Package 'pagoda2'

April 1, 2026

Title Single Cell Analysis and Differential Expression

Version 1.0.15

Description
Analyzing and interactively exploring large-scale single-cell RNA-seq datasets. 'pagoda2' primarily performs normalization and differential gene expression analysis, with an interactive application for exploring single-cell RNA-seq datasets. It performs basic tasks such as cell size normalization, gene variance normalization, and can be used to identify subpopulations and run differential expression within individual samples. 'pagoda2' was written to rapidly process modern large-scale scRNAseq datasets of approximately 1e6 cells. The companion web application allows users to explore which gene expression patterns form the different subpopulations within your data. The package also serves as the primary method for preprocessing data for conos, <https://github.com/kharchenkolab/conos>. This package interacts with data available through the 'p2data' package, which is available in a 'drat' repository. To access this data package, see the instructions at <https://github.com/kharchenkolab/pagoda2>. The size of the 'p2data' package is approximately 6 MB.

License GPL-3

Copyright See the file COPYRIGHTS for various pagoda2 copyright details

Encoding UTF-8

Depends R (>= 3.5.0), Matrix, igraph

Imports dendsort, drat, fastcluster, graphics, grDevices, irlba, magrittr, MASS, mgcv, methods, N2R, parallel, plyr, R.utils, Rcpp, rjson, rlang, R6, RMTstat, Rook, Rtsne, sccore (>= 0.1.1), stats, urltools, utils

RoxygenNote 7.3.2

Suggests AnnotationDbi, base64enc, BiocGenerics, BiocParallel, colorRamps, data.table, dbscan, dplyr, ggplot2, GO.db, gridExtra, KernSmooth, knitr, org.Dr.eg.db, org.Hs.eg.db, org.Mm.eg.db, pcaMethods, pheatmap, rgl, rmarkdown, robustbase, scde, testthat, uwot

URL https://github.com/kharchenkolab/pagoda2

BugReports https://github.com/kharchenkolab/pagoda2/issues

NeedsCompilation yes

LinkingTo Rcpp, RcppArmadillo, RcppProgress, RcppEigen

Author Nikolas Barkas [aut],
Viktor Petukhov [aut],
Peter Kharchenko [aut],
Simon Steiger [ctb],
Rasmus Rydbirk [ctb],
Evan Biederstedt [cre, aut]

Maintainer Evan Biederstedt <evan.biederstedt@gmail.com>

Repository CRAN

Date/Publication 2026-03-31 22:40:06 UTC

Contents

`armaCor` *armaCor - matrix column correlations. Allows faster matrix correlations with armadillo. Similar to cor() call, will calculate correlation between matrix columns*

Description

armaCor - matrix column correlations. Allows faster matrix correlations with armadillo. Similar to cor() call, will calculate correlation between matrix columns

Usage

```
armaCor(mat)
```

Arguments

`mat`	matrix

Value

matrix with columns as correlations

`basicP2proc` *Perform basic 'pagoda2' processing, i.e. adjust variance, calculate pca reduction, make knn graph, identify clusters with multilevel, and generate largeVis and tSNE embeddings.*

Description

Perform basic 'pagoda2' processing, i.e. adjust variance, calculate pca reduction, make knn graph, identify clusters with multilevel, and generate largeVis and tSNE embeddings.

Usage

```
basicP2proc(
  cd,
  n.cores = 1,
  n.odgenes = 3000,
  nPcs = 100,
  k = 30,
  perplexity = 50,
  log.scale = TRUE,
  trim = 10,
  keep.genes = NULL,
  min.cells.per.gene = 0,
  min.transcripts.per.cell = 100,
```

```
  get.largevis = TRUE,
  get.tsne = TRUE,
  make.geneknn = TRUE
)
```

Arguments

cd	count matrix whereby rows are genes, columns are cells.
n.cores	numeric Number of cores to use (default=1)
n.odgenes	numeric Number of top overdispersed genes to use (dfault=3e3)
nPcs	numeric Number of PCs to use (default=100)
k	numeric Default number of neighbors to use in kNN graph (default=30)
perplexity	numeric Perplexity to use in generating tSNE and largeVis embeddings (default=50)
log.scale	boolean Whether to use log scale normalization (default=TRUE)
trim	numeric Number of cells to trim in winsorization (default=10)
keep.genes	optional set of genes to keep from being filtered out (even at low counts) (default=NULL)
min.cells.per.gene	numeric Minimal number of cells required for gene to be kept (unless listed in keep.genes) (default=0)
min.transcripts.per.cell	numeric Minimumal number of molecules/reads for a cell to be admitted (default=100)
get.largevis	boolean Whether to caluclate largeVis embedding (default=TRUE)
get.tsne	boolean Whether to calculate tSNE embedding (default=TRUE)
make.geneknn	boolean Whether pre-calculate gene kNN (for gene search) (default=TRUE)

Value

a new 'Pagoda2' object

---

basicP2web *Generate a 'pagoda2' web application from a 'Pagoda2' object*

Description

Generate a 'pagoda2' web application from a 'Pagoda2' object

Usage

```
basicP2web(p2, app.title = "Pagoda2", extraWebMetadata = NULL, n.cores = 4)
```

Arguments

p2	a 'Pagoda2' object
app.title	name of application as displayed in the browser title (default='Pagoda2')
extraWebMetadata	additional metadata generated by p2.metadata.from.fractor (default=NULL)
n.cores	numeric Number of cores to use for differential expression calculation (default=4)

Value

a 'pagoda2' web object

---

buildWijMatrix *Rescale the weights in an edge matrix to match a given perplexity. From 'largeVis', <https://github.com/elbamos/largeVis>*

Description

Rescale the weights in an edge matrix to match a given perplexity. From 'largeVis', <https://github.com/elbamos/largeVis>

Usage

```
buildWijMatrix(x, threads = NULL, perplexity = 50)
```

Arguments

x	An edgematrix, either an 'edgematrix' object or a sparse matrix.
threads	numeric The maximum number of threads to spawn (default=NULL). Determined automatically if NULL (default=NULL)
perplexity	numeric Given perplexity (default=50)

Value

A `list` with the following components:

'dist' An [N,K] matrix of the distances to the nearest neighbors.

'id' An [N,K] matrix of the node indexes of the neartest neighbors. Note that this matrix is 1-indexed, unlike most other matrices in this package.

'k' The number of nearest neighbors.

`calcMulticlassified` *Returns a list vector with the number of cells that are present in more than one selections in the provided p2 selection object*

Description

Returns a list vector with the number of cells that are present in more than one selections in the provided p2 selection object

Usage

```
calcMulticlassified(sel)
```

Arguments

`sel`	a pagoda2 selection as genereated by readPagoda2SelectionFile

Value

list vector with the number of cells that are present in more than one selections in the provided p2 selection object

`cellsPerSelectionGroup` *Get the number of cells in each selection group*

Description

Get the number of cells in each selection group

Usage

```
cellsPerSelectionGroup(selection)
```

Arguments

`selection`	a pagoda2 selection list

Value

a named vector of cell numbers in each groups

collapse.aspect.clusters *Collapse aspect patterns into clusters*

Description

Collapse aspect patterns into clusters

Usage

```
collapse.aspect.clusters(d, dw, ct, scale = TRUE, pick.top = FALSE)
```

Arguments

d	matrix of normalized aspect patterns (rows: significant aspects, columns: cells), normally the output $xv in 'tamr', the combined pathways that show similar expression patterns
dw	corresponding weight matrix to parameter 'd'
ct	clusters, the output of fastcluster::hclust()
scale	boolean Whether to scale aspects (default=TRUE)
pick.top	boolean Whether to pick top aspects (default=FALSE)

Value

list of clusters from matrix of normalized aspect patterns and clusters from the corresponding weight matrix

compareClusterings *Compare two different clusterings provided as factors by plotting a normalised heatmap*

Description

Compare two different clusterings provided as factors by plotting a normalised heatmap

Usage

```
compareClusterings(cl1, cl2, filename = NA)
```

Arguments

cl1	clustering 1, a named factor
cl2	clustering 2, a named factor
filename	an optional filename to save the plot instead of displaying it, will be passed to pheatmap (default=NA)

Value

invisible summary table that gets plotted

`extendedP2proc` *Perform extended 'Pagoda2' processing. Generate organism specific GO environment and calculate pathway overdispersion.*

Description

Perform extended 'Pagoda2' processing. Generate organism specific GO environment and calculate pathway overdispersion.

Usage

```
extendedP2proc(p2, organism = "hs")
```

Arguments

`p2`	the 'Pagoda2' object
`organism`	character Organisms hs (Homo Sapiens), mm (M. Musculus, mouse) or dr (D. Rerio, zebrafish) (default='hs')

Value

list of a 'Pagoda2' object and go.env

`factorFromP2Selection` *Returns a factor of cell membership from a p2 selection object the factor only includes cells present in the selection. If the selection contains multiclassified cells an error is raised*

Description

Returns a factor of cell membership from a p2 selection object the factor only includes cells present in the selection. If the selection contains multiclassified cells an error is raised

Usage

```
factorFromP2Selection(sel, use.internal.name = FALSE, flatten = FALSE)
```

Arguments

`sel`	a pagoda2 selection as genereated by readPagoda2SelectionFile
`use.internal.name`	boolean Whether to use field 'internal.name' as factor names (default=FALSE)
`flatten`	boolean Whether to ignore multiclassified cells, overwriting randomly (default=FALSE)

Value

factor of cell membership from a p2 selection object. The factor only includes cells present in the selection.

`factorListToMetadata` *Converts a list of factors into 'pagoda2' metadata optionally filtering down to the cells present in the provided 'pagoda2' app.*

Description

Converts a list of factors into 'pagoda2' metadata optionally filtering down to the cells present in the provided 'pagoda2' app.

Usage

```
factorListToMetadata(factor.list, p2 = NULL)
```

Arguments

`factor.list`	list of factors named by the cell identifier
`p2`	'pagoda2' app to filter the factors by, optional (default=NULL)

Value

'pagoda2' web metadata object

`factorToP2selection` *Converts a names factor to a p2 selection object if colors are provided it assigns those, otherwise uses a rainbow palette*

Description

Converts a names factor to a p2 selection object if colors are provided it assigns those, otherwise uses a rainbow palette

Usage

```
factorToP2selection(cl, col = NULL)
```

Arguments

`cl`	factor
`col`	names vector of colors (default=NULL)

Value

a p2 selection object (list)

`gene.vs.molecule.cell.filter` *Filter cells based on gene/molecule dependency*

Description

Filter cells based on gene/molecule dependency

Usage

```
gene.vs.molecule.cell.filter(
  countMatrix,
  min.cell.size = 500,
  max.cell.size = 50000,
  p.level = min(0.001, 1/ncol(countMatrix)),
  alpha = 0.1,
  plot = TRUE,
  do.par = TRUE
)
```

Arguments

`countMatrix`	input count matrix to be filtered
`min.cell.size`	numeric Min allowed cell size (default=500)
`max.cell.size`	numeric Max allowed cell size (default=5e4)
`p.level`	numeric Statistical confidence level for deviation from the main trend, used for cell filtering (default=min(1e-3,1/ncol(countMatrix)))
`alpha`	numeric Shading of the confidence band (default=0.1)
`plot`	boolean Plot the molecule distribution and the gene/molecule dependency fit (default=TRUE)
`do.par`	boolean Reset graphical parameters prior to plotting (default=TRUE)

Value

a filtered matrix

`generateClassificationAnnotation` *Given a cell clustering (partitioning) and a set of user provided selections generate a cleaned up annotation of cluster groups that can be used for classification*

Description

Given a cell clustering (partitioning) and a set of user provided selections generate a cleaned up annotation of cluster groups that can be used for classification

Usage

```
generateClassificationAnnotation(clustering, selections)
```

Arguments

`clustering`	a factor that provides the clustering
`selections`	a p2 selection object that provided by the web interfact user

Value

a named factor that can be used for classification

`get.control.geneset` *Get a control geneset for cell scoring using the method described in Puram, Bernstein (Cell, 2018)*

Description

Get a control geneset for cell scoring using the method described in Puram, Bernstein (Cell, 2018)

Usage

```
get.control.geneset(data, signature, n.bins = 25, n.genes.per.bin = 100)
```

Arguments

`data`	matrix of expression, rows are cell, columns are genes
`signature`	character vector The signature to evaluate, a character vector of genes
`n.bins`	numeric Number of bins to put the genes in (default=25)
`n.genes.per.bin`	numeric Number of genes to get from each bin (default=100)

Value

a character vector that can be used as a background signature

`get.de.geneset` *Generate differential expression genesets for the web app given a cell grouping by calculating DE sets between each cell set and everything else*

Description

Generate differential expression genesets for the web app given a cell grouping by calculating DE sets between each cell set and everything else

Usage

```
get.de.geneset(pagObj, groups, prefix = "de_")
```

Arguments

`pagObj`	pagoda object
`groups`	named factor to do the de by
`prefix`	chararcter Prefix to assign to genesets generated (default="de_")

Value

a 'pagoda2' web object

`getCellsInSelections` *Returns all the cells that are in the designated selections. Given a pagoda2 selections object and the names of some selections in it returns the names of the cells that are in these selections removed any duplicates*

Description

Returns all the cells that are in the designated selections. Given a pagoda2 selections object and the names of some selections in it returns the names of the cells that are in these selections removed any duplicates

Usage

```
getCellsInSelections(p2selections, selectionNames)
```

Arguments

`p2selections`	a p2 selections object
`selectionNames`	the names of some selections in th p2 object

Value

a character vector of cell names

`getClusterLabelsFromSelection`

Assign names to the clusters, given a clustering vector and a set of selections. This function will use a set of pagoda2 cell seletcion to identify the clusters in a a named factor. It is meant to be used to import user defined annotations that are defined as selections into a more formal categorization of cells that are defined by cluster. To help with this the function allows a percent of cells to have been classified in the selections into multiple groups, something which may be the result of the users making wrong selections. The percent of cells allows to be multiselected in any given group is defined by multiClassCutoff. Furthermore the method will assign each cluster to a selection only if the most popular cluster to the next most popular exceed the ambiguous.ratio in terms of cell numbers. If a cluster does not satisfy this condtiion it is not assigned.

Description

Assign names to the clusters, given a clustering vector and a set of selections. This function will use a set of pagoda2 cell seletcion to identify the clusters in a a named factor. It is meant to be used to import user defined annotations that are defined as selections into a more formal categorization of cells that are defined by cluster. To help with this the function allows a percent of cells to have been classified in the selections into multiple groups, something which may be the result of the users making wrong selections. The percent of cells allows to be multiselected in any given group is defined by multiClassCutoff. Furthermore the method will assign each cluster to a selection only if the most popular cluster to the next most popular exceed the ambiguous.ratio in terms of cell numbers. If a cluster does not satisfy this condtiion it is not assigned.

Usage

```
getClusterLabelsFromSelection(
  clustering,
  selections,
  multiClassCutoff = 0.3,
  ambiguous.ratio = 0.5
)
```

Arguments

`clustering`	a named factor of clusters, where every entry is a cell
`selections`	a pagoda2 selection object
`multiClassCutoff`	numeric Percent of cells in any one cluster that can be multiassigned (default=0.3)

ambiguous.ratio
: numeric Ratio of first and second cell numbers for any cluster to produce a valid clustering (default=0.5)

Value

a data.frame with two columns, one for cluster and one for selections, each cluster appears only once

---

getColorsFromP2Selection

Retrieves the colors of each selection from a p2 selection object as a names vector of strings

---

Description

Retrieves the colors of each selection from a p2 selection object as a names vector of strings

Usage

```
getColorsFromP2Selection(sel)
```

Arguments

sel	pagoda2 selection object

Value

a named vector of hex colours

---

getIntExtNamesP2Selection

Get a mapping form internal to external names for the specified selection object

---

Description

Get a mapping form internal to external names for the specified selection object

Usage

```
getIntExtNamesP2Selection(x)
```

Arguments

x	p2 selection object

Value

list of names from the specified selection object

`hierDiffToGenesets` *Converts the output of hierarchical differential expression aspects into genesets that can be loaded into a 'pagoda2' web app to retrive the genes that make the geneset interactively*

Description

Converts the output of hierarchical differential expression aspects into genesets that can be loaded into a 'pagoda2' web app to retrive the genes that make the geneset interactively

Usage

```
hierDiffToGenesets(output)
```

Arguments

`output`	output of getHierarchicalDiffExpressionAspects

Value

a geneset that can be loaded into p2 web genesets

`make.p2.app` *Generate a Rook Server app from a 'Pagoda2' object. This generates a 'pagoda2' web object from a 'Pagoda2' object by automating steps that most users will want to run. This function is a wrapper about the 'pagoda2' web constructor. (Advanced users may wish to use that constructor directly.)*

Description

Generate a Rook Server app from a 'Pagoda2' object. This generates a 'pagoda2' web object from a 'Pagoda2' object by automating steps that most users will want to run. This function is a wrapper about the 'pagoda2' web constructor. (Advanced users may wish to use that constructor directly.)

Usage

```
make.p2.app(
  r,
  dendrogramCellGroups,
  additionalMetadata = list(),
  geneSets,
  show.depth = TRUE,
  show.batch = TRUE,
  show.clusters = TRUE,
  appname = "Pagoda2 Application",
  innerOrder = NULL,
  orderDend = FALSE,
  appmetadata = NULL
)
```

Arguments

r	a 'Pagoda2' object
dendrogramCellGroups	a named factor of cell groups, used to generate the main dendrogram, limits zoom in
additionalMetadata	a list of metadata other than depth, batch and cluster that are automatically added (default=list())
geneSets	a list of genesets to show
show.depth	boolean Include depth as a metadata row (default=TRUE)
show.batch	boolean Include batch as a metadata row (default=TRUE)
show.clusters	boolean Include clusters as a metadata row (default=TRUE)
appname	character Application name (default="Pagoda2 Application")
innerOrder	Ordering of cells inside the clusters provided in dendrogramCellGroups (default=NULL). This should be one of "odPCA", "reductdist", "graphbased", "knn". Defaults to NULL
orderDend	boolean Whether to order dendrogram (default=FALSE)
appmetadata	a 'pagoda2' web application metadata (default=NULL)

Value

a 'pagoda2' web object that presents a Rook compatible interface

`minMaxScale` *Scale the designated values between the range of 0 and 1*

Description

Scale the designated values between the range of 0 and 1

Usage

```
minMaxScale(x)
```

Arguments

`x`	values to scale

Value

the scaled values

Examples

```
example_matrix =  matrix(rep(c(1:5), 3), 5)
minMaxScale(example_matrix)
```

`namedNames` *Get a vector of the names of an object named by the names themselves. This is useful with lapply when passing names of objects as it ensures that the output list is also named.*

Description

Get a vector of the names of an object named by the names themselves. This is useful with lapply when passing names of objects as it ensures that the output list is also named.

Usage

```
namedNames(g)
```

Arguments

`g`	an objects on which we can call names()

Value

vector with names of object

p2.generate.dr.go *Generate a GO environment for human for overdispersion analysis for the the back end*

Description

Generate a GO environment for human for overdispersion analysis for the the back end

Usage

```
p2.generate.dr.go(r)
```

Arguments

r	a 'Pagoda2' object

Value

a GO environment object

p2.generate.go *Generate a GO environment for the organism specified*

Description

Generate a GO environment for the organism specified

Usage

```
p2.generate.go(
  r,
  organism = NULL,
  go2all.egs = NULL,
  eg.alias2eg = NULL,
  min.env.length = 5
)
```

Arguments

r	a 'Pagoda2' object
organism	the organism (default=NULL). Currently 'hs' (human), 'mm' (mouse) and 'dr' (zebrafish) are supported.
go2all.egs	mappings between a given GO identifier and all of the Entrez Gene identifiers annotated at that GO term or to one of its child nodes in the GO ontology (default=NULL)

eg.alias2eg	mappings between common gene symbol identifiers and entrez gene identifiers (default=NULL)
min.env.length	numeric Minimum environment length (default=5)

`p2.generate.human.go` *Generate a GO environment for human for overdispersion analysis for the the back end*

Description

Generate a GO environment for human for overdispersion analysis for the the back end

Usage

```
p2.generate.human.go(r)
```

Arguments

r	a 'Pagoda2' object

Value

a GO environment object

`p2.generate.mouse.go` *Generate a GO environment for mouse for overdispersion analysis for the the back end*

Description

Generate a GO environment for mouse for overdispersion analysis for the the back end

Usage

```
p2.generate.mouse.go(r)
```

Arguments

r	a 'Pagoda2' object

Value

a GO environment object

`p2.make.pagoda1.app` *Create 'PAGODA1' web application from a 'Pagoda2' object 'PAGODA1' found here, with 'SCDE': <https://www.bioconductor.org/packages/release/bioc/html/scde.html>*

Description

Create 'PAGODA1' web application from a 'Pagoda2' object 'PAGODA1' found here, with 'SCDE': <https://www.bioconductor.org/packages/release/bioc/html/scde.html>

Usage

```
p2.make.pagoda1.app(
  p2,
  col.cols = NULL,
  row.clustering = NULL,
  title = "pathway clustering",
  zlim = NULL,
  embedding = NULL,
  inner.clustering = TRUE,
  groups = NULL,
  clusterType = NULL,
  embeddingType = NULL,
  veloinfo = NULL,
  type = "PCA",
  min.group.size = 1,
  batch.colors = NULL,
  n.cores = 10
)
```

Arguments

`p2`	'Pagoda2' object
`col.cols`	Matrix of column colors (default=NULL). Useful for visualizing cell annotations such as batch labels.
`row.clustering`	Row dendrogram (default=NULL)
`title`	character Title to use (default="pathway clustering")
`zlim`	Range of the normalized gene expression levels (default=NULL). Input as a list: c(lower_bound, upper_bound). Values outside this range will be Winsorized. Useful for increasing the contrast of the heatmap visualizations. If NULL, set to the 5th and 95th percentiles.
`embedding`	A 2-D embedding of the cells (PCA, tSNE, etc.), passed as a data frame with two columns (two dimensions) and rows corresponding to cells (row names have to match cell names) (default=NULL).
`inner.clustering`	boolean Whether to get overall cell clustering (default=TRUE).

groups	factor describing grouping of different cells. If provided, the cross-fits and the expected expression magnitudes will be determined separately within each group. The factor should have the same length as ncol(counts) (default=NULL).
clusterType	cluster type (default=NULL). If NULL, takes the latest cluster in the 'Pagoda2' object using 'p2$clusters[[type]][[1]]'
embeddingType	embedding type (default=NULL). If NULL, takes the latest embedding in the 'Pagoda2' object using p2$embeddings[[type]][[1]]
veloinfo	cell velocity information, cell velocities (grid and cell) (default=NULL)
type	character Either 'counts' or a name of a 'reduction' in the 'Pagoda2' object (default='PCA')
min.group.size	integer Minimum group size (default=1)
batch.colors	colors of the batches, i.e. the factor (corresponding to rows of the model matrix) specifying batch assignment of each cell(default=NULL)
n.cores	numeric Number of cores (default=10)

Value

'PAGODA1' web application

`p2.metadata.from.factor`

Generate a list metadata structure that can be passed to a 'pagoda2' web object constructor as additional metadata given a named factor

---

Description

Generate a list metadata structure that can be passed to a 'pagoda2' web object constructor as additional metadata given a named factor

Usage

```
p2.metadata.from.factor(
  metadata,
  displayname = NULL,
  s = 1,
  v = 1,
  start = 0,
  end = NULL,
  pal = NULL
)
```

Arguments

metadata	named factor with metadata for individual cells, names must correspond to cells
displayname	character Name to display for the metadata (default=NULL)
s	numeric Value for rainbow palette (default=1)
v	numeric Value for rainbow palette (default=1)
start	numeric Starting value (default=0)
end	numeric Ending value (default=NULL)
pal	optional vector of colours to use, if provided overrides s,v,start and end parameters (default=NULL)

Value

list of data, levels, palette to be passed to 'pagoda2' web object constructor

---

p2.toweb.hdea *Generate a 'pagoda2' web object from a 'Pagoda2' object using hierarchical differential expression*

Description

Generate a 'pagoda2' web object from a 'Pagoda2' object using hierarchical differential expression

Usage

```
p2.toweb.hdea(p2, title = "")
```

Arguments

p2	p2 object
title	character Name of the pagoda object (default="")

Value

a 'pagoda2' web object

p2ViewPagodaApp *p2ViewPagodaApp R6 class*

Description

Modified 'PAGODA1' app (from 'SCDE') for browsing 'pagoda2' results. Refer to 'ViewPagodaAppOld' and 'make.pagoda.app()' in 'SCDE'

Public fields

`results` Result object returned by `scde.expression.difference()` (default=NULL). Note to browse group posterior levels, use `return.posteriors = TRUE` in the `scde.expression.difference()` call.

`type` Either 'counts' or a name of a 'reduction' in the 'Pagoda2' object

`genes` List of genes to display in the Detailed clustering panel (default=list())

`batch` Any batch or other known confounders to be included in the visualization as a column color track (default=NULL)

`pathways` character vector Pathway or gene names (default=NULL)

`name` App name (needs to be altered only if adding more than one app to the server using the 'server' parameter) (default=NULL)

`trim` Trim quantity used for Winsorization for visualization

`embedding` Embedding information (default=NULL)

`veloinfo` Velocity information (default=NULL)

`goenv` environment mapping pathways to genes (default=NULL)

`renv` Global environment (default=NULL)

Methods

Public methods:

- `p2ViewPagodaApp$new()`
- `p2ViewPagodaApp$getgenecldata()`
- `p2ViewPagodaApp$call()`
- `p2ViewPagodaApp$clone()`

Method `new()`**:** Initialize p2ViewPagodaApp class

Usage:

```
p2ViewPagodaApp$new(
  results,
  pathways,
  genes,
  goenv,
  batch = NULL,
  name = "pathway overdispersion",
```

```
  trim = 1.1/nrow(p2$counts),
  embedding = NULL,
  type,
  veloinfo = NULL
)
```

Arguments:

`results` Result object returned by `scde.expression.difference()`. Note to browse group posterior levels, use `return.posteriors = TRUE` in the `scde.expression.difference()` call.

`pathways` character vector Pathway or gene names (default=NULL)

`genes` list Genes to display in the Detailed clustering panel (default=list())

`goenv` Environment mapping pathways to genes (default=NULL)

`batch` Any batch or other known confounders to be included in the visualization as a column color track (default=NULL)

`name` string App name (needs to be altered only if adding more than one app to the server using the 'server' parameter) (default="pathway overdispersion")

`trim` numeric Trim quantity used for Winsorization for visualization (default=1.1/nrow(p2$counts) whereby the 'counts' from the 'Pagoda2' object is the gene count matrix, normalized on total counts (default=NULL)

`embedding` Embedding information (default=NULL)

`type` Either 'counts' or a name of a 'reduction' in the 'pagoda2' object

`veloinfo` Velocity information (default=NULL)

Returns: new 'p2ViewPagodaApp' object

Method `getgenecldata()`**:** Helper function to get the heatmap data for a given set of genes

Usage:

```
p2ViewPagodaApp$getgenecldata(genes = NULL, gcl = NULL, ltrim = 0)
```

Arguments:

`genes` character vector Gene names (default=NULL)

`gcl` pathway or gene-weighted PCA (default=NULL). If NULL, uses tp2c.view.pathways(self$genes, self$results$p2, goenv=goenv, vhc=self$results$hvc, plot=FALSE, trim=ltrim, n.genes=Inf).

`ltrim` numeric Winsorization trim that should be applied (default=0)

Returns: heatmap data for a given set of genes

Method `call()`**:** Call Rook application. Using client-side ExtJS framework and Inchlib HTML5 canvas libraries to create the graphical user interface for PAGODA

Usage:

```
p2ViewPagodaApp$call(env)
```

Arguments:

`env` The environment argument is a true R environment object which the application is free to modify. Please see the Rook documentation for more details.

Returns: modified 'PAGODA1' app

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
p2ViewPagodaApp$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

`pagoda.reduce.loading.redundancy`

Collapse aspects driven by the same combinations of genes. (Aspects are some pattern across cells e.g. sequencing depth, or PC corresponding to an undesired process such as ribosomal pathway variation.) Examines PC loading vectors underlying the identified aspects and clusters of aspects based on a product of loading and score correlation (raised to corr.power). Clusters of aspects driven by the same genes are determined based on the parameter "distance.threshold".

Description

Collapse aspects driven by the same combinations of genes. (Aspects are some pattern across cells e.g. sequencing depth, or PC corresponding to an undesired process such as ribosomal pathway variation.) Examines PC loading vectors underlying the identified aspects and clusters of aspects based on a product of loading and score correlation (raised to corr.power). Clusters of aspects driven by the same genes are determined based on the parameter "distance.threshold".

Usage

```
pagoda.reduce.loading.redundancy(
  tam,
  pwpca,
  clpca = NULL,
  plot = FALSE,
  cluster.method = "complete",
  distance.threshold = 0.01,
  corr.power = 4,
  abs = TRUE,
  n.cores = 1,
  ...
)
```

Arguments

tam	output of pagoda.top.aspects(), i.e. a list structure containing the following items: xv: a matrix of normalized aspect patterns (rows: significant aspects, columns: cells) xvw: corresponding weight matrix gw: set of genes driving the significant aspects df: text table with the significance testing results
pwpca	output of pagoda.pathway.wPCA(), i.e. a list of weighted PCA info for each valid gene set

clpca	output of pagoda.gene.clusters() (optional) (default=NULL). The output of pagoda.gene.clusters() is a list structure containing the following fields: clusters: alist of genes in each cluster values xf: extreme value distribution fit for the standardized lambda1 of a randomly generated pattern tci: index of a top cluster in each random iteration cl.goc: weighted PCA info for each real gene cluster varm: standardized lambda1 values for each randomly generated matrix cluster clvlm: a linear model describing dependency of the cluster lambda1 on a Tracy-Widom lambda1 expectation
plot	boolean Whether to plot the resulting clustering (default=FALSE)
cluster.method	string One of the standard clustering methods to be used (default="complete")
distance.threshold	numeric Similarity threshold for grouping interdependent aspects (default=0.01)
corr.power	numeric Power to which the product of loading and score correlation is raised (default=4)
abs	boolean Whether to use absolute correlation (default=TRUE)
n.cores	numeric Number of cores to use during processing (default=1)
...	additional arguments are passed to the pagoda.view.aspects() method during plotting

Value

a list structure analogous to that returned by pagoda.top.aspects(), but with addition of a $cnam element containing a list of aspects summarized by each row of the new (reduced) $xv and $xvw

---

`pagoda.reduce.redundancy`

Collapse aspects driven by similar patterns (i.e. separate the same sets of cells) Examines PC loading vectors underlying the identified aspects and clusters aspects based on score correlation. Clusters of aspects driven by the same patterns are determined based on the distance.threshold.

Description

Collapse aspects driven by similar patterns (i.e. separate the same sets of cells) Examines PC loading vectors underlying the identified aspects and clusters aspects based on score correlation. Clusters of aspects driven by the same patterns are determined based on the distance.threshold.

Usage

```
pagoda.reduce.redundancy(
  tamr,
  distance.threshold = 0.2,
  cluster.method = "complete",
  distance = NULL,
```

```
  weighted.correlation = TRUE,
  plot = FALSE,
  top = Inf,
  trim = 0,
  abs = FALSE,
  ...
)
```

Arguments

tamr	Combined pathways that show similar expression patterns, output of pagoda.reduce.loading.redundancy()
distance.threshold	numeric Similarity threshold for grouping interdependent aspects (default=0.2)
cluster.method	character One of the standard clustering methods to be used (default="complete")
distance	distance matrix (default=NULL)
weighted.correlation	boolean Whether to use a weighted correlation in determining the similarity of patterns (default=TRUE)
plot	boolean Whether to show plot (default=FALSE)
top	bololean Restrict output to the top N aspects of heterogeneity (default=Inf, i.e. no restriction)
trim	numeric Winsorization trim to use prior to determining the top aspects (default=0)
abs	boolean Whether to use absolute correlation (default=FALSE)
...	additional arguments are passed to the pagoda.view.aspects() method during plotting

Value

List structure analogous to that returned by pagoda.top.aspects(), but with addition of a $cnam element containing a list of aspects summarized by each row of the new (reduced) $xv and $xvw

---

pagoda2WebApp-class *pagoda2WebApp class to create 'pagoda2' web applications via a Rook server*

Description

pagoda2WebApp class to create 'pagoda2' web applications via a Rook server

Fields

`originalP2object` Input 'Pagoda2' object

`name` string Display name for the application

`mat` Embedding

`cellmetadata` Metadata associated with 'Pagoda2' object

`mainDendrogram` Dendrogram from hclust() of all cells in the 'Pagoda2' object

`geneSets` Gene sets in the 'Pagoda2' object

`rookRoot` Rook server root directory

`appmetadata` pagoda2 web application metadata

`pagoda2WebApp_arrayToJSON`
pagoda2WebApp_arrayToJSON

Description

Serialise an R array to a JSON object

Arguments

`arr` An array (default=NULL)

Value

Serialised version of the array in JSON, which includes dimension information as separate fields

---

`pagoda2WebApp_availableAspectsJSON`
pagoda2WebApp_availableAspectsJSON

Description

Parse pathways from originalP2object$misc$pathwayOD$xv into JSON

Value

JSON with parsed cell order from mainDendrogram$cellorder

pagoda2WebApp_call *pagoda2WebApp_call*

Description

Handle httpd server calls

Arguments

env	The environment argument is a true R environment object which the application is free to modify. Please see the Rook documentation for more details.

pagoda2WebApp_cellmetadataJSON
pagoda2WebApp_cellmetadataJSON

Description

Parse cellmetadata into JSON

Value

JSON with parsed cellmetadata

pagoda2WebApp_cellOrderJSON
pagoda2WebApp_cellOrderJSON

Description

Parse mainDendrogram$cellorder into JSON

Value

JSON with parsed cell order from mainDendrogram$cellorder

pagoda2WebApp_geneInformationJSON
pagoda2WebApp_geneInformationJSON

Description

Parse originalP2object$misc$varinfo[,c("m","qv")] into JSON

Value

JSON with parsed information from genename, dispersion, mean gene expression

pagoda2WebApp_generateDendrogramOfGroups
Generate a dendrogram of groups

Description

Generate a dendrogram of groups

Arguments

dendrogramCellGroups
Cell groups to input into hclust()

Value

List of hcGroups, cellorder, and cluster.sizes

pagoda2WebApp_generateEmbeddingStructure
pagoda2WebApp_generateEmbeddingStructure

Description

Generate information about the embeddings we are exporting

Value

List with embeddings

pagoda2WebApp_generateGeneKnnJSON
pagoda2WebApp_generateGeneKnnJSON

Description

Generate a JSON list representation of the gene kNN network

Arguments

graph	Input graph

Value

JSON with gene kNN network

pagoda2WebApp_getCompressedEmbedding
pagoda2WebApp_getCompressedEmbedding

Description

Compress the embedding

Arguments

reduc	reduction
embed	embedding

Value

compressed embedding as JSON

pagoda2WebApp_packCompressFloat64Array
pagoda2WebApp_packCompressFloat64Array

Description

Compress float64 array

Arguments

v	float64 array

Value

compressed array

pagoda2WebApp_packCompressInt32Array
pagoda2WebApp_packCompressInt32Array

Description

Compress int32 array

Arguments

v	int32 array

Value

compressed array

pagoda2WebApp_readStaticFile
pagoda2WebApp_readStaticFile

Description

Read a static file from the filesystem, and put in the response

Arguments

filename	path to filename

Value

Content to display or error page

`pagoda2WebApp_reducedDendrogramJSON`
pagoda2WebApp_reducedDendrogramJSON

Description

Parse dendrogram into JSON

Value

JSON with parsed dendrogram

`pagoda2WebApp_serializeToStaticFast`
pagoda2WebApp_serializeToStaticFast

Description

Convert serialized file to static file

Arguments

`binary.filename`	path to binary file (default=NULL)
`verbose`	boolean Whether to give verbose output (default=FALSE)

Value

static file written by WriteListToBinary(expL=exportList, outfile=binary.filename, verbose=verbose)

`pagoda2WebApp_serverLog`
pagoda2WebApp_serverLog

Description

Logging function for console

Arguments

`message`	Message to output for the console

Value

printed message

pagoda2WebApp_sparseMatList *pagoda2WebApp_sparseMatList*

Description

Create simple List from sparse Matrix with Dimnames as JSON

Arguments

matsparse	Sparse matrix

Value

List with slots i, p, x

pathway.pc.correlation.distance *Calculate correlation distance between PC magnitudes given a number of target dimensions*

Description

Calculate correlation distance between PC magnitudes given a number of target dimensions

Usage

```
pathway.pc.correlation.distance(pcc, xv, n.cores = 1, target.ndf = NULL)
```

Arguments

pcc	weighted PC magnitudes e.g. scde::pagoda.pathway.wPCA() gives the weighted PC magnitudes for each gene provided; e.g. scde::pagoda.gene.clusters() gives the weighted PC magnitudes for de novo gene sets identified by clustering on expression
xv	a matrix of normalized aspect patterns (rows: significant aspects, columns: cells)
n.cores	numeric Number of cores to use (default=1)
target.ndf	numeric Target dimensions (default=NULL)

Value

correlation distance matrix, akin to stats dist

plotMulticlassified *Plot multiclassified cells per selection as a percent barplot*

Description

Plot multiclassified cells per selection as a percent barplot

Usage

```
plotMulticlassified(sel)
```

Arguments

sel	pagoda2 selection object

Value

ggplot2 object

plotOneWithValues *Plot the embedding of a 'Pagoda2' object with the given values*

Description

Plot the embedding of a 'Pagoda2' object with the given values

Usage

```
plotOneWithValues(
  p2obj,
  values,
  title = "",
  type = "PCA",
  embeddingType = "tSNE"
)
```

Arguments

p2obj	the 'Pagoda2' object
values	the values to plot, fed into p2obj$plotEmbedding(colors=values)
title	character Title for the plot (default="")
type	character Type reduction on which the embedding is based on (default="PCA")
embeddingType	character Type of embedding to plot (default="tSNE")

Value

NULL, simply updates p2obj$plotEmbedding()

plotSelectionOverlaps *Get a dataframe and plot summarising overlaps between selection of a pagoda2 selection object ignore self overlaps*

Description

Get a dataframe and plot summarising overlaps between selection of a pagoda2 selection object ignore self overlaps

Usage

```
plotSelectionOverlaps(sel)
```

Arguments

sel	a pagoda2 selection object

Value

a list that contains a ggplot2 object and a datatable with the overlaps data

projectKNNs *Project a distance matrix into a lower-dimensional space. (from elbamos/largeVis)*

Description

Takes as input a sparse matrix of the edge weights connecting each node to its nearest neighbors, and outputs a matrix of coordinates embedding the inputs in a lower-dimensional space.

Usage

```
projectKNNs(
  wij,
  dim = 2,
  sgd_batches = NULL,
  M = 5,
  gamma = 7,
  alpha = 1,
  rho = 1,
  coords = NULL,
  useDegree = FALSE,
  momentum = NULL,
  seed = NULL,
  threads = NULL,
  verbose = getOption("verbose", TRUE)
)
```

Arguments

wij	A symmetric sparse matrix of edge weights, in C-compressed format, as created with the Matrix package.
dim	numeric The number of dimensions for the projection space (default=2)
sgd_batches	numeric The number of edges to process during SGD (default=NULL). Defaults to a value set based on the size of the dataset. If the parameter given is between 0 and 1, the default value will be multiplied by the parameter.
M	numeric (largeVis) The number of negative edges to sample for each positive edge (default=5).
gamma	numeric (largeVis) The strength of the force pushing non-neighbor nodes apart (default=7).
alpha	numeric (largeVis) The hyperparameter in the distance function (default=1). The default distance function, $1/(1 + \alpha||y_i - y_j||^2)$. The function relates the distance between points in the low-dimensional projection to the likelihood that the two points are nearest neighbors. Increasing α tends to push nodes and their neighbors closer together; decreasing α produces a broader distribution. Setting α to zero enables the alternative distance function. α below zero is meaningless.
rho	(largeVis) numeric Initial learning rate (default=1)
coords	An initialized coordinate matrix (default=NULL)
useDegree	boolean Whether to use vertex degree to determine weights in negative sampling (if TRUE) or the sum of the vertex's edges (if FALSE) (default=FALSE)
momentum	If not NULL, SGD with momentum is used, with this multiplier, which must be between 0 and 1 (default=NULL). Note that momentum can drastically speed-up training time, at the cost of additional memory consumed.
seed	numeric Random seed to be passed to the C++ functions (default=NULL). Sampled from hardware entropy pool if NULL (the default). Note that if the seed is not NULL (the default), the maximum number of threads will be set to 1 in phases of the algorithm that would otherwise be non-deterministic.
threads	numeric The maximum number of threads to spawn (default=NULL). Determined automatically if NULL (the default).
verbose	boolean Verbosity (default=getOption("verbose", TRUE))

Details

The algorithm attempts to estimate a dim-dimensional embedding using stochastic gradient descent and negative sampling.

The objective function is:

$$O = \sum_{(i,j)\in E} w_{ij}(\log f(||p(e_{ij} = 1||) + \sum_{k=1}^{M} E_{jk\ P_n(j)}\gamma \log(1 - f(||p(e_{ij_k} - 1||)))$$

where $f()$ is a probabilistic function relating the distance between two points in the low-dimensional projection space, and the probability that they are nearest neighbors.

The default probabilistic function is $1/(1 + \alpha||x||^2)$. If α is set to zero, an alternative probabilistic function, $1/(1 + \exp(x^2))$ will be used instead.

Note that the input matrix should be symmetric. If any columns in the matrix are empty, the function will fail.

Value

A dense [N,D] matrix of the coordinates projecting the w_ij matrix into the lower-dimensional space.

Note

If specified, `seed` is passed to the C++ and used to initialize the random number generator. This will not, however, be sufficient to ensure reproducible results, because the initial coordinate matrix is generated using the `R` random number generator. To ensure reproducibility, call `set.seed` before calling this function, or pass it a pre-allocated coordinate matrix.

The original paper called for weights in negative sampling to be calculated according to the degree of each vertex, the number of edges connecting to the vertex. The reference implementation, however, uses the sum of the weights of the edges to each vertex. In experiments, the difference was imperceptible with small (MNIST-size) datasets, but the results seems aesthetically preferrable using degree. The default is to use the edge weights, consistent with the reference implementation.

`read.10x.matrices` *Quick loading of 10X CellRanger count matrices*

Description

Quick loading of 10X CellRanger count matrices

Usage

```
read.10x.matrices(matrixPaths, version = "V3", n.cores = 1, verbose = TRUE)
```

Arguments

`matrixPaths`	a single path to the folder containing matrix.mtx, genes.tsv and barcodes.tsv files, OR a named list of such paths
`version`	string Version of 10x output to read (default='V3'). Must be one of 'V2' or 'V3'.
`n.cores`	numeric Cores to utilize in parallel (default=1)
`verbose`	boolean Whether to output verbose output (default=TRUE)

Value

a sparse matrix representation of the data (or a list of sparse matrices if a list of paths was passed)

`read10xMatrix` *This function reads a matrix generated by the 10x processing pipeline from the specified directory and returns it. It aborts if one of the required files in the specified directory do not exist.*

Description

This function reads a matrix generated by the 10x processing pipeline from the specified directory and returns it. It aborts if one of the required files in the specified directory do not exist.

Usage

```
read10xMatrix(path, version = "V3", transcript.id = "SYMBOL", verbose = TRUE)
```

Arguments

`path`	string Location of 10x output
`version`	string Version of 10x output to read (default='V3'). Must be one of 'V2' or 'V3'.
`transcript.id`	string Transcript identifier to use (default='SYMBOL'). Must be either 'SYMBOL' (e.g. "Sox17") or 'ENSEMBL' (e.g. "ENSMUSG00000025902"). This value is case-sensitive.
`verbose`	boolean Whether to return verbose output

Value

parsed 10x outputs into a matrix

`readPagoda2SelectionAsFactor` *Read a pagoda2 cell selection file and return it as a factor while removing any mutliclassified cells*

Description

Read a pagoda2 cell selection file and return it as a factor while removing any mutliclassified cells

Usage

```
readPagoda2SelectionAsFactor(filepath, use.internal.name = FALSE)
```

Arguments

filepath	name of the selection file
use.internal.name	boolean Use field 'internal.name' as factor names (default=FALSE). Passed to factorFromP2Selection

Value

a name factor with the membership of all the cells that are not multiclassified

---

readPagoda2SelectionFile

Reads a 'pagoda2' web app exported cell selection file exported as a list of list objects that contain the name of the selection, the color (as a hex string) and the identifiers of the individual cells

---

Description

Reads a 'pagoda2' web app exported cell selection file exported as a list of list objects that contain the name of the selection, the color (as a hex string) and the identifiers of the individual cells

Usage

```
readPagoda2SelectionFile(filepath)
```

Arguments

filepath	the path of the file load

---

removeSelectionOverlaps

Remove cells that are present in more than one selection from all the selections they are in

---

Description

Remove cells that are present in more than one selection from all the selections they are in

Usage

```
removeSelectionOverlaps(selections)
```

Arguments

selections	a pagoda2 selections list

Value

a new list with the duplicated cells removed

`score.cells.nb0` *Score cells by getting mean expression of genes in signatures*

Description

Score cells by getting mean expression of genes in signatures

Usage

```
score.cells.nb0(data, signature)
```

Arguments

`data`	matrix
`signature`	the genes in the signature

Value

cell scores

`score.cells.puram` *Puram, Bernstein (Cell, 2018) Score cells as described in Puram, Bernstein (Cell, 2018)*

Description

Puram, Bernstein (Cell, 2018) Score cells as described in Puram, Bernstein (Cell, 2018)

Usage

```
score.cells.puram(data, signature, correct = TRUE, show.plot = FALSE, ...)
```

Arguments

`data`	matrix of expression, rows are cell, columns are genes
`signature`	character vector The signature to evaluate, a character vector of genes
`correct`	boolean Perform background correction by getting a semi-random geneset (default=TRUE)
`show.plot`	boolean If corrected values are calculated show plot of corrected vs original scores (default=FALSE)
`...`	options for get.control.geneset()

Value

a score for each cell

`sgdBatches` *Calculate the default number of batches for a given number of vertices and edges. The formula used is the one used by the 'largeVis' reference implementation. This is substantially less than the recommendation* $E * 10000$ *in the original paper.*

Description

Calculate the default number of batches for a given number of vertices and edges. The formula used is the one used by the 'largeVis' reference implementation. This is substantially less than the recommendation $E * 10000$ in the original paper.

Usage

```
sgdBatches(N, E = 150 * N/2)
```

Arguments

N	Number of vertices
E	Number of edges (default = 150*N/2)

Value

The recommended number of sgd batches.

Examples

```
# Observe that increasing K has no effect on processing time

N <- 70000 # MNIST
K <- 10:250
plot(K, sgdBatches(rep(N, length(K)), N * K / 2))

# Observe that processing time scales linarly with N
N <- c(seq(from = 1, to = 10000, by = 100), seq(from = 10000, to = 10000000, by = 1000))
plot(N, sgdBatches(N))
```

show.app *Directly open the 'pagoda2' web application and view the 'p2web' application object from our R session*

Description

Directly open the 'pagoda2' web application and view the 'p2web' application object from our R session

Usage

```
show.app(app, name, port, ip, browse = TRUE, server = NULL)
```

Arguments

app	'pagoda2' application object
name	character Name of the application to view
port	numeric Port number
ip	numeric IP address
browse	boolean Whether to load the app into an HTML browser (default=TRUE)
server	server If NULL, will grab server with get.scde.server(port=port, ip=ip) (default=NULL)

Value

application within browser

subsetSignatureToData *Subset a gene signature to the genes in the given matrix with optional warning if genes are missing*

Description

Subset a gene signature to the genes in the given matrix with optional warning if genes are missing

Usage

```
subsetSignatureToData(data, signature, raise.warning = TRUE)
```

Arguments

data	matrix
signature	character vector The gene signature from which to subset a character vector of genes
raise.warning	boolean Warn if genes are missing (default=TRUE)

Value

The filtered subset of gene signatures

`tp2c.view.pathways` *View pathway or gene-weighted PCA 'Pagoda2' version of the function pagoda.show.pathways() Takes in a list of pathways (or a list of genes), runs weighted PCA, optionally showing the result.*

Description

View pathway or gene-weighted PCA 'Pagoda2' version of the function pagoda.show.pathways() Takes in a list of pathways (or a list of genes), runs weighted PCA, optionally showing the result.

Usage

```
tp2c.view.pathways(
  pathways,
  p2,
  goenv = NULL,
  batch = NULL,
  n.genes = 20,
  two.sided = TRUE,
  n.pc = rep(1, length(pathways)),
  colcols = NULL,
  zlim = NULL,
  labRow = NA,
  vhc = NULL,
  cexCol = 1,
  cexRow = 1,
  nstarts = 50,
  row.order = NULL,
  show.Colv = TRUE,
  plot = TRUE,
  trim = 1.1/nrow(p2$counts),
  showPC = TRUE,
  ...
)
```

Arguments

`pathways`	character vector of pathway or gene names
`p2`	'Pagoda2' object
`goenv`	environment mapping pathways to genes (default=NULL)
`batch`	factor (corresponding to rows of the model matrix) specifying batch assignment of each cell, to perform batch correction (default=NULL).

n.genes	integer Number of genes to show (default=20)
two.sided	boolean If TRUE, the set of shown genes should be split among highest and lowest loading (default=TRUE). If FALSE, genes with highest absolute loading should be shown.
n.pc	integer vector Number of principal component to show for each listed pathway(default=rep(1, length(pathways)))
colcols	column color matrix (default=NULL)
zlim	numeric z color limit (default=NULL)
labRow	row labels (default=NA)
vhc	cell clustering (default=NULL)
cexCol	positive numbers, used as cex.axis in for the row or column axis labeling(default=1)
cexRow	positive numbers, used as cex.axis in for the row or column axis labeling(default=1)
nstarts	integer Number of random starts to use (default=50)
row.order	row order (default=NULL). If NULL, uses order from hclust.
show.Colv	boolean Whether to show cell dendrogram (default=TRUE)
plot	boolean Whether to plot (default=TRUE)
trim	numeric Winsorization trim that should be applied (default=1.1/nrow(p2$counts)). Note that p2 is a 'Pagoda2' object.
showPC	boolean (default=TRUE)
...	parameters to pass to my.heatmap2. Only if plot is TRUE.

Value

cell scores along the first principal component of shown genes (returned as invisible)

`validateSelectionsObject`

Validates a pagoda2 selection object

Description

Validates a pagoda2 selection object

Usage

```
validateSelectionsObject(selections)
```

Arguments

selections	the pagoda2 selection object to be validated

Value

a logical value indicating if the object is valid

webP2proc *Generate a 'pagoda2' web object*

Description

Generate a 'pagoda2' web object

Usage

```
webP2proc(
  p2,
  additionalMetadata = NULL,
  title = "Pagoda2",
  make.go.sets = TRUE,
  make.de.sets = TRUE,
  go.env = NULL,
  make.gene.graph = TRUE,
  appmetadata = NULL
)
```

Arguments

p2	a 'Pagoda2' object
additionalMetadata	'pagoda2' web metadata object (default=NULL)
title	character string Title for the web app (default='Pagoda2')
make.go.sets	boolean Whether GO sets should be made (default=TRUE)
make.de.sets	boolean Whether differential expression sets should be made (default=TRUE)
go.env	the GO environment used for the overdispersion analysis (default=NULL)
make.gene.graph	logical specifying if the gene graph should be make, if FALSE the find similar genes functionality will be disabled on the web app
appmetadata	'pagoda2' web application metadata (default=NULL)

Value

a 'pagoda2' web application

winsorize.matrix *Sets the ncol(mat)*trim top outliers in each row to the next lowest value same for the lowest outliers*

Description

Sets the ncol(mat)*trim top outliers in each row to the next lowest value same for the lowest outliers

Usage

```
winsorize.matrix(mat, trim)
```

Arguments

mat	Numeric matrix
trim	numeric Fraction of outliers (on each side) that should be Winsorized, or (if the value is >= 1) the number of outliers to be trimmed on each side

Value

Winsorized matrix

Examples

```
set.seed(0)
mat <- matrix( c(rnorm(5*10,mean=0,sd=1), rnorm(5*10,mean=5,sd=1)), 10, 10)  # random matrix
mat[1,1] <- 1000  # make outlier
range(mat)  # look at range of values
win.mat <- winsorize.matrix(mat, 0.1)
range(win.mat)  # note outliers removed
```

writeGenesAsPagoda2Selection *Writes a list of genes as a gene selection that can be loaded in the web interface*

Description

Writes a list of genes as a gene selection that can be loaded in the web interface

Usage

```
writeGenesAsPagoda2Selection(name, genes, filename)
```

Arguments

name	the name of the selection
genes	a string vector of the gene names
filename	the filename to save to

Value

NULL, writes to filepath the list of genes as a gene selection that can be loaded in the web interface

---

writePagoda2SelectionFile

Writes a pagoda2 selection object as a p2 selection file that be be loaded to the web interface

---

Description

Writes a pagoda2 selection object as a p2 selection file that be be loaded to the web interface

Usage

```
writePagoda2SelectionFile(sel, filepath)
```

Arguments

sel	pagoda2 selection object
filepath	name of file to which to write

Value

NULL, writes to filepath the pagoda2 selection object as a p2 selection file that be be loaded to the web interface

Index