Package ‘TSCAN’

March 30, 2021

Type Package

Title Tools for Single-Cell Analysis

Version 1.28.0

Date 2020-10-18

Description Provides methods to perform trajectory analysis based on a minimum spanning tree constructed from cluster centroids. Computes pseudotemporal cell orderings by mapping cells in each cluster (or new cells) to the closest edge in the tree. Uses linear modelling to identify differentially expressed genes along each path through the tree. Several plotting and interactive visualization functions are also implemented.

License GPL(>=2)

Imports ggplot2, shiny, plyr, grid, fastICA, igraph, combinat, mgcv, mclust, gplots, methods, stats, Matrix, SummarizedExperiment, SingleCellExperiment, DelayedArray, S4Vectors

VignetteBuilder knitr

Suggests knitr, testthat, beachmat, scuttle, scran, BiocParallel, BiocNeighbors, batchelor

biocViews GeneExpression, Visualization, GUI

RoxygenNote 7.1.1

git_url https://git.bioconductor.org/packages/TSCAN

git_branch RELEASE_3_12

git_last_commit 7c1c17e

git_last_commit_date 2020-10-27

Date/Publication 2021-03-29

Author Zhicheng Ji [aut, cre],
Hongkai Ji [aut],
Aaron Lun [ctb]

Maintainer Zhicheng Ji <zji4@jhu.edu>

R topics documented:

`createClusterMST` *Minimum spanning trees on cluster centroids*

Description

Build a MST where each node is a cluster centroid and each edge is weighted by the Euclidean distance between centroids. This represents the most parsimonious explanation for a particular trajectory and has the advantage of being directly intepretable with respect to any pre-existing clusters.

Usage

```
createClusterMST(x, ...)

## S4 method for signature 'ANY'
createClusterMST(
  x,
  clusters,
  outgroup = FALSE,
  outscale = 3,
  columns = NULL,
  with.mnn = FALSE,
  mnn.k = 50,
  BNPARAM = NULL,
  BPPARAM = NULL
)

## S4 method for signature 'SummarizedExperiment'
createClusterMST(x, ..., assay.type = "logcounts")

## S4 method for signature 'SingleCellExperiment'
createClusterMST(
  x,
  clusters = colLabels(x, onAbsence = "error"),
  ...,
  use.dimred = NULL
)
```

Arguments

`x`	A numeric matrix of coordinates where each row represents a cell/sample and each column represents a dimension (usually a PC or another low-dimensional embedding, but features or genes can also be used). Alternatively, a SummarizedExperiment or SingleCellExperiment object containing such a matrix in its `assays`, as specified by `assay.type`. This will be transposed prior to use. Alternatively, for SingleCellExperiments, this matrix may be extracted from its `reducedDims`, based on the `use.dimred` specification. In this case, no transposition is performed. Alternatively, if `clusters=NULL`, a numeric matrix of coordinates for cluster centroids, where each row represents a cluster and each column represents a dimension Each row should be named with the cluster name. This mode can also be used with assays/matrices extracted from SummarizedExperiments and SingleCellExperiments.
`...`	For the generic, further arguments to pass to the specific methods. For the SummarizedExperiment method, further arguments to pass to the ANY method. For the SingleCellExperiment method, further arguments to pass to the SummarizedExperiment method (if `use.dimred` is specified) or the ANY method (otherwise).
`clusters`	A factor-like object of the same length as `nrow(x)`, specifying the cluster identity for each cell in `x`. If `NULL`, `x` is assumed to already contain coordinates for the cluster centroids.
`outgroup`	A logical scalar indicating whether an outgroup should be inserted to split unrelated trajectories. Alternatively, a numeric scalar specifying the distance threshold to use for this splitting.
`outscale`	A numeric scalar specifying the scaling to apply to the median distance between centroids to define the threshold for outgroup splitting. Only used if `outgroup=TRUE`.
`columns`	A character, logical or integer vector specifying the columns of `x` to use. If `NULL`, all provided columns are used by default.
`with.mnn`	A logical scalar indicating whether to use distances computed from mutual nearest neighbor pairs, see Details.
`mnn.k`	An integer scalar specifying the number of nearest neighbors to consider for the MNN-based distance calculation. See `findMutualNN` for more details.
`BNPARAM`	A BiocNeighborParam object specifying how the nearest-neighbor search should be performed when `with.mnn=TRUE`, see the **BiocNeighbors** package for more details.
`BPPARAM`	A BiocParallelParam object specifying whether the nearest neighbor search should be parallelized when `with.mnn=TRUE`, see the **BiocNeighbors** package for more details.
`assay.type`	An integer or string specifying the assay to use from a SummarizedExperiment `x`.
`use.dimred`	An integer or string specifying the reduced dimensions to use from a SingleCellExperiment `x`.

Value

A graph object containing an MST computed on `centers`. Each node corresponds to a cluster centroid and has a numeric vector of coordinates in the `coordinates` attribute. The edge weight is set to the Euclidean distance and the confidence is stored as the `gain` attribute.

Introducing an outgroup

If `outgroup=TRUE`, we add an outgroup to avoid constructing a trajectory between "unrelated" clusters. This is done by adding an extra row/column to the distance matrix corresponding to an artificial outgroup cluster, where the distance to all of the other real clusters is set to $\omega/2$. Large jumps in the MST between real clusters that are more distant than ω will then be rerouted through the outgroup, allowing us to break up the MST into multiple subcomponents by removing the outgroup.

The default ω value is computed by constructing the MST from the original distance matrix, computing the median edge length in that MST, and then scaling it by `outscale`. This adapts to the magnitude of the distances and the internal structure of the dataset while also providing some margin for variation across cluster pairs. Alternatively, `outgroup` can be set to a numeric scalar in which case it is used directly as ω.

Confidence on the edges

For the MST, we obtain a measure of the confidence in each edge by computing the distance gained if that edge were not present. Ambiguous parts of the tree will be less penalized from deletion of an edge, manifesting as a small distance gain. In contrast, parts of the tree with clear structure will receive a large distance gain upon deletion of an obvious edge.

For each edge, we divide the distance gain by the length of the edge to normalize for cluster resolution. This avoids overly penalizing edges in parts of the tree involving broad clusters while still retaining sensitivity to detect distance gain in overclustered regions. As an example, a normalized gain of unity for a particular edge means that its removal requires an alternative path that increases the distance travelled by that edge's length.

The normalized gain is reported as the `"gain"` attribute in the edges of the MST from `createClusterMST`. Note that the `"weight"` attribute represents the edge length.

An alternative distance calculation

While distances between centroids are usually satisfactory for gauging cluster "closeness", they do not consider the behavior at the boundaries of the clusters. Two clusters that are immediately adjacent (i.e., intermingling at the boundaries) may have a large distance between their centroids if the clusters themselves span a large region of the coordinate space. This may preclude the obvious edge from forming in the MST.

In such cases, we can use an alternative distance calculation based on the distance between mutual nearest neighbors (MNNs). An MNN pair is defined as two cells in separate clusters that are each other's nearest neighbors in the other cluster. For each pair of clusters, we identify all MNN pairs and compute the median distance between them. This distance is then used in place of the distance between centroids to construct the MST. In this manner, we focus on cluster pairs that are close at their boundaries rather than at their centers.

This mode can be enabled by setting `with.mnn=TRUE`, while the stringency of the MNN definition can be set with `mnn.k`. Similarly, the performance of the nearest neighbor search can be controlled with `BPPARAM` and `BSPARAM`. Note that this mode performs a cell-based search and so cannot be used when `x` already contains aggregated profiles.

Author(s)

Aaron Lun

References

Ji Z and Ji H (2016). TSCAN: Pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. *Nucleic Acids Res.* 44, e117

Examples

```
# Mocking up a Y-shaped trajectory.
centers <- rbind(c(0,0), c(0, -1), c(1, 1), c(-1, 1))
rownames(centers) <- seq_len(nrow(centers))
clusters <- sample(nrow(centers), 1000, replace=TRUE)
cells <- centers[clusters,]
cells <- cells + rnorm(length(cells), sd=0.5)

# Creating the MST:
mst <- createClusterMST(cells, clusters)
plot(mst)

# We could also do it on the centers:
mst2 <- createClusterMST(centers, clusters=NULL)
plot(mst2)

# Works if the expression matrix is in a SE:
library(SummarizedExperiment)
se <- SummarizedExperiment(t(cells), colData=DataFrame(group=clusters))
mst3 <- createClusterMST(se, se$group, assay.type=1)
plot(mst3)
```

`difftest` *difftest*

Description

testing differentially expressed genes

Usage

```
difftest(data, TSCANorder, df = 3)
```

Arguments

`data`	The raw single_cell data, which is a numeric matrix or data.frame. Rows represent genes/features and columns represent single cells.
`TSCANorder`	The TSCAN ordering generated by function `TSCANorder`.
`df`	Numeric value specifying the degree of freedom used in the GAM model.

Details

This function tests whether a gene is significantly expressed given pseudotime ordering. Likelihood ratio test is performed to compare a generalized additive model (GAM) with a constant fit to get the p-values. The p-values are adjusted for multiple testing by fdr.

Value

Data frame containing pvalues and qvalues of testing differentially expression.

Author(s)

Zhicheng Ji, Hongkai Ji <zji4@zji4.edu>

Examples

```
data(lpsdata)
procdata <- preprocess(lpsdata)
lpsorder <- TSCANorder(exprmclust(procdata))
diffval <- difftest(procdata,lpsorder)
#Selected differentially expressed genes under qvlue cutoff of 0.05
row.names(diffval)[diffval$qval < 0.05]
```

`exprmclust` *exprmclust*

Description

Perform model-based clustering on expression values

Usage

```
exprmclust(data, clusternum = 2:9, modelNames = "VVV", reduce = T)
```

Arguments

`data`	The raw single_cell data, which is a numeric matrix or data.frame. Rows represent genes/features and columns represent single cells.
`clusternum`	An integer vector specifying all possible cluster numbers. The best cluster number will be picked using BIC. The minimum value should be two other
`modelNames`	model to be used in model-based clustering. By default "ellipsoidal, varying volume, shape, and orientation" is used.
`reduce`	Whether to perform the PCA on the expression data.

Details

By default, this function first uses principal component analysis (PCA) to reduce dimensionality of original data. It then performs model-based clustering on the transformed expression values. A minimum-spanning-tree is constructed to link the cluster centers. The clustering results will be used for TSCAN ordering.

Value

if more than one cluster detected, a list containing

- pcareduceres Numeric matrix containing the transformed expression values after PCA.
- MSTtree igraph object which is the result of constructing MST.
- clusterid A named vector specifying which cluster the cells belong to.
- clucenter Numeric matrix of the cluster centers.

if only one cluster detected, a list containing

- pcareduceres Numeric matrix containing the transformed expression values after PCA.

Author(s)

Zhicheng Ji, Hongkai Ji <zji4@zji4.edu>

References

Fraley, C., & Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. Journal of the American Statistical Association, 97(458), 611-631.

Examples

```
data(lpsdata)
procdata <- preprocess(lpsdata)
exprmclust(procdata)
```

`lpsdata` *Sinlge-cell RNA-seq data for BMDC cells before and after LPS stimulation*

Description

The dataset contains 16776 rows and 131 columns. Each row represent a gene and each column represent a single cell. This dataset is a subset of single-cell RNA-seq data provided by GEO GSE48968. Only unstimulated cells and cells after 6h of LPS stimulation are retained for the purpose of demonstration. Genes which have raw expression values of greater than zero in at least one cell are retained. For the original dataset please refer to GSE48968 on GEO (http://www.ncbi.nlm.nih.gov/geo/query/acc.

Format

A matrix with 16776 rows and 131 variables

Source

http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE48968

References

Shalek, A. K., Satija, R., Shuga, J., Trombetta, J. J., Gennert, D., Lu, D., ... & Regev, A. (2014). Single-cell RNA-seq reveals dynamic paracrine control of cellular variation. Nature.

`mapCellsToEdges` *Map cells to edges*

Description

Map each cell to the closest edge on the MST, reporting also the distance to the corresponding vertices.

Usage

```
mapCellsToEdges(x, ...)

## S4 method for signature 'ANY'
mapCellsToEdges(x, mst, clusters, columns = NULL)

## S4 method for signature 'SummarizedExperiment'
mapCellsToEdges(x, ..., assay.type = "logcounts")

## S4 method for signature 'SingleCellExperiment'
mapCellsToEdges(
  x,
  clusters = colLabels(x, onAbsence = "error"),
  ...,
  use.dimred = NULL
)
```

Arguments

`x`	A numeric matrix of coordinates where each row represents a cell/sample and each column represents a dimension (usually a PC or another low-dimensional embedding, but features or genes can also be used). Alternatively, a SummarizedExperiment or SingleCellExperiment object containing such a matrix in its `assays`, as specified by `assay.type`. This will be transposed prior to use. Alternatively, for SingleCellExperiments, this matrix may be extracted from its `reducedDims`, based on the `use.dimred` specification. In this case, no transposition is performed.
`...`	For the generic, further arguments to pass to the specific methods. For the SummarizedExperiment method, further arguments to pass to the ANY method. For the SingleCellExperiment method, further arguments to pass to the SummarizedExperiment method (if `use.dimred` is specified) or the ANY method (otherwise).
`mst`	A graph object containing a MST, constructed from the same coordinate space as the values in `x` (e.g., same PC space, same set of features).
`clusters`	A factor-like object of the same length as `nrow(x)`, specifying the cluster identity for each cell in `x`. This can also be `NULL`, see details.
`columns`	A character, logical or integer vector specifying the columns of `x` to use. If `NULL`, all provided columns are used by default.

`assay.type`	An integer or string specifying the assay to use from a SummarizedExperiment `x`.
`use.dimred`	An integer or string specifying the reduced dimensions to use from a SingleCell-Experiment `x`.

Details

For each cluster, we consider all edges of the MST involving that cluster. Each cell of that cluster is then mapped to the closest of these edges (where proximity is defined by Euclidean distance). The identity of and distance from each ends of the edge is reported; this can be useful for downstream pseudo-time calculations or to subset cells by those lying on a particular edge.

If `clusters=NULL`, each cell can be mapped to *any* edge of the MST. This is useful if the `mst` was constructed from a different set of cells than those in `x`, allowing us to effectively project new datasets onto an existing MST. Note, however, that the new `x` must lie in the same coordinate space as the `x` used to make `mst`.

Some cells may simply be mapped to the edge endpoints. This manifests as values of zero for the distances from either end of the edge. For analyses focusing on a specific edge, it may be advisable to filter out such cells as their edge assignments are arbitrarily assigned and they do not contribute to any transitional process along the edge.

Value

A DataFrame with one row per row of `x`, containing the fields:

- `left.cluster`, the cluster on one end of the edge to which the cell was assigned.
- `right.cluster`, the cluster on the other end of the edge to which the cell was assigned.
- `left.distance`, the distance to the cluster centroid on one end.
- `right.distance`, the distance to the cluster centroid on the other end.

Note that the sum of the distances will always yield the edge length.

Author(s)

Aaron Lun

References

Ji Z and Ji H (2016). TSCAN: Pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. *Nucleic Acids Res.* 44, e117

See Also

`createClusterMST`, to generate `mst`.

`quickPseudotime`, a wrapper to quickly perform these calculations.

Examples

```
# Mocking up a Y-shaped trajectory.
centers <- rbind(c(0,0), c(0, -1), c(1, 1), c(-1, 1))
rownames(centers) <- seq_len(nrow(centers))
clusters <- sample(nrow(centers), 1000, replace=TRUE)
cells <- centers[clusters,]
cells <- cells + rnorm(length(cells), sd=0.5)
```

```
# Creating the MST first:
mst <- createClusterMST(cells, clusters=clusters)
plot(mst)

# Mapping cells to the MST:
mapping <- mapCellsToEdges(cells, mst, clusters=clusters)
head(mapping)

# Also works with some random extra cells:
extras <- matrix(rnorm(1000), ncol=2)
emapping <- mapCellsToEdges(extras, mst, clusters=NULL)
head(emapping)
```

`orderCells` *Compute pseudotimes from the MST*

Description

Compute a pseudotime for each cell lying on each path through the MST from a given starting node.

Usage

```
orderCells(mapping, mst, start = NULL)
```

Arguments

`mapping`	A DataFrame of MST-mapping information for each cell, usually obtained by running `mapCellsToEdges` with the per-cell coordinate matrix and `mst`.
`mst`	A graph object containing a MST from `createClusterMST`. This need not be generated from the same cells in `mapping`.
`start`	A character vector specifying the starting node from which to compute pseudotimes in each component of `mst`. Defaults to an arbitrarily chosen node of degree 1 or lower in each component.

Details

The pseudotimes are returned as a matrix where each row corresponds to cell in `x` and each column corresponds to a path through the MST from `start` to all nodes of degree 1. (If `start` is itself a node of degree 1, then paths are only considered to all other such nodes.) This format is inspired by that from the **slingshot** package and provides a compact representation of branching events.

Each branching event in the MST results in a new path and thus a new column in the pseudotime matrix. An `NA` entry for a cell indicates that it is not assigned to that particular path. All non-`NA` entries for any given cell are guaranteed to be identical. This reflects the fact that multiple paths will share a section of the MST for which the pseudotimes are the same.

The starting node in `start` is *completely arbitrarily chosen* by `orderClusterMST`, as directionality is impossible to infer from the expression matrix alone. However, it is often possible to use prior biological knowledge to pick an appropriate cluster as the starting node.

Value

A numeric matrix containing the pseudotimes of all cells (rows) across all paths (columns) through `mst`.

Author(s)

Aaron Lun

References

Ji Z and Ji H (2016). TSCAN: Pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. *Nucleic Acids Res.* 44, e117

See Also

`mapCellsToEdges`, to compute `mapping`.

`quickPseudotime`, a wrapper to quickly perform these calculations.

Examples

```
# Mocking up a Y-shaped trajectory.
centers <- rbind(c(0,0), c(0, -1), c(1, 1), c(-1, 1))
rownames(centers) <- seq_len(nrow(centers))
clusters <- sample(nrow(centers), 1000, replace=TRUE)
cells <- centers[clusters,]
cells <- cells + rnorm(length(cells), sd=0.5)

# Creating the MST and mapping the cells.
mst <- createClusterMST(cells, clusters=clusters)
mapping <- mapCellsToEdges(cells, mst, clusters=clusters)

# Obtaining pseudo-time orderings.
ordering <- orderCells(mapping, mst)
unified <- rowMeans(ordering, na.rm=TRUE)
plot(cells[,1], cells[,2], col=topo.colors(21)[cut(unified, 21)], pch=16)
```

`orderscore` *orderscore*

Description

Calculate pseudotemporal ordering scores for orders

Usage

```
orderscore(subpopulation, orders)
```

Arguments

`subpopulation`	Data frame with two columns. First column: cell names. Second column: subpopulation codes.
`orders`	A list with various length containing pseudotime orderings.

Details

This function calculates pseudotemporal ordering scores (POS) based on the sub-population information and order information given by users. Cells should come from at least two cell subpopulations. These sub-population should be coded as 0,1,2,...

Value

a numeric vector of calculated POS.

Author(s)

Zhicheng Ji, Hongkai Ji <zji4@zji4.edu>

Examples

```
data(lpsdata)
procdata <- preprocess(lpsdata)
subpopulation <- data.frame(cell = colnames(procdata), sub = ifelse(grepl("Unstimulated",colnames(procdata)),
lpsmclust <- exprmclust(procdata)
#Comparing default TSCAN ordering and tuned TSCAN ordering
order1 <- TSCANorder(lpsmclust)
order2 <- TSCANorder(lpsmclust, c(1,2,3))
orders <- list(order1,order2)
orderscore(subpopulation, orders)
```

`perCellEntropy` *Compute the per-cell entropy*

Description

Compute the entropy of each cell, using this as a proxy for the differentiation status.

Usage

```
perCellEntropy(x, ...)

## S4 method for signature 'ANY'
perCellEntropy(x, BPPARAM = NULL)

## S4 method for signature 'SummarizedExperiment'
perCellEntropy(x, ..., assay.type = "counts")
```

Arguments

`x`	A numeric matrix-like object containing counts for each cell (column) and feature (row). Alternatively, a SummarizedExperiment object containing such a matrix.
`...`	For the generic, further arguments to pass to specific methods. For the SummarizedExperiment method, further arguments to pass to the ANY method.

`BPPARAM`	A BiocParallelParam object from **BiocParallel**, specifying how calculations should be parallelized.
`assay.type`	An integer or string specifying the assay to use from a SummarizedExperiment `x`.

Details

Entropy values are computed from the proportion of counts assigned to each feature within a given cell. The central idea is that undifferentiated cells have higher entropies because they are not yet committed to a single lineage, and thus have low but persistent activity of the transcriptional programs for all lineages. The cluster with the highest entropy values can be used to determine the `start` cluster in `orderCells`.

Value

A numeric vector of entropies for all cells in `x`. Cells with all-zero values in `x` will be assigned `NA` entropies.

Author(s)

Aaron Lun

References

Grun D et al. (2016). De novo prediction of stem cell identity using single-cell transcriptome data. *Cell Stem Cell* 19, 266-77

Gulati GS et al. (2020). Single-cell transcriptional diversity is a hallmark of developmental potential. *Science* 367, 405-11

Guo M et al. (2017) SLICE: determining cell differentiation and lineage based on single cell entropy. *Nucleic Acids Res.* 45, e54

Examples

```
sce <- scuttle::mockSCE()
ent <- perCellEntropy(sce)
summary(ent)

# Compute average entropy over mock clusters.
clusters <- sample(ncol(sce), 5)
by.cluster <- split(ent, clusters)
mean.cluster.ent <- vapply(by.cluster, mean, 0)
```

`plotmclust` *plotmclust*

Description

Plot the model-based clustering results

Usage

```
plotmclust(
  mclustobj,
  x = 1,
  y = 2,
  MSTorder = NULL,
  show_tree = T,
  show_cell_names = T,
  cell_name_size = 3,
  markerexpr = NULL
)
```

Arguments

`mclustobj`	The exact output of `exprmclust` function.
`x`	The column of data after dimension reduction to be plotted on the horizontal axis.
`y`	The column of data after dimension reduction to be plotted on the vertical axis.
`MSTorder`	The arbitrary order of cluster to be shown on the plot.
`show_tree`	Whether to show the links between cells connected in the minimum spanning tree.
`show_cell_names`	Whether to draw the name of each cell in the plot.
`cell_name_size`	The size of cell name labels if show_cell_names is TRUE.
`markerexpr`	The gene expression used to define the size of nodes.

Details

This function will plot the gene expression data after dimension reduction and show the clustering results.

Value

A ggplot2 object.

Author(s)

Zhicheng Ji, Hongkai Ji <zji4@zji4.edu>

Examples

```
data(lpsdata)
procdata <- preprocess(lpsdata)
lpsmclust <- exprmclust(procdata)
plotmclust(lpsmclust)
```

preprocess *preprocess*

Description

preprocess the raw single-cell data

Usage

```
preprocess(
  data,
  clusternum = NULL,
  takelog = TRUE,
  logbase = 2,
  pseudocount = 1,
  minexpr_value = 1,
  minexpr_percent = 0.5,
  cvcutoff = 1
)
```

Arguments

`data`	The raw single_cell data, which is a numeric matrix or data.frame. Rows represent genes/features and columns represent single cells.
`clusternum`	The number of clusters for doing cluster, typically 5 percent of number of all genes. The clustering will be done after all the transformation and trimming. If NULL no clustering will be performed.
`takelog`	Logical value indicating whether to take logarithm
`logbase`	Numeric value specifiying base of logarithm
`pseudocount`	Numeric value to be added to the raw data when taking logarithm
`minexpr_value`	Numeric value specifying the minimum cutoff of log transformed (if takelog is TRUE) value
`minexpr_percent`	Numeric value specifying the lowest percentage of highly expressed cells (expression value bigger than minexpr_value) for the genes/features to be retained.
`cvcutoff`	Numeric value specifying the minimum value of coefficient of variance for the genes/features to be retained.

Details

This function first takes logarithm of the raw data and then filters out genes/features in which too many cells are low expressed. It also filters out genes/features with low coefficient of variance which indicates the genes/features does not contain much information. The default setting will first take log2 of the raw data after adding a pseudocount of 1. Then genes/features in which at least half of cells have expression values are greater than 1 and the coefficeints of variance across all cells are at least 1 are retained.

Value

Matrix or data frame with the same format as the input dataset.

Author(s)

Zhicheng Ji, Hongkai Ji <zji4@zji4.edu>

Examples

```
data(lpsdata)
procdata <- preprocess(lpsdata)
```

---

quickPseudotime *Quick MST-based pseudotime*

Description

A convenience wrapper to quickly compute a minimum spanning tree (MST) on the cluster centroids to obtain a pseudotime ordering of the cells.

Usage

```
quickPseudotime(x, ...)

## S4 method for signature 'ANY'
quickPseudotime(x, clusters, others = NULL, ..., start = NULL, columns = NULL)

## S4 method for signature 'SummarizedExperiment'
quickPseudotime(x, ..., assay.type = "logcounts")

## S4 method for signature 'SingleCellExperiment'
quickPseudotime(
  x,
  clusters = colLabels(x, onAbsence = "error"),
  ...,
  others = NULL,
  use.dimred = NULL,
  other.dimreds = TRUE
)
```

Arguments

`x`	A numeric matrix of coordinates where each row represents a cell/sample and each column represents a dimension (usually a PC or another low-dimensional embedding, but features or genes can also be used). Alternatively, a SummarizedExperiment or SingleCellExperiment object containing such a matrix in its `assays`, as specified by `assay.type`. This will be transposed prior to use. Alternatively, for SingleCellExperiments, this matrix may be extracted from its `reducedDims`, based on the `use.dimred` specification. In this case, no transposition is performed.
`...`	For the generic, further arguments to pass to the specific methods. For the ANY method, further arguments to pass to `createClusterMST`.

	For the SummarizedExperiment method, further arguments to pass to the ANY method. For the SingleCellExperiment method, further arguments to pass to the SummarizedExperiment method (if `use.dimred` is specified) or the ANY method (otherwise).
`clusters`	A vector or factor of length equal to the number of cells in `x`, specifying the cluster assignment for each cell.
`others`	List of numeric matrices with the same number of rows as `x`, to be passed to `reportEdges`. This typically contains dimensionality reduction results, for use in visualizing the edges of the MST. If `NULL`, defaults to a list containing `x`.
`start`	Arguments passed to `orderCells`.
`columns`	A character, logical or integer vector specifying the columns of `x` to use. If `NULL`, all provided columns are used by default.
`assay.type`	An integer or string specifying the assay to use from a SummarizedExperiment `x`.
`use.dimred`	An integer or string specifying the reduced dimensions to use from a SingleCell-Experiment `x`.
`other.dimreds`	Logical scalar indicating whether all dimensionality reduction results in `x` should be appended onto the `others` list.

Details

This function simply calls, in order:

- `rowmean`, to compute the average low-dimensional coordinates for each cluster.
- `createClusterMST` on the average coordinates created from `x`.
- `reportEdges` on the average coordinates for all entries of `other`.
- `mapCellsToEdges` on the per-cell coordinates in `x` with the constructed MST.
- `orderCells` on the mappings generated from `x` onto the MST.

Value

A List containing:

- `centered`, a list of numeric matrices containing the averaged coordinates for each cluster. Each matrix corresponds to a dimensionality reduction result in `x`.
- `mst`, a graph object containing the cluster-level MST computed on the coordinates from `use`.
- `ordering`, a numeric matrix of pseudotimes for various paths through the MST computed from `use`.
- `connected`, a list of data.frames containing the edge coordinates between centers. Each data.frame corresponds to a dimensionality reduction result in `x`.

Author(s)

Aaron Lun

See Also

`createClusterMST` and friends, for the functions that do the actual work.

Examples

```
# Mocking up an SCE object:
ncells <- 100
u <- matrix(rpois(20000, 5), ncol=ncells)
pca <- matrix(runif(ncells*5), ncells)
tsne <- matrix(rnorm(ncells*2), ncells)

library(SingleCellExperiment)
sce <- SingleCellExperiment(assays=list(counts=u),
    reducedDims=SimpleList(PCA=pca, tSNE=tsne))

# Clustering on our pretend PCA values:
clusters <- kmeans(pca, 3)$cluster

# Quickly computing the pseudotime:
out <- quickPseudotime(sce, clusters, use.dimred="PCA")
out$mst
head(out$ordering)
```

`reportEdges` *Report MST edge coordinates*

Description

Provides the coordinates of the start and end of every edge in the MST, possibly on a different coordinate space from that used to construct the MST. This is mostly useful for plotting purposes in `segments` or the equivalent **ggplot2** functionality.

Usage

```
reportEdges(x, ...)

## S4 method for signature 'ANY'
reportEdges(x, mst, clusters, combined = TRUE, columns = NULL)

## S4 method for signature 'SummarizedExperiment'
reportEdges(x, ..., assay.type = "logcounts")

## S4 method for signature 'SingleCellExperiment'
reportEdges(
  x,
  clusters = colLabels(x, onAbsence = "error"),
  ...,
  use.dimred = NULL
)
```

Arguments

`x`	A numeric matrix of coordinates where each row represents a cell/sample and each column represents a dimension (usually a PC or another low-dimensional embedding, but features or genes can also be used).

	Alternatively, a SummarizedExperiment or SingleCellExperiment object containing such a matrix in its `assays`, as specified by `assay.type`. This will be transposed prior to use. Alternatively, for SingleCellExperiments, this matrix may be extracted from its `reducedDims`, based on the `use.dimred` specification. In this case, no transposition is performed. Alternatively, if `clusters=NULL`, a numeric matrix of coordinates for cluster centroids, where each row represents a cluster and each column represents a dimension Each row should be named with the cluster name. This mode can also be used with assays/matrices extracted from SummarizedExperiments and SingleCellExperiments.
`...`	For the generic, further arguments to pass to the specific methods. For the SummarizedExperiment method, further arguments to pass to the ANY method. For the SingleCellExperiment method, further arguments to pass to the SummarizedExperiment method (if `use.dimred` is specified) or the ANY method (otherwise).
`mst`	A graph object containing a MST, typically the output of `createClusterMST`. This need not be constructed from the same coordinates as those in `x`.
`clusters`	A factor-like object of the same length as `nrow(x)`, specifying the cluster identity for each cell in `x`. If `NULL`, `x` is assumed to already contain coordinates for the cluster centroids.
`combined`	Logical scalar indicating whether a single data.frame of edge coordinates should be returned.
`columns`	A character, logical or integer vector specifying the columns of `x` to use. If `NULL`, all provided columns are used by default.
`assay.type`	An integer or string specifying the assay to use from a SummarizedExperiment `x`.
`use.dimred`	An integer or string specifying the reduced dimensions to use from a SingleCellExperiment `x`.

Details

It is entirely possibly to supply, say, t-SNE coordinates in `x` along with a MST constructed from the PCA coordinates. This allows us to visualize the edges of the MST on other low-dimensional embeddings. The coordinates in `x` can be per-cell or, if `clusters=NULL`, they are assumed to already be per-cluster means. `x` may also be `NULL`, in which case the center coordinates are obtained from the `coordinates` vertex attribute of `mst`.

Value

A data.frame containing the start and end coordinates of segments representing all the edges in `mst`. If `combined=FALSE`, a list of two data.frames is returned where corresponding rows represent the start and end coordinates of the same edge.

Author(s)

Aaron Lun

References

Ji Z and Ji H (2016). TSCAN: Pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. *Nucleic Acids Res.* 44, e117

See Also

`createClusterMST`, to generate `mst`.

`quickPseudotime`, a wrapper to quickly perform these calculations.

Examples

```
# Re-using the example from ?createClusterMST.
example(createClusterMST, ask=FALSE)

# Plotting the MST on top of existing visualizations:
edges <- reportEdges(x=NULL, mst, combined=FALSE)
plot(cells[,1], cells[,2], col=clusters)
segments(edges$start$dim1, edges$start$dim2, edges$end$dim1,
     edges$end$dim2, lwd=5)

# Use with coordinates other than those used to make the MST:
shifted.cells <- cells + 10

shift.edges <- reportEdges(shifted.cells, mst,
    clusters=clusters, combined=FALSE)
plot(shifted.cells[,1], shifted.cells[,2], col=clusters)
segments(shift.edges$start$dim1, shift.edges$start$dim2,
    shift.edges$end$dim1, shift.edges$end$dim2, lwd=5)

# Also works for ggplot2:
df <- data.frame(shifted.cells, cluster=factor(clusters))
shift.edges2 <- reportEdges(shifted.cells, mst, clusters=clusters)

library(ggplot2)
ggplot(df) +
   geom_point(aes(x=X1, y=X2, color=cluster)) +
   geom_line(data=shift.edges2, mapping=aes(x=dim1, y=dim2, group=edge))
```

`rowmean` *Compute column means of a matrix based on a grouping variable*

Description

A utility that is equivalent to `rowsum`, but computes the mean instead of the sum of each subset of rows.

Usage

```
rowmean(x, group, ...)
```

Arguments

`x`	A numeric matrix or matrix-like object.
`group`	A vector or factor specifying the group assignment for each row of `x`.
`...`	Further arguments to pass to `rowsum`, e.g., `reorder`.

Value

A numeric matrix with one row per level of `group`, where the value for each column contains the average value across the subset of rows corresponding that level.

Author(s)

Aaron Lun

Examples

```
x <- matrix(runif(100), ncol = 5)
group <- sample(1:8, 20, TRUE)
(xmean <- rowmean(x, group))
```

`singlegeneplot` *singlegeneplot*

Description

plot expression values of individual genes against pseudotime axis

Usage

```
singlegeneplot(geneexpr, TSCANorder, cell_size = 2)
```

Arguments

`geneexpr`	The gene expression values. Names should agree with the pseudotime information.
`TSCANorder`	The output of function `TSCANorder`.
`cell_size`	Size of cells in the plot.

Details

This function plots the expression values of individual genes against given pseudotime

Value

ggplot2 object.

Author(s)

Zhicheng Ji, Hongkai Ji <zji4@zji4.edu>

Examples

```
data(lpsdata)
procdata <- preprocess(lpsdata)
lpsmclust <- exprmclust(procdata)
lpsorder <- TSCANorder(lpsmclust,orderonly=FALSE,flip=TRUE)
#Choose STAT1 gene expression to plot
STAT2expr <- log2(lpsdata["STAT2",]+1)
singlegeneplot(STAT2expr, lpsorder)
```

`testPseudotime` *Test for differences along pseudotime*

Description

Implements a simple method of testing for significant differences with respect to pseudotime, based on fitting linear models with a spline basis matrix.

Usage

```
testPseudotime(x, ...)

## S4 method for signature 'ANY'
testPseudotime(
  x,
  pseudotime,
  df = 5,
  get.lfc = TRUE,
  get.spline.coef = FALSE,
  trend.only = TRUE,
  block = NULL,
  BPPARAM = NULL
)

## S4 method for signature 'SummarizedExperiment'
testPseudotime(x, ..., assay.type = "logcounts")
```

Arguments

`x`	A numeric matrix-like object containing log-expression values for cells (columns) and genes (rows). Alternatively, a SummarizedExperiment containing such a matrix.
`...`	For the generic, further arguments to pass to specific method. For the SummarizedExperiment method, further arguments to pass to the ANY method.
`pseudotime`	A numeric vector of length equal to the number of columns of `x`.
`df`	Integer scalar specifying the degrees of freedom for the splines.
`get.lfc`	Logical scalar indicating whether to return an overall log-fold change along each path.

`get.spline.coef`	Logical scalar indicating whether to return the estimates of the spline coefficients.
`trend.only`	Logical scalar indicating whether only differences in the trend should be considered when testing for differences between paths.
`block`	Factor of length equal to the number of cells in `x`, specifying the blocking factor.
`BPPARAM`	A BiocParallelParam object from the **BiocParallel** package, used to control parallelization.
`assay.type`	String or integer scalar specifying the assay containing the log-expression matrix.

Details

Tis function fits a natural spline to the expression of each gene with respect to pseudotime. It then does an ANOVA to test whether any of the spline coefficients are non-zero. In this manner, genes exhibiting a significant (and potentially non-linear) trend with respect to the pseudotime can be detected as those with low p-values.

For trajectories with multiple paths, only one path should be tested at a time. This usually involves passing a single column of the matrix returned from `orderCells`. Cells with `NA` values in `pseudotime` are assumed to be assigned to a different path and are ignored.

By default, estimates of the spline coefficients are not returned as they are difficult to interpret. Rather, a log-fold change of expression along each path is estimated to provide some indication of the overall magnitude and direction of any change.

`block` can be used to fit a separate linear model to each of multiple batches, after which the statistics are combined across batches as described in `testLinearModel`. This avoids potential confounding effects from batch-specific differences in the distribution of cells across pseudotime.

Value

A DataFrame is returned containing the statistics for each gene (row), including the p-value and its BH-adjusted equivalent. If `get.lfc=TRUE`, an overall log-fold change is returned for each path.

If `get.spline.coef=TRUE`, the estimated spline coefficients are also returned (single path) or the differences in the spline fits to the first path are returned (multiple paths).

Author(s)

Aaron Lun

See Also

`orderCells`, to generate the pseudotime matrix.

`testLinearModel`, which performs the tests under the hood.

Examples

```
y <- matrix(rnorm(10000), ncol=100)
u <- runif(100)
testPseudotime(y, u)

# Handling a blocking factor.
b <- gl(2, 50)
```

```
testPseudotime(y, u, block=b)
```

TSCAN *TSCAN: Tools for Single-Cell ANalysis*

Description

This package provides essential tools used in analyzing data from single-cell experiments

Details

TSCAN enables users to easily construct and tune pseudotemporal cell ordering as well as analyzing differentially expressed genes. TSCAN comes with a user-friendly GUI written in shiny. More functions will come in the future.

TSCANorder *TSCANorder*

Description

Construct TSCAN order after exprmclust

Usage

```
TSCANorder(mclustobj, MSTorder = NULL, orderonly = T, flip = F, listbranch = F)
```

Arguments

`mclustobj`	The exact output of the `exprmclust` function.
`MSTorder`	A numeric vector specifying the order of clusters.
`orderonly`	Only return the ordering. State or pseudotime information will not be returned
`flip`	whether to flip the ordering
`listbranch`	whether to list the ordering results of all possible branches in MST. Only works if MSTorder in NULL.

Details

This function takes the exact output of exprmclust function and construct TSCAN order by mapping all cells onto the path that connects cluster centers. Users can also specify their own path.

Value

if orderonly = F, a vector of ordered cell names. if orderonly = T, a data frame of ordered cell names, cell states and pseudotime.

Author(s)

Zhicheng Ji, Hongkai Ji <zji4@zji4.edu>

Examples

```
data(lpsdata)
procdata <- preprocess(lpsdata)
lpsmclust <- exprmclust(procdata)
TSCANorder(lpsmclust)
```

`TSCANui` *TSCANui*

Description

Launch the TSCAN user interface in local machine

Usage

```
TSCANui()
```

Details

This function will automatically launch the TSCAN user interface in a web browser. The user interface provides many powerful functions which is not available by command line programming. It also provides a much easier and more convenient way to quickly explore single cell data and construct pseudotime analysis. The user interface can also be accessed by http://zhiji.shinyapps.io/TSCAN. Neither R nor any packages are required in this online version. However, it is highly recommended that the user interface be launched locally for faster running speed.

Author(s)

Zhicheng Ji, Hongkai Ji <zji4@zji4.edu>

Examples

```
## Not run:
   TSCANui()

## End(Not run)
```

Index