Package 'ILoReg'

April 7, 2026

Type Package

Title ILoReg: a tool for high-resolution cell population identification from scRNA-Seq data

Version 1.21.0

Description ILoReg is a tool for identification of cell populations from scRNA-seq data. In particular, ILoReg is useful for finding cell populations with subtle transcriptomic differences. The method utilizes a self-supervised learning method, called Iteratitive Clustering Projection (ICP), to find cluster probabilities, which are used in noise reduction prior to PCA and the subsequent hierarchical clustering and t-SNE steps. Additionally, functions for differential expression analysis to find gene markers for the populations and gene expression visualization are provided.

License GPL-3

Imports Matrix, parallel, foreach, aricode, LiblineaR, SparseM, ggplot2, cowplot, RSpectra, umap, Rtsne, fastcluster, parallelDist, cluster, dendextend, DescTools, plyr, scales, pheatmap, reshape2, dplyr, doRNG, SingleCellExperiment, SummarizedExperiment, S4Vectors, methods, stats, doSNOW, utils

Depends R (>= 4.0.0)

Encoding UTF-8

LazyData TRUE

RoxygenNote 7.1.1

Suggests knitr, rmarkdown, BiocStyle

VignetteBuilder knitr

biocViews SingleCell, Software, Clustering, DimensionReduction, RNASeq, Visualization, Transcriptomics, DataRepresentation, DifferentialExpression, Transcription, GeneExpression

NeedsCompilation no

URL https://github.com/elolab/ILoReg

BugReports https://github.com/elolab/ILoReg/issues

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

git_branch devel

git_last_commit 676b5a1

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2026-04-06

Author Johannes Smolander [cre, aut],
Sini Junttila [aut],
Mikko S Venäläinen [aut],
Laura L Elo [aut]

Maintainer Johannes Smolander <johannes.smolander@gmail.com>

Contents

AnnotationScatterPlot *Visualiation of a custom annotation over nonlinear dimensionality reduction*

Description

The AnnotationScatterPlot enables visualizing arbitrary class labels over the nonliner dimensionality reduction, e.g. t-SNE or UMAP.

Usage

```
AnnotationScatterPlot.SingleCellExperiment(
  object,
  annotation,
  return.plot,
  dim.reduction.type,
  point.size,
  show.legend
)

## S4 method for signature 'SingleCellExperiment'
AnnotationScatterPlot(
  object,
  annotation = NULL,
  return.plot = FALSE,
  dim.reduction.type = "",
  point.size = 0.7,
  show.legend = FALSE
)
```

Arguments

`object`	of `SingleCellExperiment` class
`annotation`	a character vector, factor or numeric for the class labels.
`return.plot`	return.plot whether to return the ggplot2 object or just draw it. Default is `FALSE`.
`dim.reduction.type`	"tsne" or "umap". Default is `tsne`.
`point.size`	point size. Default is `0.7`.
`show.legend`	a logical denoting whether to show the legend on the right side of the plot. Default is `TRUE`.

Value

ggplot2 object if return.plot=TRUE

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- RunTSNE(sce)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
## Change the names to the first five alphabets and Visualize the annotation.
custom_annotation <- plyr::mapvalues(metadata(sce)$iloreg$clustering.manual,
                                     c(1,2,3,4,5),
                                     LETTERS[1:5])
AnnotationScatterPlot(sce,
                      annotation = custom_annotation,
                      return.plot = FALSE,
                      dim.reduction.type = "tsne",
                      show.legend = FALSE)
```

`CalcSilhInfo` *Estimating optimal K using silhouette*

Description

The function estimates the optimal number of clusters K from the dendrogram of the hierarhical clustering using the silhouette method.

Usage

```
CalcSilhInfo.SingleCellExperiment(object, K.start, K.end)

## S4 method for signature 'SingleCellExperiment'
CalcSilhInfo(object, K.start = 2, K.end = 50)
```

Arguments

`object`	of `SingleCellExperiment` class
`K.start`	a numeric for the smallest K value to be tested. Default is `2`.
`K.end`	a numeric for the largest K value to be tested. Default is `50`.

Value

object of `SingleCellExperiment` class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- CalcSilhInfo(sce)
```

ClusteringScatterPlot *Visualize the clustering over nonliner dimensionality reduction*

Description

ClusteringScatterPlot function enables visualizing the clustering over nonliner dimensionality reduction (t-SNE or UMAP).

Usage

```
ClusteringScatterPlot.SingleCellExperiment(
  object,
  clustering.type,
  return.plot,
  dim.reduction.type,
  point.size,
  title,
  show.legend
)

## S4 method for signature 'SingleCellExperiment'
ClusteringScatterPlot(
  object,
  clustering.type = "manual",
  return.plot = FALSE,
  dim.reduction.type = "",
  point.size = 0.7,
  title = "",
  show.legend = TRUE
)
```

Arguments

object	of SingleCellExperiment class

clustering.type
: "manual" or "optimal". "manual" refers to the clustering formed using the "SelectKClusters" function and "optimal" to the clustering formed using the "CalcSilhInfo" function. Default is "manual".

return.plot
: a logical denoting whether to return the ggplot2 object. Default is FALSE.

dim.reduction.type
: "tsne" or "umap". Default is "tsne".

point.size
: point size. Default is Default is 0.7.

title
: text to write above the plot

show.legend
: whether to show the legend on the right side of the plot. Default is TRUE.

Value

ggplot2 object if return.plot=TRUE

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
sce <- RunTSNE(sce)
ClusteringScatterPlot(sce,"manual",dim.reduction.type="tsne")
sce <- RunUMAP(sce)
ClusteringScatterPlot(sce,"manual",dim.reduction.type="umap")
```

DownOverSampling *Down- and oversample data*

Description

The function implements a script down- and oversamples data to include n cells.

Usage

```
DownOverSampling(x, n = 50)
```

Arguments

x
: A character or numeric vector of data to down-and oversample.

n
: How many cells to include per cluster.

Value

a list containing the output of the LiblineaR prediction

`FindAllGeneMarkers` *identification of gene markers for all clusters*

Description

FindAllGeneMarkers enables identifying gene markers for all clusters at once. This is done by differential expresission analysis where cells from one cluster are compared against the cells from the rest of the clusters. Gene and cell filters can be applied to accelerate the analysis, but this might lead to missing weak signals.

Usage

```
FindAllGeneMarkers.SingleCellExperiment(
  object,
  clustering.type,
  test,
  log2fc.threshold,
  min.pct,
  min.diff.pct,
  min.cells.group,
  max.cells.per.cluster,
  pseudocount.use,
  return.thresh,
  only.pos
)

## S4 method for signature 'SingleCellExperiment'
FindAllGeneMarkers(
  object,
  clustering.type = "manual",
  test = "wilcox",
  log2fc.threshold = 0.25,
  min.pct = 0.1,
  min.diff.pct = NULL,
  min.cells.group = 3,
  max.cells.per.cluster = NULL,
  pseudocount.use = 1,
  return.thresh = 0.01,
  only.pos = FALSE
)
```

Arguments

object	of SingleCellExperiment class
clustering.type	"manual" or "optimal". "manual" refers to the clustering formed using the "SelectKClusters" function and "optimal" to the clustering formed using the "CalcSilhInfo" function. Default is "manual".
test	Which test to use. Only "wilcoxon" (the Wilcoxon rank-sum test, AKA Mann-Whitney U test) is supported at the moment.
log2fc.threshold	Filters out genes that have log2 fold-change of the averaged gene expression values (with the pseudo-count value added to the averaged values before division if pseudocount.use > 0) below this threshold. Default is 0.25.
min.pct	Filters out genes that have dropout rate (fraction of cells expressing a gene) below this threshold in both comparison groups Default is 0.1.
min.diff.pct	Filters out genes that do not have this minimum difference in the dropout rates (fraction of cells expressing a gene) between the two comparison groups. Default is NULL.
min.cells.group	The minimum number of cells in the two comparison groups to perform the DE analysis. If the number of cells is below the threshold, then the DE analysis of this cluster is skipped. Default is 3.
max.cells.per.cluster	The maximun number of cells per cluster if downsampling is performed to speed up the DE analysis. Default is NULL, i.e. no downsampling.
pseudocount.use	A positive integer, which is added to the average gene expression values before calculating the fold-change, assuring that no divisions by zero occur. Default is 1.
return.thresh	If only.pos=TRUE, then return only genes that have the adjusted p-value (adjusted by the Bonferroni method) below or equal to this threshold. Default is 0.01.
only.pos	Whether to return only genes that have an adjusted p-value (adjusted by the Bonferroni method) below or equal to the threshold. Default is FALSE.

Value

a data frame of the results if positive results were found, else NULL

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
```

```
sce <- SelectKClusters(sce,K=5)
gene_markers <- FindAllGeneMarkers(sce)
```

FindGeneMarkers *Identification of gene markers for a cluster or two arbitrary combinations of clusters*

Description

FindGeneMarkers enables identifying gene markers for one cluster or two arbitrary combinations of clusters, e.g. 1_2 vs. 3_4_5. Gene and cell filters can be applied to accelerate the analysis, but this might lead to missing weak signals.

Usage

```
FindGeneMarkers.SingleCellExperiment(
  object,
  clusters.1,
  clusters.2,
  clustering.type,
  test,
  logfc.threshold,
  min.pct,
  min.diff.pct,
  min.cells.group,
  max.cells.per.cluster,
  pseudocount.use,
  return.thresh,
  only.pos
)

## S4 method for signature 'SingleCellExperiment'
FindGeneMarkers(
  object,
  clusters.1 = NULL,
  clusters.2 = NULL,
  clustering.type = "",
  test = "wilcox",
  logfc.threshold = 0.25,
  min.pct = 0.1,
  min.diff.pct = NULL,
  min.cells.group = 3,
  max.cells.per.cluster = NULL,
  pseudocount.use = 1,
  return.thresh = 0.01,
  only.pos = FALSE
)
```

Arguments

object	of SingleCellExperiment class
clusters.1	a character or numeric vector denoting which clusters to use in the first group (named group.1 in the results)
clusters.2	a character or numeric vector denoting which clusters to use in the second group (named group.2 in the results)
clustering.type	"manual" or "optimal". "manual" refers to the clustering formed using the "SelectKClusters" function and "optimal" to the clustering formed using the "CalcSilhInfo" function. Default is "manual".
test	Which test to use. Only "wilcoxon" (the Wilcoxon rank-sum test, AKA Mann-Whitney U test) is supported at the moment.
logfc.threshold	Filters out genes that have log2 fold-change of the averaged gene expression values (with the pseudo-count value added to the averaged values before division if pseudocount.use > 0) below this threshold. Default is 0.25.
min.pct	Filters out genes that have dropout rate (fraction of cells expressing a gene) below this threshold in both comparison groups Default is 0.1.
min.diff.pct	Filters out genes that do not have this minimum difference in the dropout rates (fraction of cells expressing a gene) between the two comparison groups. Default is NULL.
min.cells.group	The minimum number of cells in the two comparison groups to perform the DE analysis. If the number of cells is below the threshold, then the DE analysis is not performed. Default is 3.
max.cells.per.cluster	The maximun number of cells per cluster if downsampling is performed to speed up the DE analysis. Default is NULL, i.e. no downsampling.
pseudocount.use	A positive integer, which is added to the average gene expression values before calculating the fold-change. This makes sure that no divisions by zero occur. Default is 1.
return.thresh	If only.pos=TRUE, then return only genes that have the adjusted p-value (adjusted by the Bonferroni method) below or equal to this threshold. Default is 0.01.
only.pos	Whether to return only genes that have an adjusted p-value (adjusted by the Bonferroni method) below or equal to the threshold. Default is FALSE.

Value

a data frame of the results if positive results were found, else NULL

Examples

```
library(SingleCellExperiment)
```

```
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
gene_markes_1 <- FindGeneMarkers(sce,clusters.1=1)
gene_markes_1_vs_2 <- FindGeneMarkers(sce,clusters.1=1,clusters.2=2)
```

GeneHeatmap *Heatmap visualization of the gene markers identified by FindAllGeneMarkers*

Description

The GeneHeatmap function enables drawing a heatmap of the gene markers identified by FindAllGeneMarkers, where the cell are grouped by the clustering.

Usage

```
GeneHeatmap.SingleCellExperiment(object, clustering.type, gene.markers)

## S4 method for signature 'SingleCellExperiment'
GeneHeatmap(object, clustering.type = "manual", gene.markers = NULL)
```

Arguments

`object`	of `SingleCellExperiment` class
`clustering.type`	"manual" or "optimal". "manual" refers to the clustering formed using the "SelectKClusters" function and "optimal" to the clustering using the "CalcSilhInfo" function. Default is "manual".
`gene.markers`	a data frame of the gene markers generated by FindAllGeneMarkers function. To accelerate the drawing, filtering the dataframe by selecting e.g. top 10 genes is recommended.

Value

nothing

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
```

```
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,r=1,k=5) # Use L=200
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
gene_markers <- FindAllGeneMarkers(sce,log2fc.threshold = 0.5,min.pct = 0.5)
top10_log2FC <- SelectTopGenes(gene_markers,top.N=10,
criterion.type="log2FC",inverse=FALSE)
GeneHeatmap(sce,clustering.type = "manual",
 gene.markers = top10_log2FC)
```

GeneScatterPlot *Visualize gene expression over nonlinear dimensionality reduction*

Description

GeneScatterPlot enables visualizing gene expression of a gene over nonlinear dimensionality reduction with t-SNE or UMAP.

Usage

```
GeneScatterPlot.SingleCellExperiment(
  object,
  genes,
  return.plot,
  dim.reduction.type,
  point.size,
  title,
  plot.expressing.cells.last,
  nrow,
  ncol
)

## S4 method for signature 'SingleCellExperiment'
GeneScatterPlot(
  object,
  genes = "",
  return.plot = FALSE,
  dim.reduction.type = "tsne",
  point.size = 0.7,
  title = "",
  plot.expressing.cells.last = FALSE,
  nrow = NULL,
  ncol = NULL
)
```

Arguments

object	of SingleCellExperiment class
genes	a character vector of the genes to be visualized
return.plot	whether to return the ggplot2 object or just draw it (default FALSE)
dim.reduction.type	"tsne" or "umap" (default "tsne")
point.size	point size (default 0.7)
title	text to write above the plot
plot.expressing.cells.last	whether to plot the expressing genes last to make the points more visible
nrow	a positive integer that specifies the number of rows in the plot grid. Default is NULL.
ncol	a positive integer that specifies the number of columns in the plot grid. Default is NULL.

Value

ggplot2 object if return.plot=TRUE

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- RunTSNE(sce)
GeneScatterPlot(sce,"CD14",dim.reduction.type="tsne")
sce <- RunUMAP(sce)
GeneScatterPlot(sce,"CD14",dim.reduction.type="umap")
```

HierarchicalClustering

Hierarchical clustering using the Ward's method

Description

Perform Hierarchical clustering using the Ward's method.

Usage

```
HierarchicalClustering.SingleCellExperiment(object)

## S4 method for signature 'SingleCellExperiment'
HierarchicalClustering(object)
```

Arguments

object	of SingleCellExperiment class

Value

object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
```

LogisticRegression *Clustering projection using logistic regression from the LiblineaR R package*

Description

The function implements a script that downsamples data a dataset, trains a logistic regression classifier model and then projects its clustering onto itself using a trained L1-regularized logistic regression model.

Usage

```
LogisticRegression(
  training.sparse.matrix = NULL,
  training.ident = NULL,
  C = 0.3,
  reg.type = "L1",
  test.sparse.matrix = NULL,
  d = 0.3
)
```

Arguments

training.sparse.matrix	A sparse matrix (dgCMatrix) containing training sample's gene expression data with genes in rows and cells in columns. Default is NULL.
training.ident	A named factor containing sample's cluster labels for each cell in training.sparse.matrix. Default is NULL.

C — Cost of constraints violation in L1-regularized logistic regression (C). Default is 0.3.

reg.type — "L1" for LASSO and "L2" for Ridge. Default is "L1".

test.sparse.matrix — A sparse matrix (dgCMatrix) containing test sample's gene expression data with genes in rows and cells in columns. Default is NULL.

d — A numeric smaller than 1 and greater than 0 that determines how many cells per cluster should be down- and oversampled (d in N/k*d), where N is the total number of cells and k the number of clusters. Default is 0.3.

Value

a list containing the output of the LiblineaR prediction

MergeClusters *Merge clusters*

Description

MergeClusters function enables merging clusters and naming the newly formed cluster.

Usage

```
MergeClusters.SingleCellExperiment(object, clusters.to.merge, new.name)

## S4 method for signature 'SingleCellExperiment'
MergeClusters(object, clusters.to.merge = "", new.name = "")
```

Arguments

object — of SingleCellExperiment class

clusters.to.merge — a character or numeric vector for the names of the clusters to merge

new.name — a character for the new name of the merged cluster. If left empty, the new cluster name is formed by separating the cluster names by "_".

Value

object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
sce <- MergeClusters(sce,clusters.to.merge=c(1,2),new.name="merged1")
```

---

pbmc3k_500 *A toy dataset with 500 cells downsampled from the pbmc3k dataset.*

Description

The preprocessing was done using Cell Ranger v2.2.0 and the GRCh38.p12 human reference genome. The Normalization was done using the LogNormalize method of Seurat v3 R package. The sampling was done using the sample() function without replacement and set.seed(1) as initialization.

Usage

```
data(pbmc3k_500)
```

Format

pbmc3k_500, dgCMatrix object

Source

https://support.10xgenomics.com/single-cell-gene-expression

Examples

```
data(pbmc3k_500)
```

PCAElbowPlot *Elbow plot of the standard deviations of the principal components*

Description

Draw an elbow plot of the standard deviations of the principal components to deduce an appropriate value for p.

Usage

```
PCAElbowPlot.SingleCellExperiment(object, return.plot)

## S4 method for signature 'SingleCellExperiment'
PCAElbowPlot(object, return.plot = FALSE)
```

Arguments

object	object of class 'iloreg'
return.plot	logical indicating if the ggplot2 object should be returned (default FALSE)

Value

ggplot2 object if return.plot=TRUE

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
PCAElbowPlot(sce)
```

PrepareILoReg *Prepare* SingleCellExperiment *object for* ILoReg *analysis*

Description

This function prepares the SingleCellExperiment object for ILoReg analysis. The only required input is an object of class SingleCellExperiment with at least data in the logcounts slot.

Usage

```
PrepareILoReg.SingleCellExperiment(object)

## S4 method for signature 'SingleCellExperiment'
PrepareILoReg(object)
```

Arguments

object	an object of SingleCellExperiment class

Value

an object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
```

RenameAllClusters *Renaming all clusters at once*

Description

RenameAllClusters function enables renaming all cluster at once.

Usage

```
RenameAllClusters.SingleCellExperiment(object, new.cluster.names)

## S4 method for signature 'SingleCellExperiment'
RenameAllClusters(object, new.cluster.names = "")
```

Arguments

object	of SingleCellExperiment class
new.cluster.names	object of class 'iloreg'

Value

object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
sce <- RenameAllClusters(sce,new.cluster.names=LETTERS[seq_len(5)])
```

RenameCluster *Renaming one cluster*

Description

RenameCluster function enables renaming a cluster in ‘clustering.manual‘ slot.

Usage

```
RenameCluster.SingleCellExperiment(object, old.cluster.name, new.cluster.name)

## S4 method for signature 'SingleCellExperiment'
RenameCluster(object, old.cluster.name = "", new.cluster.name = "")
```

Arguments

`object`	of `SingleCellExperiment` class
`old.cluster.name`	a character variable denoting the old name of the cluster
`new.cluster.name`	a character variable the new name of the cluster

Value

object of `SingleCellExperiment` class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
sce <- RenameCluster(sce,1,"cluster1")
```

RunICP *Iterative Clustering Projection (ICP) clustering*

Description

The function implements Iterative Clustering Projection (ICP): a supervised learning -based clustering, which maximizes clustering similarity between the clustering and its projection by logistic regression.

Usage

```
RunICP(
  normalized.data = NULL,
  k = 15,
  d = 0.3,
  r = 5,
  C = 5,
  reg.type = "L1",
  max.iter = 200
)
```

Arguments

normalized.data	A sparse matrix (dgCMatrix) containing normalized gene expression data with genes in rows and cells in columns. Default is NULL.
k	A positive integer greater or equal to 2, denoting the number of clusters in ICP. Default is 15.
d	A numeric that defines how many cells per cluster should be down- and oversampled (d in ceiling(N/k*d)), when stratified.downsampling=FALSE, or what fraction should be downsampled in the stratified approach ,stratified.downsampling=TRUE. Default is 0.3.
r	A positive integer that denotes the number of reiterations performed until the algorithm stops. Default is 5.
C	Cost of constraints violation (C) for L1-regulatization. Default is 0.3.
reg.type	"L1" for LASSO and "L2" for Ridge. Default is "L1".
max.iter	A positive integer that denotes the maximum number of iterations performed until the algorithm ends. Default is 200.

Value

A list that includes the probability matrix and the clustering similarity measures: ARI, NMI, etc.

RunParallelICP *Run ICP runs parallerly*

Description

This functions runs in parallel `L` ICP runs, which is the computational bottleneck of ILoReg. With ~ 3,000 cells this step should be completed in ~ 2 h and ~ 1 h with 3 and 12 logical processors (threads), respectively.

Usage

```
RunParallelICP.SingleCellExperiment(
  object,
  k,
  d,
  L,
  r,
  C,
  reg.type,
  max.iter,
  threads
)

## S4 method for signature 'SingleCellExperiment'
RunParallelICP(
  object,
  k = 15,
  d = 0.3,
  L = 200,
  r = 5,
  C = 0.3,
  reg.type = "L1",
  max.iter = 200,
  threads = 0
)
```

Arguments

`object`	An object of `SingleCellExperiment` class.
`k`	A positive integer greater or equal to `2`, denoting the number of clusters in Iterative Clustering Projection (ICP). Decreasing `k` leads to smaller cell populations diversity and vice versa. Default is `15`.
`d`	A numeric greater than `0` and smaller than `1` that determines how many cells `n` are down- or oversampled from each cluster into the training data (`n=N/k*d`), where `N` is the total number of cells, `k` is the number of clusters in ICP. Increasing above 0.3 leads greadually to smaller cell populations diversity. Default is `0.3`.

L	A positive integer greater than 1 denoting the number of the ICP runs to run. Default is 200. Increasing recommended with a significantly larger sample size (tens of thousands of cells). Default is 200.
r	A positive integer that denotes the number of reiterations performed until the ICP algorithm stops. Increasing recommended with a significantly larger sample size (tens of thousands of cells). Default is 5.
C	A positive real number denoting the cost of constraints violation in the L1-regularized logistic regression model from the LIBLINEAR library. Decreasing leads to more stringent feature selection, i.e. less genes are selected that are used to build the projection classifier. Decreasing to a very low value (~ 0.01) can lead to failure to identify central cell populations. Default 0.3.
reg.type	"L1" or "L2". L2-regularization was not investigated in the manuscript, but it leads to a more conventional outcome (less subpopulations). Default is "L1".
max.iter	A positive integer that denotes the maximum number of iterations performed until ICP stops. This parameter is only useful in situations where ICP converges extremely slowly, preventing the algorithm to run too long. In most cases, reaching the number of reiterations (r=5) terminates the algorithm. Default is 200.
threads	A positive integer that specifies how many logical processors (threads) to use in parallel computation. Set 1 to disable parallelism altogether or 0 to use all available threas except one. Default is 0.

Value

an object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,r=1,k=5)
```

RunPCA *PCA transformation of the joint probability matrix*

Description

Perform the PCA transformation of the joint probability matrix, which reduces the dimensionality from k*L to p

Usage

```
RunPCA.SingleCellExperiment(object, p, scale, threshold)

## S4 method for signature 'SingleCellExperiment'
RunPCA(object, p = 50, scale = FALSE, threshold = 0)
```

Arguments

object	object of SingleCellExperiment class
p	a positive integer denoting the number of principal components to calculate and select. Default is 50.
scale	a logical specifying whether the probabilities should be standardized to unit-variance before running PCA. Default is FALSE.
threshold	a thresfold for filtering out ICP runs before PCA with the lower terminal projection accuracy below the threshold. Default is 0.

Value

object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
```

RunTSNE *Barnes-Hut implementation of t-Distributed Stochastic Neighbor Embedding (t-SNE)*

Description

Run nonlinear dimensionality reduction using t-SNE with the PCA-transformed consensus matrix as input.

Usage

```
RunTSNE.SingleCellExperiment(object, perplexity)

## S4 method for signature 'SingleCellExperiment'
RunTSNE(object, perplexity = 30)
```

Arguments

object	of SingleCellExperiment class
perplexity	perplexity of t-SNE

Value

object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- RunTSNE(sce)
```

RunUMAP *Uniform Manifold Approximation and Projection (UMAP)*

Description

Run nonlinear dimensionality reduction using UMAP with the PCA-transformed consensus matrix as input.

Usage

```
RunUMAP.SingleCellExperiment(object)

## S4 method for signature 'SingleCellExperiment'
RunUMAP(object)
```

Arguments

object	of SingleCellExperiment class

Value

object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- RunUMAP(sce)
```

SelectKClusters *Selecting K clusters from hierarchical clustering*

Description

Selects K clusters from the dendrogram.

Usage

```
SelectKClusters.SingleCellExperiment(object, K)

## S4 method for signature 'SingleCellExperiment'
SelectKClusters(object, K = NULL)
```

Arguments

object	of SingleCellExperiment class
K	a positive integer denoting how many clusters to select

Value

object of SingleCellExperiment class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
```

SelectTopGenes *Select top or bottom N genes based on a selection criterion*

Description

The SelectTopGenes function enables selecting top or bottom N genes based on a criterion (e.g. log2FC or adj.p.value).

Usage

```
SelectTopGenes(
  gene.markers = NULL,
  top.N = 10,
  criterion.type = "log2FC",
  inverse = FALSE
)
```

Arguments

gene.markers	A data frame of the gene markers found by FindAllGeneMarkers function.
top.N	How many top or bottom genes to select. Default is 10.
criterion.type	Which criterion to use for selecting the genes. Default is "log2FC".
inverse	Whether to select bottom instead of top N genes. Default is FALSE.

Value

an object of ‘data.frame‘ class

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
gene_markers <- FindAllGeneMarkers(sce)
## Select top 10 markers based on log2 fold-change
top10_log2FC <- SelectTopGenes(gene_markers,
                               top.N = 10,
                               criterion.type = "log2FC",
                               inverse = FALSE)
```

SilhouetteCurve *Silhouette curve*

Description

Draw the silhouette curve: the average silhouette value across the cells for a range of different K values.

Usage

```
SilhouetteCurve.SingleCellExperiment(object, return.plot)

## S4 method for signature 'SingleCellExperiment'
SilhouetteCurve(object, return.plot = FALSE)
```

Arguments

object	of SingleCellExperiment class
return.plot	a logical denoting whether the ggplot2 object should be returned. Default is FALSE.

Value

ggplot2 object if return.plot=TRUE

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- CalcSilhInfo(sce)
SilhouetteCurve(sce)
```

VlnPlot *Gene expression visualization using violin plots*

Description

The VlnPlot function enables visualizing expression levels of a gene, or multiple genes, across clusters using Violin plots.

Usage

```
VlnPlot.SingleCellExperiment(
  object,
  clustering.type,
  genes,
  return.plot,
  rotate.x.axis.labels
)

## S4 method for signature 'SingleCellExperiment'
```

```
VlnPlot(
  object,
  clustering.type = "manual",
  genes = NULL,
  return.plot = FALSE,
  rotate.x.axis.labels = FALSE
)
```

Arguments

object	of SingleCellExperiment class
clustering.type	"manual" or "optimal". "manual" refers to the clustering formed using the "SelectKClusters" function and "optimal" to the clustering formed using the "CalcSilhInfo" function. Default is "manual".
genes	a character vector denoting the gene names that are visualized
return.plot	return.plot whether to return the ggplot2 object
rotate.x.axis.labels	a logical denoting whether the x-axis labels should be rotated 90 degrees. or just draw it. Default is FALSE.

Value

ggplot2 object if return.plot=TRUE

Examples

```
library(SingleCellExperiment)
sce <- SingleCellExperiment(assays = list(logcounts = pbmc3k_500))
sce <- PrepareILoReg(sce)
## These settings are just to accelerate the example, use the defaults.
sce <- RunParallelICP(sce,L=2,threads=1,C=0.1,k=5,r=1)
sce <- RunPCA(sce,p=5)
sce <- HierarchicalClustering(sce)
sce <- SelectKClusters(sce,K=5)
VlnPlot(sce,genes=c("CD3D","CD79A","CST3"))
```

Index