Package 'RaceID'

July 21, 2025

Title Identification of Cell Types, Inference of Lineage Trees, and Prediction of Noise Dynamics from Single-Cell RNA-Seq Data

Version 0.3.9

Date 2024-11-24

Maintainer Dominic Grün <dominic.gruen@gmail.com>

Description Application of 'RaceID' allows inference of cell types and prediction of lineage trees by the 'StemID2' algorithm (Herman, J.S., Sagar, Grun D. (2018) <DOI:10.1038/nmeth.4662>). 'VarID2' is part of this package and allows quantification of biological gene expression noise at single-cell resolution (Rosales-Alvarez, R.E., Rettkowski, J., Herman, J.S., Dumbovic, G., Cabezas-Wallscheid, N., Grun, D. (2023) <DOI:10.1186/s13059-023-02974-1>).

Depends R (>= 3.5.0)

Imports coop, compiler, cluster, FateID, FNN, fpc, ggplot2, grDevices, harmony, ica, igraph, irlba, leiden, locfit, methods, MASS, Matrix, matrixStats, parallel, pheatmap, princurve, quadprog, randomForest, runner, Rcpp, RColorBrewer, Rtsne, umap, vegan

LinkingTo Rcpp (>= 0.11.0)

Suggests batchelor, DESeq2, knitr, rmarkdown, SingleCellExperiment, slingshot, SummarizedExperiment

VignetteBuilder knitr

SystemRequirements C++17

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

NeedsCompilation yes

Author Dominic Grün [aut, cre]

Repository CRAN

Date/Publication 2024-11-24 22:50:02 UTC

Contents

`RaceID-package` *Identification of Cell Types, Inference of Lineage Trees, and Prediction of Noise Dynamics from Single-Cell RNA-Seq Data*

Description

RaceID is a clustering algorithm for the identification of cell types from single-cell RNA-sequencing data. It was specifically designed for the detection of rare cells which correspond to outliers in conventional clustering methods. The package contains RaceID3, the most recently published version of this algorithm, and StemID2, an algorithm for the identification of lineage trees based on RaceID3 analysis. RaceID3 utilizes single cell expression data, and was designed to work well with quantitative single-cell RNA-seq data incorporating unique molecular identifiers. It requires a gene-by-cell expression matrix as input and produces a clustering partition representing cell types. StemID2 assembles these cell types into a lineage tree. The RaceID package (>= v0.1.4) also contains functions for a VarID analysis. VarID comprises a sensitive clustering method utilizing pruned k-nearest neighbor networks, connecting only cells with links supported by a background model of gene expression. These pruned k-nearest neighbor networks further enable the definition of homogenous neighborhoods for the quantification of local gene expression variability in cell state space.

Details

For details please see vignette.

Author(s)

Dominic Grun, dominic.gruen@gmail.com.

Maintainer: Dominic Grun <dominic.gruen@gmail.com>

References

Herman, J.S., Sagar, Grun D. (2018) <DOI:10.1038/nmeth.4662> Rosales-Alvarez, R.E., Rettkowski, J., Herman, J.S., Dumbovic, G., Cabezas-Wallscheid, N., Grun, D. (2023) <DOI:10.1186/s13059-023-02974-1>

`barplotgene` *Gene Expression Barplot*

Description

This functions generates a barplot of gene expression across all clusters.

Usage

```
barplotgene(object, g, n = NULL, logsc = FALSE)
```

Arguments

`object`	`SCseq` class object.
`g`	Individual gene name or vector with a group of gene names corresponding to a subset of valid row names of the `ndata` slot of the `SCseq` object.
`n`	String of characters representing the title of the plot. Default is `NULL` and the first element of `g` is chosen.
`logsc`	logical. If `TRUE`, then gene expression values are log2-transformed after adding a pseudo-count of 0.1. Default is `FALSE` and untransformed values are shown.

Value

None

`baseLineVar` *Baseline gene expression variability*

Description

This function returns the base line variability as a function of the

Usage

```
baseLineVar(x, y)
```

Arguments

x	mean expression. The corresponding corrected variance is returned.
y	object returned by compNoise, noiseBaseFit, pruneKnn or fitBackVar. Depending on the input the funtion returns either the background variability (for pruneKnn or fitBackVar) or the base line variability.

Value

Base line (or background) variability.

Examples

```
y <- noiseBaseFit(intestinalDataSmall,step=.01,thr=.05)
x <- apply(intestinalDataSmall,1,mean)
baseLineVar(x,y)
```

---

branchcells *Differential Gene Expression between Links*

Description

This function computes expression z-score between groups of cells from the same cluster residing on different links

Usage

```
branchcells(object, br)
```

Arguments

object	Ltree class object.
br	List containing two branches, where each component has to be two valid cluster numbers seperated by a . and with one common cluster in the two components. The lower number precedes the larger one, i.e. 1.3. For each component, the cluster number need to be ordered in increasing order.

Value

A list ot four components:

n	a vector with the number of significant links for each cluster.
scl	a vector with the delta entropy for each cluster.
k	a vector with the StemID score for each cluster.
diffgenes	a vector with the StemID score for each cluster.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- comptsne(sc)
ltr <- Ltree(sc)
ltr <- compentropy(ltr)
ltr <- projcells(ltr)
ltr <- lineagegraph(ltr)
ltr <- comppvalue(ltr)
x <- branchcells(ltr,list("1.3","3.6"))
head(x$diffgenes$z)
plotmap(x$scl)
plotdiffgenes(x$diffgenes,names(x$diffgenes$z)[1])
```

`calcAlphaG` *Function for calculating an aggregated dispersion parameter*

Description

This function calculates an aggregated dispersion parameter comprising global cell-to-cell variability of transcript counts and biological variability.

Usage

```
calcAlphaG(noise)
```

Arguments

`noise`	List of noise parameters returned by `compTBNoise`.

Value

Matrix of aggregated dispersion parameters.

calcVar *Function for calculating total variance from VarID fit*

Description

This function calculates the total variance from a local VarID fit.

Usage

```
calcVar(w)
```

Arguments

w	List object returned by fitNBtb.

Value

Vector of total variance estimates.

calcVarFit *Function for calculating the total variance fit*

Description

This function calculates a total variance fit comprising sampling noise, global cell-to-cell variability of transcript counts, and biological variability.

Usage

```
calcVarFit(noise, norm = FALSE)
```

Arguments

noise	List of noise parameters returned by compTBNoise.
norm	Logical. If TRUE then total variance is normalized by the technical noise component (i.e., sampling noise plus global cell-to-cell variability in transcript counts.). Default is FALSE.

Value

Matrix of total variance fits.

CCcorrect *Dimensional Reduction by PCA or ICA*

Description

This functions performs dimensional reduction by PCA or ICA and removes components enriched for particular gene sets, e.g. cell cycle related genes genes associated with technical batch effects.

Usage

```
CCcorrect(
  object,
  vset = NULL,
  CGenes = NULL,
  ccor = 0.4,
  pvalue = 0.01,
  quant = 0.01,
  nComp = NULL,
  dimR = FALSE,
  mode = "pca",
  logscale = FALSE,
  FSelect = TRUE
)
```

Arguments

object	SCseq class object.
vset	List of vectors with genes sets. The loadings of each component are tested for enrichment in any of these gene sets and if the lower quant or upper 1 - quant fraction of genes ordered by loading is enriched at a p-value < pvalue the component is discarded. Default is NULL.
CGenes	Vector of gene names. If this argument is given, gene sets to be tested for enrichment in PCA- or ICA-components are defined by all genes with a Pearson's correlation of >ccor to a gene in CGenes. The loadings of each component are tested for enrichment in any of these gene sets and if the lower quant or upper 1 - quant fraction of genes ordered by loading is enriched at a p-value < pvalue the component is discarded. Default is NULL.
ccor	Positive number between 0 and 1. Correlation threshold used to detrmine correlating gene sets for all genes in CGenes. Default is 0.4.
pvalue	Positive number between 0 and 1. P-value cutoff for determining enriched components. See vset or CGenes. Default is 0.01.
quant	Positive number between 0 and 1. Upper and lower fraction of gene loadings used for determining enriched components. See vset or CGenes. Default is 0.01.
nComp	Number of PCA- or ICA-components to use. Default is NULL and the maximal number of components is computed.

dimR	logical. If TRUE, then the number of principal components to use for downstream analysis is derived from a saturation criterion. See function plotdimsat. Default is FALSE and all nComp components are used.
mode	"pca" or "ica" to perform either principal component analysis or independent component analysis. Default is pca.
logscale	logical. If TRUE data are log-transformed prior to PCA or ICA. Default is FALSE.
FSelect	logical. If TRUE, then PCA or ICA is performed on the filtered expression matrix using only the features stored in slotcluster$features as computed in the function filterdata. See FSelect for function filterdata. Default is TRUE.

Value

The function returns an updated SCseq object with the principal or independent component matrix written to the slot dimRed$x of the SCseq object. Additional information on the PCA or ICA is stored in slot dimRed.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- CCcorrect(sc,dimR=TRUE,nComp=3)
```

---

cc_genes *Cell cycle markers for Mus Muscuus*

Description

This dataset contains official gene symbols for markers of the S phase and G2/M phase of the cell cycle in mouse.

Usage

```
cc_genes
```

Format

A list of two components with S phase marker (s) and G2M phase marker (g2m) gene symbols.

Value

None

cellsfromtree *Extract Cells on Differentiation Trajectory*

Description

This function extracts a vector of cells on a given differentiation trajectory in pseudo-temporal order determined from the projection coordinates.

Usage

```
cellsfromtree(object, z)
```

Arguments

object	Ltree class object.
z	Vector of valid cluster numbers ordered along the trajectory.

Value

A list ot four components:

f	a vector of cells ids ordered along the trajectory defined by z.
g	a vector of integer number. Number i indicates that a cell resides on the link between the i-th and (i+1)-th cluster in z.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- comptsne(sc)
ltr <- Ltree(sc)
ltr <- compentropy(ltr)
ltr <- projcells(ltr)
ltr <- lineagegraph(ltr)
ltr <- comppvalue(ltr)
x <- cellsfromtree(ltr,c(1,3,6,2))
```

cleanNN *Function for pruning k-nearest neighborhoods based on neighborhood overlap*

Description

This function compares the neighborhood of a cell with the neighorhoods of all of its k nearest neighors and prunes links to neighbors that do not co-occur in a defined minimum number of neighborhoods by setting their link p-value (entry in `pvM` data.frame of `res` input object) to 0.

Usage

```
cleanNN(res, minN = 2, no_cores = NULL)
```

Arguments

`res`	List object with k nearest neighbour information returned by `pruneKnn` function.
`minN`	Positive integer number. Minimum of neighborhoods across the k nearest neighbours of a cell expected to share a neighbor with the cell. Default is 2.
`no_cores`	Positive integer number. Number of cores for multithreading. If set to `NULL` then the number of available cores minus two is used. Default is `NULL`.

Value

A `res` object with update pvalue entries (`pvM` element).

clustdiffgenes *Inference of differentially expressed genes in a cluster*

Description

This functions computes differentially expressed genes in a (set of) cluster(s) by comparing to all remaining cells outside of the cluster (or a given background set of clusters) based on a negative binomial model of gene expression

Usage

```
clustdiffgenes(object, cl, bgr = NULL, pvalue = 0.01)
```

Arguments

object	SCseq class object.
cl	A valid set of cluster numbers from the final cluster partition stored in the cpart slot of the SCseq object.
bgr	Ordered vector of cluster numbers to be used as background set. If NULL then all clusters not in cl are used as background set.
pvalue	Positive real number smaller than one. This is the p-value cutoff for the inference of differential gene expression. Default is 0.01.

Value

A list of two components. The first component dg contains a a data.frame of differentially expressed genes ordered by p-value in increasing order, with four columns:

mean.ncl	mean expression across cells outside of cluster cl.
mean.cl	mean expression across cells within cluster cl.
fc	fold-change of mean expression in cluster cl versus the remaining cells.
pv	inferred p-value for differential expression.
padj	Benjamini-Hochberg corrected FDR.

The second component de contains the conventional output of diffexpnb, where set B corresponds to all clusters in cl and B to the background set (all clusters in bgr or not in cl). This component can be used for plotting by plotdiffgenesnb.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
x <- clustdiffgenes(sc,1)
head(x$dg[x$dg$fc>1,])
```

---

clustexp *Clustering of single-cell transcriptome data*

Description

This functions performs the initial clustering of the RaceID3 algorithm.

Usage

```
clustexp(
  object,
  sat = TRUE,
  samp = NULL,
  cln = NULL,
  clustnr = 30,
  bootnr = 50,
  rseed = 17000,
  FUNcluster = "kmedoids",
  verbose = TRUE
)
```

Arguments

object	SCseq class object.
sat	logical. If TRUE, then the number of clusters is determined based on finding the saturation point of the mean within-cluster dispersion as a function of the cluster number. Default is TRUE. If FALSE, then cluster number needs to be given as cln.
samp	Number of random sample of cells used for the inference of cluster number and for inferring Jaccard similarities. Default is 1000.
cln	Number of clusters to be used. Default is NULL and the cluster number is inferred by the saturation criterion.
clustnr	Maximum number of clusters for the derivation of the cluster number by the saturation of mean within-cluster-dispersion. Default is 30.
bootnr	Number of booststrapping runs for clusterboot. Default is 50.
rseed	Integer number. Random seed to enforce reproducible clustering results. Default is 17000.
FUNcluster	Clustering method used by RaceID3. One of "kmedoids", "kmeans", "hclust". Default is "kmedoids".
verbose	logical. If FALSE then status output messages are disabled. Default is TRUE.

Value

SCseq object with clustering data stored in slot cluster and slot clusterpar. The clustering partition is stored in cluster$kpart.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
```

clustheatmap *Plotting a Heatmap of the Distance Matrix*

Description

This functions plots a heatmap of the distance matrix grouped by clusters.

Usage

```
clustheatmap(object, final = TRUE, hmethod = "single")
```

Arguments

object	SCseq class object.
final	logical. If TRUE, then cells are grouped based on final clusters after outlier identification. If FALSE, then initial clusters prior to outlier identification are used for grouping. Default is TRUE.
hmethod	Agglomeration method used for determining the cluster order from hierarchical clustering of the cluster medoids. See hclust function.

Value

Returns a vector of cluster numbers ordered as determined by herarchical clustering of cluster the cluster medoids as depicted in the heatmap.

compdist *Computing a distance matrix for cell type inference*

Description

This functions computes the distance matrix used for cell type inference by RaceID3.

Usage

```
compdist(
  object,
  metric = "pearson",
  FSelect = TRUE,
  knn = NULL,
  alpha = 1,
  no_cores = 1
)
```

Arguments

object	SCseq class object.
metric	Distances are computed from the filtered expression matrix after optional feature selection, dimensional reduction, and/or transformation (batch correction). Possible values for metric are spearman, pearson, logpearson, euclidean, kendall. Default is "pearson". In case of the correlation based methods, the distance is computed as 1 – correlation.
FSelect	Logical parameter. If TRUE, then feature selection is performed prior to RaceID3 analysis. Default is TRUE.
knn	Positive integer number of nearest neighbours used for imputing gene expression values. Default is NULL and no imputing is done.
alpha	Positive real number. Relative weight of a cell versus its k nearest neigbour applied for imputing gene expression. A cell receives a weight of alpha while the weight of its k nearest neighbours is determined by quadratic programming. The sum across all weights is normalized to one, and the wieghted mean expression is used for computing the joint probability of a cell and each of its k nearest neighbours. These probabilities are applied for the derivation of the imputed gene expression for each cell. Default is 1. Larger values give more weight to the gene expression observed in a cell versus its neighbourhood.
no_cores	Positive integer number. Number of cores for multithreading during imputation. If set to NULL then the number of available cores minus two is used. Default is 1.

Value

SCseq object with the distance matrix in slot distances. If FSelect=TRUE, the genes used for computing the distance object are stored in slot cluster$features.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
```

---

compentropy *Compute transcriptome entropy of each cell*

Description

This function computes the transcriptome entropy for each cell.

Usage

```
compentropy(object)
```

Arguments

object	Ltree class object.

Value

An Ltree class object with a vector of entropies for each cell in the same order as column names in slot sc@ndata.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- comptsne(sc)
ltr <- Ltree(sc)
ltr <- compentropy(ltr)
```

---

compfr *Computation of a two dimensional Fruchterman-Rheingold representation*

Description

This functions performs the computation of a Fruchterman-Rheingold graph layout based on an adjacency matrix derived from the distance object in slot distances using the **igraph** package.

Usage

```
compfr(object, knn = 10, rseed = 15555)
```

Arguments

object	SCseq class object.
knn	Positive integer number of nearest neighbours used for the inference of the Fruchterman-Rheingold layout. Default is 10.
rseed	Integer number. Random seed to enforce reproducible layouts.

Value

SCseq object with layout coordinates stored in slot fr.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- compfr(sc)
```

---

compMean *Function for computing local gene expression averages*

Description

This function performs computation of locally averaged gene expression across the pruned k nearest neighbours at given link probability cutoff.

Usage

```
compMean(
  x,
  res,
  pvalue = 0.01,
  genes = NULL,
  regNB = FALSE,
  batch = NULL,
  regVar = NULL,
  offsetModel = TRUE,
  thetaML = FALSE,
  theta = 10,
  ngenes = NULL,
  span = 0.75,
  no_cores = NULL,
  seed = 12345
)
```

Arguments

x	Matrix of gene expression values with genes as rows and cells as columns. The matrix need to contain the same cell IDs as columns like the input matrix used to derive the pruned k nearest neighbours with the pruneKnn function. However, it may contain a different set of genes.
res	List object with k nearest neighbour information returned by pruneKnn function.
pvalue	Positive real number between 0 and 1. All nearest neighbours with link probability < pvalue are discarded. Default is 0.01.
genes	Vector of gene names corresponding to a subset of rownames of x. Only for these genes local gene expression averages are computed. Default is NULL and values for all genes are returned.

regNB — logical. If TRUE then gene expression averages are computed from the pearson residuals obtained from a negative binomial regression to eliminate the dependence of the expression variance on the mean. If FALSE then averages are computed from raw UMI counts. Default is FALSE.

batch — vector of batch variables. Component names need to correspond to valid cell IDs, i.e. column names of expData. If regNB is TRUE, than the batch variable will be regressed out simultaneously with the log UMI count per cell.An interaction term is included for the log UMI count with the batch variable. Default value is NULL.

regVar — data.frame with additional variables to be regressed out simultaneously with the log UMI count and the batch variable (if batch is TRUE). Column names indicate variable names (name beta is reserved for the coefficient of the log UMI count), and rownames need to correspond to valid cell IDs, i.e. column names of expData. Interaction terms are included for each variable in regVar with the batch variable (if batch is TRUE). Default value is NULL.

offsetModel — Logical parameter. Only considered if regNB is TRUE. If TRUE then the beta (log UMI count) coefficient is set to 1 and the intercept is computed analytically as the log ration of UMI counts for a gene and the total UMI count across all cells. Batch variables and additional variables in regVar are regressed out with an offset term given by the sum of the intercept and the log UMI count. Default is TRUE.

thetaML — Logical parameter. Only considered if offsetModel equals TRUE. If TRUE then the dispersion parameter is estimated by a maximum likelihood fit. Otherwise, it is set to theta. Default is FALSE.

theta — Positive real number. Fixed value of the dispersion parameter. Only considered if theaML equals FALSE.

ngenes — Positive integer number. Randomly sampled number of genes (from rownames of expData) used for predicting regression coefficients (if regNB=TRUE). Smoothed coefficients are derived for all genes. Default is NULL and all genes are used.

span — Positive real number. Parameter for loess-regression (see regNB) controlling the degree of smoothing. Default is 0.75.

no_cores — Positive integer number. Number of cores for multithreading. If set to NULL then the number of available cores minus two is used. Default is NULL.

seed — Integer number. Random number to initialize stochastic routines. Default is 12345.

Value

List object of three components:

mean — matrix with local gene expression averages, computed from Pearson residuals (if regNB=TRUE) or normalized UMI counts (if regNB=FALSE). In the latter case, the average UMI count for a local neighbourhood is normalized to one and rescaled by the median UMI count across neighborhoods.

regData — If regNB=TRUE this argument contains a list of four components: component pearsonRes contains a matrix of the Pearson Residual computed from the negative binomial regression, component nbRegr contains a matrix with the regression coefficients, component nbRegrSmooth contains a matrix with the smoothed regression coefficients, and log_umi is a vector with the total log UMI count for each cell. The regression coefficients comprise the dispersion parameter theta, the intercept, the regression coefficient beta for the log UMI count, and the regression coefficients of the batches (if batch is not NULL).

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
mexp <- compMean(intestinalDataSmall,res,pvalue=0.01,genes = NULL,no_cores=1)
```

---

compmedoids *Computes Medoids from a Clustering Partition*

Description

This functions computes cluster medoids given an SCseq object and a clustering partition. The medoids are either derived from the distance matrix or, if the slot distances is empty, from the dimensionally reduced feature matrix in slot dimRed$x using the euclidean metric.

Usage

```
compmedoids(object, part)
```

Arguments

object — SCseq class object.

part — Clustering partition. A vector of cluster numbers for (a subset of) cells (i.e. column names) of slot ndata from the SCseq object.

Value

Returns a list of medoids (column names of slot ndata from the SCseq object) ordered by increasing cluster number.

compNoise *Function for computing local gene expression variability*

Description

This function performs computation of the local gene expression variability across the pruned k nearest neighbours at given link probability cutoff. The estimated variance is corrected for the mean dependence utilizing the baseline model of gene expression variance.

Usage

```
compNoise(
  x,
  res,
  pvalue = 0.01,
  genes = NULL,
  regNB = FALSE,
  batch = NULL,
  regVar = NULL,
  offsetModel = TRUE,
  thetaML = FALSE,
  theta = 10,
  ngenes = NULL,
  span = 0.75,
  step = 0.01,
  thr = 0.05,
  no_cores = NULL,
  seed = 12345
)
```

Arguments

x	Matrix of gene expression values with genes as rows and cells as columns. The matrix need to contain the same cell IDs as columns like the input matrix used to derive the pruned k nearest neighbours with the pruneKnn function. However, it may contain a different set of genes.
res	List object with k nearest neighbour information returned by pruneKnn function.
pvalue	Positive real number between 0 and 1. All nearest neighbours with link probability < pvalue are discarded. Default is 0.01.
genes	Vector of gene names corresponding to a subset of rownames of x. Only for these genes local gene expression variability is computed. Default is NULL and values for all genes are returned.
regNB	logical. If TRUE then gene expression variability is derived from the pearson residuals obtained from a negative binomial regression to eliminate the dependence of the expression variance on the mean. If FALSE then the mean dependence is regressed out from the raw variance using the baseline variance estimate. Default is FALSE.

batch — vector of batch variables. Component names need to correspond to valid cell IDs, i.e. column names of expData. If regNB is TRUE, than the batch variable will be regressed out simultaneously with the log UMI count per cell. An interaction term is included for the log UMI count with the batch variable. Default value is NULL.

regVar — data.frame with additional variables to be regressed out simultaneously with the log UMI count and the batch variable (if batch is TRUE). Column names indicate variable names (name beta is reserved for the coefficient of the log UMI count), and rownames need to correspond to valid cell IDs, i.e. column names of expData. Interaction terms are included for each variable in regVar with the batch variable (if batch is TRUE). Default value is NULL.

offsetModel — Logical parameter. Only considered if regNB is TRUE. If TRUE then the beta (log UMI count) coefficient is set to 1 and the intercept is computed analytically as the log ration of UMI counts for a gene and the total UMI count across all cells. Batch variables and additional variables in regVar are regressed out with an offset term given by the sum of the intercept and the log UMI count. Default is TRUE.

thetaML — Logical parameter. Only considered if offsetModel equals TRUE. If TRUE then the dispersion parameter is estimated by a maximum likelihood fit. Otherwise, it is set to theta. Default is FALSE.

theta — Positive real number. Fixed value of the dispersion parameter. Only considered if theaML equals FALSE.

ngenes — Positive integer number. Randomly sampled number of genes (from rownames of expData) used for predicting regression coefficients (if regNB=TRUE). Smoothed coefficients are derived for all genes. Default is NULL and all genes are used.

span — Positive real number. Parameter for loess-regression (see regNB) controlling the degree of smoothing. Default is 0.75.

step — Positive real number between 0 and 1. See function noiseBaseFit. Default is 0.01.

thr — Positive real number between 0 and 1. See function noiseBaseFit. Default is 0.05.

no_cores — Positive integer number. Number of cores for multithreading. If set to NULL then the number of available cores minus two is used. Default is NULL.

seed — Integer number. Random number to initialize stochastic routines. Default is 12345.

Value

List object of three components:

model — the baseline noise model as computed by the noiseBaseFit function.

data — matrix with local gene expression variability estimates, corrected for the mean dependence.

regData — If regNB=TRUE this argument contains a list of four components: component pearsonRes contains a matrix of the Pearson Residual computed from the negative binomial regression, component nbRegr contains a matrix with the regression coefficients, component nbRegrSmooth contains a matrix with the smoothed

regression coefficients, and log_umi is a vector with the total log UMI count for each cell. The regression coefficients comprise the dispersion parameter theta, the intercept, the regression coefficient beta for the log UMI count, and the regression coefficients of the batches (if batch is not NULL).

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
noise <- compNoise(intestinalDataSmall,res,pvalue=0.01,genes = NULL,no_cores=1)
```

---

comppvalue *Computing P-values for Link Significance*

Description

This function computes a p-value for the significance (i.e. over-representation of assigned cells) of each inter-cluster link.

Usage

```
comppvalue(object, pthr = 0.01, sensitive = FALSE)
```

Arguments

object	Ltree class object.
pthr	p-value cutoff for link significance. This threshold is applied for the calculation of link scores reflecting how uniformly a link is occupied by cells.
sensitive	logical. Only relevant when nmode=TRUE in function projcell. If TRUE, then all cells on the most highly significant link are and the link itself are disregard to test significance of the remaining links with a binomial p-value. Default is FALSE.

Value

An Ltree class object with link p-value and occupancy data stored in slot cdata.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- comptsne(sc)
ltr <- Ltree(sc)
ltr <- compentropy(ltr)
ltr <- projcells(ltr)
ltr <- lineagegraph(ltr)
ltr <- comppvalue(ltr)
```

`compscore` *Compute StemID2 score*

Description

This function extracts the number of links connecting a given cluster to other cluster, the delta median entropy of each cluster (median entropy of a cluster after subtracting the minimum median entropy across all clusters), and the StemID2 score which is the product of both quantities for each cluster.

Usage

```
compscore(object, nn = 1, scthr = 0, show = TRUE)
```

Arguments

`object`	`Ltree` class object.
`nn`	Positive integer number. Number of higher order neighbors to be included for the determination of links: indirect connections via `n-1` intermittant neighbors are allowed. Default is 1.
`scthr`	Real number between zero and one. Score threshold for links to be included in the calculation. For `scthr=0` all significant links are included. The maximum score is one.
`show`	logical. If `TRUE`, then plot heatmap of projections. Default is `TRUE`.

Value

A list ot three components:

`links`	a vector with the number of significant links for each cluster.
`entropy`	a vector with the delta entropy for each cluster.
`StemIDscore`	a vector with the StemID score for each cluster.

`compTBNoise` *Function for fitting a negative binomial noise model of technical and biological variability across cells in pruned k-nearest neighbourhoods.*

Description

This function fits negative binomial models to transcript counts of pruned k-nearest neighbourhoods inferred by `pruneKnn` thereby deconvoluting variability into sampling noise, global cell-to-cell variability of transcript counts, and residual variability, which corresponds to biological noise.

Usage

```
compTBNoise(
  res,
  expData,
  pvalue = 0.01,
  genes = NULL,
  minN = 5,
  no_cores = NULL,
  gamma = 0.5,
  x0 = 0,
  lower = 0,
  upper = 100
)
```

Arguments

res	List object with k nearest neighbour information returned by pruneKnn function.
expData	Matrix of gene expression values with genes as rows and cells as columns. These values have to correspond to unique molecular identifier counts.
pvalue	Positive real number between 0 and 1. All nearest neighbours with link probability < pvalue are discarded. Default is 0.01.
genes	Vector of gene names corresponding to a subset of rownames of expData. Only for these genes local gene expression variability is computed. Default is NULL and values for all genes are returned.
minN	Positive integer number. Noise inference is only done for k-nearest neighbourhoods with at least minN neighbours remaining after pruning.
no_cores	Positive integer number. Number of cores for multithreading. If set to NULL then the number of available cores minus two is used. Default is NULL.
gamma	Positive real number. Scale paramter of the cauchy prior. Default is 0.5.
x0	Real number greater or equal to zero. Location parameter of the cauchy prior.
lower	Real number greater or equal to zero. Lower bound for the maximum a posterior inference of the biological noise. Default is 0.
upper	Real number greater or equal to zero. Upper bound for the maximum a posterior inference of the biological noise. Default is 100.

Value

List object of three components:

mu	Vector of mean expression for all k-nearest neighbourhoods. Componenets are set to NA if less than minN neighbours are present in pruned neighbourhood.
rt	Vector of dispersion parameters capturing global cell-to-cell variability of transcript counts for all k-nearest neighbourhoods. Componenets are set to NA if less than minN neighbours are present in pruned neighbourhood.

epsilon	Matrix of biological noise estimates for all genes across for all k-nearest neighbourhoods. Componenets are set to NA if less than minN neighbours present in pruned neighbourhood.
pars	List of parameters.

Examples

```
## Not run:
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
noise <- compTBNoise(res,intestinalDataSmall,pvalue=0.01,genes = NULL,no_cores=1)

## End(Not run)
```

---

comptsne *Computation of a two dimensional t-SNE representation*

Description

This functions performs the computation of a t-SNE map from the distance object in slot distances using the **Rtsne** package.

Usage

```
comptsne(
  object,
  dimRed = FALSE,
  initial_cmd = TRUE,
  perplexity = 30,
  rseed = 15555
)
```

Arguments

object	SCseq class object.
dimRed	logical. If TRUE then the t-SNE is computed from the feature matrix in slot dimRed$x (if not equal to NULL). Default is FALSE and the t-SNE is computed from the distance matrix stored in slot distances. If slot distances equals NULL dimRed is automatially set to TRUE.
initial_cmd	logical. If TRUE, then the t-SNE map computation is initialized with a configuration obtained by classical multidimensional scaling. Default is TRUE.
perplexity	Positive number. Perplexity of the t-SNE map. Default is 30.
rseed	Integer number. Random seed to enforce reproducible t-SNE map.

Value

SCseq object with t-SNE coordinates stored in slot tsne.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- comptsne(sc)
```

---

compumap *Computation of a two dimensional umap representation*

Description

This functions performs the computation of a two-dimensional umap representation based on the distance matrix in slot `distances` using the **umap** package.

Usage

```
compumap(
  object,
  dimRed = FALSE,
  n_neighbors = 15,
  metric = "euclidean",
  n_epochs = 200,
  min_dist = 0.1,
  local_connectivity = 1,
  spread = 1
)
```

Arguments

`object`	`SCseq` class object.
`dimRed`	logical. If `TRUE` then the umap is computed from the feature matrix in slot `dimRed$x` (if not equal to `NULL`). Default is `FALSE` and the umap is computed from the distance matrix stored in slot `distances`. If slot `distances` equals `NULL` `dimRed` is automatially set to `TRUE`.
`n_neighbors`	Umap parameter. See `help(umap.defaults)` after loading package **umap**. Default is 15.
`metric`	Umap parameter. See `help(umap.defaults)` after loading package **umap**. Default is "euclidean".
`n_epochs`	Umap parameter. See `help(umap.defaults)` after loading package **umap**. Default is 200.
`min_dist`	Umap parameter. See `help(umap.defaults)` after loading package **umap**. Default is 0.1.

local_connectivity
: Umap parameter. See help(umap.defaults) after loading package **umap**. Default is 1.

spread
: Umap parameter. See help(umap.defaults) after loading package **umap**. Default is 1.

Value

SCseq object with umap coordinates stored in slot umap.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- compumap(sc)
```

---

corrVar

Function for regressing out the mean-variance dependence. This function corrects for the systematic dependence of the variance on the mean by a local regression.

Description

Function for regressing out the mean-variance dependence. This function corrects for the systematic dependence of the variance on the mean by a local regression.

Usage

```
corrVar(m, v, span = 0.75, degree = 2)
```

Arguments

m
: Vector of mean expression estimates for a set of genes.

v
: Vector of variance etsimates for a set of genes.

span
: Parameter for the local regression. See help(loess). Default value is 0.75.

degree
: Parameter for the local regression. See help(loess). Default value is 2.

Value

Vector of corrected variance estimates.

createKnnMatrix *Function to create a knn matrix*

Description

This creates an adjacency matrix, keeping only nearest neighbour with a link probability above a minimum probability

Usage

```
createKnnMatrix(res, pvalue = 0.01)
```

Arguments

res	List object with k nearest neighbour information returned by pruneKnn function.
pvalue	Positive real number between 0 and 1. All nearest neighbours with link probability < pvalue are discarded. Default is 0.01.

Value

Adjacency matrix in sparse matrix format (see package **Matrix**) with positive non-zero entries only for k nearest neighours with link probability >= pvalue. The value of these entries equals the link probability.

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
y <- createKnnMatrix(res,pvalue=0.01)
```

diffexpnb *Function for differential expression analysis*

Description

This function performs differential expression analysis between two sets of single cell transcriptomes. The inference is based on a noise model or relies on the DESeq2 approach.

Usage

```
diffexpnb(
  x,
  A,
  B,
  DESeq = FALSE,
  method = "pooled",
  norm = FALSE,
```

```
  vfit = NULL,
  locreg = FALSE,
  ...
)
```

Arguments

x	expression data frame with genes as rows and cells as columns. Gene IDs should be given as row names and cell IDs should be given as column names. This can be a reduced expression table only including the features (genes) to be used in the analysis. This input has to be provided if g (see below) is given and corresponds to a valid gene ID, i. e. one of the rownames of x. The default value is NULL. In this case, cluster identities are highlighted in the plot.
A	vector of cell IDs corresponding column names of x. Differential expression in set A versus set B will be evaluated.
B	vector of cell IDs corresponding column names of x. Differential expression in set A versus set B will be evaluated.
DESeq	logical value. If TRUE, then **DESeq2** is used for the inference of differentially expressed genes. In this case, it is recommended to provide non-normalized input data x. The **DESeq2** package needs to be installed from bioconductor. Default value is FALSE.
method	either "per-condition" or "pooled". If DESeq is not used, this parameter determines, if the noise model is fitted for each set separately ("per-condition") or for the pooled set comprising all cells in A and B. Default value is "pooled".
norm	logical value. If TRUE then the total transcript count in each cell is normalized to the minimum number of transcripts across all cells in set A and B. Default value is FALSE.
vfit	function describing the background noise model. Inference of differentially expressed genes can be performed with a user-specified noise model describing the expression variance as a function of the mean expression. Default value is NULL.
locreg	logical value. If FALSE then regression of a second order polynomial is perfomed to determine the relation of variance and mean. If TRUE a local regression is performed instead. Default value is FALSE.
...	additional arguments to be passed to the low level function DESeqDataSetFromMatrix.

Value

If DESeq equals TRUE, the function returns the output of **DESeq2**. In this case list of the following two components is returned:

cds	object returned by the **DESeq2** function DESeqDataSetFromMatrix.
res	data frame containing the results of the **DESeq2** analysis.

Otherwise, a list of three components is returned:

vf1	a data frame of three columns, indicating the mean m, the variance v and the fitted variance vm for set A.

vf2 a data frame of three columns, indicating the mean m, the variance v and the fitted variance vm for set B.

res a data frame with the results of the differential gene expression analysis with the structure of the DESeq output, displaying mean expression of the two sets, fold change and log2 fold change between the two sets, the p-value for differential expression (pval) and the Benjamini-Hochberg corrected false discovery rate (padj).

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
A <- names(sc@cpart)[sc@cpart %in% c(1,2)]
B <- names(sc@cpart)[sc@cpart %in% c(3)]
y <- diffexpnb(getfdata(sc,n=c(A,B)), A=A, B=B )
```

---

diffgenes *Compute Expression Differences between Clusters*

Description

This functions computes expression differences between clusters and ranks genes by z-score differences.

Usage

```
diffgenes(object, cl1, cl2, mincount = 1)
```

Arguments

object SCseq class object.

cl1 A vector of valid cluster numbers (contained in the cpart slot of the SCseq object). Represents the first group of the comparison.

cl2 A vector of valid cluster numbers (contained in the cpart slot of the SCseq object). Represents the second group of the comparison.

mincount Minimal normalized expression level of a gene to be included into the analysis. A gene needs to be expressed at this level in at least a single cell.

Value

A list with four components:

z a vector of z-scores in decreasing order with genes up-regulated in cl1 appearing at the top of the list.

cl1	a data.frame with expression values for cells in cl1.
cl2	a data.frame with expression values for cells in cl2.
cl1n	a vector of cluster numbers for cells in cl1.
cl2n	a vector of cluster numbers for cells in cl2.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
x <- diffgenes(sc,1,2)
head(x$z)
plotdiffgenes(x,names(x$z)[1])
```

---

diffNoisyGenes *Function for extracting genes with elevated variability in a cluster*

Description

This function extracts genes with significantly elevated variability in a cluster on a basis of a Wilcoxon rank sum-test between cells in a cluster and all remaining cells.

Usage

```
diffNoisyGenes(noise, cl, set, bgr = NULL, no_cores = 1)
```

Arguments

noise	List object with the background noise model and a variability matrix, returned by the compNoise function.
cl	List object with clustering information, returned by the graphCluster function.
set	Postive integer number or vector of integers corresponding to valid cluster numbers. The function reports genes with elevated variability in all clusters contained in set.
bgr	Postive integer number or vector of integers corresponding to valid cluster numbers. Background set for comparison. The function reports genes with elevated variability in all clusters contained in set compared to clusters in bgr. Default is NULL and the comparison is against all clusters not in set.
no_cores	Positive integer number. Number of cores for multithreading. If set to NULL then the number of available cores minus two is used. Default is NULL.

Value

Data.frame reporting the log2 fold change between clusters in set and the remaining clusters and the p-value for elevated variability for each genes. Rows are ordered by decreasing log2 fold change.

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
noise <- compNoise(intestinalDataSmall,res,pvalue=0.01,genes = NULL,no_cores=1)
cl <- graphCluster(res,pvalue=0.01)
ngenes <- diffNoisyGenes(noise,cl,c(1,2),no_cores=1)
```

`diffNoisyGenesTB` *Function for extracting genes with differential biological variability in a cluster*

Description

This function infers genes with differential biological variability in a cluster versus a background set of clusters on the basis of a Wilcoxon rank sum-test between cells in a cluster and in the background set.

Usage

```
diffNoisyGenesTB(
  noise,
  cl,
  set,
  bgr = NULL,
  no_cores = 1,
  minobs = 5,
  ps = 0.1,
  rseed = 17000
)
```

Arguments

`noise`	List object with noise parameters returned by the `compTBNoise` function.
`cl`	List object with clustering information, returned by the `graphCluster` function.
`set`	Postive integer number or vector of integers corresponding to valid cluster numbers. The function reports genes with differential variability in all clusters contained in `set` versus vlusters in `bgr`.
`bgr`	Postive integer number or vector of integers corresponding to valid cluster numbers. Background set for comparison. The function reports genes with differential variability in all clusters contained in `set` compared to clusters in `bgr`. Default is `NULL` and `bgr` equals the set of all clusters not in `bgr`.
`no_cores`	Positive integer number. Number of cores for multithreading. If set to `NULL` then the number of available cores minus two is used. Default is `NULL`.
`minobs`	Positive integer number. Only genes with at least `minobs` neighbourhoods with non-zero biological noise levels in `set` are included for the p-value computation. Otherwise, a p-value or 0.5 is reported. Default is 5.

ps	Real number greater or equal to zero. A small random variable sampled from a uniform distribution in the interval [0,ps] is added to the noise quantification to avoid inclusion of genes with small noise differences. Default is 0.1.
rseed	Integer number. Random seed to enforce reproducible results. Default is 17000.

Value

Data.frame with five columns:

mu.set	Mean expression across clusters in set.
mu.bgr	Mean expression across clusters in bgr (or all clusters not in set).
mu.all	Mean expression across clusters in set and bgr (or all clusters).
eps.set	Average variability across clusters in set.
eps.bgr	Average variability across clusters in bgr (or all clusters not in set).
eps.all	Average variability across clusters in set and bgr (or all clusters).
log2FC	log2 fold change of variability between between clusters in set and clusters in bgr (or all clusters).
pvalue	Banjamini-Hochberg corrected Wilcoxon rank sum test p-value for differential variability.

Rows are ordered by decreasing log2 fold change of variability.

Examples

```
## Not run:
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
noise <- compTBNoise(res,intestinalDataSmall,pvalue=0.01,genes = NULL,no_cores=1)
cl <- graphCluster(res,pvalue=0.01)
ngenes <- diffNoisyGenesTB(noise,cl,c(1,2),no_cores=1)

## End(Not run)
```

---

extractCounts *Function for filtering count data*

Description

This function discards lowly expressed genes from a count matrix stored in an SCseq object, and returns (normalized or non-normalized) gene expression or noise values.

Usage

```
extractCounts(
  object,
  minexpr = 5,
  minnumber = 5,
  noise = FALSE,
  pt = NULL,
  n = NULL,
  g = NULL,
  norm = TRUE
)
```

Arguments

object	SCseq class object.
minexpr	Integer number greater or equal to zero. Minimum expression of a gene in at least minnumber cells to not be discarded. Default is 5.
minnumber	Integer number greater or equal to zero. Minimum number of cells required to have at least minexpr transcript counts for a gene to not be discarded. Default is 5.
noise	logical. If TRUE, then noise (in object@noise) is returned for the filtered genes and cells. Default is FALSE and gene expression counts are returned.
pt	List object returned by function pseudoTime. If given, then feature matrix is returned for cells in pt$ord and ordered by pseudo-time. Default is NULL and feature matrix is returned for all cells in object$ndata.
n	Vector of valid column names corresponding to a subset of valid column names of the object@ndata. Default is NULL filtering is done on all cells in object@ndata. Only considered if pt is NULL.
g	Vector of gene IDs (valid row names of object@ndata). If given, then all genes not in g are discarded prior to filtering. Default is NULL and filtering is done on all genes in object@ndata.
norm	logical. If TRUE, then transcipt counts are normalized to the minimum number of total transcript counts across all cells in the feature matrix.

Value

Filtered expression matrix.

---

filterdata *Data filtering*

Description

This function allows filtering of genes and cells to be used in the RaceID3 analysis. It also can perform batch effect correction using an internal method or a recently published alternative mnnCorrect from the **batchelor** package.

Usage

```
filterdata(
  object,
  mintotal = 3000,
  minexpr = 5,
  minnumber = 5,
  LBatch = NULL,
  knn = 10,
  CGenes = NULL,
  FGenes = NULL,
  ccor = 0.4,
  bmode = "RaceID",
  verbose = TRUE
)
```

Arguments

object	SCseq class object.
mintotal	minimum total transcript number required. Cells with less than mintotal transcripts are filtered out. Default is 3000.
minexpr	minimum required transcript count of a gene in at least minnumber cells. All other genes are filtered out. Default is 5.
minnumber	See minexpr. Default is 5.
LBatch	List of experimental batches used for batch effect correction. Each list element contains a vector with cell names (i.e. column names of the input expression data) falling into this batch. Default is NULL, i.e. no batch correction.
knn	Number of nearest neighbors used to infer corresponding cell types in different batches. Defult is 10.
CGenes	List of gene names. All genes with correlated expression to any of the genes in CGenes are filtered out for cell type inference. Default is NULL.
FGenes	List of gene names to be filtered out for cell type inference. Default is NULL.
ccor	Correlation coefficient used as a trehshold for determining genes correlated to genes in CGenes. Only genes correlating less than ccor to all genes in CGenes are retained for analysis. Default is 0.4.
bmode	Method used for batch effect correction. Any of "RaceID","mnnCorrect". If mnnCorrect from the **batchelor** package is desired, this package needs to be installed from bioconductor. Default is "RaceID".
verbose	logical. If FALSE then status output messages are disabled. Default is TRUE.

Value

An SCseq class object with filtered and normalized expression data.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
```

findoutliers *Inference of outlier cells and final clustering*

Description

This functions performs the outlier identification based on the clusters infered with the clustexp function.

Usage

```
findoutliers(
  object,
  probthr = 0.001,
  outminc = 5,
  outlg = 2,
  outdistquant = 0.95,
  verbose = TRUE
)
```

Arguments

object	SCseq class object.
probthr	outlier probability threshold for a minimum of outlg genes to be an outlier cell. This probability is computed from a negative binomial background model of expression in a cluster. Default is 0.001.
outminc	minimal transcript count of a gene in a clusters to be tested for being an outlier gene. Default is 5.
outlg	Minimum number of outlier genes required for being an outlier cell. Default is 2.
outdistquant	Real number between zero and one. Outlier cells are merged to outlier clusters if their distance smaller than the outdistquant-quantile of the distance distribution of pairs of cells in the orginal clusters after outlier removal. Default is 0.95.
verbose	logical. If FALSE then status output messages are disabled. Default is TRUE.

Value

SCseq object with outlier data stored in slot out and slot outlierpar. The final clustering partition is stored in cpart.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
```

`fitBackVar` *Function for computing a background model of gene expression variability*

Description

This funtion fits a second order polynomial to the variance-mean dependence across all genes in log space.

Usage

```
fitBackVar(x, mthr = -1)
```

Arguments

`x`	Matrix of transcript counts with genes as rows and cells as columns.
`mthr`	Real number. Threshold of log2 mean expression. Genes with mean expression < `mthr` are discarded prior to fitting the polynomial. Default is -1.

Value

List object of four components:

`fit`	model fit as returned by the `lm` function.
`genes`	genes with expression variance greater than the polynomial fit.
`m`	mean expression of all genes
`v`	expression variance of all genes

Examples

```
bg <- fitBackVar(intestinalDataSmall)
```

`fitGammaRt` *Fitting a Gamma distribution to global cell-to-cell variability*

Description

This function fits a Gamma distribution to the total transcript counts distribution across a given group of cells. Total transcript counts are normalized by the mean total transcript count across the group. This function is used to infer a Gamma distribution of the global cell-to-cell variability across pruned nearest neighbourhoods.

Usage

```
fitGammaRt(x)
```

Arguments

x	Transcript count matrix with cells as columns and genes as rows.

Value

Shape parameter of the Gamma distribution. This parameter corresponds to the dispersion explained by the global cell-to-cell variability of UMI counts in a negative binomial model.

`fitLogVarLogMean` *Second order polynomial fit of mean-variance dependence This function corrects for the systematic dependence of the variance on the mean by a local regression.*

Description

Second order polynomial fit of mean-variance dependence This function corrects for the systematic dependence of the variance on the mean by a local regression.

Usage

```
fitLogVarLogMean(x)
```

Arguments

x	Matrix of transcript counts with genes as rows and cells as columns.

Value

Second order polynomial model as obtained by `lm`.

`fitNBtb` *Function for fitting a negative binomial noise model of technical and biological variability*

Description

This function fits a negative binomial model to transcript counts of a group of cells thereby deconvoluting variability into sampling noise, global cell-to-cell variability of transcript counts, and residual variability, which corresponds to biological noise.

Usage

```
fitNBtb(z, gamma = 2, x0 = 0, lower = 0, upper = 100, grad = TRUE)
```

Arguments

z	Transcript count matrix with cells as columns and genes as rows.
gamma	Positive real number. Scale paramter of the cauchy prior. Default is 2.
x0	Real number greater or equal to zero. Location parameter of the cauchy prior.
lower	Real number greater or equal to zero. Lower bound for the maximum a posterior inference of the biological noise. Default is 0.
upper	Real number greater or equal to zero. Upper bound for the maximum a posterior inference of the biological noise. Default is 100.
grad	Logical. If TRUE then maximum a posterior value is inferred by determining the root of the gradient function. Otherwise, the maximum of the posterior probability is determined numerically. Default is TRUE.

Value

Data.frame with four columns:

mu	Mean expression.
epsilon	Biological noise.
rt	Dispersion parameter capturing global cell-to-cell variability of transcript counts.
alphaG	Dispersion parameter capturing global cell-to-cell variability of transcript counts and biological noise.

---

fitNBtbCl *Function for fitting a negative binomial noise model of technical and biological variability*

Description

This function fits a negative binomial model to transcript counts of a group of cells thereby deconvoluting variability into sampling noise, global cell-to-cell variability of transcript counts, and residual variability, which corresponds to biological noise. Local mean and and global cell-to-cell variability of transcript counts are pre-computed arguments.

Usage

```
fitNBtbCl(z, mu, rt, gamma = 2, x0 = 0.1, lower = 0, upper = 100)
```

Arguments

z	Transcript count matrix with cells as columns and genes as rows.
mu	Vector of mean expression values across cells in z.
rt	Vector of dispersion parameters explaining global cell-to-cell variability of transcript counts across cells in z.
gamma	Positive real number. Scale paramter of the cauchy prior. Default is 2.

x0	Real number greater or equal to zero. Location parameter of the cauchy prior.
lower	Real number greater or equal to zero. Lower bound for the maximum a posterior inference of the biological noise. Default is 0.
upper	Real number greater or equal to zero. Upper bound for the maximum a posterior inference of the biological noise. Default is 100.

Value

Vector of biological noise parameters across cells in z.

---

fractDotPlot *Dotplot of gene expression across clusters or samples*

Description

This is a plotting function for visualizing gene expression across subsets of clusters or samples. The diameter of a dot reflects the fraction of cells expressing a gene, and the color indicates the expression z-score across all clusters or samples.

Usage

```
fractDotPlot(
  object,
  genes,
  cluster = NULL,
  samples = NULL,
  subset = NULL,
  zsc = FALSE,
  logscale = TRUE,
  cap = Inf,
  flo = -Inf
)
```

Arguments

object	SCseq class object.
genes	vector of valid gene names corresponding to row names of slot ndata. The expression for this genes is shown.
cluster	vector of valid cluster numbers contained in slot cpart. Default is NULL. If not given, then the samples argument is expected. If both are given, only the samples argument is considered. If both are NULL, then cluster is initialized with all clusters.
samples	vector of sample names for all cells. Length and order has to correspond to colnames of slot ndata. Default is NULL.

subset	vector of unique sample names to show in the expression dotplot. Each sample names in subset has to occur in samples. Default is NULL. If not given and samples is not NULL, the subset is intialized with all sample names occuring in samples.
zsc	logical. If TRUE then a z-score transformation is applied. Default is FALSE.
logscale	logical. If TRUE then a log2 transformation is applied. Default is TRUE.
cap	real number. Upper limit for the expression, log2 expression, or z-score. Values larges then cap are replaced by cap.
flo	real number. Lower limit for the expression, log2 expression, or z-score. Values smaller then flo are replaced by flo.

Value

None

---

getExpData *Function for extracting a filtered expression matrix from a* **RaceID** SCseq *object*

Description

This function for extracts a filtered expression matrix from a **RaceID** SCseq object. The filterdata function from the **RaceID** package has to be run on the SCseq object before.

Usage

```
getExpData(object, genes = NULL)
```

Arguments

object	**RaceID** SCseq object.
genes	Vector of valid gene identifiers corresponding to valid rownames of the input expression data. An expression matrix is returned only for these genes. Default is NULL and an expression matrix is returned for all genes retained after filtering of the SCseq object, i.e. all genes in genes slot of the SCseq object.

Value

noise Sparse Matrix with genes as rows and cells as columns after filtering.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
d <- getExpData(sc)
res <- pruneKnn(d,distM=sc@distances,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
```

getfdata *Extracting filtered expression data*

Description

This functions allows the extraction of a filtered and normalized expression matrix

Usage

```
getfdata(object, g = NULL, n = NULL)
```

Arguments

object	SCseq class object.
g	Vector of gene names to be included corresponding to a subset of valid row names of the ndata slot of the SCseq object. Default is NULL and data for all genes remaining after filtering by the filterdata function are shown.
n	Vector of valid column names corresponding to a subset of valid column names of the ndata slot of the SCseq object. Default is NULL and data for all cells remaining after filtering by the filterdata function are shown.

Value

Matrix of filtered expression data with genes as rows and cells as columns.

getFilteredCounts *Function for filtering count data*

Description

This function discards lowly expressed genes from a count matrix stored in an SCseq object.

Usage

```
getFilteredCounts(object, minnumber = 5, minexpr = 5)
```

Arguments

object	SCseq class object.
minnumber	Integer number greater or equal to zero. Minimum number of cells required to have at least minexpr transcript counts for a gene to not be discarded. Default is 5.
minexpr	Integer number greater or equal to zero. Minimum expression of a gene in at least minnumber cells to not be discarded. Default is 5.

Value

Filtered expression matrix.

`getNode` *Extract all genes for a module in a FateID self-orgaizing map*

Description

Extract a vector of all genes corresponding to a given module of a FateID self-organizing map (SOM) of pseudo-time ordered gene expression (or noise) profiles.

Usage

```
getNode(ps, n)
```

Arguments

`ps`	FateID SOM. List object.
`n`	Integer number of SOM module.

Value

Vector of gene IDs in module `n`.

`getproj` *Extract Projections of all Cells from a Cluster*

Description

This function extracts projections of all cells in a cluster and plots a heatmap of these hierarchically clustered projections (rows) to all other clusters (columns). A minimum spanning tree of the cluster centers is overlaid for comparison.

Usage

```
getproj(object, i, show = TRUE, zscore = FALSE)
```

Arguments

`object`	`Ltree` class object.
`i`	Cluster number. This number has to correspond to one of the RaceID3 clusters included for the StemID2 inference, i.e. to a number present in slot `ldata$lp`.
`show`	logical. If `TRUE`, then plot heatmap of projections. Default is `TRUE`.
`zscore`	logical. If `TRUE` and `show=TRUE`, then plot z-score-transformed projections. If `TRUE` and `show=FALSE`, then plot untransformed projections. Default is `FALSE`.

Value

A list ot two components:

pr	a data.frame of projections for all cells in cluster i (rows) onto all other clusters (columns).
prz	a data.frame of z-transformed projections for all cells in cluster i (rows) onto all other clusters (columns).

`graphCluster` *Function for infering clustering of the pruned k nearest neighbour graph*

Description

This function derives a graph object from the pruned k nearest neighbours and infers clusters by modularity optimizatio nusing the Louvain or the Leiden algorithm on this graph. A Fruchterman-Rheingold graph layout is also derived from the pruned nearest neighbours.

Usage

```
graphCluster(
  res,
  pvalue = 0.01,
  use.weights = TRUE,
  use.leiden = FALSE,
  leiden.resolution = 1,
  min.size = 2,
  rseed = 12345
)
```

Arguments

res	List object with k nearest neighbour information returned by pruneKnn function.
pvalue	Positive real number between 0 and 1. All nearest neighbours with link probability < pvalue are discarded. Default is 0.01.
use.weights	logical. If TRUE, then nearest-neighbor link probabilities are used to build a graph as input for Louvain clustering. If FALSE, then all links have equal weight. Default is TRUE.
use.leiden	logical. If TRUE, then the Leiden algorithm is used. If FALSE, the Louvain algorithm is used. Default is FALSE.
leiden.resolution	Positive real number. Resolution parameter for the Leiden algorithm.
min.size	Positive integer number. Minimum cluster size. All clusters with less than min.size elements are aggregated into one cluster, to which the largest cluster number is assigned. See output value residual.cluster. Default value is 2.

rseed — Integer number. Random seed to enforce reproducible clustering results. Default is 12345.

Value

List object of three components:

partition — Vector with clustering partition.

fr — Data.frame with Fruchterman-Rheingold graph layout.

residual.cluster — In case clusters with less than min.size elements occur in the cluster partition, these are grouped into a common cluster, to which the largest cluster number is assigned. If this grouping was done, the cluster number is given by this value. Otherwise, the value of this object is NULL.

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
cl <- graphCluster(res,pvalue=0.01)
```

---

imputeexp *Imputed expression matrix*

Description

This functions returns an imputed expression matrix based on the imputing computed with compdist.

Usage

```
imputeexp(object, genes = NULL)
```

Arguments

object — SCseq class object.

genes — vector of valid gene names corresponding to row names of slot ndata. Default is NULL and imputing is done for all genes.

Value

An expression matrix with imputed expression values after size normalization. Genes are in rows and cells in columns.

inspectKNN *Function for inspecting pruned k-nearest neighbourhoods*

Description

This function allows inspection of the local background model and the pruning of nearest neighbours for a given cell. A dimensional reduction representation is plotted where k nearest neighours and outliers are highlighted. Alternatively, the dependence of the transcript count variance or, alternatively, the coefficient of variation (CV) on the mean in log2 space is plotted. The mean-variance dependence is plotted along with a loess-regression, a second order polynomial fit, and the background model of the local variability. The CV plot also highlights the local variability associated with cell-to-cell variability of total transcript counts, as calculated directly from the mean and variance of total transcript counts (turquoise) or from a local fit of a gamma distribution (orange).

Usage

```
inspectKNN(
  i,
  expData,
  res,
  cl,
  object = NULL,
  nb = res$pars$nb,
  pvalue = 0.01,
  backModel = NULL,
  alpha = res$alpha[i],
  plotSymbol = FALSE,
  id = NULL,
  degree = 2,
  span = 0.75,
  cv = FALSE,
  ...
)
```

Arguments

i	Either integer column index or column name of expData. Pruning is inspected for the neighbourhood of this cell.
expData	Matrix of gene expression values with genes as rows and cells as columns. These values have to correspond to unique molecular identifier counts.
res	List object with k nearest neighbour information returned by pruneKnn function.
cl	List object with clustering information, returned by the graphCluster function.
object	SCseq class object. Required if plotSymbol is TRUE. Default is NULL.
nb	Input parameter of pruneKnn. See help(pruneKnn). Default is res$pars$nb.
pvalue	Positive real number between 0 and 1. All nearest neighbours with link probability < pvalue are pruned. Default is 0.01.

backModel — Optional background model. Second order polynomial fitting the mean-variance dpendence on log2 scales as returned by lm. Default is NULL and the local background model is computed as in pruneKnn.

alpha — Input parameter of pruneKnn. See help(pruneKnn). Default is res$pars$alpha.

plotSymbol — Logical. If TRUE then a dimensional reduction representation is plotted highlighting cell i, all k nearest neighbours, all outliers, and the stringest outlier in different colours. Function plotsymbolsmap is used. Additional parameter for this function, such as um=TRUE can be given. Default is FALSE, and the local mean-variance dependence is plotted along with a second order polynomial fit and a local regression. See plotMV.

id — Valid column name of expData. If plotSymbol=TRUE this corresponding cell is highlighted in the dimensional reduction representation.

degree — Input parameter for mean-variance fit. See plotMV.

span — Input parameter for mean-variance fit. See plotMV.

cv — Input parameter for mean-variance fit. See plotMV.

... — Additional parameters for plotsymbolsmap.

Value

List object with six components:

pv.neighbours.cell
: Vector of outlier p-values (Bonferroni-corrected) for each of the k-nearest neighbours.

cluster.neighours
: Vector of cluster numbers for central cell and each of the k-nearest neighbours.

alpha
: alpha parameter used for pruning.

expr.neighbours
: Matrix of normalized transcript counts for the central cell and each of the k-nearest neighbours (normalized to the minimum number of total trascript counts across all neighours). Additional columns indicate inferred local mean, standard deviation, and strongest outlier p-value. Rows are sorted by p-values of the strongest outlier cell in increasing order.

pv.neighbours
: Matrix of outlier p-values of all genes for the central cells and each of the k-nearest neighbours. Rows are sorted by p-values of the strongest outlier cell in increasing order.

strongest.outlier
: Column name of strongest outlier.

intestinalData *Single-cell transcriptome data of intestinal epithelial cells*

Description

This dataset contains gene expression values, i. e. transcript counts, of 278 intestinal epithelial cells.

Usage

```
intestinalData
```

Format

A sparse matrix (using the **Matrix**) with cells as columns and genes as rows. Entries are raw transcript counts.

Value

None

References

Grün et al. (2016) Cell Stem Cell 19(2): 266-77 <DOI:10.1016/j.stem.2016.05.010> (PubMed)

intestinalDataSmall *Single-cell transcriptome data of intestinal epithelial cells*

Description

This dataset is a smaller subset of the original dataset, which contains gene expression values, i. e. transcript counts, of 278 intestinal epithelial cells. The dataset is included for quick testing and examples. Only cells with >10,000 transcripts per cell and only genes with >20 transcript counts in >10 cells were retained.

Usage

```
intestinalDataSmall
```

Format

A sparse matrix (using the **Matrix**) with cells as columns and genes as rows. Entries are raw transcript counts.

Value

None

References

Grün et al. (2016) Cell Stem Cell 19(2): 266-77 <DOI:10.1016/j.stem.2016.05.010> (PubMed)

`lineagegraph` *Inference of a Lineage Graph*

Description

This function assembles a lineage graph based on the cell projections onto inter-cluster links.

Usage

```
lineagegraph(object, verbose = TRUE)
```

Arguments

`object`	`Ltree` class object.
`verbose`	logical. If `FALSE` then status output messages are disabled. Default is `TRUE`.

Value

An Ltree class object with lineage graph-related data stored in slots `ltcoord`, `prtree`, and `cdata`.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- comptsne(sc)
ltr <- Ltree(sc)
ltr <- compentropy(ltr)
ltr <- projcells(ltr)
ltr <- lineagegraph(ltr)
```

Ltree-class *The Ltree Class*

Description

The Ltree class is the central object storing all information generated during lineage tree inference by the StemID algorithm. It comprises a number of slots for a variety of objects.

Arguments

object	An Ltree object.

Slots

sc An SCseq object with the RaceID3 analysis of the single-cell RNA-seq data for which a lineage tree should be derived.

ldata List object storing information on the clustering partition, the distance matrix, and the cluster centers in dimensionally-reduced input space and in two-dimensional t-sne space. Elements: lp: vector with the filtered partition into clusters after discarding clusters with cthr cells or less. pdi:matrix with the coordinates of all cells in the embedded space. Clusters with cthr transcripts or less were discarded (see function projcells). Rows are medoids and columns are coordinates. cn: data.frame with the coordinates of the cluster medoids in the embedded space. Clusters with cthr transcripts or less were discarded. Rows are medoids and columns are coordinates. m: vector with the numbers of the clusters which survived the filtering. pdil: data.frame with coordinates of cells in the two-dimensional t-SNE representation computed by RaceID3. Clusters with cthr transcripts or less were discarded. Rows are cells and columns are coordinates. cnl: data.frame with the coordinates of the cluster medoids in the two-dimensional t-SNE representation computed by RaceID3. Clusters with cthr transcripts or less were discarded. Rows are medoids and columns are coordinates.

entropy Vector with transcriptome entropy computed for each cell.

trproj List containing two data.frames. Elements: res: data.frame with three columns for each cell. The first column o shows the cluster of a cell, the second column l shows the cluster number for the link the cell is assigned to, and the third column h shows the projection as a fraction of the length of the inter-cluster link. Parallel projections are positive, while anti-parallel projections are negative. rma: data.frame with all projection coordinates for each cell. Rows are cells and columns are clusters. Projections are given as a fraction of the length of the inter-cluster link. Parallel projections are positive, while anti-parallel projections are negative. The column corresponding to the originating cluster of a cell shows NA.

par List of parameters used for the StemID2 analysis.

prback data.frame of the same structure as the trproj$res. In case randomizations are used to compute significant projections, the projections of all pdishuff randomizations are appended to this data.frame and therefore the number of rows corresponds to the number of cells multiplied by pdishuf. See function projback.

prbacka data.frame reporting the aggregated results of the randomizations with four columns. Column n denotes the number of the randomization sample, column o and l contain the numbers

of the originating and the terminal cluster, respectively, for each inter-cluster link and column count shows the number of cells assigned to this link in randomization sample n. The discrete distribution for the computation of the link p-value is given by the data contained in this object (if nmode=FALSE).

ltcoord Matrix storing projection coordinates of all cells in the two-dimensional t-SNE space, used for visualization.

prtree List with two elements. The first element l stores a list with the projection coordinates for each link. The name of each element identifies the link and is composed of two cluster numbers separated by a dot. The second element n is a list of the same structure and contains the cell names corresponding to the projection coordinates stored in l.

cdata list of data.frames, each with cluster ids as rows and columns: counts data.frame indicating the number of cells on the links connecting the cluster of origin (rows) to other clusters (columns). counts.br data.frame containing the cell counts on cluster connections averaged across the randomized background samples (if nmode = FALSE) or as derived from sampling statistics (if nmode = TRUE). pv.e matrix of enrichment p-values estimated from sampling statistics (if nmode = TRUE); entries are 0 if the observed number of cells on the respective link exceeds the (1 - pethr)-quantile of the randomized background distribution and 0.5 otherwise (if nmode = FALSE). pv.d matrix of depletion p-values estimated from sampling statistics (if nmode = TRUE); entries are 0 if the observed number of cells on the respective link is lower than the pethr-quantile of the randomized background distribution and 0.5 otherwise (if nmode = FALSE). pvn.e matrix of enrichment p-values estimated from sampling statistics (if nmode = TRUE); 1- quantile, with the quantile estimated from the number of cells on a link as derived from the randomized background distribution (if nmode = FALSE). pvn.d matrix of depletion p-values estimated from sampling statistics (if nmode = TRUE); quantile estimated from the number of cells on a link as derived from the randomized background distribution (if nmode = FALSE).

---

maxNoisyGenes *Function for extracting genes maximal variability*

Description

This function extracts genes with maximal variability in a cluster or in the entire data set.

Usage

```
maxNoisyGenes(noise, cl = NULL, set = NULL)
```

Arguments

noise	List object with the background noise model and a variability matrix, returned by the compNoise function.
cl	List object with clustering information, returned by the graphCluster function. Default is NULL.
set	Postive integer number or vector of integers corresponding to valid cluster numbers. Noise levels are computed across all cells in this subset of clusters. Default is NULL and noise levels are computed across all cells.

Value

Vector with average gene expression variability in decreasing order, computed across all cells or only cells in a set of clusters (if `cl` and `set` are given.

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
noise <- compNoise(intestinalDataSmall,res,pvalue=0.01,genes = NULL,no_cores=1)
mgenes <- maxNoisyGenes(noise)
```

`maxNoisyGenesTB` *Function for extracting genes maximal variability*

Description

This function extracts genes with maximal variability in a cluster or in the entire data set.

Usage

```
maxNoisyGenesTB(noise, cl = NULL, set = NULL, minobs = 5)
```

Arguments

`noise`	List object with noise parameters returned by the `compTBNoise` function.
`cl`	List object with clustering information, returned by the `graphCluster` function. Default is `NULL`.
`set`	Postive integer number or vector of integers corresponding to valid cluster numbers. Noise levels are computed across all cells in this subset of clusters. Default is `NULL` and noise levels are computed across all cells.
`minobs`	Positive integer number. Only genes with at least `minobs` neighbourhoods with non-zero biological noise levels in `set` are included. Default is 5.

Value

Vector with average gene expression variability in decreasing order, computed across all cells or only cells in a set of clusters (if `cl` and `set` are given.

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
noise <- compNoise(intestinalDataSmall,res,pvalue=0.01,genes = NULL,no_cores=1)
mgenes <- maxNoisyGenes(noise)
```

`noiseBaseFit` *Function for computing a fit to the baseline of gene expression variability*

Description

This function fits a second order polynomial to the baseline variance-mean dependence across all genes in log space.

Usage

```
noiseBaseFit(x, step = 0.01, thr = 0.05)
```

Arguments

`x`	Matrix of gene expression values with genes as rows and cells as columns.
`step`	Positive real number between 0 and 1. Bin size for the computation. The interval of mean gene expression values is divided into bins with equal number of data points and `step` equals the fraction of data points in each bin. Default is 0.01.
`thr`	Positive real number between 0 and 1. In each mean expression bin defined by `step` the lowest `thr`-quantile of the gene expression variance distribution is selected. The selected data points from all bins are used for a second order polynomial fit of the variance-mean dependence in log space. Default is 0.05.

Value

List object of three components:

`nfit`	model fit as returned by the `lm` function.
`m`	mean expression of all genes
`v`	expression variance of all genes

Examples

```
x <- noiseBaseFit(intestinalDataSmall,step=.01,thr=.05)
```

plotB *Boxplots for features across clusters*

Description

Function to generate boxplots of a feature vector across different clusters.

Usage

```
plotB(x, part, cluster = NULL, set = NULL, ...)
```

Arguments

x	Vector of real numbers.
part	Vector with cluster partition, e.g., element partition returned by the graphCluster function.
cluster	Positive integer corresponing to valid cluster number or NULL. If valid cluster number, then a horizontal line is drawn indicating the median value of x for the corresponding cluster. If NULL no line is drawn. Default is NULL.
set	Ordered set of valid cluster numbers. If box equals TRUE than data will only be plotted for these clusters in the give
...	Additional parameters of boxplot.

Value

None

plotbackground *Plot Background Model*

Description

This functions produces a scatter plot showing the gene expression variance as a function of the mean and the inferred polynomial fit of the background model computed by RaceID3. It also shows a local regression.

Usage

```
plotbackground(object)
```

Arguments

object	SCseq class object.

Value

None

plotBackVar — *Function for plottinhg the background model of gene expression variability*

Description

This function plots the variance against mean expression across all genes and a second order polynomial to the variance-mean dependence in log space. It also plots a local regression.

Usage

```
plotBackVar(x)
```

Arguments

x	List object returned by function `fitBackVar` or list object returned by function `pruneKnn` (if it was run with `FSelect=TRUE`).

Value

None

Examples

```
bg <- fitBackVar(intestinalDataSmall)
plotBackVar(bg)
```

plotdiffgenes — *Barplot of differentially expressed genes*

Description

This functions produces a barplot of differentially expressed genes derived by the function `diffgenes`

Usage

```
plotdiffgenes(z, gene)
```

Arguments

z	Output of `diffgenes`
gene	Valid gene name. Has to correspond to one of the rownames of the `ndata` slot of the `SCseq` object.

Value

None

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
x <- diffgenes(sc,1,2)
head(x$z)
plotdiffgenes(x,names(x$z)[1])
```

`plotdiffgenesnb` *Function for plotting differentially expressed genes*

Description

This is a plotting function for visualizing the output of the `diffexpnb` or `clustdiffgenes` function as MA plot.

Usage

```
plotdiffgenesnb(
  x,
  pthr = 0.05,
  padj = TRUE,
  lthr = 0,
  mthr = -Inf,
  Aname = NULL,
  Bname = NULL,
  show_names = TRUE,
  ...
)
```

Arguments

`x`	output of the function `diffexpnb`.
`pthr`	real number between 0 and 1. This number represents the p-value cutoff applied for displaying differentially expressed genes. Default value is 0.05. The parameter `padj` (see below) determines if this cutoff is applied to the uncorrected p-value or to the Benjamini-Hochberg corrected false discovery rate.
`padj`	logical value. If `TRUE`, then genes with a Benjamini-Hochberg corrected false discovery rate lower than `pthr` are displayed. If `FALSE`, then genes with a p-value lower than `pthr` are displayed.
`lthr`	real number between 0 and Inf. Differentially expressed genes are displayed only for log2 fold-changes greater than `lthr`. Default value is 0.
`mthr`	real number between -Inf and Inf. Differentially expressed genes are displayed only for log2 mean expression greater than `mthr`. Default value is -Inf.

Aname	name of expression set A, which was used as input to diffexpnb. If provided, this name is used in the axis labels. Default value is NULL.
Bname	name of expression set B, which was used as input to diffexpnb. If provided, this name is used in the axis labels. Default value is NULL.
show_names	logical value. If TRUE then gene names displayed for differentially expressed genes. Default value is FALSE.
...	Additional arguments for function plot.

Value

None

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
A <- names(sc@cpart)[sc@cpart %in% c(1,2)]
B <- names(sc@cpart)[sc@cpart %in% c(3)]
y <- diffexpnb(getfdata(sc,n=c(A,B)), A=A, B=B )
plotdiffgenesnb(y)
```

---

plotDiffNoise *Function for plotting differentially variable genes*

Description

This is a plotting function for visualizing the output of the diffNoisyGenesTB function as MA plot.

Usage

```
plotDiffNoise(
  x,
  pthr = 0.05,
  mu = TRUE,
  lthr = 0,
  ps = 0.01,
  mthr = -Inf,
  set.name = NULL,
  bgr.name = NULL,
  show_names = TRUE
)
```

Arguments

x	output of the function diffNoisyGenesTB.
pthr	real number between 0 and 1. This number represents the p-value cutoff applied for displaying differentially variable genes. Default value is 0.05.
mu	logical value. If TRUE then the log2 fold change in variability is plotted as a function of log2 average expresion. Otherwise, it is plotted as a function of mean variability.
lthr	real number between 0 and Inf. Differentially variable genes are displayed only for log2 fold-changes greater than lthr. Default value is 0.
ps	positive real number. Pseudo-count added to component mu.all and epsilon.all of argument x to avoid taking logarithm of zero. Default is 0.01.
mthr	real number between -Inf and Inf. Differentially variable genes are displayed only for log2 mean expression (or mean noise, if mu equals FALSE) greater than mthr. Default value is -Inf.
set.name	name of set, which was used as input to diffNoisyGenesTB. If provided, this name is used in the axis labels. Default value is NULL.
bgr.name	name of bgr, which was used as input to diffNoisyGenesTB. If provided, this name is used in the axis labels. Default value is NULL.
show_names	logical value. If TRUE then gene names displayed for differentially variable genes. Default value is FALSE.

Value

None

---

plotdimsat *Plotting the Saturation of Explained Variance*

Description

This functions plots the explained variance as a function of PCA/ICA components computed by the function CCcorrect. The number of components where the change in explained variability upon adding further components approaches linear behaviour demarcates the saturation point and is highlighted in blue.

Usage

```
plotdimsat(object, change = TRUE, lim = NULL)
```

Arguments

object	SCseq class object.
change	logical. If TRUE then the change in explained variance is plotted. Default is FALSE and the explained variance is shown.
lim	Number of components included for he calculation and shown in the plot. Default is NULL and all components are included.

Value

None

`plotdistanceratio` *Histogram of Cell-to-Cell Distances in Real versus Embedded Space*

Description

This function plots a histogram of the ratios of cell-to-cell distances in the original versus the high-dimensional embedded space used as input for the StemID2 inferences. The embedded space approximates correlation-based distances by Euclidean distances obtained by classical multi-dimensional scaling. A minimum spanning tree of the cluster centers is overlaid for comparison.

Usage

```
plotdistanceratio(object)
```

Arguments

`object`	`Ltree` class object.

Value

None.

`plotexpmap` *Highlighting gene expression in a dimensional reduction representation*

Description

This functions highlights gene expression in a two-dimensional t-SNE map, UMAP, or a Fruchterman-Rheingold graph layout of the singe-cell transcriptome data.

Usage

```
plotexpmap(
  object,
  g,
  n = NULL,
  logsc = FALSE,
  imputed = FALSE,
  fr = FALSE,
  um = FALSE,
  cells = NULL,
```

```
  cex = 0.5,
  map = TRUE,
  leg = TRUE,
  noise = FALSE
)
```

Arguments

object	SCseq class object.
g	Individual gene name or vector with a group of gene names corresponding to a subset of valid row names of the ndata slot of the SCseq object.
n	String of characters representing the title of the plot. Default is NULL and the first element of g is chosen.
logsc	logical. If TRUE, then gene expression values are log2-transformed after adding a pseudo-count of 0.1. Default is FALSE and untransformed values are shown.
imputed	logical. If TRUE and imputing was done by calling compdist with knn > 0, then imputed expression values are shown. If FALSE, then raw counts are shown. Default is FALSE.
fr	logical. If TRUE then plot Fruchterman-Rheingold layout. Default is FALSE.
um	logical. If TRUE then plot umap dimensional reduction representation. Default is FALSE.
cells	Vector of valid cell names corresponding to column names of slot ndata of the SCseq object. Gene expression is ony shown for this subset.
cex	size of data points. Default value is 0.5.
map	logical. If TRUE then data points are shown. Default value is TRUE.
leg	logical. If TRUE then the legend is shown. Default value is TRUE.
noise	logical. If TRUE then display local gene expression variability instead of gene expression (requires VarID analysis)/ Default value is FALSE.

Value

None

---

plotExpNoise *Noise-expression scatter plot*

Description

Plotting noise (epsilon) as a function of normalized or non-normalized expression for a given gene.

Usage

```
plotExpNoise(g, object, noise, set = NULL, ps = 0.1, norm = TRUE, ...)
```

Arguments

g	Valid gene ID with available expression and noise estimates.
object	**RaceID** SCseq object.
noise	List object returned by the compTBNoise function.
set	Set of valid cluster numbers. Default is NULL and data are plotted for cells from all clusters.
ps	Real number. Pseudo-count added to noise and expression estimates. Default is 0.1.
norm	logical. If FALSE, then noise is plotted versus non-normalized expression. Default is TRUE and noise is plotted against normalized expression.
...	Additional arguments of plot function.

Value

None.

---

plotfeatmap *Highlighting feature values in a dimensional reduction representation*

Description

This functions highlights feature values in a two-dimensional t-SNE map, UMAP, or a Fruchterman-Rheingold graph layout of the singe-cell transcriptome data.

Usage

```
plotfeatmap(
  object,
  g,
  n = NULL,
  logsc = FALSE,
  fr = FALSE,
  um = FALSE,
  cells = NULL,
  cex = 1,
  map = TRUE,
  leg = TRUE,
  flo = NULL,
  ceil = NULL
)
```

Arguments

object	SCseq class object.
g	Vector of real numbered features to highlight in the dimensional reduction representation, NAs will be highlighted in grey.
n	String of characters representing the title of the plot. Default is NULL and the first element of g is chosen.
logsc	logical. If TRUE, then feature values are log2-transformed. Default is FALSE. and untransformed values are shown.
fr	logical. If TRUE then plot Fruchterman-Rheingold layout. Default is FALSE.
um	logical. If TRUE then plot umap dimensional reduction representation. Default is FALSE.
cells	Vector of valid cell names corresponding to column names of slot ndata of the SCseq object. Gene expression is ony shown for this subset.
cex	size of data points. Default value is 1.
map	logical. If TRUE then data points are shown. Default value is TRUE.
leg	logical. If TRUE then the legend is shown. Default value is TRUE.
flo	Numeric. Lower bound for feature values. All values smaller then flo are replaced by flo. #' Default is NULL and no fllo is applied.
ceil	Numeric. Upper bound for feature values. All values larger then ceil are replaced by ceil. Default is NULL and no ceil is applied.

Value

None

---

plotgraph *StemID2 Lineage Graph*

Description

This function plots a graph of lineage trajectories connecting RaceID3 cluster medoids as inferred by StemID2 to approximate the lineage tree. The plot highlights significant links, where colour indicates the level of significance and width indicates the link score. The node colour reflects the level of transcriptome entropy.

Usage

```
plotgraph(
  object,
  showCells = FALSE,
  showMap = TRUE,
  tp = 0.5,
  scthr = 0,
  cex = 1
)
```

Arguments

object	Ltree class object.
showCells	logical. If TRUE, then projections of cells are shown in the plot. Default is FALSE.
showMap	logical. Tf TRUE, then show transparent t-SNE map (with transparency tp) of cells in the background. Default is TRUE.
tp	Real number between zero and one. Level of transparency of the t-SNE map. Deafault is 0.5. See showMap.
scthr	Real number between zero and one. Score threshold for links to be shown in the graph. For scthr=0 all significant links are shown. The maximum score is one. Default is 0.
cex	real positive number. Size of data points. Deault is 1.

Value

None.

---

plotjaccard *Plot Jaccard Similarities*

Description

This functions plots a barchart of Jaccard similarities for the RaceID3 clusters before outlier identification

Usage

```
plotjaccard(object)
```

Arguments

object	SCseq class object.

Value

None

plotlabelsmap *Plot labels in a dimensional reduction representation*

Description

This functions plots cell labels into a two-dimensional t-SNE map, UMAP, or a Fruchterman-Rheingold graph layout of the singe-cell transcriptome data.

Usage

```
plotlabelsmap(object, labels = NULL, fr = FALSE, um = FALSE, cex = 0.5)
```

Arguments

object	SCseq class object.
labels	Vector of labels for all cells to be highlighted in the t-SNE map. The order has to be the same as for the columns in slot ndata of the SCseq object. Default is NULL and cell names are highlighted.
fr	logical. If TRUE then plot Fruchterman-Rheingold layout. Default is FALSE.
um	logical. If TRUE then plot umap dimensional reduction representation. Default is FALSE.
cex	positive real number. Size of the labels. Default is 0.5.

Value

None

plotlinkpv *Heatmap of Link P-values*

Description

This function plots a heatmap of link p-values.

Usage

```
plotlinkpv(object)
```

Arguments

object	Ltree class object.

Value

None.

plotlinkscore *Heatmap of Link Scores*

Description

This function plots a heatmap of link score.

Usage

```
plotlinkscore(object)
```

Arguments

object	Ltree class object.

Value

None.

plotmap *Plotting a dimensional reduction representation*

Description

This functions plots a two-dimensional t-SNE map, UMAP, or a Fruchterman-Rheingold graph layout of the singe-cell transcriptome data.

Usage

```
plotmap(object, final = TRUE, tp = 1, fr = FALSE, um = FALSE, cex = 0.5)
```

Arguments

object	SCseq class object.
final	logical. If TRUE, then highlight final clusters after outlier identification. If FALSE, then highlight initial clusters prior to outlier identification. Default is TRUE.
tp	Number between 0 and 1 to change transparency of dots in the map. Default is 1.
fr	logical. If TRUE then plot Fruchterman-Rheingold layout. Default is FALSE.
um	logical. If TRUE then plot umap dimensional reduction representation. Default is FALSE.
cex	size of data points. Default value is 0.5.

Value

None

plotmarkergenes *Plotting a Heatmap of Marker Gene Expression*

Description

This functions generates a heatmap of expression for defined group of genes and can highlight the clustering partition and another sample grouping, e.g. origin or cell type.

Usage

```
plotmarkergenes(
  object,
  genes,
  imputed = FALSE,
  cthr = 0,
  cl = NULL,
  cells = NULL,
  order.cells = FALSE,
  aggr = FALSE,
  norm = FALSE,
  cap = NULL,
  flo = NULL,
  samples = NULL,
  cluster_cols = FALSE,
  cluster_rows = TRUE,
  cluster_set = FALSE,
  samples_col = NULL,
  zsc = FALSE,
  logscale = TRUE,
  noise = FALSE,
  fontsize = 10
)
```

Arguments

object	SCseq class object.
genes	A vector with a group of gene names corresponding to a subset of valid row names of the ndata slot of the SCseq object.
imputed	logical. If TRUE and imputing was done by calling compdist with knn > 0, then imputed expression values are shown. If FALSE, then raw counts are shown. Default is FALSE
cthr	Interger number greater or equal zero. Only clusters with >cthr cells are included in the t-SNE map. Default is 0.
cl	Vector of valid cluster numbers contained in slot cpart of the SCseq object. Default is NULL and all clusters with >cthr cells are included.

cells	Vector of valid cell names corresponding to column names of slot ndata of the SCseq object. Gene expression is only shown for this subset. Default is NULL and all cells are included. The set of cells is intersected with the subset of clusters in cl if given.
order.cells	logical. If TRUE, then columns of the heatmap are ordered by cell name and not by cluster number. If cells are given, then columns are ordered as in cells.
aggr	logical. If TRUE, then only average expression is shown for each cluster. Default is FALSE and expression in individual cells is shown.
norm	logical. If TRUE, then expression of each gene across clusters is normalized to 1, in order to depict all genes on the same scale. Default is FALSE.
cap	Numeric. Upper bound for gene expression. All values larger then cap are replaced by cap. Default is NULL and no cap is applied.
flo	Numeric. Lower bound for gene expression. All values smaller then flo are replaced by flo. Default is NULL and no flo is applied.
samples	A vector with a group of sample names for each cell in the same order as the column names of the ndata slot of the SCseq object.
cluster_cols	logical. If TRUE, then columns are clustered. Default is FALSE.
cluster_rows	logical. If TRUE, then rows are clustered. Default is TRUE.
cluster_set	logical. If TRUE then clusters are ordered by hierarchical clustering of the cluster medoids.
samples_col	Vector of colors used for highlighting all samples contained in samples in the heatmap. Default is NULL.
zsc	logical. If TRUE then a z-score transformation is applied. Default is FALSE.
logscale	logical. If TRUE then a log2 transformation is applied. Default is TRUE.
noise	logical. If TRUE then display local gene expression variability instead of gene expression (requires VarID analysis)/ Default value is FALSE.
fontsize	postive real number. Font size of gene name labels. Default is 10.

Value

Object with clustering information for rows and columns returned by the function pheatmap from the package **pheatmap**.

---

plotMV *Plot of Mean-Variance dependence and various fits*

Description

This functions plots the dependence of the transcript count variance or, alternatively, the coefficient of variation (CV) on the mean in log2 space. The mean-variance dependence is plotted along with a loess-regression, a second order polynomial fit, and the background model of the local variability. The CV plot also highlights the local variability associated with cell-to-cell variability of total transcript counts, as calculated directly from the mean and variance of total transcript counts (turquoise) or from a local fit of a gamma distribution (orange).

Usage

```
plotMV(x, cv = FALSE, ret = FALSE, span = 0.75, degree = 2, ...)
```

Arguments

x	Transcript count matrix.
cv	Logical. If TRUE then the coefficient of variation is plotted instead of the variance. Default is FALSE.
ret	Logical. If TRUE then a second order polynomial fit is returned. Default is FALSE
span	Parameter for the local regression. See help(loess). Default value is 0.75.
degree	Parameter for the local regression. See help(loess). Default value is 2.
...	Additional arguments for plot.

Value

If ret=FALSE second order polynomial fit as returned by lm.

---

plotNoiseModel *Function for plotting the baseline model of gene expression variability*

Description

This function plots the variance against mean expression across all genes and a second order polynomial to the base line of the variance-mean dependence in log space.

Usage

```
plotNoiseModel(x, corrected = FALSE)
```

Arguments

x	List object returned by function noiseBaseFit or function compNoise.
corrected	logical value. If TRUE, then the variance is plotted after regressing our the mean dependence.

Value

None

Examples

```
x <- noiseBaseFit(intestinalDataSmall,step=.01,thr=.05)
plotNoiseModel(x)
```

plotoutlierprobs *Plot Outlier Probabilities*

Description

This functions plots a barchart of outlier probabilities across all cells in each cluster.

Usage

```
plotoutlierprobs(object)
```

Arguments

object	SCseq class object.

Value

None

plotPC *Function to plot the selected number of principal components*

Description

This functions plots the percentage variability explained the first one hundred (or pcaComp) pricipal components of the PCA performed in the function pruneKnn if the parameter large was TRUE. The selected number of principal components (if pcaComp was NULL) is determined by an elbow criterion and highlighted by a blue circle.

Usage

```
plotPC(res, logDiff = FALSE)
```

Arguments

res	List object with k nearest neighbour information returned by pruneKnn function.
logDiff	logical. If TRUE, then plot log2 of the difference in variability explained by PC i and PC i+1.

plotPearsonRes *Function for plotting the variance of Pearson residuals*

Description

This function plots the variance versus the mean of the Pearson residuals obtained by the negative binomial regression computed by the function `compY` if `regNB` is `TRUE`. A local regression is also shown.

Usage

```
plotPearsonRes(y, log = FALSE, ...)
```

Arguments

`y`	List object returned by the `compNoise` or `pruneKnn` function (if run with `regNB=TRUE`).
`log`	logical. If `TRUE` then the y-axis is log-transformed. Default is `FALSE`.
`...`	Additional arguments for `plot`.

Value

None

Examples

```
res <- pruneKnn(intestinalDataSmall,no_cores=1)
plotPearsonRes(res,log=TRUE)
```

plotPP *Plotting function for posterior checks*

Description

This function plots various statistics for the posterior check

Usage

```
plotPP(pp, y = NULL, umi.eps = FALSE, i = 1, log.scale = TRUE)
```

Arguments

pp	List object returned by testPrior function.
y	One of "mean", "median", "var", "cor", or NULL. If NULL then the ratios between the predicted and the actual variances across all sampled genes and neighbourhoods are shown as boxplots for all tested values of the prior parameter gamma. If y equals "mean", "median", or "var", the mean, median, or variance is plotted for all gamma values. If y equal "cor", then the correlation between the total transcript count of a cell and the local noise estimate epsilon is plotted for all values of gamma. Default is NULL.
umi.eps	Logical. If TRUE then a scatter plot of the local noise estimate epsilon and the total transcript count is produced for a given element i of the pp$noise corresponding to a value of the prior parameter gamma. Default is FALSE.
i	Positive integer number. Index of pp$noise, corresponding to a value of the prior parameter gamma to be used for plotting is umi.eps=TRUE. Default is 1.
log.scale	Logical. If TRUE then the ratio between the predicted and the actual variance is transformed to a log2-scale prior to computations and plotting. If umi.eps=TRUE, total transcript counts and epsilon estimates are log2-transformed for plotting. Default is TRUE.

---

plotPT *Plotting pseudo-time in dimensional reduction representation*

Description

Highlight clusters or pseudotime in dimensional reduction representation and indicate trajectory derived by **slingshot**.

Usage

```
plotPT(pt, object, clusters = TRUE, lineages = FALSE)
```

Arguments

pt	List object returned by function pseudoTime.
object	**RaceID** SCseq object.
clusters	logical. If TRUE, then clusters are highlighted. Otherwise, pseudotime is highlighted. Default is TRUE.
lineages	logical. If TRUE, then lineages as linear connections of clusters are hghlighted. Otherwise, continuous trajectories are shown. Default is FALSE.

Value

None

plotQQ — *Scatter plot of two noise-related quantaties of local pruned k-nearest neighbourhoods*

Description

Displaying two noise-related quantaties of local pruned k-nearest neighbourhoods in a scatterplot highlighting VarID clusters.

Usage

```
plotQQ(x, m, n, object, cluster = NULL, cex = 0.5, show.cor = TRUE, ...)
```

Arguments

x	List object returned by quantKnn function.
m	Component name of x. One of "noise.av", "noise.ratio", "local.corr", "umi".
n	Component name of x. One of "noise.av", "noise.ratio", "local.corr", "umi".
object	SCseq class object.
cluster	Valid cluster number or vector of cluster numbers, contained in object@cpart. If given, then cells of clusters in cluster are circled in black.
cex	Real positive number. Size of data points. Default is 0.5.
show.cor	logical. If TRUE then Pearson's correlation is shown in the legend. Default is TRUE.
...	Additional parameters of plot (e.g., log, see help(plot)).

Value

None

plotQuantMap — *Plotting noise-related quantaties of local pruned k-nearest neighbourhoods*

Description

Plotting noise-related quantaties of local pruned k-nearest neighbourhoods in the dimensional reduction representation chosen for quantKnn or as boxplot across clusters.

Usage

```
plotQuantMap(
  x,
  n,
  object,
  box = FALSE,
  cluster = NULL,
  set = NULL,
  logsc = FALSE,
  cex = 0.5,
  ...
)
```

Arguments

x	List object returned by quantKnn function.
n	Component name of x. One of "noise.av", "noise.ratio", "local.corr", "umi".
object	SCseq class object.
box	Logical. If TRUE, then data are shown as boxplot across clusers. Default is FALSE and a dimensional reduction representation is shown.
cluster	Valid cluster number or vector of cluster numbers, contained in object@cpart. If given and box=TRUE then the median of the feature values across clusters in cluster is indicated as a black solid line in the boxplot. Default is NULL.
set	Ordered set of valid cluster numbers. If box equals TRUE than data will only be plotted for these clusters in the given order. Default is NULL and data for all clutsers will be shown.
logsc	logical. If TRUE, then feature values are log2-transformed. Default is FALSE.
cex	Real positive number. Size of data points. Default is 0.5.
...	Additional parameters of plotfeatmap if box=FALSE (e.g., um or fr to select dimensional reduction representation, see help(plotfeatmap)), or of plotB (e.g., ylim, see help(plotB)).

Value

None

---

plotRegNB *Function for plotting negative binomial regression*

Description

This function plots the parameters obatined by the negative binomial regression of the transcript counts on the total transcript count in each cells. Smoothed parameter estimates are also shown.

Usage

```
plotRegNB(expData, y, par.nb = NULL, span = 0.75, ...)
```

Arguments

expData	Matrix of gene expression values with genes as rows and cells as columns. The matrix needs to contain the same cell IDs as columns like the input matrix. used to derive the pruned k nearest neighbours with the pruneKnn function.
y	List object returned by the compNoise or pruneKnn function (if run with regNB=TRUE).
par.nb	Parameter to be plotted, i.e. valid column of res$regData$nbRegr. of the log total UMI count. intercept is the intercept inferred by the regression. Default is NULL and theta is shown.
span	Positive real number. Parameter for loess-regression (see large) controlling the degree of smoothing. Default is 0.75.
...	Additional arguments for plot.

Value

None

Examples

```
res <- pruneKnn(intestinalDataSmall,no_cores=1)
plotRegNB(intestinalDataSmall,res,"theta")
```

---

plotsaturation *Plot Saturation of Within-Cluster Dispersion*

Description

This functions plots the (change in the) mean within-cluster dispersion as a function of the cluster number and highlights the saturation point inferred based on the saturation criterion applied by RaceID3: The number of clusters where the change in within-cluster dispersion upon adding further clusters approaches linear behaviour demarcates the saturation point and is highlighted in blue.

Usage

```
plotsaturation(object, disp = FALSE)
```

Arguments

object	SCseq class object.
disp	logical. If FALSE, then the change of the within-cluster dispersion is plotted. if TRUE the actual dispersion is plotted. Default is FALSE

Value

None

plotsensitivity *Plot Sensitivity*

Description

This functions plots the number of outliers as a function of the outlier probability.

Usage

```
plotsensitivity(object)
```

Arguments

object	SCseq class object.

Value

None

plotsilhouette *Plot Cluster Silhouette*

Description

This functions produces a silhouette plot for RaceID3 clusters prior or post outlier identification.

Usage

```
plotsilhouette(object, final = FALSE)
```

Arguments

object	SCseq class object.
final	logical. If TRUE, then plot silhouette coefficients for final clusters after outlier identification. Default is FALSE and silhouette coefficients are plotted for initial clusters.

Value

None

plotspantree *Minimum Spanning Tree of RaceID3 clusters*

Description

This function plots a minimum spanning tree of the RaceID3 cluster medoids in a two-dimensional reduction representation.

Usage

```
plotspantree(object, tp = 0.5, cex = 1, projections = FALSE)
```

Arguments

object	Ltree class object.
tp	Real number between zero and one. Level of transparency of the t-SNE map. Deafault is 0.5.
cex	real positive number. Size of data points. Deault is 1.
projections	logical. If TRUE, then the projections of the cells onto the inter-medoid links as computed by StemID are shown. Default is FALSE

Value

None.

plotsymbolsmap *Plotting groups as different symbols in a dimensional reduction representation*

Description

This functions highlights groups of cells by different symbols in a two-dimensional t-SNE map, UMAP, or a Fruchterman-Rheingold graph layout of the singe-cell transcriptome data.

Usage

```
plotsymbolsmap(
  object,
  types,
  subset = NULL,
  samples_col = NULL,
  cex = 0.5,
  fr = FALSE,
  um = FALSE,
  leg = TRUE,
```

```
  map = TRUE,
  cex.legend = 0.75,
  leg.pos = "topleft"
)
```

Arguments

object	SCseq class object.
types	Vector assigning each cell to a type to be highlighted in the t-SNE map. The order has to be the same as for the columns in slot ndata of the SCseq object. Default is NULL and each cell is highlighted by a different symbol.
subset	Vector containing a subset of types from types to be highlighted in the map. Default is NULL and all types are shown.
samples_col	Vector of colors used for highlighting all samples contained in samples in the map. Default is NULL.
cex	size of data points. Default value is 0.5.
fr	logical. If TRUE then plot Fruchterman-Rheingold layout. Default is FALSE.
um	logical. If TRUE then plot umap dimensional reduction representation. Default is FALSE.
leg	logical. If TRUE then the legend is shown. Default value is TRUE.
map	logical. If TRUE then data points are shown. Default value is TRUE.
cex.legend	Positive real number. Size of data points and text in the legend. Default is 0.75.
leg.pos	Position of the legend. a single keyword from the list ‘"bottomright"’, ‘"bottom"’, ‘"bottomleft"’,‘"left"’, ‘"topleft"’, ‘"top"’, ‘"topright"’, ‘"right"’ and‘"center"’. This places the legend on the inside of the plot frame at the given location.

Value

None

---

plotTrProbs *Function for plotting transition probabilities between clusters*

Description

This function plots the transitions probabilities in a dimensional reduction representation of a **RaceID** SCseq object updates with the updateSC function. in order to utilize **RaceID** functions for visualization.

Usage

```
plotTrProbs(
  object,
  probs,
  tp = 0.5,
  prthr = 0,
  cthr = 0,
  fr = FALSE,
  um = FALSE,
  cex = 0.5
)
```

Arguments

object	**RaceID** SCseq object, updated with the updateSC function.
probs	Matrix of transition probabilities between clusters, returned by the transitionProbs function.
tp	Positive real number between 0 and 1. Transparency of the data points in the dimensional reduction map. Default is 0.5.
prthr	Positive real number between 0 and 1. Threshold of transition probabilities. Only transitions with probability >prthr are displayed in the map. Default is 0.
cthr	Integer number greater or equal 0 defining the minimum clusters size for inclusion into the map. Default is 0.
fr	Logical. If TRUE, then a Fruchterman-Rheingold graph layout is shown (in case it has been computed for the **RaceID** bject), otherwise a t-SNE map is shown. Default is FALSE.
um	Logical. If TRUE then plot umap dimensional reduction representation. Default is FALSE.
cex	Real positive number. Size of data points. Default is 0.5.

Value

None

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
d <- getExpData(sc)
res <- pruneKnn(d,distM=sc@distances,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
cl <- graphCluster(res,pvalue=0.01)
sc <- updateSC(sc,res=res,cl=cl)
sc <- comptsne(sc)
probs <-transitionProbs(res,cl,pvalue=0.01)
plotTrProbs(sc,probs,tp=.5,prthr=0,cthr=0,fr=FALSE)
```

plotUMINoise *Plotting noise dependence on total UMI count*

Description

This function plots the dependence of mean noise per cell on the total UMI count per cell. It serves as a basis for choosing the prior parameter gamma (see function compTBNoise). With a proper parameter choice, there should be no correlation between the two quantities. If a positive correlation is observed, gamma should be increased in order to weaken the prior. If the correlation is negative, gamma should be decreased in order to increase the strength of the prior.

Usage

```
plotUMINoise(object, noise, log.scale = TRUE)
```

Arguments

object	**RaceID** SCseq object.
noise	object returned by compTBNoise function.
log.scale	Logical. If TRUE total transcript counts and epsilon estimates are log2-transformed for plotting. Default is TRUE.

postfntb *Posterior probability*

Description

Non-normalized negative log posterior probability with a negative binomial likelihood and Cauchy prior.

Usage

```
postfntb(eps, z, x0, gamma, mu, rt)
```

Arguments

eps	Positive real number. Residual (biological) noise.
z	Vector of integer number greater or equal zero. Transcript counts.
x0	Real number. Location parameter.
gamma	Positive real number. Scale parameter.
mu	Positive real number. Mean expression.
rt	Positive real number. Technical noise parameter. See help(fitGammaRt).

Value

Negative non-normalized log posterior probability fro maximum a posterior inference.

priorfn *Prior function for maximum a posterior inference*

Description

A prior function specified as Cauchy probability density.

Usage

```
priorfn(x, x0, gamma)
```

Arguments

x	Vector or real numbers (quantiles)
x0	Real number. Location parameter.
gamma	Positive real number. Scale parameter.

Value

Vector of probabilities

projback *Compute Cell Projections for Randomized Background Distribution*

Description

This function computes the projections of cells onto inter-cluster links for randomized cell positions in a high-dimensional embedded space. Significance of link based on an increased number of cells on a link is inferred based on this background model.

Usage

```
projback(object, pdishuf = 500, fast = FALSE, rseed = 17000, verbose = TRUE)
```

Arguments

object	Ltree class object.
pdishuf	Number of randomizations of cell positions for which to compute projections of cells on inter-cluster links. Default is 2000. No randomizations are needed in this mode and the function will do nothing. Default is TRUE.
fast	logical. If TRUE and nmode=FALSE cells will still be assigned to links based on maximum projections but a fast approximate background model will be used to infer significance. The function will do nothing in this case. Default is FALSE.
rseed	Integer number used as seed to ensure reproducibility of randomizations. Defaut is 17000.
verbose	logical. If FALSE then status output messages are disabled. Default is TRUE.

Value

An Ltree class object with all information on randomized cell projections onto links stored in the prbacka slot.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- comptsne(sc)
ltr <- Ltree(sc)
ltr <- compentropy(ltr)
ltr <- projcells(ltr,nmode=FALSE)
ltr <- projback(ltr,pdishuf=50)
```

---

projcells *Compute transcriptome entropy of each cell*

Description

This function computes the projections of cells onto inter-cluster links in a high-dimensional embedded space.

Usage

```
projcells(object, cthr = 5, nmode = TRUE, knn = 3, fr = FALSE, um = FALSE)
```

Arguments

object	Ltree class object.
cthr	Positive integer number. Clusters to be included into the StemID2 analysis must contain more than cthr cells. Default is 5.
nmode	logical. If TRUE, then a cell of given cluster is assigned to the link to the cluster with the smallest average distance of the knn nearest neighbours within this cluster. Default is TRUE.
knn	Positive integer number. See nmode. Default is 3.
fr	logical. Use Fruchterman-Rheingold layout instead of t-SNE for dimensional-reduction representation of the lineage graph. Default is FALSE.
um	logical. Use umap representation instead of t-SNE for dimensional-reduction representation of the lineage graph. Default is FALSE.

Value

An Ltree class object with all information on cell projections onto links stored in the ldata slot.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- comptsne(sc)
ltr <- Ltree(sc)
ltr <- compentropy(ltr)
ltr <- projcells(ltr)
```

---

projenrichment *Enrichment of cells on inter-cluster links*

Description

This function plots a heatmap of the enrichment ratios of cells on significant links.

Usage

```
projenrichment(object)
```

Arguments

object	Ltree class object.

Value

None.

---

pruneKnn *Function inferring a pruned knn matrix*

Description

This function determines k nearest neighbours for each cell in gene expression space, and tests if the links are supported by a negative binomial joint distribution of gene expression. A probability is assigned to each link which is given by the minimum joint probability across all genes.

Usage

```
pruneKnn(
  expData,
  distM = NULL,
  large = TRUE,
  regNB = TRUE,
  bmethod = NULL,
  batch = NULL,
  regVar = NULL,
  offsetModel = TRUE,
  thetaML = FALSE,
  theta = 10,
  ngenes = 2000,
  span = 0.75,
  pcaComp = NULL,
  tol = 1e-05,
  algorithm = "kd_tree",
  metric = "pearson",
  genes = NULL,
  knn = 25,
  do.prune = TRUE,
  alpha = 1,
  nb = 3,
  no_cores = NULL,
  FSelect = FALSE,
  pca.scale = FALSE,
  ps = 1,
  seed = 12345,
  theta.harmony = NULL,
  ...
)
```

Arguments

expData	Matrix of gene expression values with genes as rows and cells as columns. These values have to correspond to unique molecular identifier counts. Alternatively, a Seurat object could be used as input, after normalization, PCA-dimensional reduction, and shared-nearest neighbour inference.
distM	Optional distance matrix used for determining k nearest neighbours. Default is NULL and the distance matrix is computed using a metric given by the parameter metric.
large	logical. If TRUE then no distance matrix is required and nearest neighbours are inferred by the **FNN** package based on a reduced feature matrix computed by a principle component analysis. Only the first pcaComp principle components are considered. Prior to principal component analysis a negative binomial regression is performed to eliminate the dependence on the total number of transcripts per cell. The pearson residuals of this regression serve as input for the principal component analysis after smoothing the parameter dependence on the mean

by a loess regression. Deafult is TRUE. Recommended mode for very large datasets, where storing a distance matrix requires too much memory. distM will be ignored if large is TRUE.

regNB — logical. If TRUE then gene a negative binomial regression is performed to prior to the principle component analysis if large = TRUE. See large. Otherwise, transcript counts in each cell are normalized to one, multipled by the minimal total transcript count across all cells, followed by adding a pseudocount of 0.1 and taking the logarithm. Default is TRUE.

bmethod — Character string indicating the batch correction method. If "harmony", then batch correction is performed by the **harmony** package. Default is NULL and batch correction will be done by negative binomial regression.

batch — vector of batch variables. Component names need to correspond to valid cell IDs, i.e. column names of expData. If regNB is TRUE, than the batch variable will be regressed out simultaneously with the log UMI count per cell. An interaction term is included for the log UMI count with the batch variable. Default value is NULL.

regVar — data.frame with additional variables to be regressed out simultaneously with the log UMI count and the batch variable (if batch is TRUE). Column names indicate variable names (name beta is reserved for the coefficient of the log UMI count), and rownames need to correspond to valid cell IDs, i.e. column names of expData. Interaction terms are included for each variable in regVar with the batch variable (if batch is TRUE). Default value is NULL.

offsetModel — Logical parameter. Only considered if regNB is TRUE. If TRUE then the beta (log UMI count) coefficient is set to 1 and the intercept is computed analytically as the log ration of UMI counts for a gene and the total UMI count across all cells. Batch variables and additional variables in regVar are regressed out with an offset term given by the sum of the intercept and the log UMI count. Default is TRUE.

thetaML — Logical parameter. Only considered if offsetModel equals TRUE. If TRUE then the dispersion parameter is estimated by a maximum likelihood fit. Otherwise, it is set to theta. Default is FALSE.

theta — Positive real number. Fixed value of the dispersion parameter. Only considered if theaML equals FALSE.

ngenes — Positive integer number. Randomly sampled number of genes (from rownames of expData) used for predicting regression coefficients (if regNB=TRUE). Smoothed coefficients are derived for all genes. Default is 2000.

span — Positive real number. Parameter for loess-regression (see large) controlling the degree of smoothing. Default is 0.75.

pcaComp — Positive integer number. Number of princple components to be included if large is TRUE. Default is NULL and the number of principal components used for dimensionality reduction of the feature matrix is derived by an elbow criterion. However, the minimum number of components will be set to 15 if the elbow criterion results in a smaller number. The derived number can be be plotted using the plotPC function.

tol — Numerical value greater than zero. Tolerance for numerical PCA using **irlba**. Default value is 1e-6.

algorithm	Algorithm for fast k nearest neighbour inference, using the get.knn function from the **FNN** package. See help(get.knn). Deafult is "kd_tree".
metric	Distances are computed from the expression matrix x after optionally including only genes given as argument genes or after optional feature selection (see FSelect). Possible values for metric are "pearson", "spearman", "logpearson", "euclidean". Default is "pearson". In case of the correlation based methods, the distance is computed as 1 – correlation. This parameter is only used if large is FALSE and distM is NULL.
genes	Vector of gene names corresponding to a subset of rownames of x. Only these genes are used for the computation of a distance matrix and for the computation of joint probabilities of nearest neighbours. Default is NULL and all genes are used.
knn	Positive integer number. Number of nearest neighbours considered for each cell. Default is 25.
do.prune	Logical parameter. If TRUE, then pruning of k-nearest neighbourhoods is performed. If FALSE, then no pruning is done. Default is TRUE.
alpha	Positive real number. Relative weight of a cell versus its k nearest neigbour applied for the derivation of joint probabilities. A cell receives a weight of alpha while the weights of its k nearest neighbours as determined by quadratic programming sum up to one. The sum across all weights and alpha is normalized to one, and the weighted mean expression is used for computing the link porbabilities for each of the k nearest neighbours. Larger values give more weight to the gene expression observed in a cell versus its neighbourhood. Typical values should be in the range of 0 to 10. Default is value is 1. If alpha is set to NULL it is inferred by an optimization, i.e., alpha is minimized under the constraint that the gene expression in a cell does not deviate more then one standard deviation from the predicted weigthed mean, where the standard deviation is calculated from the predicted mean using the background model (the average dependence of the variance on the mean expression). This procedure is coputationally more intense and inceases the run time of the function significantly.
nb	Positive integer number. Number of genes with the lowest outlier probability included for calculating the link probabilities for the knn pruning. The link probability is computed as the geometric mean across these genes. Default is 3.
no_cores	Positive integer number. Number of cores for multithreading. If set to NULL then the number of available cores minus two is used. Default is NULL.
FSelect	Logical parameter. If TRUE, then feature selection is performed prior to distance matrix calculation and VarID analysis. Default is FALSE.
pca.scale	Logical parameter. If TRUE, then input features are scaled prior to PCA transformation. Default is FALSE.
ps	Real number greater or equal to zero. Pseudocount to be added to counts within local neighbourhoods for outlier identification and pruning. Default is 1.
seed	Integer number. Random number to initialize stochastic routines. Default is 12345.
theta.harmony	theta parameter of RunHarmony function from the **harmony** package (to avoid collision with the dispersion parameter theta). Default is NULL.

... Additional parameters for RunHarmony function from the **harmony** package, if batch is not NULL and bmethod="harmony".

Value

List object of six components:

distM Distance matrix.

dimRed PCA transformation of expData including the first pcaComp principle components, computed on including genes or variable genes only if Fselect equals TRUE. Is is set to NULL if large equals FALSE.

pvM Matrix of link probabilities between a cell and each of its k nearest neighbours (Bonferroni-corrected p-values). Column i shows the k nearest neighbour link probabilities for cell i in matrix x.

pvM.raw Matrix of uncorrected link probabilities between a cell and each of its k nearest neighbours (without multiple-testing correction). Column i shows the k nearest neighbour link probabilities for cell i in matrix x.

NN Matrix of column indices of k nearest neighbours for each cell according to input matrix x. First entry corresponds to index of the cell itself. Columns contain the k nearest neighbour indices for cell i in matrix x.

B List object with background model of gene expression as obtained by fitBackVar function.

regData If regNB=TRUE this argument contains a list of four components: component pearsonRes contains a matrix of the Pearson Residual computed from the negative binomial regression, component nbRegr contains a matrix with the regression coefficients, component nbRegrSmooth contains a matrix with the smoothed regression coefficients, and log_umi is a vector with the total log UMI count for each cell. The regression coefficients comprise the dispersion parameter theta, the intercept, the regression coefficient beta for the log UMI count, and the regression coefficients of the batches (if batch is not NULL).

alpha Vector of inferred values for the alpha parameter for each neighbourhood (if input parameter alpha is NULL; otherwise all values are equal to the input parameter).

pars List object storing the run parameters.

pca Principal component analysis of the of the input data, if large is TRUE. Output or the function irlba from the **irlba** package with pcaComp principal components, or 100 principal components if pcaComp is NULL.

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
```

pseudoTime *Extract pseudo-time order of cells along a trajectory*

Description

Extract pseudo-time order of cells along a trajectory defined by a set of clusters using the **slingshot** algorithm. If the **slingshot** package is unavailable, the function falls back to inference by principal curve analysis using the **princurve** package.

Usage

```
pseudoTime(
  object,
  set,
  m = NULL,
  useSlingshot = TRUE,
  map = "umap",
  x = NULL,
  n_neighbors = 15,
  metric = "euclidean",
  n_epochs = 200,
  min_dist = 0.1,
  local_connectivity = 1,
  spread = 1,
  initial_cmd = TRUE,
  perplexity = 30,
  rseed = 15555,
  ...
)
```

Arguments

object	**RaceID** SCseq object.
set	Set of valid ordered cluster numbers (in object@cpart) defining the trajectory for which the pseudo-temporal order of cells should be computed computed. Only clusters on a single, linear trajectory should be given.
m	Existing dimensional reduction representation of RaceID object. Either "fr", "tsne" or "umap". Default is NULL and dimensional reduction representation is computed for all cells in set.
useSlingshot	logical. If TRUE and the **slingshot** package is available, trajectory inference will be done using slingshot. If FALSE, inference will be done by principal curve analysis using the **princurve** package. Default is TRUE.
map	Either "tsne" or "umap". If m is NULL this argument determines the algorithm (UMAP or t-SNE) for computing the dimensional reduction representation of all cells set used for pseudo-temporal ordering by the Bioconductor package slingshot. Default is "umap".

x	Optional feature matrix, which will be directly used for computation of the dimensional reduction representation. Default is NULL and object@dimRed$x is used, unless empty. In this case, getfdata(object) is used.
n_neighbors	Umap parameter (used if map="umap" and m=NULL). See help(umap.defaults) after loading package **umap**. Default is 15.
metric	Umap parameter (used if map="umap" and m=NULL). See help(umap.defaults) after loading package **umap**. Default is "euclidean".
n_epochs	Umap parameter (used if map="umap" and m=NULL). See help(umap.defaults) after loading package **umap**. Default is 200.
min_dist	Umap parameter (used if map="umap" and m=NULL). See help(umap.defaults) after loading package **umap**. Default is 0.1.
local_connectivity	Umap parameter (used if map="umap" and m=NULL). See help(umap.defaults) after loading package **umap**. Default is 1.
spread	Umap parameter (used if map="umap" and m=NULL). See help(umap.defaults) after loading package **umap**. Default is 1.
initial_cmd	logical. t-SNE parameter (used if map="tsne" and m=NULL). If TRUE, then the t-SNE map computation is initialized with a configuration obtained by classical multidimensional scaling. Default is TRUE.
perplexity	Positive number. t-SNE parameter (used if map="tsne" and m=NULL). Perplexity of the t-SNE map. Default is 30.
rseed	Integer number. Random seed to enforce reproducible dimensional reduction computation.
...	Additional arguments to be passed to the getCurves function of the **slingshot** package.

Value

List object of six components:

pt	Vector of pseudo-time value obtained by **slingshot**.
ord	Vector of cells in set ordered by pseudo-time, starting with the first cluster in set.
set	Vector of cluster numbers defining the trajectory used for pseudo-time inference.
part	Vector of cluster numbers of all cells in set.
rd	Two-dimensional matrix with x- and y-coordinates of dimensional reduction representation used for slingshot.
sls	slingshot data object.

quantKnn *Noise-related quantaties of local pruned k-nearest neighbourhoods*

Description

This function computes a number of noise-related quantities for all pruned k-nearest neighbourhoods.

Usage

```
quantKnn(res, noise, object, pvalue = 0.01, minN = 5, no_cores = NULL)
```

Arguments

res	List object with k nearest neighbour information returned by pruneKnn function.
noise	List of noise parameters returned by compTBNoise.
object	SCseq class object.
pvalue	Positive real number between 0 and 1. All nearest neighbours with link probability < pvalue are discarded. Default is 0.01.
minN	Positive integer number. Noise inference is only done for k-nearest neighbourhoods with at least minN neighbours remaining after pruning.
no_cores	Positive integer number. Number of cores for multithreading. If set to NULL then the number of available cores minus two is used. Default is NULL.

Value

List object with eight components:

noise.av	Vector of biological noise average across all genes for each k-nearest neighbourhood.
noise.ratio	Vector of ratio between total noise and technical noise averaged across all genes for each k-nearest neighbourhood.
local.corr	Vector of average Spearman's correlation coefficient between all cell in a pruned k-nearest neighourhood.
umi	Vector of total UMI counts for all cells.

rcpp_hello_world *Simple function using Rcpp*

Description

Simple function using Rcpp

Usage

```
rcpp_hello_world()
```

Examples

```
## Not run:
rcpp_hello_world()

## End(Not run)
```

rfcorrect *Random Forests-based Reclassification*

Description

This functions applies random forests-based reclassification of cell clusters to enhance robustness of the final clusters.

Usage

```
rfcorrect(
  object,
  rfseed = 12345,
  nbtree = NULL,
  final = TRUE,
  nbfactor = 5,
  ...
)
```

Arguments

object	SCseq class object.
rfseed	Seed for enforcing reproducible results. Default is 12345.
nbtree	Number of trees to be built. Default is NULL and the number of tree is given by the number of cells times nbfactor.
final	logical. If TRUE, then reclassification of cell types using out-of-bag analysis is performed based on the final clusters after outlier identification. If FALSE, then the cluster partition prior to outlier idenitifcation is used for reclassification.

nbfactor	Positive integer number. See nbtree.
...	additional input arguments to the randomForest function of the **randomForest** package.

Value

The function returns an updated SCseq object with random forests votes written to slot out$rfvotes. The clustering partition prior or post outlier identification (slot cluster$kpart or cpart, if parameter final equals FALSE or TRUE, respectively) is overwritten with the partition derived from the reclassification.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
sc <- clustexp(sc)
sc <- findoutliers(sc)
sc <- rfcorrect(sc)
```

SCseq *The SCseq Class*

Description

The SCseq class is the central object storing all information generated during cell type identification with the RaceID3 algorithm. It comprises a number of slots for a variety of objects.

Arguments

object	An SCseq object.

Slots

expdata The raw expression data matrix with cells as columns and genes as rows in sparse matrix format.

ndata Filtered data with expression normalized to one for each cell.

noise Matrix with local gene expression noise estimates (used for VarID analysis)

counts Vector with total transcript counts for each cell in ndata remaining after filtering.

genes Vector with gene names of all genes in ndata remaining after filtering.

dimRed list object object storing information on a feature matrix obtained by dimensional reduction, batch effect correction etc. Component x stores the actual feature matrix.

distances distance (or dis-similarity) matrix computed by RaceID3.

imputed list with two matrices computed for imputing gene expression. The first matrix nn contains the cell indices of the knn nearest neighbours, the second matrix contains the probabilities at which each cell contributes to thye imputed gene expression value for the cell correponding to the columns.

`tsne` data.frame with coordinates of two-dimensional tsne layout computed by RaceID3.

`fr` data.frame with coordinates of two-dimensional Fruchterman-Rheingold graphlayout computed by RaceID3.

`umap` data.frame with coordinates of two-dimensional umap representation computed by RaceID3.

`cluster` list storing information on the initial clustering step of the RaceID3 algorithm

`background` list storing the polynomial fit for the background model of gene expression variability computed by RaceID3, which is used for outlier identification.

`out` list storing information on outlier cells used for the prediction of rare cell types by RaceID3

`cpart` vector containing the final clustering (i.e. cell type) partition computed by RaceID3

`fcol` vector contaning the colour scheme for the RaceID3 clusters

`medoids` vector containing the cell ids for th cluster medoids

`filterpar` list containing the parameters used for cell and gene filterung

`clusterpar` list containing the parameters used for clustering

`outlierpar` list containing the parameters used for outlier identification

---

`Seurat2SCseq` *Converting a Seurat object to a RaceID/VarID object*

Description

This function expects a class `Seurat` object from the **Seurat** package as input and converts this into a **RaceID** `SCseq` object. The function transfers the counts, initializes `ndata` and `fdata` without further filtering, transfers the PCA cell embeddings from the `Seurat` object to `dimRed`, transfers the clustering partition, and `umap` and `tsne` dimensional reduction (if available). CAUTION: Cluster numbers in RaceID start at 1 by default. Hence, all Seurat cluster numbers are shifted by 1.

Usage

```
Seurat2SCseq(Se, rseed = 12345)
```

Arguments

`Se`	**Seurat** object.
`rseed`	Integer number. Random seed for sampling cluster colours.

Value

RaceID `SCseq` object.

`testPrior` *Posterior check of the model*

Description

This functions compares variance estimates obtained from the maximum a posterior estimate with a given prior to the data. The ratio between the predicted variance and the actual variance for a random subset of genes is computed across all pruned k nearest neighbourhoods.

Usage

```
testPrior(
  res,
  expData,
  gamma = c(0.2, 0.5, 1, 5, 1000),
  rseed = 12345,
  ngenes = 200,
  pvalue = 0.01,
  minN = 5,
  no_cores = NULL,
  x0 = 0,
  lower = 0,
  upper = 100
)
```

Arguments

`res`	List object with k nearest neighbour information returned by `pruneKnn`.
`expData`	Matrix of gene expression values with genes as rows and cells as columns. These values have to correspond to unique molecular identifier counts.
`gamma`	Vector of `gamma`-values to test for the Cauchy prior distribution. Default is `c(0.2,0.5,1,5,1000)`. Large values correspond to weak priors (`gamma=1000` corresponds to a maximum likelihood estimate).
`rseed`	Integer number. Random seed to enforce reproducible gene sampling. Default is 12345.
`ngenes`	Positive integer number. Randomly sampled number of genes (from rownames of `expData`) used for noise estimation. Genes are sampled uniformly across the entire expression range. Default is 200.
`pvalue`	Input parameter for `compTBNoise`. See `help(compTBNoise)`.
`minN`	Input parameter for `compTBNoise`. See `help(compTBNoise)`.
`no_cores`	Input parameter for `compTBNoise`. See `help(compTBNoise)`.
`x0`	Input parameter for `compTBNoise`. See `help(compTBNoise)`.
`lower`	Input parameter for `compTBNoise`. See `help(compTBNoise)`.
`upper`	Input parameter for `compTBNoise`. See `help(compTBNoise)`.

Value

List of three components:

pp.var.ratio	List of vectors for each gamma value of ratios between predicted and actual variances across all sampled genes and neighbourhoods.
noise	List of noise objects obtained from compTBNoise for each gamma value.
tc	Vector of total transcript counts for all cells

---

transitionProbs *Function for the computation of transition probabilities between clusters*

Description

This function computes transition probabilities between clusters based on the link probabilities of the pruned k nearest neighbour graph.

Usage

```
transitionProbs(res, cl, pvalue = 0.01)
```

Arguments

res	List object with k nearest neighbour information returned by pruneKnn function.
cl	List object with clustering information, returned by the graphCluster function. If an aggregated cluster of tiny clusters (singletons) exists, stored in residual.cluster, this cluster is disregarded, and no links with this clusters are inferred.
pvalue	Positive real number between 0 and 1. All nearest neighbours with link probability < pvalue are discarded. Default is 0.01.

Value

Matrix of transition probabilities between clusters.

Examples

```
res <- pruneKnn(intestinalDataSmall,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
cl <- graphCluster(res,pvalue=0.01)
probs <-transitionProbs(res,cl,pvalue=0.01)
```

updateSC *Function for updating a RaceID SCseq object with VarID results*

Description

This function updates a **RaceID** SCseq object with a distance matrix or dimensionally reduced feature matrix, a clustering partition, and/or a matrix of gene expression variability, in order to utilize **RaceID** functions for visualization.

Usage

```
updateSC(object, res = NULL, cl = NULL, noise = NULL, flo = NULL)
```

Arguments

object	**RaceID** SCseq object.
res	List object returned by pruneKnn function to update SCseq with distance matrix and/or dimensionally reduced feature matrix in res. Default is NULL
cl	List object with clustering information, returned by the graphCluster function to update SCseq object with clustering partition and Fruchterman-Rheingold layout. Default is NULL.
noise	List object with the background noise model and a variability matrix, returned by the compNoise or compTBNoise function, to update SCseq object with a noise matrix. Default is NULL.
flo	Real number. Lower cutoff for the gene expression variability. All values < flo in the variability matrix are set to this level. Default is NULL and values are not reset.

Value

SCseq object with a distance matrix (slot distances) and a dimensionally reduced feature matrix (slot dimRed$x), or clustering partition (slot cpart and cluster$kpart) derived from the VarID analysis, and/or with a gene expression variability matrix in slot noise.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
sc <- compdist(sc)
d <- getExpData(sc)
res <- pruneKnn(d,distM=sc@distances,knn=10,alpha=1,no_cores=1,FSelect=FALSE)
cl <- graphCluster(res,pvalue=0.01)
sc <- updateSC(sc,res=res,cl=cl)
sc <- comptsne(sc)
plotmap(sc)
```

varRegression *Linear Regression of Sources of Variability*

Description

This functions regresses out variability associated with particular sources.

Usage

```
varRegression(object, vars = NULL, logscale = FALSE, Batch = FALSE)
```

Arguments

object	SCseq class object.
vars	data.frame of variables to be regressed out. Each column corresponds to a variable and each variable corresponds to a cell. The object must contain all cells, i.e. column names of the slot ndata from the SCseq object.
logscale	logical. If TRUE data are log-transformed prior to regression. Default is FALSE.
Batch	logical. If TRUE, then the function will regress out batch-associated variability based on genes stored in the filterpar$BGenes slot of the SCseq object. This requires prior batch correction with the filterdata function using bmode="RaceID".

Value

The function returns an updated SCseq object with the corrected expression matrix written to the slot dimRed$x of the SCseq object.

Examples

```
sc <- SCseq(intestinalDataSmall)
sc <- filterdata(sc)
b <- sub("(\\_\\d+)$","",colnames(intestinalData))
vars <- data.frame(row.names=colnames(intestinalData),batch=b)
sc   <- varRegression(sc,vars)
```

violinMarkerPlot *Violin plot of marker gene expression or noise*

Description

Displaying violin plots of gene expression or gene expression noise (epsilon) across (a set of) clusters

Usage

```
violinMarkerPlot(g, object, noise = NULL, set = NULL, ti = NULL)
```

Arguments

g	Valid gene ID corresponding to a (set of) rownames of object@ndata or noise.
object	**RaceID** SCseq object.
noise	List of noise parameters returned by compTBNoise. If this argument is given, then the distribution of noise (epsilon) is plotted. Default is NULL and normalized gene expression (normalized by median count across all clusters in set) is plotted.
set	Postive integer number or vector of integers corresponding to valid cluster numbers. Violin plots are shown for all clusters in set. Default is NULL and data are shown for all clusters in object@cpart.
ti	String of characters representing the title of the plot. Default is NULL and the first element of g is chosen.

Value

None

Index