", "Aggl", "Furthest", "LocalSearch"),
  improve = TRUE,
  distThresh_B = 0.5,
  distThresh_A = 0.8
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clust" the single source clustering is skipped. Type should be one of "data", "dist" or "clust".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in `Normalization`.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".
nrclusters	The number of clusters to divide each individual dendrogram in. Default is c(7,7) for two data sets.
gap	Logical. Whether the optimal number of clusters should be determined with the gap statistic. Default is FALSE.
maxK	The maximal number of clusters to investigate in the gap statistic. Default is 15.
agglMethod	The method to be performed: "Balls","Aggl","Furthest" or "LocalSearch".

improve	Logical. If TRUE, a local search is performed to improve the obtained results. Default is TRUE.
distThresh_B	A distance threshold for the Balls algoritme. Default is 0.5.
distThresh_A	A distance threshold for the Aggl. algoritme. Default is 0.8.

Details

Gionis, Mannila and Tsaparas (2007) propose heuristic algorithms in order to find a solution for the consensus problem. In a first step, a weighted graph is built from the objects with weights between two vertices determined by the fraction of clusterings that place the two vertices in different clusters. In a second step, an algorithm searches for the partition that minimizes the total number of disagreements with the given partitions. The Balls algorithm is an iterative process which finds a cluster for the consensus partition in each iteration. For each object, all objects at a distance of at most 0.5 are collected and the average distance of this set to the object of interest is calculated. If the average distance is less or equal to a parameter the objects form a cluster; otherwise the object forms a singleton. The Agglomerative (Aggl.) algorithm starts by considering every object as a singleton cluster. Next, the two closest clusters are merged if the average distance between the clusters is less than 0.5. If there are no two clusters with an average distance smaller than 0.5, the algorithm stops and returns the created clusters as a solution. The Furthest algorithm starts by placing all objects into a single cluster. In each iteration, the pair of objects that are the furthest apart are considered as new cluster centers. The remaining objects are appointed to the center that increases the cost of the partition the least and the new cost is computed. The cost is the sum of the all distances between the obtained partition and the partitions in the ensemble. The iteration continues until the cost of the new partition is higher than the previous partition.

Value

The returned value is a list of two elements:

DistM	A NULL object
Clust	The resulting clustering

The value has class 'Ensemble'.

References

Gionis, A., Mannila, H. and Tsaparas, P. (2007). Clustering aggregation. ACM Transactions on Knowledge Discovery from Data, 1(1), 4-es.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
Aggl_toy <- ClusteringAggregation(List = L, type = "data",
  distmeasure = c("euclidean", "euclidean"), normalize = c(FALSE, FALSE),
  method = c(NULL, NULL), clust = "agnes", linkage = c("ward", "ward"),
  alpha = 0.625, nrclusters = c(7, 7), agglMethod = "Aggl",
  improve = TRUE, distThresh_B = 0.5, distThresh_A = 0.8)

## End(Not run)
```

clustering_concordance *Compare clustering solutions to each other (no ground truth needed)*

Description

Given several labellings of the same objects (e.g. different weightings or methods), compute their pairwise agreement (ARI) and each object's stability – how consistently it is co-clustered across the solutions. Useful for real data where no ground truth is available.

Usage

```
clustering_concordance(labels_list)
```

Arguments

`labels_list`	A named list of label vectors, all of the same length.

Value

A list with `agreement` (pairwise ARI matrix), `stability` (mean pairwise co-membership per object) and `changed` (objects whose label is not constant across solutions).

See Also

`cluster_agreement`, `compare_clusterings`

Examples

```
data(mosaic_toy)
v <- mosaic_labels(M_ABC(mosaic_toy$List, numsim = 40, NC = 3,
                         weighting = c("var", "var")), 3)
n <- mosaic_labels(M_ABC(mosaic_toy$List, numsim = 40, NC = 3,
                         weighting = c("nugget", "nugget")), 3)
clustering_concordance(list(var = v, nugget = n))$agreement
```

ClusterPlot *Colouring clusters in a dendrogram*

Description

Plot a dendrogram with leaves colored by a result of choice.

Usage

```
ClusterPlot(
  Data1,
  Data2 = NULL,
  nrclusters = NULL,
  cols = NULL,
  colorComps = NULL,
  hangdend = 0.02,
  plottype = "new",
  location = NULL,
  ...
)
```

Arguments

Data1	The resulting clustering of a method which contains the dendrogram to be colored.
Data2	Optional. The resulting clustering of another method , i.e. the resulting clustering on which the colors should be based. Default is NULL.
nrclusters	Optional. The number of clusters to cut the dendrogram in. If not specified the dendrogram will be drawn without colours to discern the different clusters. Default is NULL.
cols	The colours for the clusters if nrclusters is specified. Default is NULL.
colorComps	If only a specific set of objects needs to be highlighted, this can be specified here. The objects should be given in a character vector. If specified, all other compound labels will be colored black. Default is NULL.
hangdend	A specification for the length of the brances of the dendrogram. Default is 0.02.
plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document. Default is "new".
location	If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.
...	Other options which can be given to the plot function.

Value

A plot of the dendrogram of the first clustering result with colored leaves. If a second clustering result is given in Data2, the colors are based on this clustering result.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(Colors1)
```

```
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

ClusterPlot(MCF7_T ,nrclusters=7,cols=Colors1,plottype="new",location=NULL,
main="Clustering on Target Predictions: Dendrogram",ylim=c(-0.1,1.8))

## End(Not run)
```

`cluster_agreement` *Agreement between two partitions*

Description

Computes the adjusted Rand index (ARI), normalised mutual information (NMI) and pairwise Jaccard between two label vectors of equal length.

Usage

```
cluster_agreement(a, b)
```

Arguments

a, b	Integer/character/factor label vectors of equal length.

Details

ARI rule of thumb: `>0.90` excellent; `0.75-0.90` good; `0.50-0.75` moderate; `<0.50` weak.

Value

A named numeric vector `c(ARI, NMI, Jaccard)`.

Examples

```
cluster_agreement(c(1, 1, 2, 2), c(2, 2, 1, 1))   # identical up to relabel
```

cluster_biology_panel *Panel of cluster characterisation plots*

Description

Lay out plot_cluster_profile for every external variable in a single figure – boxplots for continuous variables, bar plots for categorical ones – ready for the manuscript.

Usage

```
cluster_biology_panel(labels, metadata, vars = NULL, ncol = 3, base_size = 13)
```

Arguments

labels	Cluster labels.
metadata	Data frame of external variables (rows = objects).
vars	Optional subset of columns to show (default: all).
ncol	Number of columns in the panel grid.
base_size	Base font size passed to the plot theme.

Value

Invisibly NULL.

See Also

characterize_clusters

cluster_k_sweep *Show the clustering at every k (see how it deteriorates)*

Description

Rather than only reporting the best k, project the objects to a *fixed* 2-D embedding and recolour them by the k-cut for each candidate k in a grid, annotating the average silhouette per panel. This makes it visible how the structure splits and degrades as k grows.

Usage

```
cluster_k_sweep(
  x,
  k_range = 2:6,
  linkage = "ward.D2",
  embed = c("pca", "umap", "tsne"),
  ncol = 3,
  point_size = 2.2,
  alpha = 0.9,
  palette = "Dark 3",
  base_size = 13
)
```

Arguments

x	A fitted result (`$DistM`), a dissimilarity, or a `List`.
k_range	Candidate cluster counts to display.
linkage	Agglomeration method used to cut the fused dissimilarity.
embed	Embedding for the (fixed) 2-D coordinates: `"pca"` (default), `"umap"` or `"tsne"`.
ncol	Columns in the panel grid.
point_size	Diameter of the plotted points (default `2.2`); raise it for larger, clearer markers.
alpha	Point opacity (default `0.9`).
palette	Discrete `hcl.colors` palette used to colour the clusters (default `"Dark 3"`).
base_size	Base font size passed to the **ggplot2** theme.

Details

The panels are drawn with **ggplot2** for a clean, high-resolution result whose quality is preserved when the figure is rescaled – save the returned object to a vector device (PDF/SVG) or a high-DPI raster for publication, e.g. `ggplot2::ggsave("ksweep.pdf", cluster_k_sweep(fit), width = 9, height = 6)`.

Value

Invisibly the **ggplot** object (also printed, so it draws on an open device); the average silhouette per k is attached as the attribute `"silhouette"`.

See Also

`cluster_selection_plot`, `embedding_plot`

Examples

```
data(mosaic_toy)
fit <- M_ABC(mosaic_toy$List, numsim = 40, NC = 3)
cluster_k_sweep(fit, k_range = 2:6, point_size = 3)
```

`cluster_selection_plot` *Justify the number of clusters (silhouette + elbow / scree)*

Description

For a fused dissimilarity (or a fitted result, or a `List` of sources) compute, over a range of k, the average silhouette width and the total within-cluster dispersion (an elbow / scree curve), and plot them so the chosen k can be justified in the manuscript.

Usage

```
cluster_selection_plot(
  x,
  k_range = 2:8,
  linkage = "ward.D2",
  select = c("both", "elbow", "silhouette"),
  plot = TRUE,
  point_size = 2.6,
  base_size = 13
)
```

Arguments

`x`	A fitted result (with `$DistM`), a dissimilarity matrix, or a `List` of data sources (objects in rows).
`k_range`	Integer vector of candidate cluster counts.
`linkage`	Agglomeration method used to cut the dissimilarity.
`select`	Which criterion to show *and* use for `best_k`: `"both"` (default) draws both panels and picks k from the elbow; `"elbow"` draws only the within-cluster dispersion (scree) curve and uses its knee; `"silhouette"` draws only the silhouette curve and uses its maximum. Passing a single criterion therefore shows a single panel.
`plot`	Logical; draw the figure.
`point_size`	Marker size for the curve points.
`base_size`	Base font size for the **ggplot2** theme.

Value

Invisibly a data frame with `k`, `silhouette` and `within_dispersion`, plus attributes `"best_k"`, `"elbow_k"`, `"silhouette_k"` and `"plot"` (the **ggplot** object).

See Also

`embedding_plot`, `SelectnrClusters`

Examples

```
data(mosaic_toy)
fit <- M_ABC(mosaic_toy$List, numsim = 40, NC = 3)
cluster_selection_plot(fit, k_range = 2:6, select = "elbow")   # one panel
```

ColorPalette *Create a color palette to be used in the plots*

Description

In order to facilitate the visualization of the influence of the different methods on the clustering of the objects, colours can be used. The function `ColorPalette` is able to pick out as many colours as there are clusters. This is done with the help of the `ColorRampPalette` function of the grDevices package

Usage

```
ColorPalette(colors = c("red", "green"), ncols = 5)
```

Arguments

`colors`	A vector containing the colors of choice
`ncols`	The number of colors to be specified. If higher than the number of colors, it specifies colors in the region between the given colors.

Value

A vector containing the hex codes of the chosen colors.

Examples

```
Colors1<-ColorPalette(c("cadetblue2","chocolate","firebrick2",
"darkgoldenrod2", "darkgreen","blue2","darkorchid3","deeppink2"), ncols=8)
```

ColorsNames *Function that annotates colors to their names*

Description

The `ColorsNames` function is used on the output of the `ReorderToReference` and matches the cluster numbers indicated by the cell with the names of the colors. This is necessary to produce the plot of the `ComparePlot` function and is therefore an internal function of this function but can also be applied separately.

Usage

```
ColorsNames(matrixColors, cols = NULL)
```

Arguments

matrixColors	The output of the ReorderToReference function.
cols	A character vector with the names of the colours to be used. Default is NULL.

Value

A matrix containing the hex code of the color that corresponds to each cell of the matrix to be colored. This function is called upon by the ComparePlot function.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(Colors1)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)
names=c("FP","TP")

MatrixColors=ReorderToReference(List=L,nrclusters=7,fusionsLog=TRUE,weightclust=TRUE,
names=names)

Names=ColorsNames(matrixColors=MatrixColors,cols=Colors1)

## End(Not run)
```

CompareInteractive *Interactive comparison of single and multiple source clustering results*

Description

The function CompareInteractive produces an interactive plot of the multiple source clustering results in ListM. By identifying a cluster or a method, a comparison with the single source clustering results in ListS is shown in a new plotting window.

Usage

```
CompareInteractive(
  ListM,
  ListS,
  nrclusters = NULL,
  cols = NULL,
  fusionsLogM = FALSE,
  fusionsLogS = FALSE,
  weightclustM = FALSE,
  weightclustS = FALSE,
  namesM = NULL,
  namesS = NULL,
  marginsM = c(2, 2.5, 2, 2.5),
  marginsS = c(8, 2.5, 2, 2.5),
  Interactive = TRUE,
  n = 1
)
```

Arguments

ListM	A list of the outputs from the multiple source clusterings to be compared.
ListS	A list of the outputs from the single source clusterings to be compared.
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.
cols	A character vector with the names of the colours. Default is NULL.
fusionsLogM	The fusionsLog parameter for the elements in ListM. Default is FALSE.
fusionsLogS	The fusionsLog parameter for the elements in ListS. Default is FALSE.
weightclustM	The weightclust parameter for the elements in ListM. Default is FALSE.
weightclustS	The weightclust parameter for the elements in ListS. Default is FALSE.
namesM	Optional. Names of the multiple source clusterings. Default is NULL.
namesS	Optional. Names of the single source clusterings. Default is NULL.
marginsM	Optional. Margins to be used for the ListM plot. Default is c(2,2.5,2,2.5).
marginsS	Optional. Margins to be used for the ListS plot. Default is c(8,2.5,2,2.5).
Interactive	Optional. Do you want an interactive plot? Default is TRUE.
n	The number of methods/clusters you want to identify. Default is 1.

Value

A plot of the comparison of the elements of ListM, on which multiple clusters and/or methods can be identified to compare them to the elements in ListS.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
```

```
data(Colors1)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(fingerprintMat,targetMat)

MCF7_W=WeightedClust(List=L,type="data",distmeasure=c("tanimoto","tanimoto"),
normalize=c(FALSE,FALSE),method=c(NULL,NULL),weight=seq(1,0,-0.1),weightclust=0.5,
clust="agnes",linkage="ward",StopRange=FALSE)

ListM=list(MCF7_W)
namesM=c(seq(1.0,0.0,-0.1))

ListS=list(MCF7_F,MCF7_T)
namesS=c("FP","TP")

CompareInteractive(ListM,ListS,nrclusters=7,cols=Colors1,fusionsLogM=FALSE,
fusionsLogS=FALSE,weightclustM=FALSE,weightclustS=TRUE,namesM,namesS,
marginsM=c(2,2.5,2,2.5),marginsS=c(8,2.5,2,2.5),Interactive=TRUE,n=1)

## End(Not run)
```

ComparePlot *Comparison of clustering results over multiple methods*

Description

A visual comparison of the clustering results of several methods. The function relies on `ReorderToReference` and `ColorsNames` and renders the resulting matrix either as a rectangular heatmap (using the `plotrix` package) or in a circular format (using the `circlize` package).

Usage

```
ComparePlot(
  List,
  nrclusters = NULL,
  cols = NULL,
  fusionsLog = FALSE,
  weightclust = FALSE,
  names = NULL,
  margins = c(8.1, 3.1, 3.1, 4.1),
  circle = FALSE,
  canvaslims = c(-1, 1, -1, 1),
  Highlight = NULL,
  substr = NULL,
  cex.highlight = 1,
```

```
  cex.labels = 0.7,
  trackheightdend = 0.5,
  plottype = "new",
  location = NULL
)
```

Arguments

List	A list of the outputs from the methods to be compared. The first element of the list is used as the reference in ReorderToReference.
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.
cols	A character vector with the colours to be used. Default is NULL.
fusionsLog	Logical. To be handed to ReorderToReference. Default is FALSE.
weightclust	Logical. To be handed to ReorderToReference. Default is FALSE.
names	Optional. Names of the methods to be used as labels. Default is NULL.
margins	Optional. Margins for the plot. Default is c(8.1,3.1,3.1,4.1).
circle	Logical. Whether the figure should be circular (TRUE) or a rectangle (FALSE). Default is FALSE.
canvaslims	The limits for the circular dendrogram. Default is c(-1.0,1.0,-1.0,1.0).
Highlight	Optional. A list of character vectors of objects to be highlighted. Default is NULL.
substr	Optional. A vector of length two giving start and stop positions to shorten labels. Default is NULL.
cex.highlight	Magnification for the highlight text. Default is 1.
cex.labels	Magnification for the labels. Default is 0.7.
trackheightdend	The height of the dendrogram track in the circular plot. Default is 0.5.
plottype	Should be one of "pdf","new" or "sweave". Default is "new".
location	Optional. If plottype is "pdf", a location should be provided. Default is NULL.

Value

A plot which translates the matrix output of ReorderToReference.

CompareSilCluster *Compares medoid clustering results based on silhouette widths*

Description

The function CompareSilCluster compares the results of two medoid clusterings. The null hypothesis is that the clustering is identical. A test statistic is calculated and a p-value obtained with bootstrapping.

Usage

```
CompareSilCluster(
  List,
  type = c("data", "dist"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  nrclusters = NULL,
  names = NULL,
  nboot = 100,
  plottype = "new",
  location = NULL
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	indicates whether the provided matrices in "List" are either data matrices or distance matrices. Type should be one of "data" or "dist".
distmeasure	A vector of the distance measures to be used on each data matrix. Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices. Default is c(FALSE, FALSE).
method	A method of normalization. Default is c(NULL,NULL).
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.
names	The labels to give to the elements in List. Default is NULL.
nboot	Number of bootstraps to be run. Default is 100.
plottype	Should be one of "pdf","new" or "sweave". Default is "new".
location	If plottype is "pdf", a location should be provided here. Default is NULL.

Value

A plot of the density of the statistic under the null hypothesis. Further a list with two elements:

Observed Statistic	The observed statistical value
P-Value	The P-value of the obtained statistic retrieved after bootstrapping

.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)

List=list(fingerprintMat,targetMat)
```

```
Comparison=CompareSilCluster(List=List,type="data",
distmeasure=c("tanimoto","tanimoto"),normalize=c(FALSE,FALSE),method=c(NULL,NULL),
nrclusters=7,names=NULL,nboot=100,plottype="new",location=NULL)

Comparison

## End(Not run)
```

`CompareSvsM` *Comparison of clustering results for the single and multiple source clustering.*

Description

The function `CompareSvsM` plots the comparison of the single source clustering results on the left and that of the multiple source clustering results on the right such that a visual comparison is possible.

Usage

```
CompareSvsM(
  ListS,
  ListM,
  nrclusters = NULL,
  cols = NULL,
  fusionsLogS = FALSE,
  fusionsLogM = FALSE,
  weightclustS = FALSE,
  weightclustM = FALSE,
  namesS = NULL,
  namesM = NULL,
  margins = c(8.1, 3.1, 3.1, 4.1),
  plottype = "new",
  location = NULL
)
```

Arguments

`ListS`	A list of the outputs from the single source clusterings to be compared.
`ListM`	A list of the outputs from the multiple source clusterings to be compared.
`nrclusters`	The number of clusters to cut the dendrogram in. Default is NULL.
`cols`	A character vector with the names of the colours. Default is NULL.
`fusionsLogS`	The fusionslog parameter for the elements in ListS. Default is FALSE.
`fusionsLogM`	The fusionsLog parameter for the elements in ListM. Default is FALSE.
`weightclustS`	The weightclust parameter for the elements in ListS. Default is FALSE.

weightclustM	The weightclust parameter for the elements in ListM. Default is FALSE.
namesS	Optional. Names of the single source clusterings. Default is NULL.
namesM	Optional. Names of the multiple source clusterings. Default is NULL.
margins	Optional. Margins to be used for the plot. Default is c(8.1,3.1,3.1,4.1).
plottype	Should be one of "pdf","new" or "sweave". Default is "new".
location	If plottype is "pdf", a location should be provided here. Default is NULL.

Value

A plot with on the left the comparison over the objects in ListS and on the right a comparison over the objects in ListM.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(Colors1)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(fingerprintMat,targetMat)

MCF7_W=WeightedClust(List=L,type="data", distmeasure=c("tanimoto","tanimoto"),
normalize=c(FALSE,FALSE),method=c(NULL,NULL),weight=seq(1,0,-0.1),weightclust=0.5
,clust="agnes",linkage="ward",StopRange=FALSE)

ListM=list(MCF7_W)
namesM=seq(1.0,0.0,-0.1)

ListS=list(MCF7_F,MCF7_T)
namesS=c("FP","TP")

CompareSvsM(ListS,ListM,nrclusters=7,cols=Colors1,fusionsLogS=FALSE,
fusionsLogM=FALSE,weightclustS=FALSE,weightclustM=FALSE,namesS,
namesM,plottype="new",location=NULL)

## End(Not run)
```

---

compare_clusterings *Compare two clusterings derived from distance matrices*

Description

Clusters two dissimilarity matrices with the same linkage and number of clusters and reports their agreement (ARI, NMI, pairwise Jaccard) and the contingency table.

Usage

```
compare_clusterings(
  D1,
  D2,
  NC = NULL,
  linkage = "ward.D2",
  labels = c("D1", "D2"),
  verbose = TRUE
)
```

Arguments

D1, D2	Square dissimilarity matrices over the same objects.
NC	Number of clusters (default floor(sqrt(n))).
linkage	Linkage method. Default "ward.D2".
labels	Length-2 labels for the two solutions.
verbose	Logical; print a short report.

Value

Invisibly, a list with the two label vectors, ARI, NMI, pair_Jaccard and the contingency table.

Examples

```
set.seed(1)
D1 <- as.matrix(dist(matrix(rnorm(60), 20)))
D2 <- as.matrix(dist(matrix(rnorm(60), 20)))
compare_clusterings(D1, D2, NC = 3, verbose = FALSE)$ARI
```

---

compare_methods_plot *Visually compare clustering solutions with ComparePlot (no ground truth)*

Description

Aligns several clustering solutions of the same objects and draws them with ComparePlot so their agreement can be read off directly – which objects the methods place together and where they disagree – without any ground truth. Each solution may be a fitted result (with $Clust / $DistM) or a plain label vector; label vectors are turned into a co-membership dissimilarity so every solution is comparable.

Usage

```
compare_methods_plot(
  solutions,
  nrclusters,
  cols = NULL,
  cex.labels = 0.5,
  margins = c(5, 11, 3, 2),
  ...
)
```

Arguments

`solutions`	A named list of fitted results and/or label vectors, all over the same objects in the same order.
`nrclusters`	Number of clusters to cut each solution into.
`cols`	Optional cluster colours; defaults to a distinct palette.
`cex.labels`, `margins`	Passed to `ComparePlot`.
`...`	Further arguments to `ComparePlot`.

Value

Invisibly `NULL`; called for the plot. Needs **circlize** and **plotrix**.

See Also

`clustering_concordance`, `ComparePlot`

Examples

```
data(mosaic_toy)
sols <- list(
  var    = M_ABC(mosaic_toy$List, numsim = 40, NC = 3, weighting = c("var", "var")),
  nugget = M_ABC(mosaic_toy$List, numsim = 40, NC = 3, weighting = c("nugget", "nugget")),
  NEMO   = NEMO(mosaic_toy$List, k = 3)$cluster)
## Not run: compare_methods_plot(sols, nrclusters = 3)
```

`ConsensusClustering` *Voting-based consensus clustering*

Description

Consensus Clustering performs voting-based consensus on a set of single-source partitions. Each data matrix is clustered separately and the resulting label matrix is reconciled into a single consensus partition by an iterative voting scheme. Three voting methods are available: Iterative Voting Consensus (IVC), Iterative Probabilistic Voting Consensus (IPVC) and Iterative Pairwise Consensus (IPC).

Usage

```
ConsensusClustering(
  List,
  type = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  nrclusters = c(7, 7),
  gap = FALSE,
  maxK = 15,
  votingMethod = c("IVC", "IPVC", "IPC"),
  optimalk = 7
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	Indicates whether the provided matrices in "List" are either data matrices ("data"), distance matrices ("dist") or clustering results obtained from the data ("clust").
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".
nrclusters	The number of clusters to divide each individual dendrogram in. Default is c(7,7) for two data sets.
gap	Logical. Whether the optimal number of clusters should be determined with the gap statistic. Defaults to FALSE.
maxK	The maximal number of clusters to investigate in the gap statistic. Default is 15.
votingMethod	The voting method to be performed: one of "IVC", "IPVC" or "IPC".
optimalk	The number of clusters for the final consensus partition. Defaults to 7.

Value

The returned value is a list of two elements:

DistM	A NULL object
Clust	The resulting clustering

The value has class 'Ensemble'.

References

Nguyen N. and Caruana R. (2007). Consensus Clusterings. Seventh IEEE International Conference on Data Mining (ICDM 2007), 607-612.

Examples

```
## Not run:
data(mosaic_toy)
fit <- ConsensusClustering(List = mosaic_toy$List, type = "data",
  distmeasure = c("euclidean", "euclidean"), normalize = c(FALSE, FALSE),
  method = c(NULL, NULL), clust = "agnes", linkage = c("ward", "ward"),
  nrclusters = c(3, 3), gap = FALSE, votingMethod = "IVC", optimalk = 3)

## End(Not run)
```

ContFeaturesPlot *Plot of continuous features*

Description

The function ContFeaturesPlot plots the values of continuous features. It is possible to separate between objects of interest and the other objects.

Usage

```
ContFeaturesPlot(
  leadCpds,
  data,
  nrclusters = NULL,
  orderLab = NULL,
  colorLab = NULL,
  cols = NULL,
  ylab = "features",
  addLegend = TRUE,
  margins = c(5.5, 3.5, 0.5, 8.7),
  plottype = "new",
  location = NULL
)
```

Arguments

leadCpds	A character vector containing the objects one wants to separate from the others.
data	The data matrix.
nrclusters	Optional. The number of clusters to consider if colorLab is specified. Default is NULL.
orderLab	Optional. If the objects are to set in a specific order of a specific method. Default is NULL.
colorLab	The clustering result that determines the color of the labels of the objects in the plot. If NULL, the labels are black. Default is NULL.
cols	The colors for the labels of the objects. Default is NULL.
ylab	The lable of the y-axis. Default is "features".
addLegend	Logical. Indicates whether a legend should be added to the plot. Default is TRUE.
margins	Optional. Margins to be used for the plot. Default is c(5.5,3.5,0.5,8.7).
plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document, i.e. no new device is opened and the plot appears in the current device or document. Default is "new".
location	If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Value

A plot in which the values of the features of the leadCpds are separeted from the others.

Examples

```
## Not run:
data(Colors1)
Comps=c("Cpd1", "Cpd2", "Cpd3", "Cpd4", "Cpd5")

Data=matrix(sample(15, size = 50*5, replace = TRUE), nrow = 50, ncol = 5)
colnames(Data)=colnames(Data, do.NULL = FALSE, prefix = "col")
rownames(Data)=rownames(Data, do.NULL = FALSE, prefix = "row")
for(i in 1:50){
 rownames(Data)[i]=paste("Cpd",i,sep="")
}

ContFeaturesPlot(leadCpds=Comps,orderLab=rownames(Data),colorLab=NULL,data=Data,
nrclusters=7,cols=Colors1,ylab="features",addLegend=TRUE,margins=c(5.5,3.5,0.5,8.7),
plottype="new",location=NULL)

## End(Not run)
```

`create_data_nuggets` *Create data nuggets from a data matrix*

Description

`create_data_nuggets` compresses `n` observations into `M <= n` representative "nuggets". Each nugget stores a centre (location), a weight (the number of member observations) and a scale (within-nugget variability). Data nuggets preserve the covariance structure of the data - including the periphery - far better than a random sub-sample, which makes them a robust, memory-light surrogate for the full data set when clustering or computing feature weights.

Usage

```
create_data_nuggets(
  x,
  max_nuggets = NULL,
  min_size = 1L,
  engine = c("native", "datanugget"),
  scale_features = TRUE,
  seed = NULL,
  ...
)
```

Arguments

`x`	A numeric matrix or data frame with observations in the *rows* and variables in the *columns*.
`max_nuggets`	Target (maximum) number of nuggets M. Defaults to a data-driven value (see Details). Ignored when `engine = "datanugget"`.
`min_size`	Minimum number of observations per nugget (native engine only). Defaults to `1`.
`engine`	Either `"native"` (a self-contained k-means based partition, no extra dependencies) or `"datanugget"` (delegates to the **datanugget** package via `datanugget::create.DN` / `datanugget::refine.DN` when it is installed).
`scale_features`	Logical; if `TRUE` (default) the partition is computed on standardised features so that no single high-variance variable dominates the nugget assignment. Centres and scales are always reported on the original scale.
`seed`	Optional integer seed for reproducible partitioning.
`...`	Additional arguments forwarded to `datanugget::create.DN` when `engine = "datanugget"`.

Details

With the native engine the number of nuggets defaults to `min(nrow(x), max(2, floor(nrow(x) / 10)))` when `n > 50`, and to `n` otherwise (in which case every observation is its own nugget and the

nugget-weighted variance reduces to the ordinary variance - a safe fallback for small samples). The partition is obtained with `stats::kmeans`; missing values are mean-imputed for the partition step only.

Value

An object of class `"data_nugget"`: a list with elements

`centers`	$M \times p$ matrix of nugget centres.
`weights`	Length-M integer vector of nugget sizes n_i.
`scales`	$M \times p$ matrix of within-nugget variances.
`membership`	Length-n integer vector mapping each observation to its nugget.
`x_scale`	Length-M vector of total within-nugget variability $\mathrm{tr}(\mathrm{Cov})$.
`engine`	The engine used.

References

Cherasia KE, Cabrera J, Fernholz LT, Fernholz R (2023). "Data Nuggets in Supervised Learning." In *Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler*, 429–449. Springer.

Dey T, Duan Y, Cabrera J, Cheng X (2025). *WCluster: Clustering and PCA with Weights, and Data Nuggets Clustering*. R package version 1.3.0.

See Also

`nugget_feature_weights`, `M_ABC`

Examples

```
set.seed(1)
x <- rbind(matrix(rnorm(200 * 5,  0), ncol = 5),
           matrix(rnorm(200 * 5,  4), ncol = 5))
dn <- create_data_nuggets(x, max_nuggets = 40)
dn
```

`CVAA` *Cumulative Voting Aggregation (CVAA / W-CVAA)*

Description

`CVAA` performs cumulative voting consensus clustering. A reference partition is used to align the individual source partitions, which are then aggregated into a consensus stochastic membership matrix. Two variants are available: cumulative voting (`"CVAA"`) and the information-weighted variant (`"W-CVAA"`).

Usage

```
CVAA(
  Reference = NULL,
  nrclustersR = 7,
  List,
  typeL = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  nrclusters = c(7, 7),
  gap = FALSE,
  maxK = 15,
  votingMethod = c("CVAA", "W-CVAA"),
  optimalk = nrclustersR
)
```

Arguments

Reference	A reference partition (a clustering result, optionally an object produced by a weighted/CEC method). Required.
nrclustersR	The number of clusters of the reference partition. Default is 7.
List	A list of data matrices. It is assumed the rows correspond with the objects.
typeL	Indicates whether the provided matrices in "List" are either data matrices ("data"), distance matrices ("dist") or clustering results ("clust").
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE).
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL).
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible","flexible").
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625.
nrclusters	The number of clusters to divide each individual dendrogram in. Default is c(7,7).
gap	Logical. Whether the optimal number of clusters should be determined with the gap statistic. Defaults to FALSE.
maxK	The maximal number of clusters to investigate in the gap statistic. Default is 15.
votingMethod	The method to be performed: "CVAA" or "W-CVAA".
optimalk	An estimate of the final optimal number of clusters. Defaults to nrclustersR.

Details

Cumulative voting aggregation aligns each source membership matrix to a running consensus by least-squares voting, then averages cumulatively. The weighted variant orders the partitions by average information content and weights their contribution accordingly. When `optimalk` is smaller than `nrclustersR` the JS-ALink agglomeration is used to merge reference clusters.

Value

A list of two elements:

`DistM`	A NULL object
`Clust`	The resulting clustering (order, order.lab and Clusters)

The value has class 'Ensemble'.

References

Ayad H.G. and Kamel M.S. (2008). Cumulative voting consensus method for partitions with variable number of clusters. IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(1), 160-173.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
Ref <- Cluster(L[[1]], type = "data", distmeasure = "euclidean")
attr(Ref, "method") <- "Single"
res <- CVAA(Reference = Ref, nrclustersR = 5, List = L, typeL = "data",
  distmeasure = c("euclidean","euclidean"), votingMethod = "CVAA", optimalk = 5)

## End(Not run)
```

`Cyclogram` *Comparison of clustering results over multiple results in circular format*

Description

A visual comparison of all methods is handy to see which objects will always cluster together independent of the applied methods. To this aid the function `Cyclogram` has been written. The function relies on methods of the `circlize` package.

Usage

```
Cyclogram(
  List,
  nrclusters = NULL,
  cols = NULL,
  fusionsLog = FALSE,
  weightclust = FALSE,
  names = NULL,
  canvaslims = c(-1, 1, -1, 1),
  margins = c(8.1, 3.1, 3.1, 4.1),
  Highlight = NULL,
  cex.highlight = 1,
  substr = NULL,
  cex.labels = 0.9,
  plottype = "new",
  location = NULL
)
```

Arguments

List	A list of the outputs from the methods to be compared. The first element of the list will be used as the reference in ReorderToReference.
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.
cols	A character vector with the colours to be used. Default is NULL.
fusionsLog	Logical. To be handed to ReorderToReference: indicator for the fusion of clusters. Default is TRUE
weightclust	Logical. To be handed to ReorderToReference: to be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is TRUE.
names	Optional. Names of the methods to be used as labels for the columns. Default is NULL.
canvaslims	The limits for the circular dendrogam. Default is c(-1.0,1.0,-1.0,1.0).
margins	Optional. Margins to be used for the plot. Default is c(8.1,3.1,3.1,4.1).
Highlight	Optional. A list of character vectors of objects to be highlighted. Default is NULL.
cex.highlight	Magnification for the highlight text. Default is 1.
substr	Optional. A vector of length two giving start and stop positions to shorten labels. Default is NULL.
cex.labels	Magnification for the labels. Default is 0.7.
plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document, i.e. no new device is opened and the plot appears in the current device or document. Default is "new".
location	Optional. If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Details

This function makes use of the functions `ReorderToReference` and `Colorsnames`. Given a list with the outputs of several methods, the first step is to call upon `ReorderToReference` and to produce a matrix of which the columns are ordered according to the ordering of the objects of the first method in the list. Each cell represent the number of the cluster the object belongs to for a specific method indicated by the rows. The clusters are arranged in such a way that these correspond to that one cluster of the referenced method that they have the most in common with. The `circlize` package is used to visualize the matrix is a circular format. The inner element of the circle is the first element of the List parameter, the second inner element is the second element of the list and so on. The object names are written on the circular dendrogram portayed in the center of the circle.

Value

A plot which translates the matrix output of the function `ReorderToReference` into a circular format.

`DetermineWeight_SilClust` *Determines an optimal weight for weighted clustering by silhouettes widths.*

Description

The function `DetermineWeight_SilClust` determines an optimal weight for weighted similarity clustering by calculating silhouettes widths. For each given weight, a linear combination of the distance matrices of the single data sources is obtained, medoid clustering is performed and the silhouette widths are regressed against the cluster memberships. A statistic is derived and a p-value obtained via bootstrapping.

Usage

```
DetermineWeight_SilClust(
  List,
  type = c("data", "dist", "clusters"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  weight = seq(0, 1, by = 0.01),
  nrclusters = NULL,
  names = NULL,
  nboot = 10,
  StopRange = FALSE,
  plottype = "new",
  location = NULL
)
```

Arguments

List	A list of matrices of the same type. It is assumed the rows are corresponding with the objects.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. Type should be one of "data", "dist" or "clusters".
distmeasure	A vector of the distance measures to be used on each data matrix. Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices. Default is c(FALSE, FALSE).
method	A method of normalization. Default is c(NULL,NULL).
weight	Optional. A list of different weight combinations for the data sets in List. Defaults to seq(0,1,by=0.01).
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.
names	The labels to give to the elements in List. Default is NULL.
nboot	Number of bootstraps to be run. Default is 10.
StopRange	Logical. Indicates whether the distance matrices with values not between zero and one should be standardized to have so. Default is FALSE.
plottype	Should be one of "pdf","new" or "sweave". Default is "new".
location	If plottype is "pdf", a location should be provided here. Default is NULL.

Value

Two plots and a list with two elements:

Result	A data frame with the statistic for each weight combination
Weight	The optimal weight

.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)

MC7_Weight=DetermineWeight_SilClust(List=L,type="clusters",distmeasure=
c("tanimoto","tanimoto"),normalize=c(FALSE,FALSE),method=c(NULL,NULL),
weight=seq(0,1,by=0.01),nrclusters=c(7,7),names=c("FP","TP"),nboot=10,
StopRange=FALSE,plottype="new",location=NULL)
```

```
## End(Not run)
```

`DetermineWeight_SimClust` *Determines an optimal weight for weighted clustering by similarity weighted clustering.*

Description

The function `DetermineWeight_SimClust` determines an optimal weight for performing weighted similarity clustering. For each given weight, each separate clustering is compared to the clustering on a weighted dissimilarity matrix and a Jaccard coefficient is calculated. The ratio of the Jaccard coefficients closest to one indicates an optimal weight.

Usage

```
DetermineWeight_SimClust(
  List,
  type = c("data", "dist", "clusters"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  weight = seq(0, 1, by = 0.01),
  nrclusters = NULL,
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  linkageF = "ward",
  alpha = 0.625,
  gap = FALSE,
  maxK = 15,
  names = NULL,
  StopRange = FALSE,
  plottype = "new",
  location = NULL
)
```

Arguments

`List`	A list of matrices of the same type. It is assumed the rows are corresponding with the objects.
`type`	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. Type should be one of "data", "dist" or "clusters".
`distmeasure`	A vector of the distance measures to be used on each data matrix. Defaults to c("tanimoto","tanimoto").

normalize	Logical. Indicates whether to normalize the distance matrices. Default is c(FALSE, FALSE).
method	A method of normalization. Default is c(NULL,NULL).
weight	Optional. A list of different weight combinations for the data sets in List. Defaults to seq(0,1,by=0.01).
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity for the individual clusterings. Defaults to c("flexible","flexible").
linkageF	Choice of inter group dissimilarity for the final clustering. Defaults to "ward".
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625.
gap	Logical. Whether or not to calculate the gap statistic. Only if type="data". Default is FALSE.
maxK	The maximal number of clusters to consider in calculating the gap statistic. Default is 15.
names	The labels to give to the elements in List. Default is NULL.
StopRange	Logical. Indicates whether the distance matrices with values not between zero and one should be standardized to have so. Default is FALSE.
plottype	Should be one of "pdf","new" or "sweave". Default is "new".
location	If plottype is "pdf", a location should be provided here. Default is NULL.

Value

The returned value is a list with three elements:

ClustSep	The result of `Cluster` for each single element of List
Result	A data frame with the Jaccard coefficients and their ratios for each weight
Weight	The optimal weight

.

References

Perualila-Tan, N. et al. (2016). Weighted similarity-based clustering of chemical structures and bioactivity data in early drug discovery. Journal of Bioinformatics and Computational Biology.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",alpha=0.625,gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
```

```
method=NULL,clust="agnes",linkage="flexible",alpha=0.625,gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)

MCF7_Weight=DetermineWeight_SimClust(List=L,type="clusters",weight=seq(0,1,by=0.01),
nrclusters=c(7,7),distmeasure=c("tanimoto","tanimoto"),normalize=c(FALSE,FALSE),
method=c(NULL,NULL),clust="agnes",linkage=c("flexible","flexible"),linkageF="ward",
alpha=0.625,gap=FALSE,maxK=50,names=c("FP","TP"),StopRange=FALSE,plottype="new",location=NULL)

## End(Not run)
```

`DiffGenes` *Find differentially expressed genes*

Description

The `DiffGenes` function looks for differentially expressed genes for each cluster of a clustering result by comparing the objects of a cluster against all other objects. The limma method is applied to find the differentially expressed genes.

Usage

```
DiffGenes(
  List,
  Selection = NULL,
  geneExpr = NULL,
  nrclusters = NULL,
  method = "limma",
  sign = 0.05,
  topG = NULL,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL
)
```

Arguments

`List`	A list of the clustering outputs to be compared. The first element of the list will be used as the reference in `ReorderToReference`.
`Selection`	If differential gene expression should be investigated for a specific selection of objects, this selection can be provided here. Selection can be of the type "character" (names of the objects) or "numeric" (the number of specific cluster). Default is NULL.
`geneExpr`	The gene expression matrix or ExpressionSet of the objects. The rows should correspond with the genes.
`nrclusters`	Optional. The number of clusters to cut the dendrogram in. Default is NULL.

method	The method to applied to look for DE genes. For now, only the limma method is available. Default is "limma".
sign	The significance level to be handled. Default is 0.05.
topG	Overrules sign. The number of top genes to be shown. Default is NULL.
fusionsLog	Logical. To be handed to ReorderToReference: indicator for the fusion of clusters. Default is TRUE
weightclust	Logical. To be handed to ReorderToReference: to be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is TRUE.
names	Optional. Names of the methods. Default is NULL.

Details

This function relies on the suggested Bioconductor package 'limma'.

Value

The result is a list with an element per method. Each element is again a list with for each cluster the found differentially expressed genes.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(geneMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)
names=c('FP','TP')

MCF7_FT_DE = DiffGenes(List=L,geneExpr=geneMat,nrclusters=7,method="limma",sign=0.05,
topG=10,fusionsLog=TRUE,weightclust=TRUE,names=names)

## End(Not run)
```

DiffGenesSelection *Differential expression for a selection of objects*

Description

Internal function of DiffGenes.

Usage

```
DiffGenesSelection(
  List,
  Selection,
  geneExpr = NULL,
  nrclusters = NULL,
  method = "limma",
  sign = 0.05,
  topG = NULL,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL
)
```

Arguments

List	A list of the clustering outputs to be compared. The first element of the list will be used as the reference in ReorderToReference.
Selection	If differential gene expression should be investigated for a specific selection of objects, this selection can be provided here. Selection can be of the type "character" (names of the objects) or "numeric" (the number of specific cluster). Default is NULL.
geneExpr	The gene expression matrix or ExpressionSet of the objects. The rows should correspond with the genes.
nrclusters	Optional. The number of clusters to cut the dendrogram in. The number of clusters should not be specified if the interest lies only in a specific selection of objects which is known by name. Otherwise, it is required. Default is NULL.
method	The method to applied to look for DE genes. For now, only the limma method is available. Default is "limma".
sign	The significance level to be handled. Default is 0.05.
topG	Overrules sign. The number of top genes to be shown. Default is NULL.
fusionsLog	Logical. To be handed to ReorderToReference: indicator for the fusion of clusters. Default is TRUE
weightclust	Logical. To be handed to ReorderToReference: to be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is TRUE.
names	Optional. Names of the methods. Default is NULL.

Examples

```
## Not run:
MCF7_FT_DE = DiffGenes(List=L,Selection=c("Cpd1","Cpd2"),geneExpr=geneMat,method="limma")

## End(Not run)
```

`Distance` *Compute a distance matrix*

Description

Dispatcher computing an object-by-object distance matrix under a range of measures suitable for continuous, binary and mixed-type data.

Usage

```
Distance(
  Data,
  distmeasure = c("tanimoto", "jaccard", "euclidean", "hamming", "cont tanimoto",
    "MCA_coord", "gower", "chi.squared", "cosine"),
  normalize = NULL,
  method = NULL
)

Distance_v2(Data, distmeasure = "tanimoto", normalize = NULL)
```

Arguments

`Data`	A data matrix; rows are the objects.
`distmeasure`	One of `"tanimoto"`, `"jaccard"`, `"euclidean"` (true Euclidean, via `stats::dist`), `"gower"` (mixed-type, via `FD::gowdis`), `"hamming"`, `"cont tanimoto"`, `"MCA_coord"`, `"chi.squared"`, `"cosine"`.
`normalize`	Optional normalisation method (see `Normalization`), applied for the `"euclidean"` / `"cont tanimoto"` measures.
`method`	Ignored; retained for backward compatibility.

Details

`"cosine"`, `"chi.squared"` and `"MCA_coord"` require the suggested packages **lsa**, **analogue** and **FactoMineR** respectively.

Value

A symmetric numeric distance matrix.

Examples

```
m <- matrix(rbinom(40, 1, 0.5), nrow = 8)
D <- Distance(m, distmeasure = "tanimoto")
dim(D)
```

distanceheatmaps *Determine the distance in a heatmap*

Description

Internal function of HeatmapPlot

Usage

```
distanceheatmaps(Data1, Data2, names = NULL, nrclusters = 7)
```

Arguments

Data1	The resulting clustering of method 1.
Data2	The resulting clustering of method 2.
names	The names of the objects in the data sets. Default is NULL.
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.

EHC *Ensemble hierarchical clustering via graph partitioning (METIS / MST)*

Description

Ensemble Hierarchical Clustering (EHC) builds a cluster association graph from the agglomerative clustering results of each data source. Strengths of association between pairs of objects are accumulated across the dendrograms (weighted by cluster depth and intra-cluster proximity), yielding a weighted adjacency matrix that is then partitioned.

Two partitioning strategies are available through graphPartitioning:

"METIS" Delegates the partitioning of the association graph to an **external graph-partitioning executable** (METIS / gpmetis / shmetis, or the corresponding MATLAB hmetis routine). These binaries are **not bundled** with the package and must be configured through the executable argument. When executable=FALSE (the default), or when the configured executable cannot be located, the function stops with an informative error.

"MST" A pure-R minimum spanning tree partitioning that needs no external executable. It does require the **igraph** package; if **igraph** is not installed the function stops with an informative error.

Usage

```
EHC(
  List,
  type = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  gap = FALSE,
  maxK = 15,
  graphPartitioning = c("METIS", "MST"),
  optimalk = 7,
  waitingtime = 300,
  file_number = 0,
  executable = FALSE
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clust" the single source clustering is skipped. Type should be one of "data", "dist" or "clust".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in `Normalization`.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".
gap	Logical. Whether the optimal number of clusters should be determined with the gap statistic. Default is FALSE.
maxK	The maximal number of clusters to investigate in the gap statistic. Default is 15.
graphPartitioning	A character string indicating the graph partitioning strategy. One of "METIS" (external executable) or "MST" (pure R, requires igraph). Defaults to "METIS".

optimalk	An estimate of the final optimal number of clusters. Default is 7.
waitingtime	The maximum number of seconds to wait for the external program to write its result before stopping. Defaults to 300.
file_number	An identifier appended to the temporary files exchanged with the external program. Defaults to 00.
executable	Either FALSE (the default) or the path/flag identifying the external graph-partitioning executable (METIS / gpmetis / shmetis) or MATLAB to use for the "METIS" strategy. When FALSE the "METIS" path stops, since no partition can be computed in pure R. Precompiled binaries are not bundled with the package.

Value

The returned value is a list of two elements:

DistM	The weighted cluster association matrix
Clust	The resulting clustering

The value has class 'Ensemble'.

References

Mirzaei A. and Rahmati M. (2010). A novel hierarchical-clustering-combination scheme based on fuzzy-similarity relations. IEEE Transactions on Fuzzy Systems, 18(1), 27-39.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List

# METIS path requires an external graph-partitioning executable / MATLAB
# configured through the 'executable' argument.
res <- EHC(List = L, type = "data",
  distmeasure = c("euclidean", "euclidean"), normalize = c(FALSE, FALSE),
  method = c(NULL, NULL), clust = "agnes",
  linkage = c("flexible", "flexible"), alpha = 0.625, gap = FALSE,
  maxK = 15, graphPartitioning = "METIS", optimalk = 7,
  executable = "./MetisAlgorithm")

# MST path is pure R but needs the igraph package.
res2 <- EHC(List = L, type = "data", graphPartitioning = "MST")

## End(Not run)
```

`embedding_plot` *Two-dimensional cluster visualisation (PCA / UMAP / t-SNE)*

Description

Project the objects to two dimensions and colour them by cluster. `"pca"` uses classical MDS of the fused dissimilarity (always available); `"umap"` needs **uwot** and `"tsne"` needs **Rtsne**.

Usage

```
embedding_plot(
  x,
  labels,
  method = c("pca", "umap", "tsne"),
  col = NULL,
  main = NULL
)
```

Arguments

`x`	A fitted result (`$DistM`), a dissimilarity, or a `List`.
`labels`	Cluster label of each object.
`method`	`"pca"` (default), `"umap"` or `"tsne"`.
`col`	Optional cluster colours.
`main`	Plot title.

Value

Invisibly the 2-column coordinate matrix.

See Also

`cluster_selection_plot`

Examples

```
data(mosaic_toy)
fit <- M_ABC(mosaic_toy$List, numsim = 40, NC = 3)
embedding_plot(fit, mosaic_labels(fit, 3), method = "pca")
```

`enriched_features` *Tidy CharacteristicFeatures() output for plotting*

Description

Turns the output of `CharacteristicFeatures()` into a per-cluster list of compact tables (`term` / `pvalue` / `Count`) ready for `plot_pathways()`. For each cluster it keeps the characteristic features that pass a significance cut and, by default, only those that are *enriched* in the cluster (more frequent among its objects than in the remaining objects) rather than depleted. Empty clusters are dropped.

Usage

```
enriched_features(
  char,
  data,
  source = 1,
  method = 1,
  sign = 0.05,
  enriched_only = TRUE,
  clean_names = TRUE
)
```

Arguments

`char`	Output of `CharacteristicFeatures()`.
`data`	The binary data matrix whose features were tested (objects in rows, features in columns) - e.g. the target matrix. Used to set the enrichment direction and the per-cluster feature counts.
`source`	Which tested data source to use within each cluster, by name (e.g. `"Target"`) or index. Default: the first source.
`method`	Which clustering result in `char` to use, by name or index. Default `1`.
`sign`	Keep features with adjusted p-value below this. Default `0.05`.
`enriched_only`	If `TRUE` (default) keep only features more frequent in the cluster than in the rest of the objects; if `FALSE` keep every significant characteristic feature (enriched or depleted).
`clean_names`	If `TRUE` (default) tidy the feature names for display - reversing the `make.names()` mangling common in binary descriptor matrices (drop a leading `X` before a digit, turn `./_` runs into spaces), e.g. `"X3.oxo.5.alpha_steroid"` becomes `"3 oxo 5 alpha steroid"`. Set `FALSE` to keep the raw column names.

Value

A named list - one element per non-empty cluster - of data frames with columns `term`, `pvalue` (adjusted p-value) and `Count` (number of the cluster's objects carrying the feature), suitable for `plot_pathways()`.

See Also

CharacteristicFeatures(), plot_pathways()

Examples

```
## Not run:
Char <- CharacteristicFeatures(List = list(Weighted = sol),
                               binData = list(Target = targetMat),
                               datanames = "Target", nrclusters = 4)
tabs <- enriched_features(Char, targetMat)
plot_pathways(tabs, style = "circular")

## End(Not run)
```

EnsembleClustering *Ensemble clustering via graph partitioning (CSPA / HGPA / MCLA)*

Description

Cluster-based ensemble clustering that combines several single-source partitions into a single consensus partition through the graph-partitioning algorithms of Strehl and Ghosh: the Cluster-based Similarity Partitioning Algorithm (CSPA), the HyperGraph Partitioning Algorithm (HGPA) and the Meta-CLustering Algorithm (MCLA), as well as a "Best" selection strategy.

External executable required. This method does not run in pure R. It writes the stacked label matrix to disk and delegates the actual graph partitioning to an **external graph-partitioning executable** (e.g. a compiled CSPA/HGPA/MCLA binary backed by METIS / `gpmetis` / `shmetis`) or to a **MATLAB** session. These binaries are **not bundled** with the package; the user must supply and configure them. The path to the executable is selected through the `executable` argument. When `executable=FALSE` (the default), or when the configured executable cannot be located, the function stops with an informative error rather than attempting to launch an unavailable program.

Usage

```
EnsembleClustering(
  List,
  type = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  nrclusters = c(7, 7),
  gap = FALSE,
  maxK = 15,
  ensembleMethod = c("CSPA", "HGPA", "MCLA", "Best"),
  waitingtime = 300,
```

```
  file_number = 0,
  executable = FALSE
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clust" the single source clustering is skipped. Type should be one of "data", "dist" or "clust".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".
nrclusters	The number of clusters to divide each individual dendrogram in. Default is c(7,7) for two data sets.
gap	Logical. Whether the optimal number of clusters should be determined with the gap statistic. Default is FALSE.
maxK	The maximal number of clusters to investigate in the gap statistic. Default is 15.
ensembleMethod	A character string indicating the consensus function to use. One of "CSPA", "HGPA", "MCLA" or "Best". Defaults to "CSPA".
waitingtime	The maximum number of seconds to wait for the external program to write its result before stopping. Defaults to 300.
file_number	An identifier appended to the temporary files exchanged with the external program. Defaults to 00.
executable	Either FALSE (the default) or the path/flag identifying the external graph-partitioning executable (or MATLAB) to use. When FALSE the function stops, since no consensus can be computed in pure R. Precompiled binaries are not bundled with the package.

Value

The returned value is a list of two elements:

DistM	A NULL object

`Clust` The resulting consensus clustering

The value has class 'Ensemble'.

References

Strehl A. and Ghosh J. (2002). Cluster ensembles - a knowledge reuse framework for combining multiple partitions. Journal of Machine Learning Research, 3, 583-617.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List

# Requires an external graph-partitioning executable / MATLAB to be
# configured through the 'executable' argument.
res <- EnsembleClustering(List = L, type = "data",
  distmeasure = c("euclidean", "euclidean"), normalize = c(FALSE, FALSE),
  method = c(NULL, NULL), clust = "agnes",
  linkage = c("flexible", "flexible"), nrclusters = c(7, 7),
  gap = FALSE, maxK = 15, ensembleMethod = "CSPA",
  executable = "./EnsembleClusteringC")

## End(Not run)
```

`EvidenceAccumulation` *Evidence accumulation clustering (co-association)*

Description

`EvidenceAccumulation` builds a co-association (similarity) matrix from the individual source partitions and extracts a consensus partition by graph partitioning. Three graph-partitioning options are available: a minimum-spanning-tree procedure (`"MST"`, requires the **igraph** package), a thresholded single-link procedure (`"SL"`), and an agnes single-link procedure (`"SL_agnes"`).

Usage

```
EvidenceAccumulation(
  List,
  type = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  nrclusters = c(7, 7),
  gap = FALSE,
```

```
  maxK = 15,
  graphPartitioning = c("MST", "SL", "SL_agnes"),
  t = NULL
)
```

Arguments

List	A list of data matrices. It is assumed the rows correspond with the objects.
type	Indicates whether the provided matrices in "List" are either data matrices ("data"), distance matrices ("dist") or clustering results ("clust").
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE).
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL).
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible","flexible").
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625.
nrclusters	The number of clusters to divide each individual dendrogram in. Default is c(7,7).
gap	Logical. Whether the optimal number of clusters should be determined with the gap statistic. Defaults to FALSE.
maxK	The maximal number of clusters to investigate in the gap statistic. Default is 15.
graphPartitioning	The graph partitioning method to be performed: "MST", "SL" or "SL_agnes".
t	A threshold used by the "MST" and "SL" partitioning procedures. Defaults to NULL.

Details

The co-association matrix `S` records the fraction of source partitions in which two objects are co-clustered. `"MST"` prunes a minimum spanning tree of the similarity graph (optionally at threshold `t`); `"SL"` performs a thresholded single-link grouping; and `"SL_agnes"` runs agnes single linkage on the dissimilarity `1-S`. The `"MST"` path requires the suggested package **igraph**.

Value

A list of two elements:

DistM	A NULL object
Clust	The resulting clustering

The value has class 'Ensemble' (or 'Single Clustering' for "SL_agnes").

References

Fred A.L.N. and Jain A.K. (2005). Combining multiple clusterings using evidence accumulation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(6), 835-850.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
res <- EvidenceAccumulation(List = L, type = "data",
  distmeasure = c("euclidean","euclidean"), nrclusters = c(5,5),
  graphPartitioning = "SL_agnes")

## End(Not run)
```

`f.clustABC.MultiSource`

Consensus clustering from stacked ABC label matrices

Description

Turns a row-stacked matrix of per-iteration cluster labels ($K \cdot R$ rows by N columns) into a co-clustering dissimilarity and applies `cluster::agnes`. S_{ij} is the co-clustering probability over iterations selecting both objects (B6). The co-clustering counts use the compiled C++ kernel `count_coclustering_cpp` (from `src/mcpp_sampling.cpp`) when available, otherwise a pure-R loop.

Usage

```
f.clustABC.MultiSource(
  res,
  numclust = NULL,
  dissimilarity = "mabc",
  linkage = "ward",
  alpha = 0.625,
  mds = FALSE
)
```

Arguments

`res`	Integer label matrix (`0` = object not selected).
`numclust`	Optional number of clusters to cut the tree into.
`dissimilarity`	`"mabc"` (arc-cosine, default) or `"abc"` (linear).
`linkage`	AGNES agglomeration method.
`alpha`	Flexible-linkage parameter.
`mds`	Logical; draw an MDS of the dissimilarities.

Value

A list with `DistM`, `Clust` and (if `numclust` given) `cut`.

References

(Amaratunga et al. 2008)

See Also

`M_ABC`

`FeatSelection` *Determine the characteristic features of a cluster or selection*

Description

`FeatSelection` determines the characteristic features of a selection of objects or of a specific cluster across the methods. Binary features are evaluated with Fisher's exact test, continuous features with the t-test.

Usage

```
FeatSelection(
  List,
  Selection = NULL,
  binData = NULL,
  contData = NULL,
  datanames = NULL,
  nrclusters = NULL,
  topChar = NULL,
  sign = 0.05,
  fusionsLog = TRUE,
  weightclust = TRUE
)
```

Arguments

`List`	A list of clustering outputs. Default is NULL.
`Selection`	Either a character vector of objects, or a numeric cluster number. Default is NULL.
`binData`	A list of the binary feature data matrices. Default is NULL.
`contData`	A list of continuous feature data matrices. Default is NULL.
`datanames`	A vector with the names of the data matrices. Default is NULL.
`nrclusters`	The number of clusters to cut the dendrogram in. Default is NULL.
`topChar`	The number of top characteristics to return. Default is NULL.

sign	The significance level. Default is 0.05.
fusionsLog	Logical. Indicator for the fusion of clusters. Default is TRUE.
weightclust	Logical. For the outputs of CEC, WeightedClust or WeightedSimClust, if TRUE only the result of the Clust element is considered. Default is TRUE.

Value

A list with the found (top) characteristics of the feature data.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F)
MCF7_FeatSel=FeatSelection(List=L,Selection=4,binData=list(fingerprintMat),
datanames=c("FP"),nrclusters=7,topChar=20)

## End(Not run)
```

FeaturesOfCluster *List all features present in a selected cluster of objects*

Description

The function FeaturesOfCluster lists the number of features objects of the cluster have in common. A threshold can be set selecting among how many objects of the cluster the features should be shared. An optional plot of the features is available.

Usage

```
FeaturesOfCluster(
  leadCpds,
  data,
  threshold = 1,
  plot = TRUE,
  cexB = 1.1,
  cexR = 1.1,
  cexN = 1.1,
  plottype = "new",
  location = NULL
)
```

Arguments

leadCpds	A character vector containing the objects one wants to investigate in terms of features.
data	The data matrix.
threshold	The number of objects the features at least should be shared amongst. Default is set to 1.
plot	Logical. Indicates whether or not a BinFeaturesPlot should be set up. Default is TRUE.
cexB	Font size of the bottom (object) labels in the plot. Default is 1.1.
cexR	Font size of the right (feature) labels in the plot. Default is 1.1.
cexN	Font size of the data-name label in the plot. Default is 1.1.
plottype	Should be one of "pdf","new" or "sweave". Default is "new".
location	If plottype is "pdf", a location should be provided. Default is NULL.

Value

A list with 2 items: the number of shared features among the objects, and a character vector of the plotted features.

Examples

```
## Not run:
data(fingerprintMat)

Lead=rownames(fingerprintMat)[1:5]

FeaturesOfCluster(leadCpds=Lead,data=fingerprintMat,
threshold=1,plot=TRUE,plottype="new",location=NULL)

## End(Not run)
```

FindCluster *Find a selection of objects in the output of* ReorderToReference

---

Description

FindCluster selects the objects belonging to a cluster after the results of the methods have been rearranged by ReorderToReference.

Usage

```
FindCluster(
  List,
  nrclusters = NULL,
  select = c(1, 1),
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL
)
```

Arguments

List	A list of the clustering outputs to be compared. The first element of the list will be used as the reference in ReorderToReference.
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.
select	The row (the method) and the number of the cluster to select. Default is c(1,1).
fusionsLog	Logical. To be handed to ReorderToReference: indicator for the fusion of clusters. Default is TRUE.
weightclust	Logical. To be handed to ReorderToReference: to be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is TRUE.
names	Optional. Names of the methods. Default is NULL.

Value

A character vector containing the names of the objects in the selected cluster.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
res <- list(Cluster(L[[1]], type = "data", distmeasure = "euclidean"),
            Cluster(L[[2]], type = "data", distmeasure = "euclidean"))
Comps <- FindCluster(List = res, nrclusters = 5, select = c(1, 2),
                     weightclust = FALSE)

## End(Not run)
```

FindElement *Find an element in a data structure*

Description

The function FindElement is used internally in the PreparePathway function but might come in handy for other uses as well. Given the name of an object, the function searches for that object in the data structure and extracts it. When multiple objects have the same name, all are extracted.

Usage

```
FindElement(what = NULL, object = NULL, element = list())
```

Arguments

what	A character string indicating which object to look for. Default is NULL.
object	The data structure to look into. Only the classes data frame and list are supported. Default is NULL.
element	Not to be specified by the user.

Value

The returned value is a list with an element for each object found. The element contains everything the object contained in the original data structure.

Examples

```
## Not run:
obj <- list(a = 1, b = list(TopDE = 42, c = list(TopDE = 7)))
FindElement(what = "TopDE", object = obj)

## End(Not run)
```

FindGenes *Find the genes shared across methods*

Description

The FindGenes function collects the differentially expressed genes per cluster per method (the output of DiffGenes) and determines which genes are found across the methods.

Usage

```
FindGenes(dataLimma, names = NULL)
```

Arguments

dataLimma	The output of the DiffGenes function.
names	Optional. Names of the methods. Default is NULL.

Value

The result is a list with for each cluster a list of the found genes and the methods that found them.

Examples

```
## Not run:
MCF7_SharedGenes=FindGenes(dataLimma=MCF7_FT_DE,names=c("FP","TP"))

## End(Not run)
```

`Geneset.intersect` *Intersection over resulting gene sets of* `PathwaysIter` *function*

Description

The function `Geneset.intersect` collects the results of the `PathwaysIter` function per method for each cluster and takes the intersection over the iterations per cluster per method. This is to see if over the different resamplings of the data, similar pathways were discovered.

Usage

```
Geneset.intersect(
  PathwaysOutput,
  Selection = FALSE,
  sign = 0.05,
  names = NULL,
  seperatetables = FALSE,
  separatepvals = FALSE
)
```

Arguments

`PathwaysOutput`	The output of the `PathwaysIter` function.
`Selection`	Logical. Indicates whether or not the output of the pathways function were concentrated on a specific selection of objects. If this was the case, Selection should be put to TRUE. Otherwise, it should be put to FALSE. Default is TRUE.
`sign`	The significance level to be handled for cutting of the pathways. Default is 0.05.
`names`	Optional. Names of the methods. Default is NULL.
`seperatetables`	Logical. If TRUE, a separate element is created per cluster containing the pathways for each iteration. Default is FALSE.
`separatepvals`	Logical. If TRUE, the p-values of the each iteration of each pathway in the intersection is given. If FALSE, only the mean p-value is provided. Default is FALSE.

Value

The output is a list with an element per method. For each method, it is portrayed per cluster which pathways belong to the intersection over all iterations and their corresponding mean p-values.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(geneMat)
data(GeneInfo)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)

MCF7_Paths_FandT=PathwaysIter(List=L, geneExpr = geneMat, nrclusters = 7, method =
c("limma", "MLP"), geneInfo = GeneInfo, geneSetSource = "GOBP", topP = NULL,
topG = NULL, GENESET = NULL, sign = 0.05,niter=2,fusionsLog = TRUE,
weightclust = TRUE, names =names)

MCF7_Paths_intersection=Geneset.intersect(PathwaysOutput=MCF7_Paths_FandT,
sign=0.05,names=c("FP","TP"),seperatetables=FALSE,separatepvals=FALSE)

str(MCF7_Paths_intersection)

## End(Not run)
```

Geneset.intersectSelection

Intersection over resulting gene sets of `PathwaysIter` *function for a selection of objects*

Description

Internal function of `Geneset.intersect`.

Usage

```
Geneset.intersectSelection(
  list.output,
  sign = 0.05,
  names = NULL,
  seperatetables = FALSE,
  separatepvals = FALSE
)
```

Arguments

`list.output`	The output of the `PathwaysIter` function.

sign	The significance level to be handled for cutting of the pathways. Default is 0.05.
names	Optional. Names of the methods. Default is NULL.
seperatetables	Logical. If TRUE, a separate element is created per cluster containing the pathways for each iteration. Default is FALSE.
separatepvals	Logical. If TRUE, the p-values of the each iteration of each pathway in the intersection is given. If FALSE, only the mean p-value is provided. Default is FALSE.

Examples

```
## Not run:
Geneset.intersectSelection(MCF7_Paths_FandT,sign=0.05)

## End(Not run)
```

HBGF *Hybrid Bipartite Graph Formulation*

Description

The HBGF function implements the Hybrid Bipartite Graph Formulation, a graph-based consensus clustering method. A bipartite graph is constructed from the cluster ensemble (objects on one side, clusters on the other) and partitioned with a spectral method based on the singular value decomposition followed by k-means.

Usage

```
HBGF(
  List,
  type = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  nrclusters = c(7, 7),
  gap = FALSE,
  maxK = 15,
  graphPartitioning = "Spec",
  optimalk = 7
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	Indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clust" the single source clustering is skipped. Type should be one of "data", "dist" or "clust".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".
nrclusters	The number of clusters to divide each individual dendrogram in. Default is c(7,7) for two data sets.
gap	Logical. Whether the optimal number of clusters should be determined with the gap statistic. Default is FALSE.
maxK	The maximal number of clusters to investigate in the gap statistic. Default is 15.
graphPartitioning	The graph partitioning method. Currently "Spec" (spectral) is supported.
optimalk	An estimate of the final optimal number of clusters. Default is 7.

Value

The returned value is a list of two elements:

DistM	A NULL object
Clust	The resulting clustering

The value has class 'Ensemble'.

References

Fern, X. Z. and Brodley, C. E. (2004). Solving cluster ensemble problems by bipartite graph partitioning. Proceedings of the Twenty-First International Conference on Machine Learning (ICML).

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
HBGF_toy <- HBGF(List = L, type = "data",
                 distmeasure = c("euclidean", "euclidean"),
                 normalize = c(FALSE, FALSE), method = c(NULL, NULL),
                 clust = "agnes", linkage = c("ward", "ward"),
                 nrclusters = c(7, 7), graphPartitioning = "Spec", optimalk = 7)

## End(Not run)
```

`HeatmapPlot` *A heatmap of the comparison of two clustering results.*

Description

A function to plot a heatmap of the agreement between two clustering results.

These are the clusters to which a comparison is made. A matrix is set up of which the columns are determined by the ordering of clustering of method 2 and the rows by the ordering of method 1. Every column represent one object just as every row and every column represent the color of its cluster. A function visits every cell of the matrix. If the objects represented by the cell are still together in a cluster, the color of the column is passed to the cell. This creates the distance matrix which can be given to the HeatmapCols function to create the heatmap.

Usage

```
HeatmapPlot(
  Data1,
  Data2,
  names = NULL,
  nrclusters = NULL,
  cols = NULL,
  plottype = "new",
  location = NULL
)
```

Arguments

`Data1`	The resulting clustering of method 1.
`Data2`	The resulting clustering of method 2.
`names`	The names of the objects in the data sets. Default is NULL.
`nrclusters`	The number of clusters to cut the dendrogram in. Default is NULL.
`cols`	A character vector with the colours for the clusters. Default is NULL.

plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document, i.e. no new device is opened and the plot appears in the current device or document. Default is "new".
location	If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Value

A heatmap based on the distance matrix created by the function with the dendrogram of method 2 on top of the plot and the one from method 1 on the left. The names of the objects are depicted on the bottom in the order of clustering of method 2 and on the right by the ordering of method 1. Vertically the cluster of method 2 can be seen while horizontally those of method 1 are portrayed.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(Colors1)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
clust="agnes",linkage="flexible",gap=FALSE,maxK=15)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
clust="agnes",linkage="flexible",gap=FALSE,maxK=15)


L=list(MCF7_F,MCF7_T)
names=c("FP","TP")

HeatmapPlot(Data1=MCF7_T,Data2=MCF7_F,names=rownames(fingerprintMat)
,nrclusters=7,cols=Colors1,plottype="new", location=NULL)

## End(Not run)
```

HeatmapSelection *A function to select a group of objects via the similarity heatmap.*

Description

The function HeatmapSelection plots the similarity values between objects. The plot is similar to the one produced by SimilarityHeatmap but without the dendrograms on the sides. The function is rather explorative and experimental and is to be used with some caution. By clicking in the plot, the user can select a group of objects of interest. See more in Details.

A similarity heatmap is created in the same way as in SimilarityHeatmap. The user is now free to select two points on the heatmap. It is advised that these two points are in opposite corners of

a square that indicates a high similarity among the objects. The points do not have to be the exact corners of the group of interest, a little deviation is allowed as rows and columns of the selected subset of the matrix with sum equal to 1 are filtered out. A sum equal to one, implies that the compound is only similar to itself.

The function is meant to be explorative but is experimental. The goal was to make the selection of interesting objects easier as sometimes the labels of the dendrograms are too distorted to be read. If the figure is exported to a pdf file with an appropriate width and height, the labels can be become readable again.

Usage

```
HeatmapSelection(
  Data,
  type = c("data", "dist", "clust", "sim"),
  distmeasure = "tanimoto",
  normalize = FALSE,
  method = NULL,
  linkage = "flexible",
  cutoff = NULL,
  percentile = FALSE,
  dendrogram = NULL,
  width = 7,
  height = 7
)
```

Arguments

`Data`	The data of which a heatmap should be drawn.
`type`	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clusters" the single source clustering is skipped. Type should be one of "data", "dist" or "clusters".
`distmeasure`	The distance measure. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to "tanimoto".
`normalize`	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in `Normalization`.
`method`	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is NULL.
`linkage`	Choice of inter group dissimilarity (character). Defaults to "flexible".
`cutoff`	Optional. If a cutoff value is specified, all values lower are put to zero while all other values are kept. This helps to highlight the most similar objects. Default is NULL.
`percentile`	Logical. The cutoff value can be a percentile. If one want the cutoff value to be the 90th percentile of the data, one should specify cutoff = 0.90 and percentile = TRUE. Default is FALSE.

dendrogram	Optional. If the clustering results of the data is already available and should not be recalculated, this results can be provided here. Otherwise, it will be calculated given the data. This is necessary to have the objects in their order of clustering on the plot. Default is NULL.
width	The width of the plot to be made. This can be adjusted since the default size might not show a clear picture. Default is 7.
height	The height of the plot to be made. This can be adjusted since the default size might not show a clear picture. Default is 7.

Value

A heatmap with the names of the objects on the right and bottom. Once points are selected, it will return the names of the objects that are in the selected square provided that these show similarity among each other.

Examples

```
## Not run:
data(fingerprintMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55)

HeatmapSelection(Data=MCF7_F$DistM,type="dist",cutoff=0.90,percentile=TRUE,
dendrogram=MCF7_F,width=7,height=7)

## End(Not run)
```

HierarchicalEnsembleClustering

Hierarchical ensemble clustering

Description

(Zheng et al. 2014) proposed the Hierarchical Ensemble Clustering (HEC) algorithm. For each dendrogram, the cophenetic distances between the object are calculated. The distances are aggregated across the data sets and an ultra-metric which is the closest to the distance matrix is determined. A final hierarchical clustering is based on the ultra-metric values.

Usage

```
HierarchicalEnsembleClustering(
  List,
  type = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
```

```
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clusters" the single source clustering is skipped. Type should be one of "data", "dist" or"clusters".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, default is FALSE. This is recommended if different distance types are used. More details on normalization in Normalization
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".

Value

The returned value is a list of two elements:

DistM	The resulting distance matrix
Clust	The resulting hierarchical structure

The value has class 'HEC'.

References

Zheng X, Chen BM, Zheng G (2014). "A Novel Approach for Hierarchical Ensemble Clustering." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **36**, 957–970.

intNMF *Integrative NMF clustering (intNMF)*

Description

Joint non-negative matrix factorisation across data sources: a single shared basis matrix W (objects $\times$ rank) and source-specific coefficient matrices H_k minimise $\sum_k \alpha_k \|X_k - W H_k\|_F^2$ with all factors non-negative. Sources are weighted by $\alpha_k = \max_j \overline{SS}_j / \overline{SS}_k$. Object clusters are obtained by consensus of the dominant-factor assignment across restarts.

Usage

```
intNMF(List, k, nstart = 10, max_iter = 200, seed = NULL)
```

Arguments

`List`	A list of data matrices over the same objects, **objects in rows**. Matrices are shifted to be non-negative internally.
`k`	Number of clusters (also the factorisation rank).
`nstart`	Number of random restarts forming the consensus.
`max_iter`	Maximum multiplicative-update iterations per restart.
`seed`	Optional RNG seed.

Value

A list of class `"intNMF"` with `cluster`, the consensus `DistM`, an `agnes` `Clust`, the best-fit basis `W` and source weights `alpha`.

References

Chalise, P. & Fridley, B. L. (2017). Integrative clustering of multi-level omic data based on non-negative matrix factorization. PLoS ONE 12:e0176278. Reviewed in Zhang et al. (2022), WIREs Comput Stat 14:e1553.

See Also

M_ABC, SNF

Examples

```
data(mosaic_toy)
fit <- intNMF(mosaic_toy$List, k = 3, nstart = 5, seed = 1)
table(fit$cluster)
```

LabelCols *Helper that colours specific dendrogram leaves*

Description

Internal helper applied via `dendrapply` that colours the leaves and edges of a dendrogram for one or two selections of objects. Used by `LabelPlot`.

Usage

```
LabelCols(x, Sel1, Sel2 = NULL, col1 = NULL, col2 = NULL)
```

Arguments

`x`	A node of a dendrogram (passed by `dendrapply`).
`Sel1`	The selection of objects to be colored.
`Sel2`	An optional second selection to be colored. Default is NULL.
`col1`	The color for the first selection. Default is NULL.
`col2`	The color for the optional second selection. Default is NULL.

Value

The dendrogram node `x` with coloured `nodePar` and `edgePar` attributes.

LabelPlot *Coloring specific leaves of a dendrogram*

Description

The function plots a dendrogrmam of which specific leaves are coloured.

Usage

```
LabelPlot(Data, sel1, sel2 = NULL, col1 = NULL, col2 = NULL)
```

Arguments

`Data`	The result of a method which contains the dendrogram to be colored.
`sel1`	The selection of objects to be colored. Default is NULL.
`sel2`	An optional second selection to be colored. Default is NULL.
`col1`	The color for the first selection. Default is NULL.
`col2`	The color for the optional second selection. Default is NULL.

Value

A plot of the dendrogram of which the leaves of the selection(s) are colored.

Examples

```
## Not run:
data(fingerprintMat)
MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

ClustF_6=cutree(MCF7_F$Clust,6)

SelF=rownames(fingerprintMat)[ClustF_6==6]
SelF

LabelPlot(Data=MCF7_F,sel1=SelF,sel2=NULL,col1='darkorchid')

## End(Not run)
```

LinkBasedClustering *Link-based cluster ensembles*

Description

Link-based cluster ensembles combine multiple base clusterings into a single, refined similarity matrix by exploiting the link structure between clusters of the ensemble. Three refinement schemes are available: Connected-Triple-Based Similarity ("cts"), SimRank-Based Similarity ("srs") and Approximate SimRank-Based Similarity ("asrs"). The refined object-by-object similarity matrix is turned into a dissimilarity matrix on which a final agglomerative hierarchical clustering is performed.

Usage

```
LinkBasedClustering(
  List,
  type = c("data", "dist", "clust"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625,
  nrclusters = c(7, 7),
  gap = FALSE,
  maxK = 15,
  linkBasedMethod = c("cts", "srs", "asrs"),
  decayfactor = 0.8,
```

```
  niter = 5,
  linkBasedLinkage = "ward",
  waitingtime = 300,
  file_number = 0,
  executable = FALSE
)
```

Arguments

List	A list of data matrices, distance matrices or clustering results. It is assumed the rows are corresponding with the objects.
type	Indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. Should be one of "data", "dist" or "clust".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile", "Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".
nrclusters	A vector with the number of clusters to cut each base dendrogram into. If NULL and gap=FALSE, the function stops.
gap	Logical. Whether to use the gap statistic to determine the number of clusters per data set. Defaults to FALSE.
maxK	The maximal number of clusters to consider when gap=TRUE.
linkBasedMethod	The link-based refinement scheme. One of "cts", "srs" or "asrs". Defaults to "cts".
decayfactor	The decay (confidence) factor used by the refinement scheme. Defaults to 0.8.
niter	The number of iterations for the SimRank-based scheme ("srs"). Defaults to 5.
linkBasedLinkage	The linkage used in the final agglomerative clustering of the refined similarity matrix. Defaults to "ward".
waitingtime	Retained for backward compatibility with the external executable / MATLAB workflow; ignored by the pure-R schemes.
file_number	Retained for backward compatibility with the external executable / MATLAB workflow; ignored by the pure-R schemes.

executable — Logical. If TRUE, the refined similarity matrix is computed by an external executable / MATLAB. The "cts" and "srs" schemes are pure R and work without an executable. The "asrs" scheme is only available through the external executable; if executable=FALSE it stops with an informative message.

Details

Each base clustering of the ensemble is represented as a hard partition. The clusters of all partitions form the vertices of a weighted graph; the link weights between clusters are derived from shared objects. The refined object-by-object similarity is obtained by propagating these cluster-to-cluster links ("cts", "srs", "asrs"). See Iam-On et al. (2010).

Value

The returned value is a list of two elements:

DistM — The resulting (refined) distance matrix

Clust — The resulting hierarchical clustering

The value has class 'LinkBased'.

References

Iam-On, N., Boongoen, T., Garrett, S. and Price, C. (2010). Link-based cluster ensembles for the combination of multiple clusterings. IEEE Transactions on Knowledge and Data Engineering, 23(12), 1839-1853.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
res <- LinkBasedClustering(List = L, type = "data",
  distmeasure = c("euclidean", "euclidean"), normalize = c(FALSE, FALSE),
  linkBasedMethod = "cts", decayfactor = 0.8, niter = 5,
  linkBasedLinkage = "ward", nrclusters = c(3, 3))

## End(Not run)
```

LUCID *LUCID: latent unknown clustering integrating omics with an outcome*

Description

Thin wrapper around **LUCIDus**' `estimate_lucid`. LUCID is a *model-based, supervised* integrative clustering method: it defines a categorical latent variable (the clusters) jointly from upstream exposures `G`, multi-omics data `Z` and an outcome `Y` via an EM algorithm (a quasi-mediation model). Unlike the unsupervised methods in MosaiClusteR it requires an outcome and returns posterior cluster membership.

Usage

```
LUCID(
  G,
  Z,
  Y,
  K,
  lucid_model = c("early", "parallel", "serial"),
  family = c("normal", "binary"),
  ...
)
```

Arguments

G	Exposure/predictor matrix (objects in rows).
Z	Omics matrix, or a list of matrices for the parallel/serial models (objects in rows).
Y	Outcome vector (length = number of objects).
K	Number of latent clusters.
lucid_model	One of "early", "parallel", "serial".
family	Outcome family, "normal" or "binary".
...	Further arguments forwarded to LUCIDus::estimate_lucid.

Value

A list of class "LUCID" with the fitted model and the posterior cluster assignment per object.

References

Zhao, Y., Jia, K., Goodrich, J. & Conti, D. V. (2024). LUCIDus: an R package for implementing LUCID with phenotypic traits. The R Journal 16(2).

See Also

M_ABC, intNMF

Examples

```
## Not run:
# requires the 'LUCIDus' package
data(mosaic_toy)
Z <- mosaic_toy$List[[1]]
G <- mosaic_toy$List[[2]][, 1:3]
Y <- as.numeric(mosaic_toy$truth)
fit <- LUCID(G = G, Z = Z, Y = Y, K = 3)
table(fit$cluster)

## End(Not run)
```

mosaic_labels *Extract a flat partition from a MosaiClusteR result*

Description

Cuts the hierarchical clustering carried in a MosaiClusteR result object into k groups. Works with any method returning a Clust element (an agnes/hclust object) and/or a DistM.

Usage

```
mosaic_labels(object, k, linkage = "ward")
```

Arguments

object	A MosaiClusteR result (e.g. from M_ABC), an agnes/hclust object, or a distance matrix.
k	Number of clusters.
linkage	Linkage used when object is a bare distance matrix.

Value

An integer vector of cluster labels, named by object.

Examples

```
set.seed(1)
s <- paste0("o", 1:20)
L <- list(matrix(rnorm(20 * 40), 20, dimnames = list(s, NULL)),
          matrix(rnorm(20 * 50), 20, dimnames = list(s, NULL)))
fit <- M_ABC(L, numsim = 30)
mosaic_labels(fit, k = 3)
```

mosaic_sim *Simulate a multi-source data set with planted clusters*

Description

Generates K feature-by-sample matrices that share a common set of n samples drawn from g ground-truth groups. Each source has its own informative features (shifted between groups) plus noise features, so the clusters are only partially visible in any single source - the canonical setting in which multi-source integration helps.

Usage

```
mosaic_sim(
  n = 60,
  g = 3,
  p = c(120, 150),
  informative = c(20, 25),
  effect = 2.5,
  K = 2,
  seed = NULL
)
```

Arguments

n	Number of samples (objects).
g	Number of ground-truth groups.
p	Per-source feature counts (recycled to length K).
informative	Per-source number of group-informative features.
effect	Between-group mean shift for informative features.
K	Number of sources.
seed	Optional RNG seed.

Value

A list with List (the K matrices, **objects in rows**, features in columns) and truth (the integer group label per object).

Examples

```
sim <- mosaic_sim(n = 30, g = 3, K = 2, seed = 1)
lengths(lapply(sim$List, dim))
```

---

mosaic_toy *Toy multi-omics example data*

Description

A small two-source example produced by mosaic_sim with three planted groups, bundled for examples and tests.

Usage

```
data(mosaic_toy)
```

Format

A list with two elements:

List A list of two numeric matrices with the 60 shared objects in rows and features in columns (120 and 150 features).

truth Integer vector of the 60 ground-truth group labels.

Examples

```
data(mosaic_toy)
str(mosaic_toy, max.level = 2)
```

M_ABC *Multi-source Aggregating Bundles of Clusters (M-ABC), C++-accelerated*

Description

Extends the single-source ABC algorithm of (Amaratunga et al. 2008) to $K \geq 2$ sources: runs `ABCpp.SingleInMultiple` per source, row-stacks the per-iteration label matrices, and builds a single consensus dissimilarity via `f.clustABC.MultiSource` (C++ kernel when available).

Usage

```
M_ABC(
  List,
  distmeasure = rep("euclidean", length(List)),
  weighting = rep("var", length(List)),
  normalize = rep(FALSE, length(List)),
  gr = c(),
  bag = rep(TRUE, length(List)),
  numsim = 1000,
  numvar = rep(100, length(List)),
  linkage = rep("ward.D2", length(List)),
  NC = NULL,
  dissimilarity = "mabc",
  final_linkage = "ward",
  alpha = 0.625,
  mds = FALSE,
  nfeat = NULL,
  nugget_type = "between",
  nugget_args = list()
)
```

Arguments

List	A list of K data matrices over the same objects, **objects (samples) in rows** and features in columns.
distmeasure, weighting, normalize, bag, numvar, linkage	Per-source vectors (recycled to K) forwarded to ABCpp.SingleInMultiple.
gr	Unused; kept for compatibility.
numsim	Iterations per source.
NC	Base-cluster count per source (NULL = data-driven).
dissimilarity	Consensus dissimilarity, "mabc" or "abc".
final_linkage, alpha	Final clustering controls.
mds	Logical; MDS of the consensus dissimilarities.
nfeat	Feature-subsample size forwarded to ABCpp.SingleInMultiple (NULL = $\lfloor\sqrt{G}\rfloor$).
nugget_type, nugget_args	Forwarded to the nugget weighting.

Value

A list of class "M_ABC" with DistM and Clust.

Data-nugget weighting

Set weighting = "nugget" (per source, recycled) to weight features by the data-nugget between-group variance instead of the classic variance; see ABCpp.SingleInMultiple and create_data_nuggets.

References

Amaratunga D, Cabrera J, Kovtun V (2008). "Microarray Learning with ABC." *Biostatistics*, **9**(1), 128–136. doi:10.1093/biostatistics/kxm017. https://academic.oup.com/biostatistics/article-abstract/9/1/128/254198.

See Also

ABCpp.SingleInMultiple, f.clustABC.MultiSource, create_data_nuggets

Examples

```
data(mosaic_toy)
fit <- M_ABC(mosaic_toy$List, numsim = 50, NC = 3,
             weighting = c("var", "nugget"))
table(stats::cutree(stats::as.hclust(fit$Clust), 3))
```

M_ABCdeep *Multi-source M-ABC with a deep-learning (autoencoder) Step 4 (M-ABCdeep)*

Description

The deep-learning counterpart of M_ABC: runs ABCdeep.SingleInMultiple per source (autoencoder embedding + latent clustering for the base partitions), row-stacks the per-iteration label matrices, and builds one consensus dissimilarity via f.clustABC.MultiSource. The ABC algorithm of (Amaratunga et al. 2008) is preserved end to end; only Step 4 is deep. There is no Python/torch dependency and no PCA fallback (see ABCdeep.SingleInMultiple).

Usage

```
M_ABCdeep(
  List,
  weighting = rep("var", length(List)),
  normalize = rep(FALSE, length(List)),
  gr = c(),
  bag = rep(TRUE, length(List)),
  numsim = 500,
  numvar = rep(100, length(List)),
  NC = NULL,
  dissimilarity = "mabc",
  final_linkage = "ward",
  alpha = 0.625,
  mds = FALSE,
  topk_frac = 0.2,
  latent_dim = "auto",
  hidden_units = 32L,
  ae_epochs = 80L,
  ae_lr = 0.001,
  cluster_in_latent = "ward",
  linkage = rep("ward.D2", length(List)),
  nugget_type = "between",
  nugget_args = list(),
  min_samples = 40L,
  seed = NULL
)
```

Arguments

List	A list of K data matrices over the same objects, **objects (samples) in rows** and features in columns.
weighting, normalize, bag, numvar	Per-source vectors (recycled to K) forwarded to ABCdeep.SingleInMultiple.
gr	Unused; kept for compatibility.

numsim	Iterations per source.
NC	Base-cluster count per source (NULL = data-driven).
dissimilarity	Consensus dissimilarity, "mabc" or "abc".
final_linkage, alpha	Final clustering controls.
mds	Logical; MDS of the consensus dissimilarities.
topk_frac	Fraction of top-weighted features the autoencoder focuses on (default 0.2); forwarded to ABCdeep.SingleInMultiple.
latent_dim, hidden_units, ae_epochs, ae_lr, cluster_in_latent, linkage	Autoencoder / latent-clustering controls (scalar or length-K), forwarded to ABCdeep.SingleInMultiple.
nugget_type, nugget_args	Forwarded to the nugget weighting.
min_samples	Warn (per source) when N is below this; the autoencoder is underdetermined at very small N (default 40).
seed	Optional RNG seed.

Value

A list of class "M_ABC" with DistM and Clust.

References

Amaratunga D, Cabrera J, Kovtun V (2008). "Microarray Learning with ABC." *Biostatistics*, **9**(1), 128–136. doi:10.1093/biostatistics/kxm017. https://academic.oup.com/biostatistics/article-abstract/9/1/128/254198.

See Also

ABCdeep.SingleInMultiple, M_ABC, f.clustABC.MultiSource

Examples

```
data(mosaic_toy)
fit <- M_ABCdeep(mosaic_toy$List, numsim = 40, NC = 3, ae_epochs = 40, seed = 1)
table(stats::cutree(stats::as.hclust(fit$Clust), 3))
```

M_ABCdist *Multi-source M-ABC with distance accumulation*

Description

Runs ABCdist.SingleInMultiple on each source to obtain a [0, 1] dissimilarity, combines them as a weighted mean $D = \sum_k w_k D_k$ and applies a final agglomerative clustering.

Usage

```
M_ABCdist(
  List,
  source_weights = NULL,
  distmeasure = rep("euclidean", length(List)),
  weighting = rep("var", length(List)),
  normalize = rep(list(NULL), length(List)),
  gr = c(),
  bag = rep(TRUE, length(List)),
  numsim = 1000L,
  numvar = rep(100L, length(List)),
  linkage = rep("ward.D", length(List)),
  NC = NULL,
  NC2 = NULL,
  final_linkage = "ward",
  alpha = 0.625,
  mds = FALSE,
  nfeat = NULL,
  nugget_type = "between",
  nugget_args = list()
)
```

Arguments

`List`	A list of K data matrices over the same objects, **objects (samples) in rows** and features in columns.
`source_weights`	Optional non-negative source weights (normalised to sum to one); `NULL` = equal weights.
`distmeasure`, `weighting`, `normalize`, `bag`, `numvar`, `linkage`	Per-source vectors (recycled to K) forwarded to `ABCpp.SingleInMultiple`.
`gr`	Unused; kept for compatibility.
`numsim`	Iterations per source.
`NC`	Base-cluster count per source (`NULL` = data-driven).
`NC2`	Optional number of clusters to cut the final tree into.
`final_linkage`, `alpha`	Final clustering controls.
`mds`	Logical; MDS of the consensus dissimilarities.
`nfeat`	Feature-subsample size forwarded to `ABCpp.SingleInMultiple` (`NULL` = $\lfloor\sqrt{G}\rfloor$).
`nugget_type`, `nugget_args`	Forwarded to the nugget weighting.

Value

A list of class `"M_ABCdist"` with `source_D`, `DistM`, `weights` and `Clust`.

References

(Amaratunga et al. 2008)

See Also

M_ABC, ABCdist.SingleInMultiple, M_ABCdist.WC

Examples

```
data(mosaic_toy)
fit <- M_ABCdist(mosaic_toy$List, numsim = 40, weighting = c("var", "nugget"))
dim(fit$DistM)
```

M_ABCdist.WC *Multi-source M-ABC fused with WeightedClust*

Description

Runs ABCdist.SingleInMultiple on each source, then feeds the per-source dissimilarities to WeightedClust (type = "dist") to explore convex combinations over a weight grid and return the focus-weight clustering.

Usage

```
M_ABCdist.WC(
  List,
  distmeasure = rep("euclidean", length(List)),
  weighting = rep("var", length(List)),
  normalize = rep(list(NULL), length(List)),
  gr = c(),
  bag = rep(TRUE, length(List)),
  numsim = 1000L,
  numvar = rep(100L, length(List)),
  linkage = rep("ward.D", length(List)),
  NC = NULL,
  NC2 = NULL,
  StopRange = FALSE,
  wc_weight = seq(1, 0, -0.1),
  wc_weightclust = 0.5,
  wc_linkage = "ward",
  wc_alpha = 0.625,
  mds = FALSE,
  nugget_type = "between",
  nugget_args = list()
)
```

Arguments

List	A list of K data matrices over the same objects, **objects (samples) in rows** and features in columns.
distmeasure, weighting, normalize, bag, numvar, linkage	Per-source vectors (recycled to K) forwarded to ABCpp.SingleInMultiple.
gr	Unused; kept for compatibility.
numsim	Iterations per source.
NC	Base-cluster count per source (NULL = data-driven).
NC2	Optional number of clusters to cut the final tree into.
StopRange	Passed to WeightedClust.
wc_weight	Weight grid for WeightedClust.
wc_weightclust	Focus weight whose result is returned in Clust.
wc_linkage, wc_alpha	Final clustering controls for WeightedClust.
mds	Logical; MDS of the consensus dissimilarities.
nugget_type, nugget_args	Forwarded to the nugget weighting.

Value

A list of class "M_ABCdist_WC" with source_D, the full WC object, the focus DistM and Clust, and the weight_grid.

References

(Amaratunga et al. 2008)

See Also

M_ABCdist, WeightedClust

Examples

```
data(mosaic_toy)
fit <- M_ABCdist.WC(mosaic_toy$List, numsim = 40,
                    wc_weight = seq(1, 0, -0.25), wc_weightclust = 0.5)
dim(fit$DistM)
```

NEMO *NEMO: neighborhood-based multi-omics clustering*

Description

NEMO (Wang et al. 2014) integrates several omics by building a locally-scaled, k-nearest-neighbour affinity network for each one and averaging them, for every pair of objects, over only the omics in which both objects are measured. Partial / missing modalities are therefore handled without imputation. The integrated affinity is clustered spectrally.

Usage

```
NEMO(List, k = NULL, NN = 20, normalize = TRUE)
```

Arguments

`List`	A list of data matrices over the same objects, **objects in rows**. Sources may cover different subsets of objects (matched by row name) for the partial-data setting.
`k`	Number of clusters (`NULL` = eigengap estimate).
`NN`	Number of neighbours for the affinity graphs and local scaling.
`normalize`	Logical; row-normalise each affinity to a transition matrix before integration (NEMO default `TRUE`).

Value

A list of class `"NEMO"` with `FusedM` (integrated affinity), `DistM` (`1 - FusedM`), `cluster` (labels) and `Clust` (an `agnes` object on `DistM` for interoperability).

References

(Wang et al. 2014); Rappoport, N. & Shamir, R. (2019). NEMO: cancer subtyping by integration of partial multi-omic data. Bioinformatics 35:3348-3356.

See Also

SNF, spectral_clustering

Examples

```
data(mosaic_toy)
fit <- NEMO(mosaic_toy$List, k = 3, NN = 15)
table(fit$cluster)
```

`Normalization` *Normalisation of features*

Description

When data of different scales are combined it is recommended to normalise so that the structures are comparable.

Usage

```
Normalization(
  Data,
 method = c("Quantile", "Fisher-Yates", "Standardize", "Range", "Q", "q", "F", "f", "S",
    "s", "R", "r")
)
```

Arguments

`Data`	A data matrix; rows are assumed to correspond to the objects.
`method`	One of `"Quantile"`, `"Fisher-Yates"`, `"Standardize"`, `"Range"` (or any unambiguous first letter).

Details

`"Quantile"` performs quantile normalisation (common in omics); `"Fisher-Yates"` uses normal scores based only on the number of rows; `"Standardize"` centres and scales each column (base `scale`); `"Range"` maps the matrix to `[0, 1]`.

Value

The normalised matrix.

Examples

```
x <- matrix(rnorm(100), 10, 10)
Norm_x <- Normalization(x, method = "R")
```

nugget_cluster *Data-nugget clustering*

Description

Cluster a (potentially large) data set by first compressing its observations into weighted data nuggets (create_data_nuggets), clustering the nugget centres with a weighted clusterer, and mapping the nugget labels back to the original observations. This is the clustering analogue of the data-nugget weighting used by M_ABC and follows DN.Wkmeans / DN.Whclust from the WCluster package.

Usage

```
nugget_cluster(
  data,
  k,
  method = c("kmeans", "ward"),
  max_nuggets = NULL,
  nugget_args = list()
)
```

Arguments

data	A numeric matrix (objects in rows) *or* a precomputed create_data_nuggets{data_nugget} object. For multi-source data, column-bind the per-source matrices first (the direct/ADC strategy) or pass a single fused matrix.
k	Number of clusters.
method	Weighted clusterer for the nugget centres: "kmeans" (WWCSS) or "ward" (weighted Ward).
max_nuggets, nugget_args	Passed to create_data_nuggets when data is a raw matrix.

Value

A list of class "nugget_cluster" with cluster (labels for the original objects), nugget_cluster (labels for the nuggets) and the nuggets object.

References

Cherasia et al. (2023); Dey et al. (2025), WCluster.

See Also

create_data_nuggets, Wkmeans

Examples

```
data(mosaic_toy)
X <- do.call(cbind, mosaic_toy$List)        # fuse sources (objects in rows)
fit <- nugget_cluster(X, k = 3, max_nuggets = 30)
table(fit$cluster)
```

`nugget_feature_weights`

Feature weights derived from data nuggets

Description

Computes a per-feature importance score from a data-nugget (`create_data_nuggets`) object built over the *observations* (samples). This score is the data-nugget analogue of the feature variance used by the ABC / M-ABC family: it rewards features that separate the representative groups while discounting within-nugget noise, giving a robust, Big-Data-friendly weighting.

Usage

```
nugget_feature_weights(nuggets, type = c("between", "ratio", "inv_scale"))
```

Arguments

`nuggets`	A `"data_nugget"` object from `create_data_nuggets` (built with samples in the rows).
`type`	One of:
	`"between"` (default) size-weighted between-nugget variance of each feature: $\sum_i w_i (c_{ij} - \bar{c}_j)^2 / \sum_i w_i$.
	`"ratio"` a separability / pseudo-F score, $\text{between}_j / (\text{between}_j + \overline{\text{within}}_j)$.
	`"inv_scale"` inverse mean within-nugget variance, which down-weights noisy features.

Value

A named numeric vector of length `ncol(centers)` of non-negative feature scores.

References

Cherasia KE, Cabrera J, Fernholz LT, Fernholz R (2023). "Data Nuggets in Supervised Learning." In *Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler*, 429–449. Springer.

See Also

`create_data_nuggets`, `M_ABC`

Examples

```
set.seed(1)
x <- cbind(signal = c(rnorm(50, 0), rnorm(50, 5)),  # separates groups
           noise  = rnorm(100))                      # pure noise
dn <- create_data_nuggets(x, max_nuggets = 20)
nugget_feature_weights(dn, type = "between")
```

PathwayAnalysis *Pathway analysis with intersection over iterations*

Description

The PathwayAnalysis function performs pathway analysis multiple times via PathwaysIter and takes the intersection over the iterations via Geneset.intersect.

Usage

```
PathwayAnalysis(
  List,
  Selection = NULL,
  geneExpr = NULL,
  nrclusters = NULL,
  method = c("limma", "MLP"),
  geneInfo = NULL,
  geneSetSource = "GOBP",
  topP = NULL,
  topG = NULL,
  GENESET = NULL,
  sign = 0.05,
  niter = 10,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL,
  seperatetables = FALSE,
  separatepvals = FALSE
)
```

Arguments

List	A list of clustering outputs or output of theDiffGenes function.
Selection	If pathway analysis should be conducted for a specific selection of objects, this selection can be provided here. Default is NULL.
geneExpr	The gene expression matrix or ExpressionSet of the objects.
nrclusters	Optional. The number of clusters to cut the dendrogram in. Default is NULL.
method	The methods for gene and pathway analysis. Default is c("limma","MLP").

geneInfo	A data frame with at least the columns ENTREZID and SYMBOL. Default is NULL.
geneSetSource	The source for the getGeneSets function, defaults to "GOBP".
topP	Overrules sign. The number of pathways to display for each cluster. Default is NULL.
topG	Overrules sign. The number of top genes to be returned. Default is NULL.
GENESET	Optional. Can provide own candidate gene sets. Default is NULL.
sign	The significance level to be handled. Default is 0.05.
niter	The number of times to perform pathway analysis. Default is 10.
fusionsLog	Logical. To be handed to ReorderToReference. Default is TRUE
weightclust	Logical. To be handed to ReorderToReference. Default is TRUE.
names	Optional. Names of the methods. Default is NULL.
seperatetables	Logical. Default is FALSE.
separatepvals	Logical. Default is FALSE.

Details

This function relies on the suggested Bioconductor packages 'MLP', 'biomaRt' and 'org.Hs.eg.db'.

Value

The intersection over the iterations of the pathway analysis.

Examples

```
## Not run:
MCF7_PathsFandT=PathwayAnalysis(List=L, geneExpr = geneMat, nrclusters = 7,
method = c("limma","MLP"), geneInfo = GeneInfo, geneSetSource = "GOBP",niter=2)

## End(Not run)
```

Pathways *Pathway analysis for multiple clustering results*

Description

A pathway analysis per cluster per method is conducted.

Usage

```
Pathways(
  List,
  Selection = NULL,
  geneExpr = NULL,
  nrclusters = NULL,
  method = c("limma", "MLP"),
  geneInfo = NULL,
  geneSetSource = "GOBP",
  topP = NULL,
  topG = NULL,
  GENESET = NULL,
  sign = 0.05,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL
)
```

Arguments

List	A list of clustering outputs or output of theDiffGenes function. The first element of the list will be used as the reference in ReorderToReference. The output of ChooseFeatures is also accepted.
Selection	If pathway analysis should be conducted for a specific selection of objects, this selection can be provided here. Selection can be of the type "character" (names of the objects) or "numeric" (the number of specific cluster). Default is NULL.
geneExpr	The gene expression matrix or ExpressionSet of the objects. The rows should correspond with the genes.
nrclusters	Optional. The number of clusters to cut the dendrogram in. The number of clusters should not be specified if the interest lies only in a specific selection of objects which is known by name. Otherwise, it is required. Default is NULL.
method	The method to applied to look for differentially expressed genes and related pathways. For now, only the limma method is available for gene analysis and the MLP method for pathway analysis. Default is c("limma","MLP").
geneInfo	A data frame with at least the columns ENTREZID and SYMBOL. This is necessary to connect the symbolic names of the genes with their EntrezID in the correct order. The order of the gene is here not in the order of the rownames of the gene expression matrix but in the order of their significance. Default is NULL.
geneSetSource	The source for the getGeneSets function, defaults to "GOBP".
topP	Overrules sign. The number of pathways to display for each cluster. If not specified, only the significant genes are shown. Default is NULL.
topG	Overrules sign. The number of top genes to be returned in the result. If not specified, only the significant genes are shown. Defaults is NULL.
GENESET	Optional. Can provide own candidate gene sets. Default is NULL.

sign	The significance level to be handled. Default is 0.05.
fusionsLog	Logical. To be handed to ReorderToReference: indicator for the fusion of clusters. Default is TRUE
weightclust	Logical. To be handed to ReorderToReference: to be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is TRUE.
names	Optional. Names of the methods. Default is NULL.

Details

After finding differently expressed genes, it can be investigated whether pathways are related to those genes. This can be done with the help of the function Pathways which makes use of the MLP function of the MLP package. Given the output of a method, the cutree function is performed which results into a specific number of clusters. For each cluster, the limma method is performed comparing this cluster to the other clusters. This to obtain the necessary p-values of the genes. These are used as the input for the MLP function to find interesting pathways. By default the candidate gene sets are determined by the AnnotateEntrezIDtoGO function. The default source will be GOBP, but this can be altered. Further, it is also possible to provide own candidate gene sets in the form of a list of pathway categories in which each component contains a vector of Entrez Gene identifiers related to that particular pathway. The default values for the minimum and maximum number of genes in a gene set for it to be considered were used. For MLP this is respectively 5 and 100. If a list of outputs of several methods is provided as data input, the cluster numbers are rearranged according to a reference method. The first method is taken as the reference and ReorderToReference is applied to get the correct ordering. When the clusters haven been re-appointed, the pathway analysis as described above is performed for each cluster of each method.

This function relies on the suggested Bioconductor packages 'MLP', 'biomaRt' and 'org.Hs.eg.db'.

Value

The returned value is a list with an element per cluster per method.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(geneMat)
data(GeneInfo)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)
names=c('FP','TP')

MCF7_PathsFandT=Pathways(List=L, geneExpr = geneMat, nrclusters = 7, method = c("limma",
"MLP"), geneInfo = GeneInfo, geneSetSource = "GOBP", topP = NULL,
```

```
topG = NULL, GENESET = NULL, sign = 0.05,fusionsLog = TRUE, weightclust = TRUE,
 names =names)

## End(Not run)
```

PathwaysIter *Iterations of the pathway analysis*

Description

The MLP method to perform pathway analysis is based on resampling of the data. Therefore it is recommended to perform the pathway analysis multiple times to observe how much the results are influenced by a different resample. The function PathwaysIter performs the pathway analysis as described in Pathways a specified number of times. The input can be one data set or a list as in Pathway.2 and Pathways.

Usage

```
PathwaysIter(
  List,
  Selection = NULL,
  geneExpr = NULL,
  nrclusters = NULL,
  method = c("limma", "MLP"),
  geneInfo = NULL,
  geneSetSource = "GOBP",
  topP = NULL,
  topG = NULL,
  GENESET = NULL,
  sign = 0.05,
  niter = 10,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL
)
```

Arguments

List	A list of clustering outputs or output of theDiffGenes function. The first element of the list will be used as the reference in ReorderToReference. The output of ChooseFeatures is also accepted.
Selection	If pathway analysis should be conducted for a specific selection of objects, this selection can be provided here. Selection can be of the type "character" (names of the objects) or "numeric" (the number of specific cluster). Default is NULL.
geneExpr	The gene expression matrix of the objects. The rows should correspond with the genes.
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.

method	The method to applied to look for differentially expressed genes and related pathways. For now, only the limma method is available for gene analysis and the MLP method for pathway analysis. Default is c("limma","MLP").
geneInfo	A data frame with at least the columns ENTREZID and SYMBOL. This is necessary to connect the symbolic names of the genes with their EntrezID in the correct order. The order of the gene is here not in the order of the rownames of the gene expression matrix but in the order of their significance. Default is NULL.
geneSetSource	The source for the getGeneSets function ("GOBP", "GOMF","GOCC", "KEGG" or "REACTOME"). Default is "GOBP".
topP	Overrules sign. The number of pathways to display for each cluster. If not specified, only the significant genes are shown. Default is NULL.
topG	Overrules sign. The number of top genes to be returned in the result. If not specified, only the significant genes are shown. Default is NULL.
GENESET	Optional. Can provide own candidate gene sets. Default is NULL.
sign	The significance level to be handled. Default is 0.05.
niter	The number of times to perform pathway analysis. Default is 10.
fusionsLog	Logical. To be handed to ReorderToReference: indicator for the fusion of clusters. Default is TRUE
weightclust	Logical. To be handed to ReorderToReference: to be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is TRUE.
names	Optional. Names of the methods. Default is NULL.

Details

This function relies on the suggested Bioconductor packages 'MLP', 'biomaRt' and 'org.Hs.eg.db'.

Value

A list with an element per iteration, each being the output of Pathways.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(geneMat)
data(GeneInfo)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)
names=c('FP','TP')
```

```
MCF7_Paths_FandT=PathwaysIter(List=L, geneExpr = geneMat, nrclusters = 7, method =
c("limma", "MLP"), geneInfo = GeneInfo, geneSetSource = "GOBP", topP = NULL,
topG = NULL, GENESET = NULL, sign = 0.05,niter=2,fusionsLog = TRUE,
weightclust = TRUE, names =names)

## End(Not run)
```

PathwaysSelection *Pathway analysis for a selection of objects*

Description

Internal function of Pathways.

Usage

```
PathwaysSelection(
  List = NULL,
  Selection,
  geneExpr = NULL,
  nrclusters = NULL,
  method = c("limma", "MLP"),
  geneInfo = NULL,
  geneSetSource = "GOBP",
  topP = NULL,
  topG = NULL,
  GENESET = NULL,
  sign = 0.05,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL
)
```

Arguments

List	A list of clustering outputs or output of theDiffGenes function. The first element of the list will be used as the reference in ReorderToReference. The output of ChooseFeatures is also accepted.
Selection	If pathway analysis should be conducted for a specific selection of objects, this selection can be provided here. Selection can be of the type "character" (names of the objects) or "numeric" (the number of specific cluster). Default is NULL.
geneExpr	The gene expression matrix or ExpressionSet of the objects. The rows should correspond with the genes.
nrclusters	Optional. The number of clusters to cut the dendrogram in. The number of clusters should not be specified if the interest lies only in a specific selection of objects which is known by name. Otherwise, it is required. Default is NULL.

method	The method to applied to look for differentially expressed genes and related pathways. For now, only the limma method is available for gene analysis and the MLP method for pathway analysis. Default is c("limma","MLP").
geneInfo	A data frame with at least the columns ENTREZID and SYMBOL. This is necessary to connect the symbolic names of the genes with their EntrezID in the correct order. The order of the gene is here not in the order of the rownames of the gene expression matrix but in the order of their significance. Default is NULL.
geneSetSource	The source for the getGeneSets function, defaults to "GOBP".
topP	Overrules sign. The number of pathways to display for each cluster. If not specified, only the significant genes are shown. Default is NULL.
topG	Overrules sign. The number of top genes to be returned in the result. If not specified, only the significant genes are shown. Defaults is NULL.
GENESET	Optional. Can provide own candidate gene sets. Default is NULL.
sign	The significance level to be handled. Default is 0.05.
fusionsLog	Logical. To be handed to ReorderToReference: indicator for the fusion of clusters. Default is TRUE
weightclust	Logical. To be handed to ReorderToReference: to be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is TRUE.
names	Optional. Names of the methods. Default is NULL.

Details

This function relies on the suggested Bioconductor packages 'MLP', 'biomaRt' and 'org.Hs.eg.db'.

Examples

```
## Not run:
PathwaysSelection(List=L,Selection=c("Cpd1","Cpd2"),geneExpr=geneMat,geneInfo=GeneInfo)

## End(Not run)
```

PlotPathways *A GO plot of a pathway analysis output.*

Description

The PlotPathways function takes an output of the PathwayAnalysis function and plots a GO graph with the help of the plotGOgraph function of the MLP package.

Usage

```
PlotPathways(
  Pathways,
  nRow = 5,
  main = NULL,
  plottype = "new",
  location = NULL
)
```

Arguments

Pathways	One element of the output list returned by `PathwayAnalysis` or `Geneset.intersect`.
nRow	Number of GO IDs for which to produce the plot. Default is 5.
main	Title of the plot. Default is NULL.
plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document. Default is "new".
location	If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Value

The output is a GO graph.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(geneMat)
data(GeneInfo)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)
names=c('FP','TP')

MCF7_PathsFandT=PathwayAnalysis(List=L, geneExpr = geneMat, nrclusters = 7, method = c("limma",
"MLP"), geneInfo = GeneInfo, geneSetSource = "GOBP", topP = NULL,
topG = NULL, GENESET = NULL, sign = 0.05,niter=2,fusionsLog = TRUE, weightclust = TRUE,
 names =names,seperatetables=FALSE,separatepvals=FALSE)

PlotPathways(MCF7_PathsFandT$FP$"Cluster 1"$Pathways,nRow=5,main=NULL)

## End(Not run)
```

plot_cluster_profile *Plot one external variable by cluster*

Description

Boxplot (continuous) or grouped bar plot of column proportions (categorical) of an external variable across clusters – the per-cluster distribution and gradient the manuscript should show.

Usage

```
plot_cluster_profile(labels, x, name = deparse(substitute(x)), col = NULL)
```

Arguments

labels	Cluster labels.
x	The variable (numeric -> boxplot; factor/character -> barplot).
name	Axis / title label for the variable.
col	Optional vector of cluster colours.

Value

Invisibly NULL; called for the plot.

See Also

characterize_clusters, cluster_biology_panel

plot_pathways *Modern Gene Ontology / pathway-enrichment plots*

Description

Publication-quality replacements for the classic green GO DAG and the horizontal bar plot. Given the output of PathwayAnalysis() (or a single MLP result table) it draws either a *dot plot* that compares the enriched terms across clusters (the enrichment-map / clusterProfiler::dotplot idiom) or a *lollipop* plot for a single cluster. Dot area encodes the number of genes in the term and colour encodes $-\log_{10}$ p on a perceptually-uniform viridis scale.

Usage

```
plot_pathways(
  pathways,
  style = c("dotplot", "lollipop", "circular"),
  top = 8,
  cluster = NULL,
  source = 1,
  title = NULL,
  palette = "D",
  max_chars = 42,
  wrap = 20,
  cluster_cols = NULL
)
```

Arguments

pathways	Output of PathwayAnalysis() (a list of sources, each a list of per-cluster results carrying a $Pathways table), a single source's per-cluster list, or a single MLP/data.frame result table.
style	"dotplot" to compare clusters side by side, "lollipop" for one cluster, or "circular" for a radial (sunburst-style) bar plot that groups the terms by cluster around a central hub; bar length encodes $-\log_{10}$ p, colour encodes the cluster, and each term is a single-line radial label that stays legible as the number of terms grows.
top	Number of top terms (by p-value) to show per cluster.
cluster	For "lollipop" (or to subset a dot plot): which cluster to show, by name (e.g. "Cluster 1") or index. NULL shows all.
source	Which source (index or name) of a full PathwayAnalysis() object to plot. Default 1.
title	Optional plot title.
palette	viridis option passed to ggplot2::scale_colour_viridis_c() ("D" viridis, "C" plasma, "B" inferno, "G" mako, ...).
max_chars	Term descriptions longer than this are wrapped onto two lines for legibility (dot plot / lollipop).
wrap	For style = "circular", the character width at which long term labels are wrapped onto successive lines, so long names stay compact and legible around the ring. Smaller values wrap more aggressively.
cluster_cols	For style = "circular", an optional vector of fill colours for the clusters (cluster bars and legend). Supply the same palette used elsewhere - e.g. the cols passed to Cyclogram() - so the clusters share one colour scheme across figures. Positional by cluster order, or a vector named by cluster. Default NULL uses the package palette.

Value

A **ggplot** object (invisibly); it is also printed so the function works directly on an open graphics device.

See Also

PathwayAnalysis(), PlotPathways()

Examples

```
## Not run:
P <- PathwayAnalysis(List = L, geneExpr = ge, nrclusters = 3,
                     method = c("limma", "MLP"), geneInfo = geneInfo,
                     geneSetSource = "GOBP", sign = 0.05, niter = 2)
plot_pathways(P, style = "dotplot", top = 8)          # compare clusters
plot_pathways(P, style = "lollipop", cluster = 1)     # one cluster

## End(Not run)
```

PreparePathway *Prepare data for pathway analysis*

Description

The PreparePathway function prepares the input for the pathway analysis by computing differentially expressed genes (via limma) for a selection of objects if these were not provided.

Usage

```
PreparePathway(Object, geneExpr, topG, sign)
```

Arguments

Object	A clustering output or DiffGenes output element.
geneExpr	The gene expression matrix or ExpressionSet of the objects.
topG	Overrules sign. The number of top genes. Default is NULL.
sign	The significance level to be handled. Default is 0.05.

Details

This function may rely on the suggested Bioconductor package 'limma'.

Value

A list with the p-values of the genes, the objects and the genes.

Examples

```
## Not run:
PreparePathway(Object=MCF7_F,geneExpr=geneMat,topG=NULL,sign=0.05)

## End(Not run)
```

`ProfilePlot` *Plotting gene profiles*

Description

In `ProfilePlot`, the gene profiles of the significant genes for a specific cluster are shown on 1 plot. Therefore, each gene is normalized by subtracting its the mean.

Usage

```
ProfilePlot(
  Genes,
  Comps,
  geneExpr = NULL,
  raw = FALSE,
  orderLab = NULL,
  colorLab = NULL,
  nrclusters = NULL,
  cols = NULL,
  addLegend = TRUE,
  margins = c(8.1, 4.1, 1.1, 6.5),
  extra = 5,
  plottype = "new",
  location = NULL
)
```

Arguments

`Genes`	The genes to be plotted.
`Comps`	The objects to be plotted or to be separated from the other objects.
`geneExpr`	The gene expression matrix or ExpressionSet of the objects.
`raw`	Logical. Should raw p-values be plotted? Default is FALSE.
`orderLab`	Optional. If the objects are to set in a specific order of a specific method. Default is NULL.
`colorLab`	The clustering result that determines the color of the labels of the objects in the plot. Default is NULL.
`nrclusters`	Optional. The number of clusters to cut the dendrogram in.
`cols`	Optional. The color to use for the objects in Clusters for each method.
`addLegend`	Optional. Whether a legend of the colors should be added to the plot.
`margins`	Optional. Margins to be used for the plot. Default is margins=c(8.1,4.1,1.1,6.5).
`extra`	The space between the plot and the legend. Default is 5.

plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document, i.e. no new device is opened and the plot appears in the current device or document. Default is "new".
location	If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Value

A plot which contains multiple gene profiles. A distinction is made between the values for the objects in Comps and the others.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)
data(geneMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
 method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
 method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)
names=c('FP','TP')

MCF7_FT_DE = DiffGenes(List=L,geneExpr=geneMat,nrclusters=7,method="limma",sign=0.05,topG=10,
fusionsLog=TRUE,weightclust=TRUE)

Comps=SharedComps(list(MCF7_FT_DE$`Method 1`$"Cluster 1",MCF7_FT_DE$`Method 2`$"Cluster 1"))[[1]]

MCF7_SharedGenes=FindGenes(dataLimma=MCF7_FT_DE,names=c("FP","TP"))

Genes=names(MCF7_SharedGenes[[1]])[-c(2,4,5)]

colscl=ColorPalette(colors=c("red","green","purple","brown","blue","orange"),ncols=9)

ProfilePlot(Genes=Genes,Comps=Comps,geneExpr=geneMat,raw=FALSE,orderLab=MCF7_F,
colorLab=NULL,nrclusters=7,cols=colscl,addLegend=TRUE,margins=c(16.1,6.1,1.1,13.5),
extra=4,plottype="sweave",location=NULL)

## End(Not run)
```

ReorderToReference *Relabel and reorder clusterings to a reference*

Description

ReorderToReference relabels and reorders the clusters of a series of clustering results so that the cluster numbers of every method are matched, via a Gale-Shapley stable-matching scheme, to those of the first (reference) method. The result is a matrix in which the columns represent the objects (ordered as for the reference method) and the rows represent the methods. Each cell contains the number of the cluster the object belongs to for that method, relabelled to the reference. It is a possibility that two or more clusters are fused together compared to the reference; in that case the function alerts the user and asks to set fusionsLog to TRUE.

Usage

```
ReorderToReference(
  List,
  nrclusters = NULL,
  fusionsLog = FALSE,
  weightclust = FALSE,
  names = NULL
)
```

Arguments

List	A list of clustering outputs to be compared. The first element of the list will be used as the reference.
nrclusters	The number of clusters to cut the dendrogram in. Default is NULL.
fusionsLog	Logical. Indicator for the fusion of clusters. Default is FALSE.
weightclust	Logical. To be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is FALSE.
names	Optional. A character vector with the names of the methods.

Value

A matrix of which the cells indicate to what cluster the objects belong to according to the rearranged methods.

Note

The ReorderToReference function was optimized for the situations presented by the data sets at hand. It is noted that the function might fail in a particular situation which results in an infinite loop.

References

Van Moerbeke, M. (2017). MoSaiClusteR: integrative clustering of multi-source data. Hasselt University.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
res <- list(Cluster(L[[1]], type = "data", distmeasure = "euclidean"),
            Cluster(L[[2]], type = "data", distmeasure = "euclidean"))
M <- ReorderToReference(res, nrclusters = 5, fusionsLog = FALSE,
                        weightclust = FALSE, names = c("S1", "S2"))

## End(Not run)
```

`SelectnrClusters` *Select the number of clusters via silhouette widths*

Description

For each element of `List`, partitions around medoids (`cluster::pam`) are computed over a range of cluster numbers and the average silhouette width is recorded. The number of clusters maximising the average silhouette width is reported per source and for the across-source average. The analytical result (the silhouette table and the chosen number of clusters) is returned and is testable without producing any plot.

Usage

```
SelectnrClusters(
  List,
  type = c("data", "dist", "pam"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  nrclusters = seq(5, 25, 1),
  names = NULL,
  StopRange = FALSE,
  plottype = "none",
  location = NULL
)
```

Arguments

`List`	A list of data matrices (objects in rows), distance matrices, or `cluster::pam` outputs, depending on `type`.
`type`	One of `"data"` (compute a distance first via `Distance`), `"dist"` (elements are dissimilarity matrices) or `"pam"` (elements are `pam` objects).
`distmeasure`	Distance measure(s) used when `type = "data"`.
`normalize, method`	Normalisation controls passed to `Distance`.

nrclusters	Sequence of cluster numbers to evaluate.
names	Optional names for the sources; used in output labels.
StopRange	Logical; if FALSE and dissimilarities fall outside [0, 1] they are range-normalised for comparability.
plottype	One of "none" (default; no plot, safe for headless runs), "new", "pdf" or "sweave". Any value other than "none" attempts to draw the silhouette curve; drawing is guarded so that a missing graphics device does not abort the computation.
location	File location (without extension) used when plottype = "pdf". Default NULL.

Value

A list with two elements: Silhoutte_Widths (a matrix of average silhouette widths per source and their mean) and Optimal_Nr_of_CLusters (a one-row data frame with the silhouette optimal number of clusters per source and overall).

References

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

Examples

```
## Not run:
data(mosaic_toy)
sel <- SelectnrClusters(mosaic_toy$List, type = "data",
                        distmeasure = c("euclidean", "euclidean"),
                        nrclusters = seq(2, 5), plottype = "none")
sel$Optimal_Nr_of_CLusters

## End(Not run)
```

SharedComps *Objects shared across clusterings*

Description

SharedComps determines, per cluster, which objects are shared across all listed clustering results. The clusterings are first rearranged to a common reference with ReorderToReference and the intersection of the lead objects of every method is taken per cluster.

Usage

```
SharedComps(
  List,
  nrclusters = NULL,
  fusionsLog = FALSE,
  weightclust = FALSE,
  names = NULL
)
```

Arguments

List	A list of the clustering outputs to be compared. The first element of the list will be used as the reference in ReorderToReference.
nrclusters	If List is the output of several clustering methods, it has to be provided in how many clusters to cut the dendrograms in. Default is NULL.
fusionsLog	Logical. To be handed to ReorderToReference: indicator for the fusion of clusters. Default is FALSE.
weightclust	Logical. To be handed to ReorderToReference: to be used for the outputs of CEC, WeightedClust or WeightedSimClust. If TRUE, only the result of the Clust element is considered. Default is FALSE.
names	Names of the methods or clusters. Default is NULL.

Value

A list containing the shared objects of all listed elements per cluster.

Examples

```
## Not run:
data(mosaic_toy)
L <- mosaic_toy$List
res <- list(Cluster(L[[1]], type = "data", distmeasure = "euclidean"),
            Cluster(L[[2]], type = "data", distmeasure = "euclidean"))
Comps <- SharedComps(List = res, nrclusters = 5, fusionsLog = FALSE,
                     weightclust = FALSE, names = c("S1", "S2"))

## End(Not run)
```

SharedGenesPathsFeat *Shared genes, pathways and features across methods*

Description

The SharedGenesPathsFeat function takes the intersection of the differentially expressed genes, the pathways and the characteristic features over the methods per cluster.

Usage

```
SharedGenesPathsFeat(
  DataLimma = NULL,
  DataMLP = NULL,
  DataFeat = NULL,
  names = NULL,
  Selection = FALSE
)
```

Arguments

DataLimma	Optional. The output of a DiffGenes function. Default is NULL.
DataMLP	Optional. The output of Geneset.intersect function. Default is NULL.
DataFeat	Optional. The output of CharacteristicFeatures function. Default is NULL.
names	Optional. Names of the methods. Default is NULL.
Selection	Logical. Whether a selection of objects was considered. Default is FALSE.

Value

A list with a Table summarizing the shared elements and a Which element detailing the shared genes, pathways and features per cluster.

Examples

```
## Not run:
SharedGenesPathsFeat(DataLimma=MCF7_FT_DE,DataMLP=MCF7_Paths_intersection,names=c("FP","TP"))

## End(Not run)
```

SharedSelection *Intersection of genes and pathways over multiple methods for a selection of objects.*

Description

Internal function of SharedGenesPathsFeat.

Usage

```
SharedSelection(
  DataLimma = NULL,
  DataMLP = NULL,
  DataFeat = NULL,
  names = NULL
)
```

Arguments

DataLimma	Optional. The output of a DiffGenes function. Default is NULL.
DataMLP	Optional. The output of Geneset.intersect function. Default is NULL.
DataFeat	Optional. The output of CharacteristicFeatures function. Default is NULL.
names	Optional. Names of the methods or "Selection" if it only considers a selection of objects. Default is NULL.

Value

A list with a Table and a Which element for the selection.

Examples

```
## Not run:
SharedSelection(DataLimma=MCF7_FT_DE,names="Selection")

## End(Not run)
```

SharedSelectionLimma *Intersection of genes over multiple methods for a selection of objects.*

Description

Internal function of SharedGenesPathsFeat.

Usage

```
SharedSelectionLimma(DataLimma = NULL, names = NULL)
```

Arguments

DataLimma	Optional. The output of a DiffGenes function. Default is NULL.
names	Optional. Names of the methods or "Selection" if it only considers a selection of objects. Default is NULL.

Value

A list with a Table and a Which element for the shared genes.

Examples

```
## Not run:
SharedSelectionLimma(DataLimma=MCF7_FT_DE)

## End(Not run)
```

SharedSelectionMLP *Intersection of pathways over multiple methods for a selection of objects.*

Description

Internal function of SharedGenesPathsFeat.

Usage

```
SharedSelectionMLP(DataMLP = NULL, names = NULL)
```

Arguments

DataMLP	Optional. The output of Geneset.intersect function. Default is NULL.
names	Optional. Names of the methods or "Selection" if it only considers a selection of objects. Default is NULL.

Value

A list with a Table and a Which element for the shared pathways.

Examples

```
## Not run:
SharedSelectionMLP(DataMLP=MCF7_Paths_intersection)

## End(Not run)
```

SimilarityHeatmap *A heatmap of similarity values between objects*

Description

The function SimilarityHeatmap plots the similarity values between objects. The darker the shade, the more similar objects are. The option is available to set a cutoff value to highlight the most similar objects.

Usage

```
SimilarityHeatmap(
  Data,
  type = c("data", "clust", "sim", "dist"),
  distmeasure = "tanimoto",
  normalize = FALSE,
  method = NULL,
  linkage = "flexible",
  cutoff = NULL,
  percentile = FALSE,
  plottype = "new",
  location = NULL
)
```

Arguments

Data	The data of which a heatmap should be drawn.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clusters" the single source clustering is skipped. Type should be one of "data", "dist" ,"sim" or "clusters".
distmeasure	The distance measure. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to "tanimoto".
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is NULL.
linkage	Choice of inter group dissimilarity (character). Defaults to "flexible".
cutoff	Optional. If a cutoff value is specified, all values lower are put to zero while all other values are kept. This helps to highlight the most similar objects. Default is NULL.
percentile	Logical. The cutoff value can be a percentile. If one want the cutoff value to be the 90th percentile of the data, one should specify cutoff = 0.90 and percentile = TRUE. Default is FALSE.
plottype	Should be one of "pdf","new" or "sweave". If "pdf", a location should be provided in "location" and the figure is saved there. If "new" a new graphic device is opened and if "sweave", the figure is made compatible to appear in a sweave or knitr document, i.e. no new device is opened and the plot appears in the current device or document. Default is "new".
location	If plottype is "pdf", a location should be provided in "location" and the figure is saved there. Default is NULL.

Details

If data is of type "clust", the distance matrix is extracted from the result and transformed to a similarity matrix. Possibly a range normalization is performed. If data is of type "dist", it is also transformed to a similarity matrix and cluster is performed on the distances. If data is of type "sim", the data is tranformed to a distance matrix on which clustering is performed. Once the similarity mattrix is obtained, the cutoff value is applied and a heatmap is drawn. If no cutoff value is desired, one can leave the default NULL specification.

Value

A heatmap with the names of the objects on the right and bottom and a dendrogram of the clustering at the left and top.

Examples

```
## Not run:
data(fingerprintMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55)

SimilarityHeatmap(Data=MCF7_F,type="clust",cutoff=0.90,percentile=TRUE)
SimilarityHeatmap(Data=MCF7_F,type="clust",cutoff=0.75,percentile=FALSE)


## End(Not run)
```

SimilarityMeasure *Similarity of a set of clusterings to a reference*

Description

Given a colour/label matrix (clusterings in rows, objects in columns) or a list of clustering outputs, the proportion of objects sharing the label of the first (reference) row is computed for every row. When a list is supplied it is first aligned with `ReorderToReference`; supplying the label matrix directly avoids that dependency.

Usage

```
SimilarityMeasure(
  List,
  nrclusters = NULL,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL
)
```

Arguments

List	A label matrix (rows are clusterings, columns are objects) or a list of clustering outputs to be aligned via ReorderToReference.
nrclusters	Number of clusters used when List is a list. Default NULL.
fusionsLog, weightclust	Logical controls passed to ReorderToReference when List is a list.
names	Optional method names passed to ReorderToReference.

Value

A numeric vector of similarities in [0, 1], one per row, where the first entry is the reference compared to itself (always 1).

References

Fodeh, S. J., Brandt, C., Luong, T. B., Haddad, A., Schultz, M., Murphy, T. and Krauthammer, M. (2013). Complementary ensemble clustering of biomedical data. Journal of Biomedical Informatics, 46(3), 436-443.

Examples

```
## Not run:
M <- rbind(c(1, 1, 2, 2, 3, 3),
           c(1, 1, 2, 3, 3, 3),
           c(2, 2, 1, 1, 3, 3))
SimilarityMeasure(M)

## End(Not run)
```

---

simulate_weighting_regime

Simulate data for evaluating feature-weighting schemes (variance vs CV)

Description

Generates a multi-source data set with a known cluster structure in which the discriminative signal is carried either by high-absolute-variance features (regime = "variance") or by high-*coefficient-of-variation* features buried under high-abundance, high-variance decoys (regime = "cv"). It is intended for benchmarking the "var", "cv" and "nugget" weightings of the ABC family (and any other feature-weighting method) on the same footing.

Usage

```
simulate_weighting_regime(
  regime = c("cv", "variance"),
  n_samples = 120,
  n_clusters = 3,
  n_sources = 2,
  n_signal = 40,
  n_decoy = 500,
  n_noise = 60,
  effect = NULL,
  seed = NULL
)
```

Arguments

`regime`	`"cv"` (default) or `"variance"`; see Details.
`n_samples`	Number of objects (rows), split evenly across clusters.
`n_clusters`	Number of clusters.
`n_sources`	Number of data sources (independent draws of the same design).
`n_signal`, `n_decoy`, `n_noise`	Numbers of discriminative, decoy (non-discriminative but high-variance in the CV regime) and pure-noise features per source.
`effect`	Between-cluster mean shift on the signal features (defaults: 1.4 for the variance regime, 0.9 for the cv regime).
`seed`	Optional RNG seed.

Value

A list with `List` (the sources, objects in rows), `truth` (cluster labels), `signal_features` (indices of the discriminative features), `regime` and `params`.

See Also

`M_ABC` (the `weighting` argument), `mosaic_sim`

Examples

```
cv  <- simulate_weighting_regime("cv", seed = 1)
# variance weighting is misled by high-abundance decoys, CV recovers the truth:
fit_var <- M_ABC(cv$List, numsim = 80, NC = 3, weighting = c("var", "var"))
fit_cv  <- M_ABC(cv$List, numsim = 80, NC = 3, weighting = c("cv",  "cv"))
c(var = cluster_agreement(mosaic_labels(fit_var, 3), cv$truth)[["ARI"]],
  cv  = cluster_agreement(mosaic_labels(fit_cv,  3), cv$truth)[["ARI"]])
```

SNF *Similarity network fusion*

Description

Similarity Network Fusion (SNF) is a similarity-based multi-source clustering technique. SNF consists of two steps. In the initial step a similarity network is set up for each data matrix. The network is the visualization of the similarity matrix as a weighted graph with the objects as vertices and the pairwise similarities as weights on the edges. In the network-fusion step, each network is iteratively updated with information of the other network which results in more alike networks every time. This eventually converges to a single network.

Usage

```
SNF(
  List,
  type = c("data", "dist", "clusters"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  StopRange = FALSE,
  NN = 20,
  mu = 0.5,
  T = 20,
  clust = "agnes",
  linkage = "ward",
  alpha = 0.625
)
```

Arguments

List	A list of data matrices of the same type. It is assumed the rows are corresponding with the objects.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clusters" the single source clustering is skipped. Type should be one of "data", "dist" or "clusters".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.

StopRange	Logical. Indicates whether the distance matrices with values not between zero and one should be standardized to have so. If FALSE the range normalization is performed. See Normalization. If TRUE, the distance matrices are not changed. This is recommended if different types of data are used such that these are comparable. Default is FALSE.
NN	The number of neighbours to be used in the procedure. Defaults to 20.
mu	The parameter epsilon. The value is recommended to be between 0.3 and 0.8. Defaults to 0.5.
T	The number of iterations.
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for the final clustering. Defaults to "ward".
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible"

Details

If r is specified and nrclusters is a fixed number, only a random sampling of the features will be performed for the t iterations (ADECa). If r is NULL and the nrclusters is a sequence, the clustering is performedon all features and the dendrogam is divided into clusters for the values of nrclusters (ADECb). If both r is specified and nrclusters is a sequence, the combination is performed (ADECc). After every iteration, either be random sampling, multiple divisions of the dendrogram or both, an incidence matrix is set up. All incidence matrices are summed and represent the distance matrix on which a final clustering is performed.

Value

The returned value is a list with the following three elements.

FusedM	The fused similarity matrix
DistM	The distance matrix computed by subtracting FusedM from one
Clust	The resulting clustering

The value has class 'SNF'.

References

Wang B, Mezlini AM, Demir F, Fiume M, Tu Z, Brudno M, Haibe-Kains B, Goldenberg A (2014). "Similarity Network Fusion for Aggregating Data Types on a Genomic Scale." *Nature Methods*, **11**, 333–337.

`spectral_clustering` *Spectral clustering of an affinity matrix*

Description

Normalised (Ng-Jordan-Weiss) spectral clustering. Given a symmetric, non-negative affinity matrix it forms the normalised Laplacian, embeds the objects in the space of its leading eigenvectors and applies k-means.

Usage

```
spectral_clustering(affinity, k = NULL, max_k = 10, nstart = 10)
```

Arguments

`affinity`	A symmetric $n \times n$ non-negative similarity/affinity matrix over the objects.
`k`	Number of clusters. If `NULL`, chosen by the largest eigengap of the normalised Laplacian (searched over `2:max_k`).
`max_k`	Upper bound for the eigengap search when `k = NULL`.
`nstart`	Number of k-means restarts.

Value

A list with `cluster` (named integer labels), `k` and the spectral `embedding`.

References

Ng, A. Y., Jordan, M. I. & Weiss, Y. (2002). On spectral clustering: analysis and an algorithm. NIPS 14.

See Also

NEMO, SNF

Examples

```
set.seed(1)
X <- rbind(matrix(rnorm(20 * 5, 0), 20), matrix(rnorm(20 * 5, 6), 20))
A <- exp(-as.matrix(dist(X))^2 / 2)
spectral_clustering(A, k = 2)$cluster[1:5]
```

`TrackCluster` *Track a cluster or a selection of objects across multiple methods*

Description

`TrackCluster` follows a selection of objects or a specific cluster across a sequence of clustering results, recording how the objects are distributed over the clusters of each method and optionally producing tracking plots.

Usage

```
TrackCluster(
  List,
  Selection,
  nrclusters = NULL,
  followMaxComps = FALSE,
  followClust = TRUE,
  fusionsLog = TRUE,
  weightclust = TRUE,
  names = NULL,
  selectionPlot = FALSE,
  table = FALSE,
  completeSelectionPlot = FALSE,
  ClusterPlot = FALSE,
  cols = NULL,
  legendposx = 0.5,
  legendposy = 2.4,
  plottype = "sweave",
  location = NULL
)
```

Arguments

`List`	A list of clustering outputs to be compared. The first element of the list is used as the reference in `ReorderToReference`.
`Selection`	Either a character vector of objects, or a numeric cluster number.
`nrclusters`	The number of clusters to cut the dendrogram in. Default is NULL.
`followMaxComps`	Logical. Whether to follow the cluster with the maximum number of objects. Default is FALSE.
`followClust`	Logical. Whether to follow the cluster of interest. Default is TRUE.
`fusionsLog`	Logical. Indicator for the fusion of clusters. Default is TRUE.
`weightclust`	Logical. For the outputs of CEC, WeightedClust or WeightedSimClust, if TRUE only the result of the Clust element is considered. Default is TRUE.
`names`	Optional. Names of the methods. Default is NULL.
`selectionPlot`	Logical. Whether to produce the selection plot. Default is FALSE.

table	Logical. Whether to produce a table of the tracking. Default is FALSE.
completeSelectionPlot	Logical. Whether to produce the complete selection plot. Default is FALSE.
ClusterPlot	Logical. Whether to produce the cluster plot. Default is FALSE.
cols	The colors to use in the plots. Default is NULL.
legendposx	The x-coordinate of the legend. Default is 0.5.
legendposy	The y-coordinate of the legend. Default is 2.4.
plottype	Should be one of "pdf","new" or "sweave". Default is "sweave".
location	If plottype is "pdf", a location should be provided. Default is NULL.

Value

A list with one element per method describing how the selection is distributed across the clusters, and optionally a tracking table.

Examples

```
## Not run:
data(fingerprintMat)
data(targetMat)

MCF7_F = Cluster(fingerprintMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)
MCF7_T = Cluster(targetMat,type="data",distmeasure="tanimoto",normalize=FALSE,
method=NULL,clust="agnes",linkage="flexible",gap=FALSE,maxK=55,StopRange=FALSE)

L=list(MCF7_F,MCF7_T)
MCF7_Track=TrackCluster(List=L,Selection=4,nrclusters=7,followMaxComps=FALSE,
followClust=TRUE,selectionPlot=TRUE,table=TRUE,cols=NULL)

## End(Not run)
```

WeightedClust *Weighted clustering*

Description

Weighted Clustering (Weighted) is a similarity-based multi-source clustering technique. Weighted clustering is performed with the function `WeightedClust`. Given a list of the data matrices, a dissimilarity matrix is computed of each with the provided distance measures. These matrices are then combined resulting in a weighted dissimilarity matrix. Hierarchical clustering is performed on this weighted combination with the agnes function and the ward link

Usage

```
WeightedClust(
  List,
  type = c("data", "dist", "clusters"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  StopRange = FALSE,
  weight = seq(1, 0, -0.1),
  weightclust = 0.5,
  clust = "agnes",
  linkage = "ward",
  alpha = 0.625
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	indicates whether the provided matrices in "List" are either data matrices, distance matrices or clustering results obtained from the data. If type="dist" the calculation of the distance matrices is skipped and if type="clusters" the single source clustering is skipped. Type should be one of "data", "dist" or "clusters".
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. This is recommended if different distance types are used. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.
StopRange	Logical. Indicates whether the distance matrices with values not between zero and one should be standardized to have values between zero and one. If FALSE the range normalization is performed. See Normalization. If TRUE, the distance matrices are not changed. This is recommended if different types of data are used such that these are comparable. Default is FALSE.
weight	Optional. A list of different weight combinations for the data sets in List. If NULL, the weights are determined to be equal for each data set. It is further possible to fix weights for some data matrices and to let it vary randomly for the remaining data sets. Defaults to seq(1,0,-0.1). An example is provided in the details.
weightclust	A weight for which the result will be put aside of the other results. This was done for comparative reason and easy access. Defaults to 0.5 (two data sets)
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for the final clustering. Defaults to "ward".

alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".

Details

The weight combinations should be provided as elements in a list. For three data matrices an example could be: weights=list(c(0.5,0.2,0.3),c(0.1,0.5,0.4)). To provide a fixed weight for some data sets and let it vary randomly for others, the element "x" indicates a free parameter. An example is weights=list(c(0.7,"x","x")). The weight 0.7 is now fixed for the first data matrix while the remaining 0.3 weight will be divided over the other two data sets. This implies that every combination of the sequence from 0 to 0.3 with steps of 0.1 will be reported and clustering will be performed for each.

Value

The returned value is a list of four elements:

DistM	A list with the distance matrix for each data structure
WeightedDist	A list with the weighted distance matrices
Results	The hierarchical clustering result for each element in WeightedDist
Clust	The result for the weight specified in Clustweight

The value has class 'Weighted'.

References

Perualila-Tan N, Kasim A, Talloen W, Göhlmann H, Shkedy Z (2016). "Weighted Similarity-Based Clustering of Chemical Structures and Bioactivity Data in Early Drug Discovery." *Journal of Bioinformatics and Computational Biology*, **14**, 1650018.

Whclust *Weighted hierarchical clustering (weighted Ward)*

Description

Agglomerative clustering of weighted observations using the weighted Ward merge cost. Intended for the small set of data-nugget centres.

Usage

```
Whclust(x, weights = rep(1, nrow(x)))
```

Arguments

x	Numeric matrix, observations (e.g. nugget centres) in rows.
weights	Non-negative weight per observation.

Value

An hclust object.

References

Dey et al. (2025). WCluster.

See Also

Wkmeans, nugget_cluster

Examples

```
set.seed(1)
x <- rbind(matrix(rnorm(8 * 3, 0), 8), matrix(rnorm(8 * 3, 5), 8))
stats::cutree(Whclust(x, weights = rep(2, 16)), k = 2)[1:4]
```

Wkmeans *Weighted k-means (minimises WWCSS)*

Description

k-means in which each observation carries a non-negative weight. Cluster centres are weighted means and the objective is the Weighted Within-Cluster Sum of Squares $\sum_c \sum_{i \in c} w_i \|x_i - \mu_c\|^2$.

Usage

```
Wkmeans(x, k, weights = rep(1, nrow(x)), nstart = 10, max_iter = 100)
```

Arguments

x	Numeric matrix, observations in rows.
k	Number of clusters.
weights	Non-negative weight per observation (default all 1).
nstart	Number of random restarts (best WWCSS kept).
max_iter	Maximum Lloyd iterations per start.

Value

A list with cluster, centers and wwcss.

References

Dey, T., Duan, Y., Cabrera, J. & Cheng, X. (2025). WCluster: Clustering and PCA with Weights, and Data Nuggets Clustering.

See Also

nugget_cluster, Whclust

Examples

```
set.seed(1)
x <- rbind(matrix(rnorm(20 * 3, 0), 20), matrix(rnorm(20 * 3, 5), 20))
Wkmeans(x, k = 2, weights = runif(40, 1, 5))$cluster[1:5]
```

WonM *Weighting on Membership clustering*

Description

Weighting on Membership (WonM) is a hierarchy-based consensus technique. Each data matrix is hierarchically clustered and the resulting dendrogram is cut into a range of K values. For every cut a co-membership matrix is built, recording which objects fall in the same cluster. These matrices are averaged within and across data sets to produce an overall consensus matrix, which is transformed into a dissimilarity and clustered a final time.

Usage

```
WonM(
  List,
  type = c("data", "dist", "clusters"),
  distmeasure = c("tanimoto", "tanimoto"),
  normalize = c(FALSE, FALSE),
  method = c(NULL, NULL),
  nrclusters = list(seq(5, 25), seq(5, 25)),
  clust = "agnes",
  linkage = c("flexible", "flexible"),
  alpha = 0.625
)
```

Arguments

List	A list of data matrices. It is assumed the rows are corresponding with the objects.
type	Indicates whether the provided matrices in "List" are either data matrices ("data"), distance matrices ("dist") or clustering results obtained from the data ("clusters").
distmeasure	A vector of the distance measures to be used on each data matrix. Should be one of "tanimoto", "euclidean", "jaccard", "hamming". Defaults to c("tanimoto","tanimoto").
normalize	Logical. Indicates whether to normalize the distance matrices or not, defaults to c(FALSE, FALSE) for two data sets. More details on normalization in Normalization.
method	A method of normalization. Should be one of "Quantile","Fisher-Yates", "standardize","Range" or any of the first letters of these names. Default is c(NULL,NULL) for two data sets.

nrclusters	A list with, for each data matrix, the sequence of numbers of clusters to cut the dendrogram in. Defaults to list(seq(5,25), seq(5,25)).
clust	Choice of clustering function (character). Defaults to "agnes".
linkage	Choice of inter group dissimilarity (character) for each data set. Defaults to c("flexible", "flexible") for two data sets.
alpha	The parameter alpha to be used in the "flexible" linkage of the agnes function. Defaults to 0.625 and is only used if the linkage is set to "flexible".

Value

The returned value is a list of two elements:

DistM	The resulting consensus dissimilarity matrix
Clust	The resulting clustering

The value has class 'WonM'.

References

Saeed F., Salim N. and Abdo A. (2012). Voting-based consensus clustering for combining multiple clusterings of chemical structures. Journal of Cheminformatics, 4, 37.

Examples

```
## Not run:
data(mosaic_toy)
fit <- WonM(List = mosaic_toy$List, type = "data",
  distmeasure = c("euclidean", "euclidean"), normalize = c(FALSE, FALSE),
  method = c(NULL, NULL), nrclusters = list(seq(3, 6), seq(3, 6)),
  clust = "agnes", linkage = c("ward", "ward"))

## End(Not run)
```

Index

∗ **datasets**
    mosaic_toy, 92

ABCdeep.SingleInMultiple, 4, *95, 96*
ABCdist.SingleInMultiple, 6, *96*, *98*
ABCpp.SingleInMultiple, *4–7*, 8, *93, 94*, *97*, *99*
ADC, 10
ADEC, 11

BinFeaturesPlot_MultipleData, 13
BinFeaturesPlot_SingleData, 15
BoxPlotDistance, 17

CEC, 19
CharacteristicFeatures, 21
CharacteristicFeatures(), *65, 66*
characterize_clusters, 22, *32*, *113*
ChooseCluster, 23
Cluster, 25
cluster_agreement, *29*, 31
cluster_biology_panel, *23*, 32, *113*
cluster_k_sweep, 32
cluster_selection_plot, *33*, 34, *64*
ClusterCols, 26
clustering_concordance, 29, *44*
ClusteringAggregation, 26
ClusterPlot, 29
ColorPalette, 35
ColorsNames, 35
compare_clusterings, *29*, 42
compare_methods_plot, 43
CompareInteractive, 36
ComparePlot, 38, *43, 44*
CompareSilCluster, 39
CompareSvsM, 41
ConsensusClustering, 44
ContFeaturesPlot, 46
create_data_nuggets, *9*, 48, *94*, *102, 103*
CVAA, 49

Cyclogram, 51
Cyclogram(), *114*

DetermineWeight_SilClust, 53
DetermineWeight_SimClust, 55
DiffGenes, 57
DiffGenesSelection, 58
Distance, *7, 8*, *25*, 60, *119*
Distance_v2 *(*Distance*)*, 60
distanceheatmaps, 61

EHC, 61
embedding_plot, *33, 34*, 64
enriched_features, 65
EnsembleClustering, 66
EvidenceAccumulation, 68

f.clustABC.MultiSource, *4*, *6*, 70, *93–96*
FeatSelection, 71
FeaturesOfCluster, 72
FindCluster, 73
FindElement, 74
FindGenes, 75

Geneset.intersect, 76
Geneset.intersectSelection, 77
ggplot2::scale_colour_viridis_c(), *114*

HBGF, 78
hcl.colors, *33*
hclust, *136*
HeatmapPlot, 80
HeatmapSelection, 81
HierarchicalEnsembleClustering, *25*, 83

intNMF, 85, *90*

LabelCols, 86
LabelPlot, 86
LinkBasedClustering, 87
LUCID, 89

M_ABC, *4*, *8, 9*, *25*, *49*, *71*, *85*, *90, 91*, 93, *95, 96*, *98*, *102, 103*, *128*
M_ABCdeep, *6*, 95
M_ABCdist, *6, 7*, 96, *99*
M_ABCdist.WC, *98*, 98
mosaic_labels, 91
mosaic_sim, 91, *92*, *128*
mosaic_toy, 92

NEMO, 100, *131*
Normalization, *5*, *7, 8*, *60*, 101
nugget_cluster, 102, *136, 137*
nugget_feature_weights, *49*, 103

PathwayAnalysis, 104
PathwayAnalysis(), *113–115*
Pathways, 105
PathwaysIter, 108
PathwaysSelection, 110
plot_cluster_profile, *23*, *32*, 113
plot_pathways, 113
plot_pathways(), *65, 66*
PlotPathways, 111
PlotPathways(), *115*
PreparePathway, 115
ProfilePlot, 116

ReorderToReference, 117

scale, *101*
SelectnrClusters, *34*, 119
SharedComps, 120
SharedGenesPathsFeat, 121
SharedSelection, 122
SharedSelectionLimma, 123
SharedSelectionMLP, 124
SimilarityHeatmap, 124
SimilarityMeasure, 126
simulate_weighting_regime, 127
SNF, *85*, *100*, 129, *131*
spectral_clustering, *100*, 131

TrackCluster, 132

WeightedClust, *98, 99*, 133
Whclust, 135, *137*
Wkmeans, *102*, *136*, 136
WonM, 137