Package ‘uSORT’

April 26, 2017

Title uSORT: A self-refining ordering pipeline for gene selection
Version 1.2.0
Author Mai Chan Lau, Hao Chen, Jinmiao Chen
Description This package is designed to uncover the intrinsic cell progression path from single-cell RNA-seq data. It incorporates data pre-processing, preliminary PCA gene selection, preliminary cell ordering, feature selection, refined cell ordering, and post-analysis interpretation and visualization.
Maintainer Hao Chen <chen_hao@immunol.a-star.edu.sg>
biocViews RNASeq, GUI, CellBiology, DNASEq
Depends R (>= 3.3.0), tcltk
VignetteBuilder knitr
Suggests knitr, RUnit, testthat, ggplot2
Imports igraph, Matrix, RANN, RSpectra, VGAM, gplots, parallel, plyr, methods, cluster, Biobase, fpc, BiocGenerics, monocle, grDevices, graphics, stats, utils
License Artistic-2.0
Encoding UTF-8
LazyData true
RoxygenNote 5.0.1
NeedsCompilation no

R topics documented:

autoSPIN ................................................................. 2
clusterGenesI ......................................................... 3
compareModelsI ...................................................... 4
differentialGeneTestI ............................................... 5
diff_test_helperI ................................................... 5
distance.function ................................................... 6
driving_force_gene_selection ....................................... 6
elbow_detection ..................................................... 7
EXP_to_CellDataSet .................................................. 8
fluidigmSC_analyzeGeneDetection ................................... 8
fluidigmSC_identifyExpOutliers .................................... 9
autoSPIN

A wrapper function for autoSPIN sorting method

Description

A wrapper function for autoSPIN method which implements optimized local refinement using the selected SPIN sorting method, i.e. STS or Neighborhood.

Usage

autoSPIN(data, data_type = c("linear", "cyclical"),
          sorting_method = c("STS", "neighborhood"), alpha = 0.2, sigma_width = 1,
          no_randomization = 20, window_perc_range = c(0.1, 0.9),
          window_size_incre_perct = 0.05)
clusterGenes1

Arguments

data An log2 transformed expression matrix containing n-rows of cells and m-cols of genes.
data_type A character string indicating the type of progression, i.e. 'linear' (strictly linear) or 'cyclical' (cyclically linear).
sorting_method A character string indicating the choice of SPIN sorting method, i.e. 'STS' (Side-to-Side) or 'Neighborhood'.
alpha A fraction value denoting the size of locality used for calculating the summed local variance.
sigma_width An integer number denoting the degree of spread of the gaussian distribution which is used for computing weight matrix for Neighborhood sorting method.
no_randomization An integer number indicating the number of repeated sorting, each of which uses randomly selected initial cell position.
window_perc_range A fraction value indicating the range of window size to be examined during local refinement.
window_size_incre_perc A fraction value indicating the step size at each iteration for incrementing window size.

Value
A data frame containing single column of ordered sample IDs.

Examples

```r
set.seed(15)
da <- iris[sample(150, 150, replace = FALSE), ]
rownames(da) <- paste0('spl', seq(1, nrow(da)))
d <- da[, 1:4]
dl <- da[, 5, drop = FALSE]
res <- autoSPIN(data = d)
dl <- dl[match(res$SampleID, rownames(dl)), ]
annot <- data.frame(id = seq(1, nrow(res)), label = dl, stringsAsFactors = FALSE)
# ggplot(annot, aes(x = id, y = id, colour = label)) + geom_point() + theme_bw()
```

clusterGenes1 A modified monocle’s function

Description
A modified monocle’s function for ’compareModels’ which identifies and removes genes whose reduced_models is better than full_models in term of likelihood

Usage

```r
clusterGenes1(expr_matrix, krange, method = function(x) { as.dist((1 - cor(t(x)))/2), ...)
```
**compareModels1**

**Arguments**

- `expr_matrix`: Expression matrix.
- `krange`: krange.
- `method`: method function.
- `...`: Other parameters.

**Value**

- `test_res`: A dataframe containing status of modeling and adjusted p-value

**Author(s)**

MaiChan Lau

---

**Description**

A modified monocle's function for 'compareModels' which identifies and removes genes whose reduced_models is better than full_models in term of likelihood

**Usage**

```r
compareModels1(full_models, reduced_models)
```

**Arguments**

- `full_models`: a Monocle's vgam full model
- `reduced_models`: a Monocle's vgam reduced/ null model

**Value**

- `test_res`: A dataframe containing status of modeling and adjusted p-value

**Author(s)**

MaiChan Lau
differentialGeneTest1  differential gene test

Description
modified from FludigmSC pacakge

Usage
differentialGeneTest1(cds,
    fullModelFormulaStr = "expression~sm.ns(Pseudotime, df=3)",
    reducedModelFormulaStr = "expression~1", cores = 1)

Arguments
    cds  Input object.
    fullModelFormulaStr  Full model formula.
    reducedModelFormulaStr  Reduced model formula.
    cores  Number of cores will be used.

Value
    test results

diff_test_helper1  A modified monocle's helper function

Description
A modified monocle's function for 'diff_test_helper1' which includes more attempts on finding models and also compute max. magnitude change in expression values predicted by GLM model

Usage
diff_test_helper1(x, fullModelFormulaStr, reducedModelFormulaStr,
    expressionFamily, lowerDetectionLimit = 0.1, type_ordering = "linear")

Arguments
    x  an expression data
    fullModelFormulaStr  a Monocle's model structure
    reducedModelFormulaStr  a Monocle's model structure
    expressionFamily  a Monocle's model structure
    lowerDetectionLimit  a Monocle's family character
    type_ordering  a threshold value
    type_ordering  a character indicating the type of underlying cell progression, i.e. linear or circular
driving_force_gene_selection

Value
d_test_res a dataframe containing status of modeling and adjusted p-value

Author(s)
MaiChan Lau

distance.function

Description
A distance function computes cell-to-cell distance matrix.

Usage
distance.function(expr, method = c("Euclidean", "Correlation", "eJaccard", "none"))

Arguments
expr
An expression matrix containing n-rows of cells and m-cols of genes.
method
A character string indicating the distance function.

Value
A matrix containing n-by-n cell distance.

driving_force_gene_selection

Description
A feature/ gene selection function (1) removes sparsely expressed genes, (2) identifies differentially expressed genes based on preliminary cell ordering, (3) removes highly dispersed genes from the identified DEGs, (4) further picks genes which are expected to have large expression difference on the 2 extreme ends of preliminary cell ordering.

Usage
driving_force_gene_selection(cds, scattering.cutoff.prob = 0.75, driving.force.cutoff = NULL, qval_cutoff = 0.05, min_expr = 0.1, data_type = c("linear", "cyclical"), nCores = 1)
elbow_detection

A elbow detection function

**Description**

A elbow detection function detects the elbow/knee of a given vector of values. Values will be sorted descendingly before detection, and the ID of those values above the elbow will be returned.

**Usage**

```r
elbow_detection(scores, if_plot = FALSE)
```

**Arguments**

- `scores`: A vector of numeric scores.
- `if_plot`: Boolean determine if plot the results.

**Value**

- a vector of selected elements IDs
**Examples**

```r
scores <- c(10, 9, 8, 6, 3, 2, 1, 0.1)
elbow_detection(scores, if_plot = TRUE)
```

---

**EXP_to_CellDataSet**  
*A function for constructing a Monocle’s CellDataSet object from an expression matrix*

---

**Description**

A function for constructing a Monocle’s CellDataSet object from an expression matrix

**Usage**

```
EXP_to_CellDataSet(log2_exp = NULL, expression_data_raw = NULL, lod = 1)
```

**Arguments**

- `log2_exp`: An log2 transformed expression matrix containing n-rows of cells and m-cols of genes.
- `expression_data_raw`: A data frame containing raw expression values, with rownames of cells and colnames of genes.
- `lod`: A value of limit of detection in the unit of TPM/CPM/RPKM.

**Value**

A CellDataSet object.

---

**fluidigmSC_analyzeGeneDetection**  
*A gene detection function*

---

**Description**

A gene detection function computes the fraction of genes detected in each cell, reproduced from FluidigmSC package.

**Usage**

```
fluidigmSC_analyzeGeneDetection(expression_data, threshold = 1)
```

**Arguments**

- `expression_data`: A data frame containing raw expression values, with rownames of genes and colnames of cells.
- `threshold`: A limit of detection in the unit of TPM/CPM/RPKM.
**fluidigmSC_identifyExpOutliers**

**Description**

An outlier detection function identifies cells with median expression below that of the bulk, reproduced from FluidigmSC package.

**Usage**

```r
fluidigmSC_identifyExpOutliers(log2ex_data, expression_data_raw, threshold, 
step, fine_step, num_fine_test, pct_goodsample_threshold = 0.5, 
quantile_threshold = 0.95, low_quantile_threshold = 0.25, 
min_gene_number = 25, lod)
```

**Arguments**

- `log2ex_data` A data frame containing log2 transformed expression values, with rownames of genes and colnames of cells.
- `expression_data_raw` A data frame containing raw expression values, with rownames of genes and colnames of cells.
- `threshold` A value in raw expression used as the starting threshold value.
- `step` An integer number indicating the increment of threshold value at each iteration.
- `fine_step` An integer number indicating the increment of threshold value at each iteration, at the refining stage.
- `num_fine_test` An integer number indicating the number of iteration of the refining stage.
- `pct_goodsample_threshold` A fraction value indicating the minimum percentage of samples on which the representative genes are detectable.
- `quantile_threshold` A probability of gene detection rate above which a sample is considered good sample.
- `low_quantile_threshold` A probability of average gene expression value below which a sample is taken as an outlier.
- `min_gene_number` An integer indicating the minimum size of representative genes.
- `lod` A value of limit of detection in the unit of TPM/CPM/RPKM.

**Value**

A vector of character stating the IDs of outlier cells.
A gene finding function looking for genes in the target set \( x \) from the source set \( y \), reproduced from FluidigmSC package.

Usage

```
fluidigmSC_isElementIgnoreCase(x, y, ignore_case = TRUE)
```

Arguments

- \( x \): A vector of characters representing gene names (target genes).
- \( y \): A vector of characters representing gene names (source genes).
- \( ignore_case \): Boolean, if TRUE ignores letter case.

Value

A vector of characters representing gene names.

An expression reading function which imports expression data from .txt file, and then computes log2 transformed data, reproduced from FluidigmSC package.

Usage

```
fluidigmSC_readLinearExp(exp_file = TRUE, lod = 1)
```

Arguments

- \( exp_file \): Input file name in txt format, with rownames of cells and colnames of genes.
- \( lod \): A value of limit of detection in the unit of TPM/CPM/RPKM. It will be used as the starting value for outlier cell detection and the basis for removing scarce genes.

Value

A list containing expression_data_raw(data frame), log2ex_data(data frame), and log2ex_avg_data(data frame).
**fluidigmSC_removeGenesByLinearExpForAllType**

*A gene trimming function*

**Description**
A gene trimming function removes genes whose average expression value is below the log2(threshold), and also present in at least 10

**Usage**
```
fluidigmSC_removeGenesByLinearExpForAllType(log2ex_data, log2ex_avg_data, threshold)
```

**Arguments**
- `log2ex_data`: A data frame containing log2 transformed expression values, with rownames of genes and colnames of cells.
- `log2ex_avg_data`: A data frame containing log2 transformed average expression values for individual gene.
- `threshold`: A limit of detection in the unit of TPM/CPM/RPMK.

**Value**
A vector of character containing gene names of those passed the filtering.

---

**fluidigmSC_removeGenesByLinearExpForAllType_log2**

*A gene trimming function*

**Description**
A gene trimming function removes genes whose average expression value is below the log2(threshold); reproduced from FluidigmSC package.

**Usage**
```
fluidigmSC_removeGenesByLinearExpForAllType_log2(log2ex_data, threshold)
```

**Arguments**
- `log2ex_data`: A data frame containing log2 transformed expression values, with rownames of genes and colnames of cells.
- `threshold`: A limit of detection in the unit of TPM/CPM/RPMK.

**Value**
A vector of character containing gene names of those passed the filtering.
monocle_wrapper  

A wrapper function for Monocle sorting method

Description

A wrapper function for Monocle sorting method

Usage

monocle_wrapper(log2_exp, expression_data_raw, lod = 1)

Arguments

log2_exp  
An log2 transformed expression matrix containing n-rows of cells and m-cols of genes.

expression_data_raw  
A data frame containing raw expression values, with rownames of cells and colnames of genes.

lod  
A value of limit of detection in the unit of TPM/CPM/RPKM.

Value

A data frame containing single column of ordered sample IDs.

Examples

set.seed(15)
da <- iris[sample(150, 150, replace = FALSE), ]rownames(da) <- paste0('spl_', seq(1, nrow(da))
d <- da[, 1:4]
dl <- da[, 5, drop = FALSE]
#res <- monocle_wrapper(log2_exp = d, expression_data_raw = d)
#dl <- dl[match(res, rownames(dl)), ]
#annot <- data.frame(id = seq(1, length(res)), label = dl, stringsAsFactors = FALSE)
#ggplot(annot, aes(x=id, y=id, colour = label)) + geom_point() + theme_bw()

neighborhood_sorting  

A sorting function using the Neighborhood algorithm

Description

A sorting function using the Neighborhood algorithm

Usage

neighborhood_sorting(d, weights_mat = NULL, max_iter = 100)
Arguments

\[
d \quad \text{A matrix containing n-by-n cell distance.}
\]
\[
\text{weights_mat} \quad \text{A weight matrix of size n-by-n.}
\]
\[
\text{max_iter} \quad \text{An integer number indicating the maximum number of iteration if sorting does not converge.}
\]

Value

A list containing ordering(a vector of re-ordered sequence) and cost(a numeric value).

Description

A cost computation function for Neighborhood algorithm

Usage

\[
\text{neighborhood_sortingcost}(\text{expr} = \text{NULL}, \text{sigma_width} = 1, \\
\text{method} = \text{c("Euclidean", "Correlation", "eJaccard", "none")})
\]

Arguments

\[
\text{expr} \quad \text{An expression matrix containing n-rows of cells and m-cols of genes.}
\]
\[
\text{sigma_width} \quad \text{An integer number determining the degree of spread of the gaussian distribution which is used for computing weight matrix for Neighborhood sorting method.}
\]
\[
\text{method} \quad \text{A character string indicating the distance function.}
\]

Value

A numeric value of sorting cost.

Examples

\[
\text{set.seed(15)}
\]
\[
da \leftarrow \text{iris[sample(150, 150, replace = FALSE), ]}
\]
\[
d \leftarrow \text{da[,] : 4}
\]
\[
\text{randomOrdering_cost} \leftarrow \text{neighborhood_sortingcost(d, method= 'Euclidean')}
\]
\[
\text{randomOrdering_cost}
\]
\[
da \leftarrow \text{iris}
\]
\[
d \leftarrow \text{da[,] : 4}
\]
\[
\text{properOrdering_cost} \leftarrow \text{neighborhood_sortingcost(d, method= 'Euclidean')}
\]
\[
\text{properOrdering_cost}
\]
neighborhood_sorting_wrapper

A wrapper function for Neighborhood sorting.

Description

A wrapper function for Neighborhood sorting as proposed in [Tsafrir et al. 2005].

Usage

neighborhood_sorting_wrapper(expr, sigma_width = 1, no_randomization = 10)

Arguments

expr An expression matrix containing n-rows of cells and m-cols of genes.
sigma_width An integer number determining the degree of spread of the gaussian distribution which is used for computing weight matrix for Neighborhood sorting method.
no_randomization An integer number indicating the number of repeated sorting, each of which uses a randomly selected initial cell ordering.

Value

A list containing permutated.expr (data frame) and best.cost (a numeric value).

pca_gene_selection Gene selection using PCA technique

Description

Gene selection using PCA technique

Usage

pca_gene_selection(data)

Arguments

data A matrix of data.frame with row.name of cells, and col.name of genes

Value

A vector of the names of selected genes.

Examples

dir <- system.file('extdata', package='uSORT')
file <- list.files(dir, pattern='.txt$', full=TRUE)
exprs <- uSORT_preProcess(exprs_file = file)
exp_trimmed <- t(exprs$exprs_log_trimed)
PCA_selected_genes <- pca_gene_selection(exp_trimmed)
Rwanderlust

R implementation of wanderlust

Description

R implementation of wanderlust

Usage

Rwanderlust(data, s, l = 15, k = 15, num_graphs = 1,
             num_waypoints = 250, waypoints_seed = 123, flock_waypoints = 2,
             metric = "euclidean", voting_scheme = "exponential",
             band_sample = FALSE, partial_order = NULL, verbose = TRUE)

Arguments

- `data` Input data matrix.
- `s` Starting point ID.
- `l` \(l\) nearest neighbours.
- `k` \(k\) nearest neighbours, \(k < l\).
- `num_graphs` Number of repeated graphs.
- `num_waypoints` Number of waypoints to guide the trajectory detection.
- `waypoints_seed` The seed for reproducing the results.
- `flock_waypoints` The number of times for flocking the waypoints, default is 2.
- `metric` Distance calculation metric for nearest neighbour detection.
- `voting_scheme` The scheme of voting.
- `band_sample` Boolean, if band the sample
- `partial_order` default NULL.
- `verbose` Boolean, if print the details

Value

a list containing Trajectory, Order, Waypoints

Author(s)

Hao Chen

Examples

```r
set.seed(15)
shuffled_iris <- iris[sample(150, 150, replace = FALSE), ]
data <- shuffled_iris[,1:4]
data_label <- shuffled_iris[,5]
wishbone <- Rwanderlust(data = data, num_waypoints = 100, waypoints_seed = 2)
pd1 <- data.frame(id = wishbone$Trajectory, label=data_label, stringsAsFactors = FALSE)
pd2 <- data.frame(id = seq_along(row.names(data)), label=data_label, stringsAsFactors = FALSE)
#ggplot(pd1, aes(x=id, y=id, colour = label)) + geom_point() + theme_bw()
#ggplot(pd2, aes(x=id, y=id, colour = label)) + geom_point() + theme_bw()
```
**scattering_quantification_per_gene**

*An expression scattering measurement function*

**Description**

An expression scattering measurement function computes the level of scattering for individual genes along the cell ordering.

**Usage**

```r
scattering_quantification_per_gene(CDS = NULL)
```

**Arguments**

- `CDS` a Monocle’s CellDataSet object

**Value**

integer

**Author(s)**

MaiChan Lau

---

**sorting_method_parameter_GUI**

*GUI for sorting method parameters*

**Description**

The parameters appeared on GUI are based on input method, this function is called by `uSORT_parameters_GUI`. For internal use only.

**Usage**

```r
sorting_method_parameter_GUI(method = c("autoSPIN", "sWanderlust", "monocle", "Wanderlust", "SPIN", "none"))
```

**Arguments**

- `method` method name.

**Value**

a list of parameters.

**Author(s)**

Hao Chen
**Description**

A wrapper function for SPIN method provides a R version of SPIN [Tsafir et al. 2005].

**Usage**

```
SPIN(data, sorting_method = c("STS", "neighborhood"), sigma_width = 1)
```

**Arguments**

- `data`: An log2 transformed expression matrix containing n-rows of cells and m-cols of genes.
- `sorting_method`: A character string indicating the choice of sorting method, i.e. 'STS' (Side-to-Side) or 'Neighborhood'.
- `sigma_width`: An integer number determining the degree of spread of the gaussian distribution which is used for computing weight matrix for Neighborhood sorting method.

**Value**

A data frame containing single column of ordered sample IDs.

**Examples**

```
set.seed(15)
da <- iris[sample(150, 150, replace = FALSE), ]
rownames(da) <- paste0('spl_',seq(1,nrow(da)))
d <- da[,1:4]
dl <- da[,5,drop=FALSE]
res <- SPIN(data = d)
dl <- dl[match(res$SampleID,rownames(dl))]
annot <- data.frame(id = seq(1,nrow(res)), label=dl, stringsAsFactors = FALSE)
#ggplot(annot, aes(x=id, y=id, colour = label)) + geom_point() + theme_bw()
```

**STS_sorting**

*A sorting function using the Side-to-Side (STS) algorithm*

**Description**

A sorting function using the Side-to-Side (STS) algorithm

**Usage**

```
STS_sorting(d, max_iter = 10)
```
Arguments

- **d**: A matrix containing n-by-n cell distance.
- **max_iter**: An integer number indicating the maximum number of iteration if sorting does not converge.

Value

A list containing `ordering` (a vector of re-ordered sequence) and `cost` (a numeric value).

---

**STS_sortingcost**  
*A cost computation function for Side-to-Side (STS) algorithm*

Description

A cost computation function for Side-to-Side (STS) algorithm

Usage

```
STS_sortingcost(expr = NULL, method = c("Euclidean", "Correlation", "eJaccard", "none"))
```

Arguments

- **expr**: An expression matrix containing n-rows of cells and m-cols of genes.
- **method**: A character string indicating the distance function.

Value

A numeric value of sorting cost.

Examples

```r
set.seed(15)
da <- iris[sample(150, 150, replace = FALSE), ]
d <- da[,1:4]
randomOrdering_cost <- STS_sortingcost(d, method= "Euclidean")
randomOrdering_cost
da <- iris
d <- da[,1:4]
properOrdering_cost <- STS_sortingcost(d, method= 'Euclidean')
properOrdering_cost
```
**STS_sorting_wrapper**  
A wrapper function for Side-to-Side (STS) sorting.

**Description**  
A wrapper function for Side-to-Side (STS) sorting as proposed in [Tsafrir et al. 2005].

**Usage**  
```r  
STS_sorting_wrapper(expr, no_randomization = 10)  
```

**Arguments**

- `expr`  
  An expression matrix containing n-rows of cells and m-cols of genes.

- `no_randomization`  
  An integer number indicating the number of repeated sorting, each of which uses a randomly selected initial cell ordering.

**Value**  
A list containing `permutated.expr` (data frame) and `best.cost` (a numeric value).

---

**summed_local_variance**  
A summed local variance function

**Description**  
A summed local variance function

**Usage**  
```r  
summed_local_variance(expr = NULL, alpha = NULL, data_type = "linear")  
```

**Arguments**

- `expr`  
  An expression matrix containing n-rows of cells and m-cols of genes.

- `alpha`  
  A fraction value indicating the size of window for local variance measurement.

- `data_type`  
  A character string indicating the type of progression, i.e. 'linear' (strictly linear) or 'cyclical' (cyclically linear).

**Value**  
A numeric value of the summed local variance.
summed_local_variance_linear

A summed local variance function for strictly linear data type

Description
A summed local variance function for strictly linear data type

Usage
summed_local_variance_linear(d, alpha = 0.3)

Arguments
- d: A cell-to-cell distance matrix.
- alpha: A fraction value indicating the size of window for local variance measurement.

Value
A numeric value of the summed local variance.

summed_local_variance_cyclical

A summed local variance function for cyclical linear data type

Description
A summed local variance function for cyclical linear data type

Usage
summed_local_variance_cyclical(d, alpha = 0.3)

Arguments
- d: A cell-to-cell distance matrix.
- alpha: A fraction value indicating the size of window for local variance measurement.

Value
A numeric value of the summed local variance.
Description

autoSPIN guided wanderlust. Specifically, we use autoSPIN to help find the starting point for wanderlust.

Usage

sWanderlust(data, data_type = c("linear", "cyclical"),
SPIN_option = c("STS", "neighborhood"), alpha = 0.2, sigma_width = 1,
diffusionmap_components = 4, l = 15, k = 15, num_waypoints = 150,
flock_waypoints = 2, waypoints_seed = 2711)

Arguments

data data Input data matrix.
data_type The data type which guides the autoSPIN sorting, including linear, cyclical.
SPIN_option SPIN contains two options including STS(default), neighborhood.
alpha alpha parameter for autoSPIN, default is 0.2.
sigma_width Sigma width parameter for SPIN, default is 1.
diffusionmap_components Number of components from diffusion map used for wanderlust analysis, default is 4.
l Number of nearest neighbors, default is 15.
k Number of nearest neighbors for repeating graphs, default is 15, should be less than or equal to l.
num_waypoints Number of waypoint used for wanderlust, default is 150.
flock_waypoints The number of times for flocking the waypoints, default is 2.
waypoints_seed The seed for reproducing the results.

Value

a vector of the sorted order.

Author(s)

Hao Chen

Examples

set.seed(15)
shuffled_iris <- iris[sample(150, 150, replace = FALSE), ]
data <- shuffled_iris[,1:4]
data_label <- shuffled_iris[,5]
wishbone <- sWanderlust(data = data, num_waypoints = 100)
trajectory_landmarks  determining initial trajectory and landmarks

Description
determining initial trajectory and landmarks

Usage

```
trajectory_landmarks(knn, data, s, partial_order = NULL, waypoints = 250,
waypoints_seed = 123, metric = "euclidean", flock_waypoints = 2,
band_sample = FALSE)
```

Arguments

- **knn**: A sparse matrix of knn.
- **data**: data.
- **s**: The ID of starting point.
- **partial_order**: A vector of IDs specified as recommended waypoints, NULL to ignore.
- **waypoints**: Either the number of waypoints, or specify the waypoint IDs.
- **waypoints_seed**: Random sampling seed, for reproducible results.
- **metric**: Distance calculation metric for nearest neighbour detection.
- **flock_waypoints**: Iteration of using nearest points around waypoint to adjust its position.
- **band_sample**: if give more chance to nearest neighbours of starting point in randomly waypoints selection.

Value

a list

uSORT  uSORT: A self-refining ordering pipeline for gene selection

Description

This package is designed to uncover the intrinsic cell progression path from single-cell RNA-seq data.

The main function of uSORT-package which provides a workflow of sorting scRNA-seq data.
Usage

```r
uSORT(exprs_file, log_transform = TRUE, remove_outliers = TRUE,
preliminary_sorting_method = c("autoSPIN", "sWanderlust", "monocle",
"Wanderlust", "SPIN", "none"), refine_sorting_method = c("autoSPIN",
"sWanderlust", "monocle", "Wanderlust", "SPIN", "none"),
project_name = "uSORT", result_directory = getwd(), nCores = 1,
save_results = TRUE, reproduce_seed = 1234,
scattering_cutoff_prob = 0.75, driving_force_cutoff = NULL,
qval_cutoff_featureSelection = 0.05, pre_data_type = c("linear",
"cyclical"), pre_SPIN_option = c("STS", "neighborhood"),
pre_SPIN_sigma_width = 1, pre_autoSPIN_alpha = 0.2,
pre_autoSPIN_randomization = 20, pre_wanderlust_start_cell = NULL,
pre_wanderlust_num_waypoints = 15, pre_wanderlust_waypoints_seed = 2711,
pre_wanderlust_flock_waypoints = 2, ref_data_type = c("linear",
"cyclical"), ref_SPIN_option = c("STS", "neighborhood"),
ref_SPIN_sigma_width = 1, ref_autoSPIN_alpha = 0.2,
ref_autoSPIN_randomization = 20, ref_wanderlust_start_cell = NULL,
ref_wanderlust_num_waypoints = 15, ref_wanderlust_flock_waypoints = 2,
ref_wanderlust_waypoints_seed = 2711)
```

Arguments

- `exprs_file` Input file name in txt format, with rownames of cells and colnames of genes.
- `log_transform` Boolean, if log transform the data.
- `remove_outliers` Boolean, if remove the outliers.
- `preliminary_sorting_method` Method name for preliminary sorting, including autoSPIN, sWanderlust, monocle, Wanderlust, SPIN, or none.
- `refine_sorting_method` Method name for refined sorting, including autoSPIN, sWanderlust, monocle, Wanderlust, SPIN, or none.
- `project_name` A character name as the prefix of the saved result file.
- `result_directory` The directory indicating where to save the results.
- `nCores` Number of cores that will be employed for drive gene selection (parallel computing), default is 1.
- `save_results` Boolean determining if save the results.
- `reproduce_seed` A seed used for reproducing the result.
- `scattering_cutoff_prob` Scattering cutoff value probability for gene selection, default 0.75.
- `driving_force_cutoff` Driving force cutoff value for gene selection, default NULL (automatically).
- `qval_cutoff_featureSelection` Q value cutoff for gene selection, default 0.05.
- `pre_data_type` The data type which guides the autoSPIN sorting, including linear, cyclical.
pre_SPIN_option
   SPIN contains two options including STS(default), neighborhood.
pre_SPIN_sigma_width
   Sigma width parameter for SPIN, default is 1.
pre_autoSPIN_alpha
   alpha parameter for autoSPIN, default is 0.2.
pre_autoSPIN_randomization
   Number of randomizations for autoSPIN, default is 20.
pre_wanderlust_start_cell
   The name of starting cell for wanderlust, default is the first cell from the data.
pre_wanderlust_dfmap_components
   Number of components from diffusion map used for wanderlust analysis, default is 4.
pre_wanderlust_l
   Number of nearest neighbors used for wanderlust, default is 15.
pre_wanderlust_num_waypoints
   Number of waypoint used for wanderlust, default is 150.
pre_wanderlust_waypoints_seed
   The seed for reproducing the wanderlust results.
pre_wanderlust_flock_waypoints
   The number of times for flocking the waypoints, default is 2.
ref_data_type
   The data type which guides the autoSPIN sorting, including linear, cyclical.
ref_SPIN_option
   SPIN contains two options including STS(default), neighborhood.
ref_SPIN_sigma_width
   Sigma width parameter for SPIN, default is 1.
ref_autoSPIN_alpha
   alpha parameter for autoSPIN, default is 0.2.
ref_autoSPIN_randomization
   Number of randomizations for autoSPIN, default is 20.
ref_wanderlust_start_cell
   The name of starting cell for wanderlust, default is the first cell from the data.
ref_wanderlust_dfmap_components
   Number of components from diffusion map used for wanderlust analysis, default is 4.
ref_wanderlust_l
   Number of nearest neighbors used for wanderlust, default is 15.
ref_wanderlust_num_waypoints
   Number of waypoint used for wanderlust, default is 150
ref_wanderlust_flock_waypoints
   The number of times for flocking the waypoints, default is 2.
ref_wanderlust_waypoints_seed
   The seed for reproducing the wanderlust results.

Details
This package incorporates data pre-processing, preliminary PCA gene selection, preliminary cell ordering, feature selection, refined cell ordering, and post-analysis interpretation and visualization. The uSORT workflow can be implemented through calling the main function or the GUI uSORT.
Value

results object (a list)

See Also

uSORT-package, uSORT_GUI

Examples

```r
dir <- system.file('extdata', package='uSORT')
file <- list.files(dir, pattern='.*txt$', full=TRUE)
# remove the # symbol of the following codes to test
# uSORT_results <- uSORT(exprs_file = file, project_name = "test",
# preliminary_sorting_method = "autoSPIN",
# refine_sorting_method = "sWanderlust",
# save_results = FALSE)
```

Description

This GUI provides an easy way for applying the uSORT package.

Usage

uSORT_GUI()

Value

the GUI for uSORT-package

Author(s)

Hao Chen

References

http://JinmiaoChenLab.github.io/uSORT/

See Also

uSORT-package, uSORT

Examples

```r
interactive()
#if(interactive()) uSORT_GUI() # remove the hash symbol to run
```
uSORT_parameters_GUI  The GUI for inputting parameters for uSORT

Description
This is a function for generating the GUI for uSORT, it’s called by uSORT_GUI. For internal use only.

Usage
uSORT_parameters_GUI()

Value
a list of parameters.

Author(s)
Hao Chen

uSORT_preProcess  A data loading and pre-processing function

Description
A data loading and pre-processing function which firstly identifies outlier cells and scarcely expressed genes.

Usage
uSORT_preProcess(exprs_file, log_transform = TRUE, remove_outliers = TRUE, lod = 1)

Arguments
exprs_file Input file name in txt format, with rownames of cells and colnames of genes.
log_transform Boolean, if TRUE log transform the data.
remove_outliers Boolean, if TRUE remove the outliers.
lod A value of limit of detection in the unit of TPM/CPM/RPKM. It will be used as the starting value for outlier cell detection and the basis for removing scarce genes.

Value
A list containing exprs_raw(data frame) and exprs_log_trimmed(data frame).

Examples
dir <- system.file('extdata', package='uSORT')
file <- list.files(dir, pattern='*.txt$', full=TRUE)
exprs <- uSORT_preProcess(exprs_file = file)
**Description**

Sorting methods include autoSPIN, sWanderlust, monocle, Wanderlust, SPIN. Any of the sorting method can be called directly using this function.

**Usage**

```r
uSORT_sorting_wrapper(data, data_raw, method = c("autoSPIN", "sWanderlust",
"monocle", "Wanderlust", "SPIN", "none"), data_type = c("linear",
"cyclical"), SPIN_option = c("STS", "neighborhood"), SPIN_sigma_width = 1,
autoSPIN_alpha = 0.2, autoSPIN_randomization = 20,
wanderlust_start_cell = NULL, wanderlust_dfmap_components = 4,
wanderlust_l = 15, wanderlust_num_waypoints = 150,
wanderlust_waypoints_seed = 2711, wanderlust_flock_waypoints = 2)
```

**Arguments**

- `data` : Input preprocessed data matrix with row.name of cells and col.name of genes.
- `data_raw` : Input raw data matrix with row.name of cells and col.name of genes, for monocle method.
- `method` : The name of the sorting method to use, including autoSPIN, sWanderlust, monocle, Wanderlust, SPIN and none.
- `data_type` : The type of the data, either linear or cyclical.
- `SPIN_option` : The runing option of SPIN, STS or neighborhood.
- `SPIN_sigma_width` : Sigma width for SPIN.
- `autoSPIN_alpha` : alpha for autoSPIN.
- `autoSPIN_randomization` : Number of randomization for autoSPIN.
- `wanderlust_start_cell` : The id of the starting cell for wanderlust.
- `wanderlust_dfmap_components` : The number of components from diffusionmap for wanderlust.
- `wanderlust_l` : The number of nearest neighbors used for wanderlust.
- `wanderlust_num_waypoints` : The number of waypoints for wanderlust.
- `wanderlust_waypoints_seed` : The seed for reproducible analysis.
- `wanderlust_flock_waypoints` : The number of flock times for wanderlust.

**Value**

return the order of sorting results.
Examples

dir <- system.file('extdata', package='uSORT')
file <- list.files(dir, pattern='\.txt$', full=TRUE)
exprs <- uSORT_preProcess(exprs_file = file)
exp_trimmed <- t(exprs$exprs_log_trimed)
PCA_selected_genes <- pca_gene_selection(exp_trimmed)
exp_PCA_genes <- exp_trimmed[, PCA_selected_genes]
#order <- uSORT_sorting_wrapper(data = exp_PCA_genes, method = 'autoSPIN')

uSORT_write_results Resluts parsing for uSORT

Description

Save result object into a RData file. Save cell to cell distance heatmap for both preliminary and refined results. Create plot of driver gene profiles on final ordering using heatmap.

Usage

uSORT_write_results(uSORT_results, project_name, result_directory)

Arguments

uSORT_results Result object from uSort function, a list.
project_name A prefix for the saving files.
result_directory The path where to save the results.

Value

save the results.

Examples

dir <- system.file('extdata', package='uSORT')
file <- list.files(dir, pattern='\.txt$', full=TRUE)
#remove the # symbol of the following codes to test
#uSORT_results <- uSORT(exprs_file = file,
#  # project_name = 'test',
#  # preliminary_sorting_method = 'autoSPIN',
#  # refine_sorting_method = 'sWanderlust',
#  # save_results = FALSE)
#uSORT_write_results(uSORT_results,
#  # project_name = 'test',
#  # result_directory = getwd())
variability_per_gene  A utility function for scattering_quantification_per_gene

Description

A utility function for scattering_quantification_per_gene which computes the degree of scattering for single gene, whereby the value is computed by summing over the local values of smaller local windows.

Usage

variability_per_gene(logExp = NULL, min_expr = 0.1, window_size_perct = 0.1, nonZeroExpr_perct = 0.1)

Arguments

logExp a log-scale expression vector of a gene
min_expr a minimum expression value
window_size_perct a window size (in dispersion level)
nonZeroExpr_perct a minimum amount of cells (in expression, otherwise the associated window will be assigned to 0 dispersion value)

Value

integer

Author(s)

MaiChan Lau

wanderlust_wrapper  a wrapper of wanderlust for sWanderlust

Description

a wrapper of wanderlust for sWanderlust

Usage

wanderlust_wrapper(data, s, diffusionmap_components = 4, 1 = 15, k = 15, num_graphs = 1, num_waypoints = 150, waypoints_seed = 123, flock_waypoints = 2)
Arguments

data: Input data matrix.
s: The ID of starting point.
diffusionmap_components: Number of components from diffusion map used for wanderlust analysis, default is 4.
l: Number of nearest neighbors, default is 15.
k: Number of nearest neighbors for repeating graphs, default is 15, should be less than or equal to l.
num_graphs: Number of repeated graphs.
num_waypoints: Number of waypoint used for wanderlust, default is 150.
waypoints_seed: The seed for reproducing the results.
flock_waypoints: The number of times for flocking the waypoints, default is 2.

Value

sorted order.

Author(s)

Hao Chen
Index

autoSPIN, 2
clusterGenes1, 3
compareModels1, 4
diff_test_helper1, 5
differentialGeneTest1, 5
distance.function, 6
driving_force_gene_selection, 6
elbow_detection, 7
EXP_to_CellDataSet, 8
fluidigmSC_analyzeGeneDetection, 8
fluidigmSC_identifyExpOutliers, 9
fluidigmSC_isElementIgnoreCase, 10
fluidigmSC_readLinearExp, 10
fluidigmSC_removeGenesByLinearExpForAllType, 11
fluidigmSC_removeGenesByLinearExpForAllType_log2, 11
monocle_wrapper, 12
neighborhood_sorting, 12
neighborhood_sorting_wrapper, 14
neighborhood_sortingcost, 13
pca_gene_selection, 14
Rwanderlust, 15
scattering_quantification_per_gene, 16
sorting_method_parameter_GUI, 16
SPIN, 17
STS_sorting, 17
STS_sorting_wrapper, 19
STS_sortingcost, 18
summed_local_variance, 19
summed_local_variance_cyclical, 20
summed_local_variance_linear, 20
swanderlust, 21
trajectory_landmarks, 22
uSORT, 22, 24, 25
uSORT-package (uSORT), 22
uSORT_GUI, 25, 26
uSORT_parameters_GUI, 16, 26
uSORT_preProcess, 26
uSORT_sorting_wrapper, 27
uSORT_write_results, 28
variability_per_gene, 29
wanderlust_wrapper, 29