

# Package ‘scGPS’

April 29, 2025

**Type** Package

**Title** A complete analysis of single cell subpopulations, from identifying subpopulations to analysing their relationship (scGPS = single cell Global Predictions of Subpopulation)

**Version** 1.23.0

**Description** The package implements two main algorithms to answer two key questions: a SCORE (Stable Clustering at Optimal REsolution) to find subpopulations, followed by scGPS to investigate the relationships between subpopulations.

**Encoding** UTF-8

**LazyData** true

**License** GPL-3

**BugReports** <https://github.com/IMB-Computational-Genomics-Lab/scGPS/issues>

**url** <https://github.com/IMB-Computational-Genomics-Lab/scGPS/>

**RoxygenNote** 7.1.1

**Depends** R (>= 3.6), SummarizedExperiment, dynamicTreeCut, SingleCellExperiment

**biocViews** SingleCell, Clustering, DataImport, Sequencing, Coverage

**Imports** glmnet (> 2.0), caret (>= 6.0), ggplot2 (>= 2.2.1), fastcluster, dplyr, Rcpp, RcppArmadillo, RcppParallel, grDevices, graphics, stats, utils, DESeq2, locfit

**Suggests** Matrix (>= 1.2), testthat, knitr, parallel, rmarkdown, RColorBrewer, ReactomePA, clusterProfiler, cowplot, org.Hs.eg.db, reshape2, xlsx, dendextend, networkD3, Rtsne, BiocParallel, e1071, WGCNA, devtools, DOSE

**VignetteBuilder** knitr

**LinkingTo** Rcpp, RcppArmadillo, RcppParallel

**SystemRequirements** GNU make

**git\_url** <https://git.bioconductor.org/packages/scGPS>

**git\_branch** devel

**git\_last\_commit** 9c713b2

**git\_last\_commit\_date** 2025-04-15

**Repository** Bioconductor 3.22

**Date/Publication** 2025-04-28

**Author** Quan Nguyen [aut, cre],  
Michael Thompson [aut],  
Anne Senabouth [aut]

**Maintainer** Quan Nguyen <quan.nguyen@uq.edu.au>

## Contents

|                                       |    |
|---------------------------------------|----|
| add_import . . . . .                  | 3  |
| annotate_clusters . . . . .           | 3  |
| bootstrap_parallel . . . . .          | 4  |
| bootstrap_prediction . . . . .        | 5  |
| calcDist . . . . .                    | 6  |
| calcDistArma . . . . .                | 7  |
| clustering . . . . .                  | 8  |
| clustering_bagging . . . . .          | 9  |
| CORE_bagging . . . . .                | 10 |
| CORE_clustering . . . . .             | 11 |
| CORE_subcluster . . . . .             | 12 |
| day_2_cardio_cell_sample . . . . .    | 13 |
| day_5_cardio_cell_sample . . . . .    | 14 |
| distvec . . . . .                     | 14 |
| find_markers . . . . .                | 15 |
| find_optimal_stability . . . . .      | 16 |
| find_stability . . . . .              | 17 |
| mean_cpp . . . . .                    | 18 |
| new_scGPS_object . . . . .            | 18 |
| new_summarized_scGPS_object . . . . . | 19 |
| PCA . . . . .                         | 20 |
| plot_CORE . . . . .                   | 21 |
| plot_optimal_CORE . . . . .           | 22 |
| plot_reduced . . . . .                | 23 |
| predicting . . . . .                  | 24 |
| PrinComp_cpp . . . . .                | 25 |
| rand_index . . . . .                  | 26 |
| rcpp_Eucl_distance_NotPar . . . . .   | 26 |
| rcpp_parallel_distance . . . . .      | 27 |
| reformat_LASSO . . . . .              | 28 |
| subset_cpp . . . . .                  | 29 |
| sub_clustering . . . . .              | 29 |
| summary_accuracy . . . . .            | 30 |
| summary_deviance . . . . .            | 31 |
| summary_prediction_lasso . . . . .    | 32 |
| summary_prediction_lda . . . . .      | 32 |
| top_var . . . . .                     | 33 |
| tp_cpp . . . . .                      | 34 |
| training . . . . .                    | 34 |
| training_gene_sample . . . . .        | 36 |
| tSNE . . . . .                        | 36 |
| var_cpp . . . . .                     | 37 |

---

|            |                   |
|------------|-------------------|
| add_import | <i>add_import</i> |
|------------|-------------------|

---

## Description

import packages to namespace

---

|                   |                                                                         |
|-------------------|-------------------------------------------------------------------------|
| annotate_clusters | <i>annotate_clusters functionally annotates the identified clusters</i> |
|-------------------|-------------------------------------------------------------------------|

---

## Description

often we need to label clusters with unique biological characters. One of the common approach to annotate a cluster is to perform functional enrichment analysis. The annotate implements ReactomePA and clusterProfiler for this analysis type in R. The function require installation of several databases as described below.

## Usage

```
annotate_clusters(
  DEgeneList,
  pvalueCutoff = 0.05,
  gene_symbol = TRUE,
  species = "human"
)
```

## Arguments

|              |                                                                  |
|--------------|------------------------------------------------------------------|
| DEgeneList   | is a vector of gene symbols, convertible to ENTREZID             |
| pvalueCutoff | is a numeric of the cutoff p value                               |
| gene_symbol  | logical of whether the geneList is a gene symbol                 |
| species      | is the selection of 'human' or 'mouse', default to 'human' genes |

## Value

write enrichment test output to a file and an enrichment test object for plotting

## Examples

```
genes <- training_gene_sample
genes <- genes$Merged_unique[seq_len(50)]
enrichment_test <- annotate_clusters(genes, pvalueCutoff=0.05,
  gene_symbol=TRUE, species = 'human')
clusterProfiler::dotplot(enrichment_test, showCategory=15)
```

---

|                    |                                                                                   |
|--------------------|-----------------------------------------------------------------------------------|
| bootstrap_parallel | <i>BootStrap runs for both scGPS training and prediction with parallel option</i> |
|--------------------|-----------------------------------------------------------------------------------|

---

## Description

same as bootstrap\_prediction, but with an multicore option

## Usage

```
bootstrap_parallel(
  ncores = 4,
  nboots = 1,
  genes = genes,
  mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2,
  c_selectID,
  listData = list(),
  cluster_mixedpop1 = NULL,
  cluster_mixedpop2 = NULL
)
```

## Arguments

|                   |                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------|
| ncores            | a number specifying how many cpus to be used for running                                    |
| nboots            | a number specifying how many bootstraps to be run                                           |
| genes             | a gene list to build the model                                                              |
| mixedpop1         | a <a href="#">SingleCellExperiment</a> object from a mixed population for training          |
| mixedpop2         | a <a href="#">SingleCellExperiment</a> object from a target mixed population for prediction |
| c_selectID        | the root cluster in mixedpop1 to be compared to clusters in mixedpop2                       |
| listData          | a list object, which contains trained results for the first mixed population                |
| cluster_mixedpop1 | a vector of cluster assignment for mixedpop1                                                |
| cluster_mixedpop2 | a vector of cluster assignment for mixedpop2                                                |

## Value

a list with prediction results written in to the index out\_idx

## Author(s)

Quan Nguyen, 2017-11-25

## Examples

```
day2 <- day_2_cardio_cell_sample
mixedpop1 <- new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2_geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
genes <- training_gene_sample
genes <- genes$Merged_unique
#prl_boots <- bootstrap_parallel(ncores = 4, nboots = 1, genes=genes,
#  mixedpop1 = mixedpop1, mixedpop2 = mixedpop2, c_selectID=1,
#  listData = list())
#prl_boots[[1]]$ElasticNetPredict
#prl_boots[[1]]$LDAPredict
```

---

bootstrap\_prediction    *BootStrap runs for both scGPS training and prediction*

---

## Description

ElasticNet and LDA prediction for each of all the subpopulations in the new mixed population after training the model for a subpopulation in the first mixed population. The number of bootstraps to be run can be specified.

## Usage

```
bootstrap_prediction(
  nboots = 1,
  genes = genes,
  mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2,
  c_selectID = NULL,
  listData = list(),
  cluster_mixedpop1 = NULL,
  cluster_mixedpop2 = NULL,
  trainset_ratio = 0.5,
  LDA_run = TRUE,
  verbose = FALSE,
  log_transform = FALSE
)
```

## Arguments

|            |                                                                                             |
|------------|---------------------------------------------------------------------------------------------|
| nboots     | a number specifying how many bootstraps to be run                                           |
| genes      | a gene list to build the model                                                              |
| mixedpop1  | a <a href="#">SingleCellExperiment</a> object from a mixed population for training          |
| mixedpop2  | a <a href="#">SingleCellExperiment</a> object from a target mixed population for prediction |
| c_selectID | the root cluster in mixedpop1 to be compared to clusters in mixedpop2                       |
| listData   | a list object, which contains trained results for the first mixed population                |

cluster\_mixedpop1  
a vector of cluster assignment for mixedpop1

cluster\_mixedpop2  
a vector of cluster assignment for mixedpop2

trainset\_ratio  
a number specifying the proportion of cells to be part of the training subpopulation

LDA\_run  
logical, if the LDA prediction is added to compare to ElasticNet

verbose  
a logical whether to display additional messages

log\_transform  
boolean whether log transform should be computed

**Value**

a list with prediction results written in to the index out\_idx

**Author(s)**

Quan Nguyen, 2017-11-25

**See Also**

[bootstrap\\_parallel](#) for parallel options

**Examples**

```
day2 <- day_2_cardio_cell_sample
mixedpop1 <- new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2_geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
genes <- training_gene_sample
genes <- genes$Merged_unique
cluster_mixedpop1 <- colData(mixedpop1)[,1]
cluster_mixedpop2 <- colData(mixedpop2)[,1]
c_selectID <- 2
test <- bootstrap_prediction(nboots = 1, mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2, genes=genes, listData =list(),
  cluster_mixedpop1 = cluster_mixedpop1,
  cluster_mixedpop2 = cluster_mixedpop2, c_selectID = c_selectID)
names(test)
test$ElasticNetPredict
test$LDAPredict
```

---

calcDist

---

*Compute Euclidean distance matrix by rows*


---

**Description**

Compute Euclidean distance matrix by rows

**Usage**

```
calcDist(x)
```

**Arguments**

x                      A numeric matrix

**Value**

a distance matrix

**Examples**

```
mat_test <-matrix(rnbinom(1000,mu=0.01, size=10),nrow=1000)
calcDist(mat_test)
```

---

|              |                                                  |
|--------------|--------------------------------------------------|
| calcDistArma | <i>Compute Euclidean distance matrix by rows</i> |
|--------------|--------------------------------------------------|

---

**Description**

Compute Euclidean distance matrix by rows

**Usage**

```
calcDistArma(x)
```

**Arguments**

x                      A numeric matrix

**Value**

a distance matrix

**Examples**

```
mat_test <-matrix(rnbinom(1000,mu=0.01, size=10),nrow=1000)
#library(microbenchmark)
#microbenchmark(calcDistArma(mat_test), dist(mat_test), times=3)
```

clustering

*HC clustering for a number of resolutions***Description**

performs 40 clustering runs or more depending on windows

**Usage**

```
clustering(
  object = NULL,
  ngenes = 1500,
  windows = seq(from = 0.025, to = 1, by = 0.025),
  remove_outlier = c(0),
  nRounds = 1,
  PCA = FALSE,
  nPCs = 20,
  verbose = FALSE,
  log_transform = FALSE
)
```

**Arguments**

|                |                                                                                                                       |
|----------------|-----------------------------------------------------------------------------------------------------------------------|
| object         | is a <a href="#">SingleCellExperiment</a> object from the train mixed population                                      |
| ngenes         | number of top variable genes to be used                                                                               |
| windows        | a numeric specifying the number of windows to test                                                                    |
| remove_outlier | a vector containing IDs for clusters to be removed the default vector contains 0, as 0 is the cluster with singletons |
| nRounds        | number of iterations to remove a selected clusters                                                                    |
| PCA            | logical specifying if PCA is used before calculating distance matrix                                                  |
| nPCs           | number of principal components from PCA dimensional reduction to be used                                              |
| verbose        | a logical whether to display additional messages                                                                      |
| log_transform  | boolean whether log transform should be computed                                                                      |

**Value**

clustering results

**Author(s)**

Quan Nguyen, 2017-11-25

**Examples**

```
day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
test <- clustering(mixedpop2, remove_outlier = c(0))
```

---

|                    |                                                  |
|--------------------|--------------------------------------------------|
| clustering_bagging | <i>HC clustering for a number of resolutions</i> |
|--------------------|--------------------------------------------------|

---

## Description

subsamples cells for each bagging run and performs 40 clustering runs or more depending on windows.

## Usage

```
clustering_bagging(
  object = NULL,
  ngenes = 1500,
  bagging_run = 20,
  subsample_proportion = 0.8,
  windows = seq(from = 0.025, to = 1, by = 0.025),
  remove_outlier = c(0),
  nRounds = 1,
  PCA = FALSE,
  nPCs = 20,
  log_transform = FALSE
)
```

## Arguments

|                      |                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------|
| object               | is a <a href="#">SingleCellExperiment</a> object from the train mixed population.                                      |
| ngenes               | number of genes used for clustering calculations.                                                                      |
| bagging_run          | an integer specifying the number of bagging runs to be computed.                                                       |
| subsample_proportion | a numeric specifying the proportion of the tree to be chosen in subsampling.                                           |
| windows              | a numeric vector specifying the ranges of each window.                                                                 |
| remove_outlier       | a vector containing IDs for clusters to be removed the default vector contains 0, as 0 is the cluster with singletons. |
| nRounds              | a integer specifying the number rounds to attempt to remove outliers.                                                  |
| PCA                  | logical specifying if PCA is used before calculating distance matrix.                                                  |
| nPCs                 | an integer specifying the number of principal components to use.                                                       |
| log_transform        | boolean whether log transform should be computed                                                                       |

## Value

a list of clustering results containing each bagging run as well as the clustering of the original tree and the tree itself.

## Author(s)

Quan Nguyen, 2017-11-25

## Examples

```
day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo, CellMetadata = day5$dat5_clusters)
test <- clustering_bagging(mixedpop2, remove_outlier = c(0),
  bagging_run = 2, subsample_proportion = .7)
```

---

|              |                                                                                                                 |
|--------------|-----------------------------------------------------------------------------------------------------------------|
| CORE_bagging | <i>Main clustering SCORE (CORE V2.0) Stable Clustering at Optimal RESolution with bagging and bootstrapping</i> |
|--------------|-----------------------------------------------------------------------------------------------------------------|

---

## Description

CORE is an algorithm to generate reproducible clustering, CORE is first implemented in ascend R package. Here, CORE V2.0 uses bagging analysis to find a stable clustering result and detect rare clusters mixed population.

## Usage

```
CORE_bagging(
  mixedpop = NULL,
  bagging_run = 20,
  subsample_proportion = 0.8,
  windows = seq(from = 0.025, to = 1, by = 0.025),
  remove_outlier = c(0),
  nRounds = 1,
  PCA = FALSE,
  nPCs = 20,
  ngenes = 1500,
  log_transform = FALSE
)
```

## Arguments

|                      |                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------|
| mixedpop             | is a <a href="#">SingleCellExperiment</a> object from the train mixed population.                                      |
| bagging_run          | an integer specifying the number of bagging runs to be computed.                                                       |
| subsample_proportion | a numeric specifying the proportion of the tree to be chosen in subsampling.                                           |
| windows              | a numeric vector specifying the ranges of each window.                                                                 |
| remove_outlier       | a vector containing IDs for clusters to be removed the default vector contains 0, as 0 is the cluster with singletons. |
| nRounds              | an integer specifying the number rounds to attempt to remove outliers.                                                 |
| PCA                  | logical specifying if PCA is used before calculating distance matrix.                                                  |
| nPCs                 | an integer specifying the number of principal components to use.                                                       |
| ngenes               | number of genes used for clustering calculations.                                                                      |
| log_transform        | boolean whether log transform should be computed                                                                       |

**Value**

a list with clustering results of all iterations, and a selected optimal resolution

**Author(s)**

Quan Nguyen, 2018-05-11

**Examples**

```
day5 <- day_5_cardio_cell_sample
cellnames<-colnames(day5$dat5_counts)
cluster <-day5$dat5_clusters
cellnames <- data.frame('cluster' = cluster, 'cellBarcodes' = cellnames)
#day5$dat5_counts needs to be in a matrix format
mixedpop2 <-new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
test <- CORE_bagging(mixedpop2, remove_outlier = c(0), PCA=FALSE,
  bagging_run = 2, subsample_proportion = .7)
```

---

CORE\_clustering

---

Main clustering CORE V2.0 updated

---

**Description**

CORE is an algorithm to generate reproducible clustering, CORE is first implemented in ascend R package. Here, CORE V2.0 introduces several new functionalities, including three key features: fast (and more memory efficient) implementation with C++ and parallelisation options allowing clustering of hundreds of thousands of cells (ongoing development), outlier removal important if singletons exist (done), a number of dimensionality reduction methods including the imputation implementation (CIDR) for confirming clustering results (done), and an option to select the number of optimisation tree height windows for increasing resolution

**Usage**

```
CORE_clustering(
  mixedpop = NULL,
  windows = seq(from = 0.025, to = 1, by = 0.025),
  remove_outlier = c(0),
  nRounds = 1,
  PCA = FALSE,
  nPCs = 20,
  ngenes = 1500,
  verbose = FALSE,
  log_transform = FALSE
)
```

**Arguments**

|          |                                                                                  |
|----------|----------------------------------------------------------------------------------|
| mixedpop | is a <a href="#">SingleCellExperiment</a> object from the train mixed population |
| windows  | a numeric specifying the number of windows to test                               |

|                             |                                                                                                                        |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------|
| <code>remove_outlier</code> | a vector containing IDs for clusters to be removed the default vector contains 0, as 0 is the cluster with singletons. |
| <code>nRounds</code>        | an integer specifying the number rounds to attempt to remove outliers.                                                 |
| <code>PCA</code>            | logical specifying if PCA is used before calculating distance matrix                                                   |
| <code>nPCs</code>           | an integer specifying the number of principal components to use.                                                       |
| <code>ngenes</code>         | number of genes used for clustering calculations.                                                                      |
| <code>verbose</code>        | a logical whether to display additional messages                                                                       |
| <code>log_transform</code>  | boolean whether log transform should be computed                                                                       |

**Value**

a list with clustering results of all iterations, and a selected optimal resolution

**Author(s)**

Quan Nguyen, 2017-11-25

**Examples**

```
day5 <- day_5_cardio_cell_sample
#day5$dat5_counts needs to be in a matrix format
cellnames <- colnames(day5$dat5_counts)
cluster <- day5$dat5_clusters
cellnames <- data.frame('Cluster'=cluster, 'cellBarcodes' = cellnames)
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo, CellMetadata = day5$dat5_clusters)
test <- CORE_clustering(mixedpop2, remove_outlier = c(0), PCA=FALSE, nPCs=20,
  ngenes=1500)
```

---

|                 |                                                            |
|-----------------|------------------------------------------------------------|
| CORE_subcluster | <i>sub_clustering (optional) after running CORE 'test'</i> |
|-----------------|------------------------------------------------------------|

---

**Description**

CORE\_subcluster allows re-cluster the CORE clustering result

**Usage**

```
CORE_subcluster(
  mixedpop = NULL,
  windows = seq(from = 0.025, to = 1, by = 0.025),
  select_cell_index = NULL,
  ngenes = 1500
)
```

**Arguments**

|                                |                                                                                  |
|--------------------------------|----------------------------------------------------------------------------------|
| <code>mixedpop</code>          | is a <a href="#">SingleCellExperiment</a> object from the train mixed population |
| <code>windows</code>           | a numeric specifying the number of windows to test                               |
| <code>select_cell_index</code> | a vector containing indexes for cells in selected clusters to be reclustered     |
| <code>ngenes</code>            | number of genes used for clustering calculations.                                |

**Value**

a list with clustering results of all iterations, and a selected optimal resolution

**Author(s)**

Quan Nguyen, 2017-11-25

**Examples**

```
day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
test <- CORE_clustering(mixedpop2, remove_outlier = c(0))
```

---

day\_2\_cardio\_cell\_sample

*One of the two example single-cell count matrices to be used for training **scGPS** model*

---

**Description**

The count data set contains counts for 16990 genes for 590 cells randomly subsampled from day-2 cardio-differentiation population. The vector of clustering information contains corresponding to cells in the count matrix

**Usage**

```
day_2_cardio_cell_sample
```

**Format**

a list instance, containing a count matrix and a vector with clustering information

**Value**

a list, with the name day2

**Author(s)**

Quan Nguyen, 2017-11-25

**Source**

Dr Joseph Powell's laboratory, IMB, UQ

---

day\_5\_cardio\_cell\_sample

*One of the two example single-cell count matrices to be used for scGPS prediction*

---

### Description

The count data set contains counts for 17402 genes for 983 cells (1 row per gene) randomly sub-sampled from day-5 cardio-differentiation population. The vector of clustering information contains corresponding to cells in the count matrix.

### Usage

day\_5\_cardio\_cell\_sample

### Format

a list instance, containing a count matrix and a vector with clustering information.

### Value

a list, with the name day5

### Author(s)

Quan Nguyen, 2017-11-25

### Source

Dr Joseph Powell's laboratory, IMB, UQ

---

distvec

*Compute Distance between two vectors*

---

### Description

Compute Distance between two vectors

### Usage

distvec(x, y)

### Arguments

|   |                  |
|---|------------------|
| x | A numeric vector |
| y | A numeric vector |

### Value

a numeric distance

**Examples**

```
x <-matrix(rnbinom(1000,mu=0.01, size=10),nrow=1000)
x <- x[1,]
y <-matrix(rnbinom(1000,mu=0.01, size=10),nrow=1000)
y <- y[1,]
distvec(x, y)
```

find\_markers

*find marker genes***Description**

Find DE genes from comparing one clust vs remaining

**Usage**

```
find_markers(
  expression_matrix = NULL,
  cluster = NULL,
  selected_cluster = NULL,
  fitType = "local",
  dispersion_method = "per-condition",
  sharing_Mode = "maximum"
)
```

**Arguments**

**expression\_matrix** is a normalised expression matrix.

**cluster** corresponding cluster information in the expression\_matrix by running CORE clustering or using other methods.

**selected\_cluster** a vector of unique cluster ids to calculate

**fitType** string specifying 'local' or 'parametric' for DEseq dispersion estimation

**dispersion\_method** one of the options c( 'pooled', 'pooled-CR', 'per-condition', 'blind' )

**sharing\_Mode** one of the options c("maximum", "fit-only", "gene-est-only")

**Value**

a list containing sorted DESeq analysis results

**Author(s)**

Quan Nguyen, 2017-11-25

**Examples**

```

day2 <- day_2_cardio_cell_sample
mixedpop1 <- new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
# depending on the data, the DESeq::estimateDispersions function requires
# suitable fitType
# and dispersion_method options
DEgenes <- find_markers(expression_matrix=assay(mixedpop1),
  cluster = colData(mixedpop1)[,1],
  selected_cluster=c(1), #can also run for more
  #than one clusters, e.g.selected_cluster = c(1,2)
  fitType = "parametric",
  dispersion_method = "blind",
  sharing_Mode="fit-only"
)

names(DEgenes)

```

---

find\_optimal\_stability

*Find the optimal cluster*


---

**Description**

from calculated stability based on Rand indexes for consecutive clustering run, find the resolution (window), where the stability is the highest

**Usage**

```

find_optimal_stability(
  list_clusters,
  run_RandIdx,
  bagging = FALSE,
  windows = seq(from = 0.025, to = 1, by = 0.025)
)

```

**Arguments**

|               |                                                                         |
|---------------|-------------------------------------------------------------------------|
| list_clusters | is a list object containing 40 clustering results                       |
| run_RandIdx   | is a data frame object from iterative clustering runs                   |
| bagging       | is a logical that is true if bagging is to be performed, changes return |
| windows       | a numeric vector specifying the ranges of each window.                  |

**Value**

bagging == FALSE => a list with optimal stability, cluster count and summary stats  
 bagging == TRUE => a list with high res cluster count, optimal cluster count and keystats

**Author(s)**

Quan Nguyen, 2017-11-25

**Examples**

```

day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
cluster_all <- clustering(object=mixedpop2)
stab_df <- find_stability(list_clusters=cluster_all$list_clusters,
  cluster_ref = cluster_all$cluster_ref)
optimal_stab <- find_optimal_stability(list_clusters =
  cluster_all$list_clusters, stab_df, bagging = FALSE)

```

---

|                |                                  |
|----------------|----------------------------------|
| find_stability | <i>Calculate stability index</i> |
|----------------|----------------------------------|

---

**Description**

from clustering results, compare similarity between clusters by adjusted Randindex

**Usage**

```
find_stability(list_clusters = NULL, cluster_ref = NULL)
```

**Arguments**

`list_clusters` is a object from the iterative clustering runs  
`cluster_ref` is a object from the reference cluster

**Value**

a data frame with stability scores and rand\_index results

**Author(s)**

Quan Nguyen, 2017-11-25

**Examples**

```

day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
cluster_all <- clustering(object=mixedpop2)
stab_df <- find_stability(list_clusters=cluster_all$list_clusters,
  cluster_ref = cluster_all$cluster_ref)

```

---

mean\_cpp

*Calculate mean*


---

### Description

Calculate mean

### Usage

```
mean_cpp(x)
```

### Arguments

x                      integer.

### Value

a scalar value

### Examples

```
mean_cpp(seq_len(10^6))
```

---

new\_scGPS\_object

*new\_scGPS\_object*


---

### Description

[new\\_scGPS\\_object](#) generates a scGPS object in the [SingleCellExperiment](#) class for use with the scGPS package. This object contains an expression matrix, associated metadata (cells, genes, clusters). The data are expected to be normalised counts.

### Usage

```
new_scGPS_object(
  ExpressionMatrix = NULL,
  GeneMetadata = NULL,
  CellMetadata = NULL,
  LogMatrix = NULL
)
```

### Arguments

ExpressionMatrix

An expression matrix in data.frame or matrix format. Rows should represent a transcript and its normalised counts, while columns should represent individual cells.

|              |                                                                                                                                                                                                                                                                                                                                                                            |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| GeneMetadata | A data frame or vector containing gene identifiers used in the expression matrix. The first column should hold the gene identifiers you are using in the expression matrix. Other columns contain information about the genes, such as their corresponding ENSEMBL transcript identifiers.                                                                                 |
| CellMetadata | A data frame containing cell identifiers (usually barcodes) and an integer representing which batch they belong to. The column containing clustering information needs to be the first column in the CellMetadata dataframe. If clustering information is not available, users can run CORE function and add the information to the scGPS before running scGPS prediction. |
| LogMatrix    | optional input for a log matrix of the data. If no log matrix is supplied one will be created for the object.                                                                                                                                                                                                                                                              |

**Value**

This function generates an scGPS object belonging to the [SingleCellExperiment](#).

**Author(s)**

Quan Nguyen, 2018-04-06

**See Also**

[SingleCellExperiment](#)

**Examples**

```
day2 <- day2_cardio_cell_sample
t <- new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2_geneInfo, CellMetadata = day2$dat2_clusters)
colData(t); show(t); colnames(t)
```

---

new\_summarized\_scGPS\_object

*new\_summarized\_scGPS\_object*

---

**Description**

[new\\_scGPS\\_object](#) generates a scGPS object in the [SingleCellExperiment](#) class for use with the scGPS package. This object contains an expression matrix, associated metadata (cells, genes, clusters). The data are expected to be normalised counts.

**Usage**

```
new_summarized_scGPS_object(
  ExpressionMatrix = NULL,
  GeneMetadata = NULL,
  CellMetadata = NULL
)
```

Arguments

|                  |                                                                                                                                                                                                                                                                                                                                                                                        |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ExpressionMatrix | An expression dataset in matrix format. Rows should represent a transcript and its normalised counts, while columns should represent individual cells.                                                                                                                                                                                                                                 |
| GeneMetadata     | A data frame or vector containing gene identifiers used in the expression matrix. The first column should hold the cell identifiers you are using in the expression matrix. Other columns contain information about the genes, such as their corresponding ENSEMBL transcript identifiers.                                                                                             |
| CellMetadata     | A data frame containing cell identifiers (usually barcodes) and clustering information (the first column of the data frame contains clustering information). The column containing clustering information needs to be named as 'Cluster'. If clustering information is not available, users can run CORE function and add the information to the scGPS before running scGPS prediction |

Value

This function generates an scGPS object belonging to the [SingleCellExperiment](#).

Author(s)

Quan Nguyen, 2017-11-25

See Also

[SingleCellExperiment](#)

Examples

```
day2 <- day_2_cardio_cell_sample
t <-new_summarized_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
colData(t); show(t); colnames(t)
```

---

|     |            |
|-----|------------|
| PCA | <i>PCA</i> |
|-----|------------|

---

Description

Select top variable genes and perform prcomp

Usage

```
PCA(expression.matrix = NULL, ngenes = 1500, scaling = TRUE, npcs = 50)
```

Arguments

|                   |                                                                  |
|-------------------|------------------------------------------------------------------|
| expression.matrix | An expression matrix, with genes in rows                         |
| ngenes            | number of genes used for clustering calculations.                |
| scaling           | a logical of whether we want to scale the matrix                 |
| npcs              | an integer specifying the number of principal components to use. |

**Value**

a list containing PCA results and variance explained

**Examples**

```
day2 <- day_2_cardio_cell_sample
mixedpop1 <- new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2_geneInfo, CellMetadata = day2$dat2_clusters)
t <- PCA(expression.matrix=assay(mixedpop1))
```

---

|           |                                             |
|-----------|---------------------------------------------|
| plot_CORE | <i>Plot dendrogram tree for CORE result</i> |
|-----------|---------------------------------------------|

---

**Description**

This function plots CORE and all clustering results underneath

**Usage**

```
plot_CORE(original.tree, list_clusters = NULL, color_branch = NULL)
```

**Arguments**

|               |                                                                                                                                                                                                     |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| original.tree | the original dendrogram before clustering                                                                                                                                                           |
| list_clusters | a list containing clustering results for each of the                                                                                                                                                |
| color_branch  | is a vector containing user-specified colors (the number of unique colors should be equal or larger than the number of clusters). This parameter allows better selection of colors for the display. |

**Value**

a plot with clustering bars underneath the tree

**Examples**

```
day5 <- day_5_cardio_cell_sample
cellnames <- colnames(day5$dat5_counts)
cluster <- day5$dat5_clusters
cellnames <- data.frame('Cluster'=cluster, 'cellBarcodes' = cellnames)
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = cellnames)
CORE_cluster <- CORE_clustering(mixedpop2, remove_outlier = c(0))
plot_CORE(CORE_cluster$tree, CORE_cluster$Cluster)
```

---

|                   |                                                                |
|-------------------|----------------------------------------------------------------|
| plot_optimal_CORE | <i>plot one single tree with the optimal clustering result</i> |
|-------------------|----------------------------------------------------------------|

---

### Description

after an optimal cluster has been identified, users may use this function to plot the resulting dendrogram with the branch colors represent cluterling results

### Usage

```
plot_optimal_CORE(
  original_tree,
  optimal_cluster = NULL,
  shift = -100,
  values = NULL
)
```

### Arguments

|                 |                                                                                                                                                                                           |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| original_tree   | a dendrogram object                                                                                                                                                                       |
| optimal_cluster | a vector of cluster IDs for cells in the dendrogram                                                                                                                                       |
| shift           | a numer specifying the gap between the dendrogram and the colored                                                                                                                         |
| values          | a vector containing color values of the branches and the colored bar underneath the tree bar underneath the dendrogram. This parameter allows better selection of colors for the display. |

### Value

a plot with colored braches and bars for the optimal clustering result

### Author(s)

Quan Nguyen, 2017-11-25

### Examples

```
day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo, CellMetadata = day5$dat5_clusters)
CORE_cluster <- CORE_clustering(mixedpop2, remove_outlier = c(0))
key_height <- CORE_cluster$optimalClust$KeyStats$Height
optimal_res <- CORE_cluster$optimalClust$OptimalRes
optimal_index = which(key_height == optimal_res)
plot_optimal_CORE(original_tree= CORE_cluster$tree,
  optimal_cluster = unlist(CORE_cluster$Cluster[optimal_index]),
  shift = -2000)
```

---

|              |                          |
|--------------|--------------------------|
| plot_reduced | <i>plot reduced data</i> |
|--------------|--------------------------|

---

### Description

plot PCA, tSNE, and CIDR reduced datasets

### Usage

```
plot_reduced(
  reduced_dat,
  color_fac = NULL,
  dims = c(1, 2),
  dimNames = c("Dim1", "Dim2"),
  palletes = NULL,
  legend_title = "Cluster"
)
```

### Arguments

|              |                                                                                                                                                                                        |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| reduced_dat  | is a matrix with genes in rows and cells in columns                                                                                                                                    |
| color_fac    | is a vector of colors corresponding to clusters to determine colors of scattered plots                                                                                                 |
| dims         | an integer of the number of dimestions                                                                                                                                                 |
| dimNames     | a vector of the names of the dimensions                                                                                                                                                |
| palletes     | can be a customised color pallete that determine colors for density plots, if NULL it will use RColorBrewer colorRampPalette(RColorBrewer::brewer.pal(sample_num, 'Set1'))(sample_num) |
| legend_title | title of the plot's legend                                                                                                                                                             |

### Value

a matrix with the top 20 CIDR dimensions

### Examples

```
day2 <- day_2_cardio_cell_sample
mixedpop1 <- new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
#CIDR_dim <- CIDR(expression.matrix=assay(mixedpop1))
#p <- plot_reduced(CIDR_dim, color_fac = factor(colData(mixedpop1)[,1]),
#  palletes = seq_len(length(unique(colData(mixedpop1)[,1]))))
#plot(p)
tSNE_dim <- tSNE(expression.mat=assay(mixedpop1))
p2 <- plot_reduced(tSNE_dim, color_fac = factor(colData(mixedpop1)[,1]),
  palletes = seq_len(length(unique(colData(mixedpop1)[,1]))))
plot(p2)
```

---

|            |                                                                                |
|------------|--------------------------------------------------------------------------------|
| predicting | <i>Main prediction function applying the optimal ElasticNet and LDA models</i> |
|------------|--------------------------------------------------------------------------------|

---

### Description

Predict a new mixed population after training the model for a subpopulation in the first mixed population. All subpopulations in the new target mixed population will be predicted, where each targeted subpopulation will have a transition score from the original subpopulation to the new subpopulation.

### Usage

```
predicting(
  listData = NULL,
  cluster_mixedpop2 = NULL,
  mixedpop2 = NULL,
  out_idx = NULL,
  standardize = TRUE,
  LDA_run = FALSE,
  c_selectID = NULL,
  log_transform = FALSE
)
```

### Arguments

|                   |                                                                                                                                                                |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| listData          | a list object containing trained results for the selected subpopulation in the first mixed population                                                          |
| cluster_mixedpop2 | a vector of cluster assignment for mixedpop2                                                                                                                   |
| mixedpop2         | a <a href="#">SingleCellExperiment</a> object from the target mixed population of importance, e.g. differentially expressed genes that are most significant    |
| out_idx           | a number to specify index to write results into the list output. This is needed for running bootstrap.                                                         |
| standardize       | a logical of whether to standardize the data                                                                                                                   |
| LDA_run           | logical, if the LDA prediction is added to compare to ElasticNet, the LDA model needs to be trained from the training before inputting to this prediction step |
| c_selectID        | a number to specify the trained cluster used for prediction                                                                                                    |
| log_transform     | boolean whether log transform should be computed                                                                                                               |

### Value

a list with prediction results written in to the index out\_idx

### Author(s)

Quan Nguyen, 2017-11-25

**Examples**

```

c_selectID<-1
out_idx<-1
day2 <- day_2_cardio_cell_sample
mixedpop1 <-new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <-new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo, CellMetadata = day5$dat5_clusters)
genes <-training_gene_sample
genes <-genes$Merged_unique
listData <- training(genes,
  cluster_mixedpop1 = colData(mixedpop1)[, 1], mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2, c_selectID, listData =list(), out_idx=out_idx)
listData <- predicting(listData =listData, mixedpop2 = mixedpop2,
  out_idx=out_idx, cluster_mixedpop2 = colData(mixedpop2)[, 1],
  c_selectID = c_selectID)

```

PrinComp\_cpp

*Principal component analysis***Description**

This function provides significant speed gain if the input matrix is big

**Usage**

```
PrinComp_cpp(X)
```

**Arguments**

**X** an R matrix (expression matrix), rows are genes, columns are cells

**Value**

a list with three list pca lists

**Examples**

```

mat_test <-matrix(rnbinom(1000000,mu=0.01, size=10),nrow=1000)
#library(microbenchmark)
#microbenchmark(PrinComp_cpp(mat_test), prcomp(mat_test), times=3)

```

---

|            |                             |
|------------|-----------------------------|
| rand_index | <i>Calculate rand index</i> |
|------------|-----------------------------|

---

**Description**

Comparing clustering results Function for calculating randindex (adapted from the function by Steve Horvath and Luohua Jiang, UCLA, 2003)

**Usage**

```
rand_index(tab, adjust = TRUE)
```

**Arguments**

|        |                                                         |
|--------|---------------------------------------------------------|
| tab    | a table containing different clustering results in rows |
| adjust | a logical of whether to use the adjusted rand index     |

**Value**

a rand\_index value

**Author(s)**

Quan Nguyen and Michael Thompson, 2018-05-11

**Examples**

```
day5 <- day_5_cardio_cell_sample
mixedpop2 <- new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
cluster_all <- clustering(object=mixedpop2)

rand_index(table(unlist(cluster_all$list_clusters[[1]]),
cluster_all$cluster_ref))
```

---

|                           |
|---------------------------|
| rcpp_Eucl_distance_NotPar |
|---------------------------|

*Function to calculate Euclidean distance matrix without parallelisation*

---

**Description**

Function to calculate Euclidean distance matrix without parallelisation

**Usage**

```
rcpp_Eucl_distance_NotPar(mat)
```

**Arguments**

mat                      an R matrix (expression matrix), with cells in rows and genes in columns

**Value**

a distance matrix

**Examples**

```
mat_test <-matrix(rnbinom(100000,mu=0.01, size=10),nrow=1000)
#library(microbenchmark)
#microbenchmark(rcpp_Eucl_distance_NotPar(mat_test), dist(mat_test), times=3)
```

---

rcpp\_parallel\_distance

*distance matrix using C++*

---

**Description**

This function provides fast and memory efficient distance matrix calculation

**Usage**

```
rcpp_parallel_distance(mat)
```

**Arguments**

mat                      an R matrix (expression matrix), rows are genes, columns are cells

**Value**

a distance matrix

**Examples**

```
mat_test <-matrix(rnbinom(1000000,mu=0.01, size=10),nrow=10000)
#library(microbenchmark)
#microbenchmark(rcpp_parallel_distance(mat_test), dist(mat_test), times=3)
```

reformat\_LASSO

*summarise bootstrap runs for Lasso model, from n bootstraps***Description**

the training and prediction results from bootstrap were written to the object LSOLDA\_dat, the reformat\_LASSO summarises prediction for n bootstrap runs

**Usage**

```
reformat_LASSO(
  c_selectID = NULL,
  mp_selectID = NULL,
  LSOLDA_dat = NULL,
  nPredSubpop = NULL,
  Nodes_group = "#7570b3",
  nboots = 2
)
```

**Arguments**

|             |                                                                                               |
|-------------|-----------------------------------------------------------------------------------------------|
| c_selectID  | is the original cluster to be projected                                                       |
| mp_selectID | is the target mixedpop to project to                                                          |
| LSOLDA_dat  | is the results from the bootstrap                                                             |
| nPredSubpop | is the number of clusters in the target mixedpop row_cluster <-length(unique(target_cluster)) |
| Nodes_group | string representation of hexadecimal color code for node                                      |
| nboots      | is an integer for how many bootstraps are run                                                 |

**Value**

a dataframe containing information for the Lasso prediction results, each column contains prediction results for all subpopulations from each bootstrap run

**Examples**

```
c_selectID<-1
day2 <- day_2_cardio_cell_sample
mixedpop1 <-new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <-new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo, CellMetadata = day5$dat5_clusters)
genes <-training_gene_sample
genes <-genes$Merged_unique
LSOLDA_dat <- bootstrap_prediction(nboots = 2, mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2, genes=genes, c_selectID, listData =list(),
  cluster_mixedpop1 = colData(mixedpop1)[,1],
  cluster_mixedpop2=colData(mixedpop2)[,1])
reformat_LASSO(LSOLDA_dat=LSOLDA_dat,
  nPredSubpop=length(unique(colData(mixedpop2)[,1])), c_selectID = 1,
  mp_selectID =2, nboots = 2)
```

---

|            |                        |
|------------|------------------------|
| subset_cpp | <i>Subset a matrix</i> |
|------------|------------------------|

---

**Description**

Subset a matrix

**Usage**

```
subset_cpp(m1in, rowidx_in, colidx_in)
```

**Arguments**

|           |                                     |
|-----------|-------------------------------------|
| m1in      | an R matrix (expression matrix)     |
| rowidx_in | a numeric vector of rows to keep    |
| colidx_in | a numeric vector of columns to keep |

**Value**

a subsetted matrix

**Examples**

```
mat_test <- matrix(rnbinom(1000000,mu=0.01, size=10),nrow=100)
subset_mat <- subset_cpp(mat_test, rowidx_in=c(1:10), colidx_in=c(100:500))
dim(subset_mat)
```

---

|                |                                          |
|----------------|------------------------------------------|
| sub_clustering | <i>sub_clustering for selected cells</i> |
|----------------|------------------------------------------|

---

**Description**

performs 40 clustering runs or more depending on windows

**Usage**

```
sub_clustering(
  object = NULL,
  ngenes = 1500,
  windows = seq(from = 0.025, to = 1, by = 0.025),
  select_cell_index = NULL
)
```

**Arguments**

|                   |                                                                                  |
|-------------------|----------------------------------------------------------------------------------|
| object            | is a <a href="#">SingleCellExperiment</a> object from the train mixed population |
| ngenes            | number of genes used for clustering calculations.                                |
| windows           | a numeric vector specifying the ranges of each window.                           |
| select_cell_index | a vector containing indexes for cells in selected clusters to be reclustered     |

**Value**

clustering results

**Author(s)**

Quan Nguyen, 2018-01-31

**Examples**

```
day5 <- day_5_cardio_cell_sample
mixedpop2 <-new_summarized_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo, CellMetadata = day5$dat5_clusters)
test_sub_clustering <-sub_clustering(mixedpop2,
  select_cell_index = c(seq_len(100)))
```

---

summary\_accuracy

*get percent accuracy for Lasso model, from n bootstraps*

---

**Description**

The training results from training were written to

**Usage**

```
summary_accuracy(object = NULL)
```

**Arguments**

**object** is a list containing the training results from the summary\_accuracy summarise n bootstraps

**Value**

a vector of percent accuracy for the selected subpopulation

**Author(s)**

Quan Nguyen, 2017-11-25

**Examples**

```
c_selectID<-1
day2 <- day_2_cardio_cell_sample
mixedpop1 <-new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <-new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo, CellMetadata = day5$dat5_clusters)
genes <-training_gene_sample
genes <-genes$Merged_unique
LSOLDA_dat <- bootstrap_prediction(nboots = 1,mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2, genes=genes, c_selectID, listData =list(),
  cluster_mixedpop1 = colData(mixedpop1)[,1],
```

```

cluster_mixedpop2=colData(mixedpop2)[,1])
summary_accuracy(LSOLDA_dat)
summary_deviance(LSOLDA_dat)

```

---

|                  |                                                                          |
|------------------|--------------------------------------------------------------------------|
| summary_deviance | <i>get percent deviance explained for Lasso model, from n bootstraps</i> |
|------------------|--------------------------------------------------------------------------|

---

## Description

the training results from training were written to the object LSOLDA\_dat, the summary\_deviance summarises deviance explained for n bootstrap runs and also returns the best deviance matrix for plotting, as well as the best matrix with Lasso genes and coefficients

## Usage

```
summary_deviance(object = NULL)
```

## Arguments

object            is a list containing the training results from training

## Value

a list containing three elements, with a vector of percent maximum deviance explained, a dataframe containg information for the full deviance, and a dataframe containing gene names and coefficients of the best model

## Author(s)

Quan Nguyen, 2017-11-25

## Examples

```

c_selectID<-1
day2 <- day_2_cardio_cell_sample
mixedpop1 <-new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <-new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo,
  CellMetadata = day5$dat5_clusters)
genes <-training_gene_sample
genes <-genes$Merged_unique
LSOLDA_dat <- bootstrap_prediction(nboots = 2,mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2, genes=genes, c_selectID, listData =list(),
  cluster_mixedpop1 = colData(mixedpop1)[,1],
  cluster_mixedpop2=colData(mixedpop2)[,1])
summary_deviance(LSOLDA_dat)

```

---

summary\_prediction\_lasso

*get percent deviance explained for Lasso model, from n bootstraps*


---

### Description

the training results from training were written to the object LSOLDA\_dat, the summary\_prediction summarises prediction for n bootstrap runs

### Usage

```
summary_prediction_lasso(LSOLDA_dat = NULL, nPredSubpop = NULL)
```

### Arguments

LSOLDA\_dat        is a list containing the training results from training  
nPredSubpop       is the number of subpopulations in the target mixed population

### Value

a dataframe containing information for the Lasso prediction results, each column contains prediction results for all subpopulations from each bootstrap run

### Examples

```
c_selectID<-1
day2 <- day_2_cardio_cell_sample
mixedpop1 <-new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2_geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <-new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
genes <-training_gene_sample
genes <-genes$Merged_unique
LSOLDA_dat <- bootstrap_prediction(nboots = 1,mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2, genes=genes, c_selectID, listData =list(),
  cluster_mixedpop1 = colData(mixedpop1)[,1],
  cluster_mixedpop2=colData(mixedpop2)[,1])
summary_prediction_lasso(LSOLDA_dat=LSOLDA_dat, nPredSubpop=4)
```

---

summary\_prediction\_lda

*get percent deviance explained for LDA model, from n bootstraps*


---

### Description

the training results from training were written to the object LSOLDA\_dat, the summary\_prediction summarises prediction explained for n bootstrap runs and also returns the best deviance matrix for plotting, as well as the best matrix with Lasso genes and coefficients

**Usage**

```
summary_prediction_lda(LSOLDA_dat = NULL, nPredSubpop = NULL)
```

**Arguments**

LSOLDA\_dat        is a list containing the training results from training  
 nPredSubpop      is the number of subpopulations in the target mixed population

**Value**

a dataframe containing information for the LDA prediction results, each column contains prediction results for all subpopulations from each bootstrap run

**Examples**

```
c_selectID<-1
day2 <- day_2_cardio_cell_sample
mixedpop1 <-new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2_geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <-new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5_geneInfo, CellMetadata = day5$dat5_clusters)
genes <-training_gene_sample
genes <-genes$Merged_unique
LSOLDA_dat <- bootstrap_prediction(nboots = 1,mixedpop1 = mixedpop1,
  mixedpop2 = mixedpop2, genes=genes, c_selectID, listData =list(),
  cluster_mixedpop1 = colData(mixedpop1)[,1],
  cluster_mixedpop2=colData(mixedpop2)[,1])
summary_prediction_lda(LSOLDA_dat=LSOLDA_dat, nPredSubpop=4)
```

---

|         |                                  |
|---------|----------------------------------|
| top_var | <i>select top variable genes</i> |
|---------|----------------------------------|

---

**Description**

subset a matrix by top variable genes

**Usage**

```
top_var(expression.matrix = NULL, ngenes = 1500)
```

**Arguments**

expression.matrix  
                   is a matrix with genes in rows and cells in columns  
 ngenes            number of genes used for clustering calculations.

**Value**

a subsetting expression matrix with the top n most variable genes

Examples

```
day2 <- day_2_cardio_cell_sample
mixedpop1 <-new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
SortedExprsMat <-top_var(expression.matrix=assay(mixedpop1))
```

---

|        |                           |
|--------|---------------------------|
| tp_cpp | <i>Transpose a matrix</i> |
|--------|---------------------------|

---

Description

Transpose a matrix

Usage

```
tp_cpp(X)
```

Arguments

X                      an R matrix (expression matrix)

Value

a transposed matrix

Examples

```
mat_test <-matrix(rnbinom(1000000,mu=0.01, size=10),nrow=100)
tp_mat <- tp_cpp(mat_test)
```

---

|          |                                                                                                   |
|----------|---------------------------------------------------------------------------------------------------|
| training | <i>Main model training function for finding the best model that characterises a subpopulation</i> |
|----------|---------------------------------------------------------------------------------------------------|

---

Description

Training a haft of all cells to find optimal ElasticNet and LDA models to predict a subpopulation

Usage

```
training(
  genes = NULL,
  cluster_mixedpop1 = NULL,
  mixedpop1 = NULL,
  mixedpop2 = NULL,
  c_selectID = NULL,
  listData = list(),
  out_idx = 1,
  standardize = TRUE,
  trainset_ratio = 0.5,
  LDA_run = FALSE,
  log_transform = FALSE
)
```

**Arguments**

|                   |                                                                                                                                                                                                                                                                                                                                 |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| genes             | a vector of gene names (for ElasticNet shrinkage); gene symbols must be in the same format with gene names in subpop2. Note that genes are listed by the order of importance, e.g. differentially expressed genes that are most significant, so that if the gene list contains too many genes, only the top 500 genes are used. |
| cluster_mixedpop1 | a vector of cluster assignment in mixedpop1                                                                                                                                                                                                                                                                                     |
| mixedpop1         | is a <a href="#">SingleCellExperiment</a> object from the train mixed population                                                                                                                                                                                                                                                |
| mixedpop2         | is a <a href="#">SingleCellExperiment</a> object from the target mixed population                                                                                                                                                                                                                                               |
| c_selectID        | a selected number to specify which subpopulation to be used for training                                                                                                                                                                                                                                                        |
| listData          | list to store output in                                                                                                                                                                                                                                                                                                         |
| out_idx           | a number to specify index to write results into the list output. This is needed for running bootstrap.                                                                                                                                                                                                                          |
| standardize       | a logical value specifying whether or not to standardize the train matrix                                                                                                                                                                                                                                                       |
| trainset_ratio    | a number specifying the proportion of cells to be part of the training subpopulation                                                                                                                                                                                                                                            |
| LDA_run           | logical, if the LDA run is added to compare to ElasticNet                                                                                                                                                                                                                                                                       |
| log_transform     | boolean whether log transform should be computed                                                                                                                                                                                                                                                                                |

**Value**

a list with prediction results written in to the indexed out\_idx

**Author(s)**

Quan Nguyen, 2017-11-25

**Examples**

```

c_selectID<-1
out_idx<-1
day2 <- day_2_cardio_cell_sample
mixedpop1 <-new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2geneInfo, CellMetadata = day2$dat2_clusters)
day5 <- day_5_cardio_cell_sample
mixedpop2 <-new_scGPS_object(ExpressionMatrix = day5$dat5_counts,
  GeneMetadata = day5$dat5geneInfo, CellMetadata = day5$dat5_clusters)
genes <-training_gene_sample
genes <-genes$Merged_unique
listData <- training(genes,
  cluster_mixedpop1 = colData(mixedpop1)[, 1],
  mixedpop1 = mixedpop1, mixedpop2 = mixedpop2, c_selectID,
  listData =list(), out_idx=out_idx, trainset_ratio = 0.5)
names(listData)
listData$Accuracy

```

---

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| training_gene_sample | <i>Input gene list for training scGPS, e.g. differentially expressed genes</i> |
|----------------------|--------------------------------------------------------------------------------|

---

**Description**

Genes should be ordered from most to least important genes (1 row per gene)

**Usage**

```
training_gene_sample
```

**Format**

a list instance, containing a count matrix and a vector with clustering information.

**Value**

a vector containing gene symbols

**Author(s)**

Quan Nguyen, 2017-11-25

**Source**

Dr Joseph Powell's laboratory, IMB, UQ

---

|      |             |
|------|-------------|
| tSNE | <i>tSNE</i> |
|------|-------------|

---

**Description**

calculate tSNE from top variable genes

**Usage**

```
tSNE(
  expression.mat = NULL,
  topgenes = 1500,
  scale = TRUE,
  thet = 0.5,
  perp = 30
)
```

**Arguments**

|                |                                                                                                                        |
|----------------|------------------------------------------------------------------------------------------------------------------------|
| expression.mat | An expression matrix, with genes in rows                                                                               |
| topgenes       | number of genes used for clustering calculations.                                                                      |
| scale          | a logical of whether we want to scale the matrix                                                                       |
| thet           | numeric; Speed/accuracy trade-off (increase for less accuracy)                                                         |
| perp           | numeric; Perplexity parameter (should not be bigger than 3 * perplexity < nrow(X) - 1, see details for interpretation) |

**Value**

a tSNE reduced matrix containing three tSNE dimensions

**Examples**

```
day2 <- day_2_cardio_cell_sample
mixedpop1 <- new_scGPS_object(ExpressionMatrix = day2$dat2_counts,
  GeneMetadata = day2$dat2_geneInfo, CellMetadata = day2$dat2_clusters)
t <- tSNE(expression.mat = assay(mixedpop1))
```

---

|         |                           |
|---------|---------------------------|
| var_cpp | <i>Calculate variance</i> |
|---------|---------------------------|

---

**Description**

Calculate variance

**Usage**

```
var_cpp(x, bias = TRUE)
```

**Arguments**

|      |                              |
|------|------------------------------|
| x    | a vector of gene expression. |
| bias | degree of freedom            |

**Value**

a variance value

**Examples**

```
var_cpp(seq_len(10^6))
```

# Index

## \* datasets

- day\_2\_cardio\_cell\_sample, [13](#)
- day\_5\_cardio\_cell\_sample, [14](#)
- training\_gene\_sample, [36](#)

add\_import, [3](#)

annotate\_clusters, [3](#)

bootstrap\_parallel, [4](#), [6](#)

bootstrap\_prediction, [5](#)

calcDist, [6](#)

calcDistArma, [7](#)

clustering, [8](#)

clustering\_bagging, [9](#)

CORE\_bagging, [10](#)

CORE\_clustering, [11](#)

CORE\_subcluster, [12](#)

day\_2\_cardio\_cell\_sample, [13](#)

day\_5\_cardio\_cell\_sample, [14](#)

distvec, [14](#)

find\_markers, [15](#)

find\_optimal\_stability, [16](#)

find\_stability, [17](#)

mean\_cpp, [18](#)

new\_scGPS\_object, [18](#), [18](#), [19](#)

new\_summarized\_scGPS\_object, [19](#)

PCA, [20](#)

plot\_CORE, [21](#)

plot\_optimal\_CORE, [22](#)

plot\_reduced, [23](#)

predicting, [24](#)

PrinComp\_cpp, [25](#)

rand\_index, [26](#)

rcpp\_Eucl\_distance\_NotPar, [26](#)

rcpp\_parallel\_distance, [27](#)

reformat\_LASSO, [28](#)

SingleCellExperiment, [4](#), [5](#), [8–12](#), [18–20](#),  
[24](#), [29](#), [35](#)

sub\_clustering, [29](#)

subset\_cpp, [29](#)

summary\_accuracy, [30](#)

summary\_deviance, [31](#)

summary\_prediction\_lasso, [32](#)

summary\_prediction\_lda, [32](#)

top\_var, [33](#)

tp\_cpp, [34](#)

training, [34](#)

training\_gene\_sample, [36](#)

tSNE, [36](#)

var\_cpp, [37](#)