Package 'SJD'

February 16, 2024

Type Package
Title Structured Joint Decomposition (SJD)
Version 1.0
Date 2021-11-16
Author c(person(``Huan", ``Chen", email = ``hzchenhuan@gmail.com", role = c(``aut", ``cre")), person(``Guangyan", ``Li", role = c(``aut")), person(``Jinrui", ``Liu", role = c(``aut")), person(``Shreyash", ``Sonthalia", role = c(``aut")), person(``Carlo", ``Colantuoni", role = c(``aut")))
Maintainer Huan Chen <hzchenhuan@gmail.com></hzchenhuan@gmail.com>
Imports tidyverse, RSpectra, fastICA, NMF, Biobase, MASS, RMTstat, dplyr, biomaRt, plotrix, ggplot2, stringr, gridExtra, googledrive
Description This is a R Package for Visualizing Biologically StructuredGene Expression Matrix Environment Based on Low Rank Models, (1) separately, (2) concatenately, (3) jointly, and (4) sequentially.
Depends R (>= $3.5.0$)
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 7.2.3
Suggests rmarkdown, knitr
VignetteBuilder knitr
<pre>URL https://github.com/CHuanSite/SJD</pre>
R topics documented:
addSJDtoSeurat

2

compNameAssign
compNameAssignSep
concatICA
concatNMF
concatPCA
configuration_setting_generation
datasetNameExtractor
filterNAValue
frameToMatrix
geneNameAssign
geneNameExtractor
geneScreen
getMatch
groupNameExtractor
jointICA
jointNMF
jointPCA
linkedPCA
marchenko_pastur_quantile
<u> </u>
mtx
NeuroGenesis4
NeuroGenesis4.afterWrap
NeuroGenesis4.info
normalizeData
Procrustes
projectiLCA
projectNMF
pveMultiple
pveSep
rebalanceData
rotate_component
sampleNameAssign
sampleNameAssignProj
sampleNameAssignSep
sampleNameExtractor
scoreNameAssign
scoreNameAssignProj
scoreNameAssignSep
score_generation
ScreePlot_LCvsPC
sepICA
sepNMF
sepPCA
simulated_data_generation
SJDScorePlotter
sjdWrap
sjdWrapProjection
twoStageiLCA
twoStageiLCA.rank
twoStageLCA
twoStageLCA.rank
umap_tsne_onLC

addSJDtoSeurat 3

Index 51

addSJDtoSeurat Add SJD loadings to a Seurat object

Description

Add SJD loadings to a Seurat object

Usage

```
addSJDtoSeurat(Seurat.obj, SJDoutput, Dataset, SJDloading, SJDmethod = NA)
```

Arguments

Seurat.obj A Seurat Object.

SJDoutput A SJD object of decomposition results.

Dataset The name of the Dataset of interests in the SJD score_list component

SJDloading The name of the SJDloading of interests in the SJD score_list component under

Dataset.

SJDmethod The name of the SJD method used for the decomposition, e.g. "twoStageLCA".

If NA, will use "SJD"

assemble.byComponent Assemble Files Based on Component

Description

Assemble individual same-component figures from multiple datasets analyzed and plotted by SJD for cross comparison

Usage

```
assemble.byComponent(SJDScorePlotter.obj, component, SJD_algorithm, group = NA)
```

Arguments

SJDScorePlotter.obj

A list outputted by the SJDScorePlotter function

component numer/order of component of interest to print out, i.e. 1 or c(1,2)

SJD_algorithm SJD_algorithm name of SJD algorithm, i.e. concatICA

group name of the weights group from the SJD_algorithm output, i.e 'Shared.All.13'

Value

A list of images filtered by component

4 assemble.byDataset

Examples

```
library(ggplot2)
data(NeuroGenesis4.afterWrap)
data(NeuroGenesis4.info)
SampleMetaNamesTable = data.frame(
   row.names = names(NeuroGenesis4),
   Type = c('Yaxis','Yaxis','2Dscatter','2Dscatter'),
   XaxisColumn = c("X", "DAYx", "tSNE_1", "tsne1:ch1"),
   YaxisColumn = c("PJDscores", "PJDscores", "tSNE_2", "tsne2:ch1"),
   COLaxisColumn = c("color","colorBYlabelsX","PJDscores","PJDscores"),
   PCHColumn = c("","","","")
)
grp = list(
Shared.All.4 = c(1 : 4),
Shared.bulk.2 = c(1, 2),
Shared.sc.2 = c(3, 4),
Hs.Meisnr.1 = c(1),
Hs.AZ.1 = c(2),
Gesch.1 = c(3),
Telley.1 = c(4)
dims = c(2, 2, 2, 2, 2, 2, 2)
lbb = "NeuroGenesis4.p2"
twoStageLCA.out = twoStageLCA(dataset = NeuroGenesis4.afterWrap, group = grp, comp_num = dims)
SJDScorePlotter.obj = SJDScorePlotter(
    SJDalg = "twoStageLCA",
    scores = twoStageLCA.out$score_list,
    1bb = 1bb,
    info = NeuroGenesis4.info,
    SampleMetaNamesTable = SampleMetaNamesTable
)
assemble.byComponent.obj = assemble.byComponent(
SJDScorePlotter.obj = SJDScorePlotter.obj,
component = c(1, 2),
SJD_algorithm = "twoStageLCA",
group = 'Shared.All.4')
```

assemble.byDataset

Assemble files based on Dataset

Description

Assemble individual figures for a single dataset analyzed and plotted by SJD

assemble.byDataset 5

Usage

```
assemble.byDataset(
   SJDScorePlotter.obj,
   dataset_name,
   SJD_algorithm,
   group = NA
)
```

Arguments

group

```
SJDScorePlotter.obj
A list outputted by the SJDScorePlotter function

dataset_name dataset/study analyzed by SJD

SJD_algorithm SJD_algorithm name of SJD algorithm, i.e. concatICA
```

group name of the weights group from the SJD_algorithm output, i.e 'Shared.All.13'

Value

a list of images filtered by dataset

```
library(ggplot2)
data(NeuroGenesis4.afterWrap)
data(NeuroGenesis4.info)
SampleMetaNamesTable = data.frame(
   row.names = names(NeuroGenesis4.afterWrap),
   Type = c('Yaxis','Yaxis','2Dscatter','2Dscatter'),
   XaxisColumn = c("X","DAYx","tSNE_1","tsne1:ch1"),
   YaxisColumn = c("PJDscores", "PJDscores", "tSNE_2", "tsne2:ch1"),
   COLaxisColumn = c("color", "colorBYlabelsX", "PJDscores", "PJDscores"),
   PCHColumn = c("","","","")
)
grp = list(
Shared.All.4 = c(1 : 4),
Shared.bulk.2 = c(1, 2),
Shared.sc.2 = c(3, 4),
Hs.Meisnr.1 = c(1),
Hs.AZ.1 = c(2),
Gesch.1 = c(3),
Telley.1 = c(4)
dims = c(2, 2, 2, 2, 2, 2, 2)
lbb = "NeuroGenesis4.p2"
twoStageLCA.out = twoStageLCA(dataset = NeuroGenesis4.afterWrap, group = grp, comp_num = dims)
SJDScorePlotter.obj = SJDScorePlotter(
    SJDalg = "twoStageLCA",
    scores = twoStageLCA.out$score_list,
    1bb = 1bb,
```

6 BEMA

```
info = NeuroGenesis4.info,
    SampleMetaNamesTable = SampleMetaNamesTable
)

assemble.byDataset.obj = assemble.byDataset(
SJDScorePlotter.obj = SJDScorePlotter.obj,
dataset_name = "Meissner.inVitro.bulk.Hs",
SJD_algorithm = "twoStageLCA",
group = NA)
```

balanceData

Balance Data

Description

Balance Data for Concatenate and Joint Analysis

Usage

```
balanceData(dataset)
```

Arguments

dataset

A list of data sets to be used

Value

A list of rebalanced data sets

Examples

```
dataset = list(matrix(c(1 : 8), nrow = 2), matrix(1 : 6, nrow = 2)) balanceData(dataset)
```

BEMA

BEMA for the standard spiked covariance model

Description

Apply BEMA algorithm for spiked covariance proposed in the paper "Estimation of the number of spiked eigenvalues in a covariance matrix by bulk eigenvalue matching analysis""

Usage

```
BEMA(eigenvalue, p, n, alpha = 0.2, beta = 0.1)
```

compNameAssign 7

Arguments

eigenvalue a list of eigenvalues to choose from

p dimension of the features

n number of samples

alpha a tuning parameter in the analysis, a default value is set to 0.2

beta a tuning parameter on computing quantile in Tracy-Widom, a default value is

set to be 0.1

Value

The total number of spikes extracted, K

Examples

```
x = matrix(rnorm(1000, 100), nrow = 1000)

eigen_x = svd(x)

eigen_out = list(eigenvalue = eigen_x$d^2 / 100, p = 1000, n = 100)

BEMA(eigen_out$eigenvalue, p = 1000, n = 100)
```

compNameAssign Components Name Assignment for concatenate, joint and

twoStageLCA analysis

Description

Assign name to components in the concatenate, joint and twoStageLCA methods

Usage

```
compNameAssign(linked_component_list, group_name)
```

Arguments

 $linked_component_list$

list of components extracted

group_name A vector of names for the datasets

Value

renamed list of linked_component_list

```
linked_component_list = list(matrix(c(1:4), nrow = 2), matrix(c(1:4), nrow = 2)) group_name = c("x", "y") compNameAssign(linked_component_list, group_name)
```

8 concatICA

compNameAssignSep

Components Name Assignment for Seperate Analysis

Description

Assign name to components in the seperate analysis, sepPCA, sepICA, sepNMF

Usage

```
compNameAssignSep(linked_component_list, dataset_name)
```

Arguments

Value

renamed list of linked_component_list

Examples

```
linked_component_list = list(matrix(c(1:4), nrow = 2), matrix(c(1:4), nrow = 2)) dataset_name = c("x", "y") compNameAssignSep(linked_component_list, dataset_name)
```

concatICA

Concatenated decomposition with Independent Component Analysis

Description

Concatenated decomposition of several linked matrices with Independent Component Analysis (ICA)

Usage

```
concatICA(
  dataset,
  group,
  comp_num,
  weighting = NULL,
  proj_dataset = NULL,
  proj_group = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

concatNMF 9

Arguments

dataset A list of dataset to be analyzed A list of grouping of the datasets, indicating the relationship between datasets group A vector indicates the dimension of each compoent comp_num Weighting of each dataset, initialized to be NULL weighting proj_dataset The dataset(s) to be projected on. proj_group A listed of boolean combinations indicating which groupings should be used for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should match the length of the parameter group. enable_normalization

An argument to decide whether to use normalization or not, default is TRUE

column_sum_normalization

An argument to decide whether to use column sum normalization or not, default

it FALSE

A vector of probabilies for genes to be chosen screen_prob

Value

A list contains the component and the score of each dataset on every component after concatPCA algorithm

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2,2)
proj_dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
proj_group = list(c(TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE))
res_concatICA = concatICA(
dataset,
group,
comp_num,
proj_dataset = proj_dataset,
proj_group = proj_group)
```

concatNMF

Concatenated decomposition with Nonnegative Matrix Factorization

Description

Concatenated decomposition of several matrices with Nonnegative Matrix Factorization (NMF)

10 concatNMF

Usage

```
concatNMF(
  dataset,
  group,
  comp_num,
  weighting = NULL,
  perturbation = 1e-04,
  proj_dataset = NULL,
  proj_group = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset A list of dataset to be analyzed

group A list of grouping of the datasets, indicating the relationship between datasets

comp_num A vector indicates the dimension of each compoent weighting Weighting of each dataset, initialized to be NULL perturbation the perturbation of the 0 element in the analysis

proj_dataset The dataset(s) to be projected on.

proj_group A listed of boolean combinations indicating which groupings should be used

for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should

match the length of the parameter group.

enable_normalization

An argument to decide whether to use normalizaiton or not, default is TRUE

column_sum_normalization

An argument to decide whether to use column sum normalization or not, default

it FALSE

screen_prob A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after concatNMF algorithm

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2,2)
proj_dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
proj_group = list(c(TRUE, TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE))
res_concatNMF = concatNMF(
dataset,
group,
```

concatPCA 11

```
comp_num,
proj_dataset = proj_dataset,
proj_group = proj_group)
```

concatPCA

Concatenated Decomposition with Principal Component Analysis

Description

Concatenated decomposition of several matrices with Principal Component Analysis (PCA)

Usage

```
concatPCA(
  dataset,
  group,
  comp_num,
  weighting = NULL,
  proj_dataset = NULL,
  proj_group = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset A list of dataset to be analyzed

group A list of grouping of the datasets, indicating the relationship between datasets

comp_num A vector indicates the dimension of each compoent weighting Weighting of each dataset, initialized to be NULL

 $proj_dataset$ The dataset(s) to be projected on.

proj_group A listed of boolean combinations indicating which groupings should be used

for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should

match the length of the parameter group.

enable_normalization

An argument to decide whether to use normalizaiton or not, default is TRUE

column_sum_normalization

An argument to decide whether to use column sum normalization or not, default

it FALSE

screen_prob A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after concatPCA algorithm

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2)
proj_dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
proj_group = list(c(TRUE, TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, FALSE))
res_concatPCA = concatPCA(
dataset,
group,
comp_num,
weighting = NULL,
proj_dataset = proj_dataset,
proj_group = proj_group)
```

configuration_setting_generation

Configuration for simulated data generation

Description

generate the configuration for the data

Usage

```
configuration_setting_generation(
  featureNum = 50,
  DataNum = c(100, 100, 100, 100),
  commonlySharedNum = 2,
  partiallySharedNum = c(2, 2, 2, 2),
  individualSharedNum = c(2, 2, 2, 2),
  noiseVariance = c(1, 1, 1, 1)
)
```

Arguments

```
featureNum number of features

DataNum number of data

commonlySharedNum
number of common component

partiallySharedNum
number of partial shared component

individualSharedNum
number of individual component

noiseVariance variance of noise
```

datasetNameExtractor 13

Examples

```
configuration_setting_generation()
```

Description

Extract data.frame name from a list of data.frames, if no name is given in the list, it will be renamed as "dataset_no.index"

Usage

```
datasetNameExtractor(dataset)
```

Arguments

dataset

A list of data sets

Value

A vector of strings of dataset name

Examples

```
dataset = list(
x = matrix(c(1 : 4), nrow = 2),
y = matrix(c(1 : 4), nrow = 2))
datasetNameExtractor(dataset)
```

filterNAValue

Filter NA Value

Description

Assign NA values to scores not in the group for each data set

Usage

```
filterNAValue(list_score, dataset, group)
```

Arguments

list_score A list of scores extracted in the analysis dataset A list of datasets used in the analysis

group A list of group assignments

14 frameToMatrix

Value

A list of scores by assigning NA to scores not in groups

Examples

```
x = list(matrix(c(1 : 4), nrow = 2), matrix(c(1 : 4), nrow = 2))
y = list(matrix(c(1 : 4), nrow = 2), matrix(c(1 : 4), nrow = 2))
list_score = list(x, y)
dataset = c(1, 2)
group = list(c(1), c(2))
filterNAValue(list_score, dataset, group)
```

frameToMatrix

Transform data.frame to matrix

Description

Function to transform a data frame into a matrix

Usage

```
frameToMatrix(dataset)
```

Arguments

dataset

A list of data sets

Value

A list of matrix transformed from the list of data frames

```
dataset = list(
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50)
)
frameToMatrix(dataset)
```

geneNameAssign 15

geneNameAssign

Gene name assignment

Description

Assign gene names to component derived in the analysis

Usage

```
geneNameAssign(linked_component_list, gene_name)
```

Arguments

```
linked_component_list
A list of extracted components
gene_name A vector of strings for gene names
```

Value

A list of components with gene name added

Examples

```
x = matrix(c(1 : 4), nrow = 2)
y = matrix(c(1 : 4), nrow = 2)
component_list = list(x, y)
gene_name = c("gene.1", "gene.2")
geneNameAssign(component_list, gene_name)
```

geneNameExtractor

Gene Name Extractor

Description

Extract gene names from the input dataset

Usage

```
geneNameExtractor(dataset)
```

Arguments

dataset

A list of data sets to be analyzed

Value

```
a vector of strings of gene name
```

16 geneScreen

Examples

```
x = matrix(c(1 : 4), nrow = 2)
rownames(x) = c("row1", "row2")
y = matrix(c(1 : 4), nrow = 2)
rownames(y) = c("row1", "row2")
dataset = list(x, y)
geneNameExtractor(dataset)
```

geneScreen

Gene Screen

Description

Screen genes based on their standard deviation

Usage

```
geneScreen(dataset, screen_prob)
```

Arguments

dataset A list of dataset to be analyzed

screen_prob A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after jointPCA algorithm

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
screen_prob = c(0.2, 0.2, 0.2, 0.2)
screened_dataset = geneScreen(dataset, screen_prob)
```

getMatch 17

getMatch	Species matching function	

Description

Use biomaRt to retrieve additional gene identifiers for a vector of gene identifiers in one species (e.g. 'human'), or gene identifiers for orthologous genes in a second species (e.g. 'mouse')

Usage

```
getMatch(
  genes,
  inSpecies,
  inType,
  newSpecies,
  useNewestVersion = FALSE,
  moreAttrIn = NA,
  moreAttrNew = NA
)
```

Arguments

genes character vector of gene identifiers in either ensembl gene identifier or gene

symbol format

inSpecies a character vector indicating the species from which 'genes' come; must be one

of: "human", "mouse", "roundworm", "fruitfly", "zebrafish", "chicken", "rat", "guinea pig", "golden hamster", "rabbit", "pig", "sheep", "cow", "dog", "cat",

"macaque", "bonobo", "chimpanzee"

inType a single character value indicating the type of gene identifiers being passed in

the "genes" argument; must be one of: "symbol", "ensembl"

newSpecies a character vector indicating the species for which orthologous gene identi-

fiers are desired; must be one of: "human", "mouse", "roundworm", "fruitfly", "zebrafish", "chicken", "rat", "guinea pig", "golden hamster", "rabbit", "pig",

"sheep", "cow", "dog", "cat", "macaque", "bonobo", "chimpanzee"

useNewestVersion

logical indicating if the function should attempt to use the latest version of ensembl, or to use the Aug 2020 archive version that is more stable. default is

FALSE.

moreAttrIn character vector of other gene attributes that you want returned for the input

species from biomaRt - to add this argument you must know the names of the fields in the species-specific biomaRt that you are requesting. default is NA.

moreAttrNew character vector of other gene attributes that you want returned for the output

species from biomaRt - to add this argument you must know the names of the fields in the species-specific biomaRt that you are requesting. default is NA.

Value

A matrix whose first column contains the exact gene identifiers (and in the same order) that were sent to the function in the "genes" argument, followed by 3 columns of gene identifiers that were retrieved from biomaRt from both the "inSpecies" and the "newSpecies" (for each species, these 3 identifiers are: gene symbol, ensembl gene ID, and text description).

18 jointICA

Examples

```
data(NeuroGenesis4)
out = getMatch(
rownames(NeuroGenesis4$Meissner.inVitro.bulk.Hs),
inSpecies = 'human',
inType = 'symbol',
newSpecies = 'mouse')
```

groupNameExtractor

Group Name Extractor

Description

Extract group name from groups, if no name is given in the input list, the output vector will be represented as "component_No.index"

Usage

```
groupNameExtractor(group)
```

Arguments

group

A list of group assignments for the data sets with group name on it

Value

A vector of strings of group name

Examples

```
dataset = list(
x = c(1, 2, 3),
y = c(1, 2, 4))
```

jointICA

Joint decomposition with Independent Component Analysis

Description

Joint decomposition of several linked matrices with Independent Component Analysis (ICA)

jointICA 19

Usage

```
jointICA(
  dataset,
  group,
  comp_num,
  weighting = NULL,
  max_ite = 100,
  max_err = 1e-04,
  proj_dataset = NULL,
  proj_group = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset A list of dataset to be analyzed

group A list of grouping of the datasets, indicating the relationship between datasets

comp_num A vector indicates the dimension of each compoent weighting Weighting of each dataset, initialized to be NULL

max_ite The maximum number of iterations for the jointPCA algorithms to run, default

value is set to 100

max_err The maximum error of loss between two iterations, or the program will termi-

nate and return, default value is set to be 0.0001

proj_dataset The dataset(s) to be projected on.

proj_group A listed of boolean combinations indicating which groupings should be used

for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should

match the length of the parameter group.

enable_normalization

An argument to decide whether to use normalization or not, default is TRUE

 $\verb"column_sum_normalization"$

An argument to decide whether to use column sum normalization or not, default

it FALSE

screen_prob A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after jointPCA algorithm

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2,2)
```

20 jointNMF

```
proj_dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
proj_group = list(c(TRUE, TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, FALSE))
res_jointICA = jointICA(
dataset,
group,
comp_num,
proj_dataset = proj_dataset,
proj_group = proj_group)
```

jointNMF

Joint Decomposition with Nonnegative Matrix Factorization

Description

Joint decomposition of several linked matrices with Nonnegative Matrix Factorization (NMF) It is based on the MSE loss, proposed by Lee, Daniel D., and H. Sebastian Seung. "Learning the parts of objects by non-negative matrix factorization." Nature 401.6755 (1999): 788-791.

Usage

```
jointNMF(
  dataset,
  group,
  comp_num,
  weighting = NULL,
  max_ite = 1000,
  max_err = 1e-04,
  proj_dataset = NULL,
  proj_group = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset	A list of dataset to be analyzed
group	A list of grouping of the datasets, indicating the relationship between datasets
comp_num	A vector indicates the dimension of each compoent
weighting	Weighting of each dataset, initialized to be NULL
max_ite	The maximum number of iterations for the jointNMF algorithms to run, default value is set to 100
max_err	The maximum error of loss between two iterations, or the program will terminate and return, default value is set to be 0.0001
proj_dataset	The dataset to be projected on.
proj_group	A boolean combination indicating which groupings should be used for the projected dataset.

jointPCA 21

```
enable_normalization
```

 $\label{thm:column_sum} An \ argument \ to \ decide \ whether \ to \ use \ normalization \ or \ not, \ default \ is \ TRUE \ column_sum_normalization$

An argument to decide whether to use column sum normalization or not, default it FALSE

screen_prob

A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after jointNMF algorithm

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2,2)
proj_dataset = matrix(runif(5000, 1, 2), nrow = 100, ncol = 50)
proj_group = c(TRUE, TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, FALSE)
res_jointNMF = jointNMF(
dataset,
group,
comp_num,
proj_dataset = proj_dataset,
proj_group = proj_group)
```

jointPCA

Joint Decomposition with Principal Component Analysis

Description

Joint decomposition of several linked matrices with Principal Component Analysis (PCA)

Usage

```
jointPCA(
  dataset,
  group,
  comp_num,
  weighting = NULL,
  max_ite = 100,
  max_err = 1e-04,
  proj_dataset = NULL,
  proj_group = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

22 jointPCA

Arguments

dataset	A list of dataset to be analyzed
group	A list of grouping of the datasets, indicating the relationship between datasets
comp_num	A vector indicates the dimension of each compoent
weighting	Weighting of each dataset, initialized to be NULL
max_ite	The maximum number of iterations for the jointPCA algorithms to run, default value is set to 100
max_err	The maximum error of loss between two iterations, or the program will terminate and return, default value is set to be 0.001
proj_dataset	The dataset(s) to be projected on.
proj_group	A listed of boolean combinations indicating which groupings should be used for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should match the length of the parameter group.
enable_normaliz	zation
	An argument to decide whether to use normalization or not, default is TRUE
column_sum_norm	nalization
	An argument to decide whether to use column sum normalization or not, default it FALSE
screen_prob	A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after jointPCA algorithm

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2,2)
proj_dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
proj_group = list(c(TRUE, TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, FALSE))
res_jointPCA = jointPCA(
dataset,
group,
comp_num,
proj_dataset = proj_dataset,
proj_group = proj_group)
```

linkedPCA 23

|--|

Description

Functions to implement the linked component analysis

Usage

```
linkedPCA(dataset, cov_list, eigen_space, group, comp_num)
```

Arguments

dataset A list of dataset to be analyzed cov_list A list of covariance of the datasets eigen_space A matrix of the space of the signal

group A list of grouping of the datasets, indicating the relationship between datasets

comp_num A vector indicates the dimension of each compoent

Value

A list of component

```
marchenko_pastur_quantile
```

Quantile for Marchenko Pastur Distribution

Description

Compute the q quantile for Marchenko Pastur Distribution with n and p

Usage

```
marchenko_pastur_quantile(q, n, p, step_size = 1e-04)
```

Arguments

q quantile to be chosen

n number of samples to be considered
p dimension of the covariance matrix
step_size step length to do numerical integration

Value

Quantile location

```
out = marchenko_pastur_quantile(0.2, 50, 500)
```

24 NeuroGenesis4

mtx

mtx sequence data

Description

Data from sequecing, it contains two datasets: HS.Nico and Nm.Nico HS.Nico is from human and Mm.Nico is from mouse

Usage

```
data(mtx)
```

Format

A list containing two dataframes

Examples

```
data(mtx)
head(mtx$HS.Nico)
head(mtx$Mm.Nico)
```

NeuroGenesis4

NeuroGenesis4

Description

A list of 4 neuro data sets sampled, containing: Meissner.inVitro.bulk.Hs, LIBD.AZ.inVitro.bulk.Hs, Geschwind.inVivo.sc.Hs, Jabaudon.inVivo.sc.Mm

Usage

```
data(NeuroGenesis4)
```

Format

A list containing 4 dataframes

```
data(NeuroGenesis4)
head(NeuroGenesis4$Meissner.inVitro.bulk.Hs)
head(NeuroGenesis4$LIBD.AZ.inVitro.bulk.Hs)
head(NeuroGenesis4$Geschwind.inVivo.sc.Hs)
head(NeuroGenesis4$Jabaudon.inVivo.sc.Mm)
```

NeuroGenesis4.afterWrap

NeuroGenesis4.afterWrap

Description

A list of 4 neuro data sets sampled after wrapper, containing: Meissner.inVitro.bulk.Hs, LIBD.AZ.inVitro.bulk.Hs, Geschwind.inVivo.sc.Hs, Jabaudon.inVivo.sc.Mm

Usage

data(NeuroGenesis4.afterWrap)

Format

A list containing 4 dataframes

Examples

```
data(NeuroGenesis4.afterWrap)
head(NeuroGenesis4.afterWrap$Meissner.inVitro.bulk.Hs)
head(NeuroGenesis4.afterWrap$LIBD.AZ.inVitro.bulk.Hs)
head(NeuroGenesis4.afterWrap$Geschwind.inVivo.sc.Hs)
head(NeuroGenesis4.afterWrap$Jabaudon.inVivo.sc.Mm)
```

NeuroGenesis4.info

NeuroGenesis4.info

Description

A list of info for 4 neuro data sets sampled, containing: Meissner.inVitro.bulk.Hs, LIBD.AZ.inVitro.bulk.Hs, Geschwind.inVivo.sc.Hs, Jabaudon.inVivo.sc.Mm

Usage

data(NeuroGenesis4)

Format

A list containing 4 dataframes

```
data(NeuroGenesis4.info)
head(NeuroGenesis4.info$Meissner.inVitro.bulk.Hs)
head(NeuroGenesis4.info$LIBD.AZ.inVitro.bulk.Hs)
head(NeuroGenesis4.info$Geschwind.inVivo.sc.Hs)
head(NeuroGenesis4.info$Jabaudon.inVivo.sc.Mm)
```

26 Procrustes

normalizeData

Data Normalization

Description

Normalize data to have mean zero and std 1

Usage

```
normalizeData(
  dataset,
  enable_normalization = TRUE,
  column_sum_normalization = TRUE,
  nonnegative_normalization = FALSE
)
```

Arguments

dataset The input list of data sets matrix enable_normalization

it TRUE

An argument to decide whether to use normalizaiton or not, default is TRUE

column_sum_normalization

An argument to decide whether to use column sum normalization or not, default

nonnegative_normalization

An argument to decide wehther it is nonnegative matrix factorization based or not, default is FALSE

Examples

```
dataset = list(
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50)
)
normalizeData(dataset)
```

Procrustes

Procrustes Projection

Description

Procrustes projection function to solve the procrustes problem $\|A - UB\|^2 U^T U = I$

Usage

```
Procrustes(A, B)
```

projectiLCA 27

Arguments

A The input matrix A as target
B the input matrix B as basis

Value

The procrustes matrix U

projectiLCA

Function to estimate sample embeddings for one dataset from a gene loading matrix derived from an iLCA analysis of another dataset.

Description

projectiLCA estimates the embeddings for samples in a new dataset when given a gene loading matrix from an iLCA analysis result of another single matrix, or set of matrices (e.g. the "list_component" from a twoStageiLCA output object)

Usage

```
projectiLCA(
  proj_dataset,
  proj_group,
  list_component,
  ica_score,
  max_ite = 1000,
  max_err = 1e-04,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE
)
```

Arguments

proj_dataset The dataset(s) to be projected on.

proj_group A logical vector indicating which groupings, i. e. which elements of list_component

should be used for each projected dataset. The length of proj_group should

match the length of list_component.

list_component a single matrix of gene loadings as a list element, or a list_component produced

from a twoStageiLCA() decomposition.

ica_score ice_score produced from a twoStageiLCA() decomposition.

max_ite The maximum number of iterations for the twoStageiLCA algorithms to run,

default value is set to 1000

max_err The maximum error of loss between two iterations, or the program will termi-

nate and return, default value is set to be 0.0001

enable_normalization

An argument to decide whether to use normalization or not, default is TRUE

column_sum_normalization

An argument to decide whether to use column sum normalization or not, default it FALSE

28 projectNMF

Value

A list that contains the projected scores of each dataset on every component.

Examples

```
proj_dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
proj_group = c(TRUE, TRUE) # which groupings in the twoStageiLCA analysis you want to project on.
list_component = twoStageiLCA_res$linked_component_list # from twoStageiLCA result
ica_score = twoStageiLCA_res$ica_score # from twoStageiLCA result
res_projiLCA = projectiLCA(
proj_dataset = proj_dataset,
proj_group = proj_group,
list_component = list_component,
ica_score = ica_score)

PLEASE MAKE SURE YOUR proj_dataset AND list_component ELEMENTS HAVE MEANINGFUL ROW(GENE) NAMES - they are matched.
#**
```

projectNMF

Function to estimate sample embeddings for one dataset from a gene loading matrix derived from an NMF decomposition of another dataset.

Description

projectNMF estimates the embeddings for samples in a new dataset when given a gene loading matrix from an NMF decomposition of another single matrix, or set of matrices (e.g. the "list_component" from a jointNMF output object)

Usage

```
projectNMF(
   proj_dataset,
   proj_group,
   list_component,
   max_ite = 1000,
   max_err = 1e-04,
   enable_normalization = TRUE,
   column_sum_normalization = FALSE
)
```

Arguments

proj_dataset The dataset(s) to be projected on.

proj_group A logical vector indicating which groupings, i. e. which elements of list_component should be used for each projected dataset. The length of proj_group should match the length of list_component.

list_component a single matrix of gene loadings as a list element, or a list_component produced from a jointNMF() decomposition.

max_ite The maximum number of iterations for the jointNMF algorithms to run, default value is set to 1000

pveMultiple 29

max_err The maximum error of loss between two iterations, or the program will terminate and return, default value is set to be 0.0001

enable_normalization

An argument to decide whether to use normalizaiton or not, default is TRUE

column_sum_normalization

An argument to decide whether to use column sum normalization or not, default it FALSE

Value

A list that contains the 1] projected scores of each dataset on every component. and 2] the log of errors as the NMF was iterated.

Examples

```
proj_dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
proj_group = c(TRUE, TRUE) # which groupings in the joint decomposition you want to project on.
list_component = jointNMF$linked_component_list # from jointNMF result
res_projNMF = projectNMF(
proj_dataset = proj_dataset,
proj_group = proj_group,
list_component = list_component)
```

PLEASE MAKE SURE YOUR proj_dataset AND list_component ELEMENTS HAVE MEANINGFUL ROW(GENE) NAMES - they are matched #'

pveMultiple

Percentage of Variance Explained for Multiple Data sets

Description

Compute the PVE (Percentage of Variance Explained) for multiple data sets on multiple components

Usage

```
pveMultiple(dataset, group, comp_num, list_score, list_component)
```

Arguments

dataset A list of dataset to be analyzed

group A list of grouping of the datasets, indicating the relationship between datasets

comp_num A vector indicates the dimension of each compoent

list_score A list of extracted scores by the corresponding algorithm

list_component A list of components comptuted by the corresponding algorithm

Value

The list of scores

30 pveSep

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2)
res_concatPCA = concatPCA(dataset, group, comp_num)
pveMultiple(dataset, group, comp_num, res_concatPCA$score_list, res_concatPCA$linked_component_list)
```

pveSep

Percentage of Variance Explained for separate data set

Description

Compute the PVE (percentage of variance explained) for each data set

Usage

```
pveSep(dataset, list_score, list_component)
```

Arguments

dataset A list of data sets for input

list_score A list of extracted scores by the corresponding algorithm

list_component A list of components comptuted by the corresponding algorithm

Value

The list of scores

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
comp_num = 2
res_sepPCA = sepPCA(dataset, comp_num)
pveSep(dataset, res_sepPCA$score_list, res_sepPCA$linked_component_list)
```

rebalanceData 31

rebalanceData Rebalance Data

Description

Rebalance scores based on the balanced data set

Usage

```
rebalanceData(list_score, group, dataset)
```

Arguments

list_score A list of scores extracted in the analysis

group A list of group assignment dataset A list of dataset in analysis

Value

A list of rebalanced scores

Examples

rotate_component

Simulated Component Rotation

Description

Rotated the simulated component generated

Usage

```
rotate_component(comp, angle = 0)
```

Arguments

comp A matrix of components

angle Rotation angle of the component

Value

Matrix of rotated component

Examples

```
component = svd(matrix(rnorm(100 * 200), nrow = 200))$u[, 1 : 2] rotate_component(component, pi / 6)
```

sampleNameAssign

Sample Name Assignment for Concatenate, Joint and TwoStageLCA Analysis

Description

Assign sample names to the scores

Usage

```
sampleNameAssign(score_list, sample_name)
```

Arguments

score_list A list of scores to do analysis

sample_name A list of names for samples to be analyzed

Value

A list of scores to analyze

Examples

```
x = list(matrix(c(1 : 4), nrow = 2), matrix(c(1 : 4), nrow = 2))
y = list(matrix(c(1 : 4), nrow = 2), matrix(c(1 : 4), nrow = 2))
score_list = list(x, y)
sample_name = list(c("x.sample.1", "x.sample.2"), c("y.sample.1", "y.sample.2"))
sampleNameAssign(score_list, sample_name)
```

sampleNameAssignProj Sample Name Assignment for projectNMF

Description

Assign sample names to dataset, it takes two arguments, the first is the list of scores, the second is the list of sample names

Usage

```
sampleNameAssignProj(score_list, sample_name)
```

Arguments

score_list List of score for each data.frame
sample_name List of names for the samples in the list

Value

A list of scores for the samples

Examples

```
x = matrix(c(1:4), nrow = 2)
y = matrix(c(1:4), nrow = 2)
score_list = list(x, y)
sample_name = list("x.sample.1", "x.sample.2")
sampleNameAssignProj(score_list, sample_name)
```

 ${\tt sample Name Assign Sep}$

Sample Name Assignment for Seperate Analysis

Description

Assign sample names to each dataset in the list of data sets, it takes two arguments, the first is the list of scores, the second is the list of sample names

Usage

```
sampleNameAssignSep(score_list, sample_name)
```

Arguments

```
score_list List of score for each data.frame
sample_name List of names for the samples in the list
```

Value

A list of scores for the samples

```
x = matrix(c(1:4), nrow = 2)
y = matrix(c(1:4), nrow = 2)
score_list = list(x, y)
sample_name = list(c("x.sample.1", "x.sample.2"), c("y.sample.1", "y.sample.2"))
sampleNameAssignSep(score_list, sample_name)
```

34 scoreNameAssign

sampleNameExtractor Sample name extractor

Description

Extract a list of sample names from input list of datasets

Usage

```
sampleNameExtractor(dataset)
```

Arguments

dataset

A list of datasets containing sample names

Value

A vector of sample names

Examples

```
x = matrix(c(1 : 4), nrow = 2)
colnames(x) = c("sp1", "sp2")
y = matrix(c(1 : 4), nrow = 2)
colnames(y) = c("sp3", "sp4")
dataset = list(x, y)
sampleNameExtractor(dataset)
```

scoreNameAssign

Score Name Assignment for Concatenate, Joint and TwoStageLCA Analysis

Description

Assign names to score lists based on dataset_name and group_name, the dataset_name is the output of the dataNameExtractor, the group_name is the output of the groupNameExtractor

Usage

```
scoreNameAssign(score_list, dataset_name, group_name)
```

Arguments

score_list A list of scores in the analysis

dataset_name A vector of datasets names extracted by dataNameExtractor function group_name A vector of group names extracted by groupNameExtractor function

Value

A list of scores assigned with names

scoreNameAssignProj 35

Examples

```
score_list = list(
list(matrix(c(1 : 4), nrow = 2), matrix(c(1 : 4), nrow = 2)),
list(matrix(c(1 : 4), nrow = 2), matrix(c(1 : 4), nrow = 2)))
dataset_name = c("dat1", "dat2")
group_name = c("comp1", "comp2")
scoreNameAssign(score_list, dataset_name, group_name)
```

scoreNameAssignProj

Score Name Assignment for projectNMF

Description

Assign names to score lists based on group_name, the group_name is the output of the groupName-Extractor

Usage

```
scoreNameAssignProj(score_list, group_name)
```

Arguments

score_list A list of scores in the analysis

group_name A vector of group names extracted by groupNameExtractor function

Value

A list of scores assigned with names

Examples

```
score_list = list(
list(matrix(c(1 : 4), nrow = 2), matrix(c(1 : 4), nrow = 2)),
list(matrix(c(1 : 4), nrow = 2), matrix(c(1 : 4), nrow = 2)))
group_name = c("comp1", "comp2")
scoreNameAssignProj(score_list, group_name)
```

scoreNameAssignSep

Score Name Assign for Seperate Analysis

Description

Assign name to scores based on the extracted name for each dataset, it takes the vector of dataset_name outputed by the 'datasetNameExtractor' function

Usage

```
scoreNameAssignSep(score_list, dataset_name)
```

36 score_generation

Arguments

score_list List of scores in sep analysis

Value

A list of scores, content same as input name changed based on dataset_name

Examples

```
score\_list = list(matrix(c(1 : 4), nrow = 2), matrix(c(5 : 8), nrow = 2)) \\ dataset\_name = c("x", "y") \\ scoreNameAssignSep(score\_list, dataset\_name)
```

score_generation

Random Score Generation

Description

Gererate random scores for the simulation

Usage

```
score_generation(dim_score, num_score, score_variance)
```

Arguments

dim_score dimension of the scores

num_score number of scores
score_variance variance of the score

Value

```
a matrix of generated random scores, with dim_score * num_score
```

```
score\_generation(2, 10, c(1,2))
```

ScreePlot_LCvsPC 37

ScreePlot_LCvsPC	genetrate scree plot using tsLCA and sepPCA embeddings for each
	matrix

Description

genetrate scree plot using tsLCA and sepPCA embeddings for each matrix

Usage

```
ScreePlot_LCvsPC(LCAobj, group, sepPCAobj)
```

Arguments

LCAobj list output of tsLCA run, list

group name of the weights group from the SJD_algorithm output, i.e 'Shared.All.13',

str

sepPCAobj list output of sepPCA run, list

Single Data Set Decomposition with Independent Component Analysis

Description

sepICA

Apply ICA (Independent Component Analysis) to a single data set

Usage

```
sepICA(
  dataset,
  comp_num,
  weighting = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset A dataframe/matrix to be decomposed

comp_num Number of ICs to be extracted

weighting Weighting of each dataset, initialized to be NULL

 $enable_normalization$

An argument to decide whether to use normalizaiton or not, default is TRUE

column_sum_normalization

An argument to decide whether to use column sum normalization or not, default

it FALSE

screen_prob A vector of probabilies for genes to be chosen

38 sepNMF

Value

A list of scores and component

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
comp_num = 2
res_sepICA = sepICA(dataset, comp_num)
```

sepNMF

Single Data Set Decomposition with Nonnegative Matrix Factorization

Description

Apply NMF (Nonnegative Matrix Factorization) to a single data set

Usage

```
sepNMF(
  dataset,
  comp_num,
  weighting = NULL,
  perturbation = 1e-04,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset A dataframe/matrix to be decomposed

comp_num Number of NMFs to be extracted

weighting Weighting of each dataset, initialized to be NULL

perturbation A small perturbation to ensure nmf works well
enable_normalization

 $\label{thm:column_sum} An \ argument \ to \ decide \ whether \ to \ use \ normalization \ or \ not, \ default \ is \ TRUE \ column_sum_normalization$

An argument to decide whether to use column sum normalization or not, default it FALSE

screen_prob A vector of probabilies for genes to be chosen

Value

A list of scores and component

sepPCA 39

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
comp_num = 2
res_sepNMF = sepNMF(dataset, comp_num)
```

sepPCA

Single Data Set Decomposition with Principal Component Analysis

Description

Apply PCA (Principal Component Analysis) to a single data set

Usage

```
sepPCA(
  dataset,
  comp_num,
  weighting = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset A dataframe/matrix to be decomposed

comp_num Number of PCs to be extracted

weighting Weighting of each dataset, initialized to be NULL

enable_normalization

An argument to decide whether to use normalization or not, default is TRUE

 ${\tt column_sum_normalization}$

An argument to decide whether to use column sum normalization or not, default

it FALSE

screen_prob A vector of probabilies for genes to be chosen

Value

A list of scores and component

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
comp_num = 2
res_sepPCA = sepPCA(dataset, comp_num)
```

40 SJDScorePlotter

```
simulated_data_generation
```

Simulated Data Generation

Description

Generate simulation data

Usage

```
simulated_data_generation(
  configuration_setting,
  amplitude = 1,
  heterogeneousNoise = FALSE
)
```

Arguments

```
configuration_setting
setting for the configuration
amplitude The amplitude of the score variance
heterogeneousNoise
Whether the noise for each dataset to be heteregeneous
```

Examples

```
configuration_setting = configuration_setting_generation()
simulated_data_generation(configuration_setting)
```

SJDScorePlotter

Plot SJD score

Description

plot dimensionality reduction scores for each SJD algorithm for dataset analyzed by SJD

Usage

SJDScorePlotter 41

Arguments

SJDalg SJD algorithm to plot i.e 'twoStageLCA'
scores score list of the SJD algorithm i.e twoStageLCA\$score_list
lbb dataset label i.e 'NeuroGenesis4'
info list of sample meta data matrices

SampleMetaNamesTable

dataframe containing column information of each sample meta data matrices

clrs2end color scale for result scores from other algorithms. Default: c("plum","purple","blue4","blue4","black"

clrs1end color scale for result scores from sepNMF, concatNMF and jointNMF algo-

rithms. Default: c("black","black","black","darkred","red","orange","yellow")

Value

A list containing ggplot object

```
library(ggplot2)
data(NeuroGenesis4.afterWrap)
data(NeuroGenesis4.info)
SampleMetaNamesTable = data.frame(
   row.names = names(NeuroGenesis4),
   Type = c('Yaxis','Yaxis','2Dscatter','2Dscatter'),
   XaxisColumn = c("X","DAYx","tSNE_1","tsne1:ch1"),
   YaxisColumn = c("PJDscores","PJDscores","tSNE_2","tsne2:ch1"),
   COLaxisColumn = c("color","colorBYlabelsX","PJDscores","PJDscores"),
   PCHColumn = c("","","",""),
   inset = c(TRUE, TRUE, TRUE, TRUE),
   insetLOC = c("topright", "topright", "topright", "topright"),
   insetZoom = c(0.3, 0.3, 0.3, 0.3),
   ordDECREASE=c(FALSE, FALSE, FALSE, FALSE),
   CLRfoldPRB=c(0.5, 0.5, 0.5, 0.5)
)
grp = list(
Shared.All.4 = c(1 : 4),
Shared.bulk.2 = c(1, 2),
Shared.sc.2 = c(3, 4),
Hs.Meisnr.1 = c(1),
Hs.AZ.1 = c(2),
Gesch.1 = c(3),
Telley.1 = c(4)
dims = c(2, 2, 2, 2, 2, 2, 2)
twoStageLCA.out = twoStageLCA(dataset = NeuroGenesis4.afterWrap, group = grp, comp_num = dims)
SJDScorePlotter.obj = SJDScorePlotter(
    SJDalg = "twoStageLCA",
    scores = twoStageLCA.out$score_list,
    lbb = "NeuroGenesis4.p2",
```

42 sjdWrap

```
info = NeuroGenesis4.info,
    SampleMetaNamesTable = SampleMetaNamesTable
)
```

sjdWrap

SJD Wrap

Description

wrapping up expression matrices (of different species) with only shared genes as SJD input

Usage

```
sjdWrap(
  data.list,
  species.vector,
  geneType.vector,
  geneType.out = "symbol",
  species.out
)
```

Arguments

```
data.list input list of expression matrices from different species i.e human and mouse datasets

species.vector character of species type of each matrix i.e c('human', 'mouse', 'mouse')

geneType.vector character of gene/rowname type of each matrix i.e c("symbol","ensembl","symbol")

geneType.out character of output gene/rowname type of each matrix i.e "symbol"

species.out character of output species type for gene/rowname
```

Value

A list of expression matrices (of different species) with only shared genes

```
data(NeuroGenesis4)
SJDdataIN = sjdWrap(
data.list = NeuroGenesis4,
species.vector=c("human","human","human","mouse"),
geneType.vector=c("symbol","ensembl","symbol","symbol"),
geneType.out="symbol",
species.out="human")
```

sjdWrapProjection 43

sjdWrapProjection

SJD Wrap for projection

Description

Wrapping up a new list of projection matrices with the list computed from the function of sjdWrap

Usage

```
sjdWrapProjection(
  data.list.template,
  data.list.projection,
  species.template,
  species.vec.projection,
  geneType.template,
  geneType.vec.projection)
```

Arguments

Value

A list of expression matrices (of different species) with only shared genes to be projected

```
## Load NeuroGenesis4 data into R
data(NeuroGenesis4)

## sjdWrap of the training data sets
SJDdataIN = sjdWrap(
data.list = NeuroGenesis4,
species.vector=c("human","human","mouse"),
geneType.vector=c("symbol","ensembl","symbol","symbol"),
geneType.out="symbol",
species.out="human")

## Sample from data, serving as the projection expression matrices
```

44 twoStageiLCA

```
NeuroGenesis4.sample = NeuroGenesis4
NeuroGenesis4.sample[[1]] = NeuroGenesis4.sample[[1]][-5,]
rownames(NeuroGenesis4.sample[[1]])[5] = paste0(rownames(NeuroGenesis4.sample[[1]])[5], ".test")
NeuroGenesis4.sample[[2]] = NeuroGenesis4.sample[[2]][-10,]
rownames(NeuroGenesis4.sample[[2]])[10] = paste0(rownames(NeuroGenesis4.sample[[2]])[10], ".test")
NeuroGenesis4.sample[[3]] = NeuroGenesis4.sample[[3]][-15,]
rownames(NeuroGenesis4.sample[[3]])[15] = paste0(rownames(NeuroGenesis4.sample[[3]])[15], ".test")
NeuroGenesis4.sample[[4]] = NeuroGenesis4.sample[[4]][-20,]
rownames(NeuroGenesis4.sample[[4]])[20] = paste0(rownames(NeuroGenesis4.sample[[4]])[20], ".test")
SJDdataProjection = sjdWrapProjection(
SJDdataIN, NeuroGenesis4.sample, "human", c("human", "human", "mouse"),
"symbol", c("symbol", "ensembl", "symbol", "symbol"))
```

twoStageiLCA

Two-staged Independent Linked Component Analysis

Description

Two-staged Independent Linked Component Analysis, a generalization based on the Two-staged Independent Linked Component Analysis

Usage

```
twoStageiLCA(
  dataset,
  group,
  comp_num,
  weighting = NULL,
  backup = 0,
  plotting = FALSE,
  proj_dataset = NULL,
  proj_group = NULL,
  enable_normalization = TRUE,
  column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset A list of dataset to be analyzed

group A list of grouping of the datasets, indicating the relationship between datasets

comp_num A vector indicates the dimension of each compoent weighting Weighting of each dataset, initialized to be NULL

backup A positive scalar to determine how many ICs to over select

plotting A boolean value to determine whether to plot the scree plot or not, default to be

False

twoStageiLCA.rank 45

proj_dataset The dataset(s) to be projected on.

proj_group A listed of boolean combinations indicating which groupings should be used

for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should

match the length of the parameter group.

enable_normalization

 $\label{thm:column_sum} An \ argument \ to \ decide \ whether \ to \ use \ normalization \ or \ not, \ default \ is \ TRUE \ column_sum_normalization$

An argument to decide whether to use column sum normalization or not, default

it FALSE

screen_prob A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after 2siLCA algorithm

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2,2)
proj_dataset = matrix(runif(5000, 1, 2), nrow = 100, ncol = 50)
proj_group = c(TRUE, TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE)
res_twoStageiLCA = twoStageiLCA(
dataset,
group,
comp_num,
proj_dataset = proj_dataset,
proj_group = proj_group)
```

twoStageiLCA.rank

Two-staged Independent LCA and automatic rank selection

Description

Two-staged decomposition of several matrices with Independent LCA, twoStageLCA is first performed on the data, the rank selection procedure is automatic based on BEMA. Then, fastICA is implemented on the score to extract the independent components.

Usage

```
twoStageiLCA.rank(
  dataset,
  group,
  weighting = NULL,
  total_number = NULL,
```

46 twoStageiLCA.rank

```
threshold,
backup = 0,
plotting = FALSE,
proj_dataset = NULL,
proj_group = NULL,
enable_normalization = TRUE,
column_sum_normalization = FALSE,
screen_prob = NULL
)
```

Arguments

dataset A list of dataset to be analyzed

group A list of grouping of the datasets, indicating the relationship between datasets

weighting Weighting of each dataset, initialized to be NULL

total_number Total number of components will be extracted, if default value is set to NA, then

BEMA will be used.

threshold The threshold used to cutoff the eigenvalues

backup A backup variable, which permits the overselection of the components by BEMA

plotting A boolean value to determine whether to plot the scree plot or not, default to be

False

proj_dataset The dataset(s) to be projected on.

proj_group A listed of boolean combinations indicating which groupings should be used

for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should

match the length of the parameter group.

enable_normalization

An argument to decide whether to use normalization or not, default is TRUE column_sum_normalization

An argument to decide whether to use column sum normalization or not, default

it FALSE

screen_prob A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after twoStageiLCA.rank algorithm

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1, 2, 3, 4), c(1, 2), c(3, 4), c(1, 3), c(2, 4), c(1), c(2), c(3), c(4))
threshold = c(3, 1.5, 1.5, 1.5, 1.5, 0.5, 0.5, 0.5, 0.5)
res_twoStageiLCA.rank = twoStageiLCA.rank(
dataset,
group,
threshold = threshold)
```

47 twoStageLCA

twoStageLCA	Two-staged Linked	Component Analysis
-------------	-------------------	--------------------

Description

Two-staged Linked Component Analysis

Usage

```
twoStageLCA(
 dataset,
 group,
 comp_num,
 weighting = NULL,
 backup = 0,
 plotting = FALSE,
 proj_dataset = NULL,
 proj_group = NULL,
 enable_normalization = TRUE,
 column_sum_normalization = FALSE,
 screen_prob = NULL
)
```

Arguments

dataset	A list of dataset to be analyzed	
group	A list of grouping of the datasets, indicating the relationship between datasets	
comp_num	A vector indicates the dimension of each compoent	
weighting	Weighting of each dataset, initialized to be NULL	
backup	A positive scalar to determine how many PCs to over select	
plotting	A boolean value to determine whether to plot the scree plot or not, default to be False	
proj_dataset	The dataset(s) to be projected on.	
proj_group	A listed of boolean combinations indicating which groupings should be used for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should match the length of the parameter group.	
enable_normalization		
	An argument to decide whether to use normalization or not default is TDIJE	

An argument to decide whether to use normalization or not, default is TRUE column_sum_normalization

> An argument to decide whether to use column sum normalization or not, default it FALSE

A vector of probabilies for genes to be chosen screen_prob

Value

A list contains the component and the score of each dataset on every component after 2sLCA algorithm

48 twoStageLCA.rank

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1,2,3,4), c(1,2), c(3,4), c(1,3), c(2,4), c(1), c(2), c(3), c(4))
comp_num = c(2,2,2,2,2,2,2,2,2)
proj_dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
proj_group = list(c(TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE))
res_twoStageLCA = twoStageLCA(
dataset,
group,
comp_num,
proj_dataset = proj_dataset,
proj_group = proj_group)
```

twoStageLCA.rank

Two-staged LCA and automatic rank selection

Description

Two-staged decomposition of several matrices with LCA, the rank selection procedure is automatic based on BEMA

Usage

```
twoStageLCA.rank(
 dataset,
 group,
 weighting = NULL,
 total_number = NULL,
  threshold,
 backup = 0,
 plotting = FALSE,
 proj_dataset = NULL,
 proj_group = NULL,
 enable_normalization = TRUE,
 column_sum_normalization = FALSE,
  screen_prob = NULL
)
```

Arguments

dataset

A list of grouping of the datasets, indicating the relationship between datasets group Weighting of each dataset, initialized to be NULL weighting

A list of dataset to be analyzed

total_number Total number of components will be extracted, if default value is set to NA, then

BEMA will be used.

threshold The threshold used to cutoff the eigenvalues

backup A backup variable, which permits the overselection of the components by BEMA umap_tsne_onLC 49

plotting A boolean value to determine whether to plot the scree plot or not, default to be

False

proj_dataset The dataset(s) to be projected on.

proj_group A listed of boolean combinations indicating which groupings should be used

for each projected dataset. The length of proj_group should match the length of proj_dataset, and the length of each concatenated boolean combination should

match the length of the parameter group.

enable_normalization

An argument to decide whether to use normalization or not, default is TRUE

column_sum_normalization

An argument to decide whether to use column sum normalization or not, default

it FALSE

screen_prob A vector of probabilies for genes to be chosen

Value

A list contains the component and the score of each dataset on every component after seqPCA algorithm

Examples

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
group = list(c(1, 2, 3, 4), c(1, 2), c(3, 4), c(1, 3), c(2, 4), c(1), c(2), c(3), c(4))
threshold = c(3, 1.5, 1.5, 1.5, 1.5, 0.5, 0.5, 0.5, 0.5)
res_twoStageLCA.rank = twoStageLCA.rank(
dataset,
group,
threshold = threshold)
```

umap_tsne_onLC

genetrate new UMAP and TSNE coordinates given LCA object

Description

genetrate new UMAP and TSNE coordinates given LCA object

Usage

```
umap_tsne_onLC(twoStageLCAobj, group, n_comp, add_to_meta, meta_list)
```

Arguments

twoStageLCAobj list output of tsLCA run, list

group name of the weights group from the SJD_algorithm output, i.e 'Shared.All.13'

n_comp number of components to make umaps and tsne, int
add_to_meta adds tsne and umap coordinates to metadata, bool
meta_list Meta.List that was passed into SJD run, str

50 weightData

weightData

Weighting Data Set

Description

To weight each data set based on input weighting vector

Usage

```
weightData(dataset, weighting)
```

Arguments

dataset A list of data sets

weighting A vector of weighting constant for each data set

Value

A list of weighted data sets

```
dataset = list(matrix(runif(5000, 1, 2), nrow = 100, ncol = 50),
matrix(runif(5000, 1, 2), nrow = 100, ncol = 50))
weighting = c(1, 2)
weighted_dataset = weightData(dataset, weighting)
```

Index

TinkedPCA, 23 PoeMultiple, 29 PoeSep, 30	linkedPCA, 23	* Analysis,	* analysis
* Component linkedPCA, 23 * ICA concatICA, 8 jointICA, 18 sepICA, 37 * Joint, jointICA, 18 * LCA twoStageiLCA, 44 twoStageiLCA, 47 twoStageiLCA, 47 twoStageiLCA, 7ank, 48 * Linked linkedPCA, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectMNF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * balanceData, 6 * component, assemble. byComponent, 3 * compNameAssign. 7 compNameAssignSep, 8 * component rotate_component, 31 * component rotate_component, 31 * compNameAssignSep, 8 * component rotate_component, 3 * compNameAssignSep, 8 * component rotate_component, 31 * compNameAssignSep, 8 * component rotate_component, 3 * compNameAssignSep, 8 * component rotate_component, assemble.byComponent, assemble.byComponent. assemble.byComponet. assemble.byComponent. assemble.byComponent. assemble.byCompon	* Component linkedPCA, 23 * ICA concatICA, 8 jointICA, 18 sepICA, 37 * Joint, jointICA, 18 * LCA twoStageiLCA, 44 twoStageiLCA, 47 twoStageiLCA, 47 twoStagelCA, 47 twoStagelCA, 48 linkedPCA, 23 * Marchenko-Pastur, marchenko-pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * geneNameExtractor, 15 * balanceData, 6 * component, assemble byComponent, 3 * compNameAssign. 7 compNameA		•
linkedPCA, 23 * ICA concatICA, 8 jointICA, 18 sepICA, 37 * Joint, joint(A, 18 * twoStageiLCA, 44 twoStageiLCA, 47 twoStagelCA, 77 * Linked linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * NMI * Multiple pveMultiple, 29 * NA, filterNaValue, 13 * NMF concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * Separate pveSepICA, 37 sepNMF, 38 * SepICA, 37 sepNMF, 38	linkedPCA, 23 * ICA concatICA, 8 jointICA, 18 sepICA, 37 * Joint, jointICA, 18 * LCA twoStageiLCA, 44 twoStageiLCA, 47 twoStageiLCA, 47 twoStagelCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNaValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepPMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * sanalysis, sepICA, 37 sepIMF, 38 * SpiKe BEMA, 6 * sanalysis, sepICA, 37 sepIMF, 38 * component rotate_component, 31 * compNameAssignSep, 8 * compNameAssign, 15 assemble.byComponent, 31 * compNameAssign Pope 8 * configuration_setting_generation, 40 * concatPCA, 11 jointPCA, 21 sepPCA, 39 * sextractor simulated_data_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNaValue, 13 * for configuration_setting_generation, 12 * sampleNameExtractor, 34 * filter filterNaValue, 13 * for configuration_setting_generation, 12 * gene, geneNameExtractor, 15		
* ICA concatICA, 8 jointICA, 18 sepICA, 37 * Joint, jointICA, 18 * sepICA, 37 * Joint, jointICA, 18 * LCA twoStageiLCA, 44 twoStageiLCA, 47 twoStagelCA, 77 twoStageLCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNaValue, 13 * PCA concatPCA, 11 jointPCA, 21 sepPVF, pveSep, 30 * Spike BEMA, 6 * semponent rotate_component, 31 * component rotate_component, 31 * comfigure configuration_setting_generation, 12 * dataset, mtx, 24 NeuroGenesis4, 24 * NeuroGenesis4, 24 * NeuroGenesis4, afterWrap, 25 NeuroGenesis4, afterWrap, 25 * NeuroGenesis4, info, 25 * dataset balanceData, 6 * data concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveWultiple, 29 pveSep, 30 * Spike BEMA, 6 * semalysis, sepICA, 37 sepNMF, 38	*ICA concatICA, 8 jointICA, 18 sepICA, 37 * Joint, jointICA, 18 sepICA, 37 * Joint, jointICA, 18 * LCA twoStageiLCA, 44 twoStageiLCA, rank, 45 twoStagelCA, rank, 48 * Linked linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNaValue, 13 * NMF concatPCA, 11 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 21 sepPCA, 39 * PVE, pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * component, assemble. byComponent, 31 * component rotate_component, 31 * component, component rotate_component, 31 * component rotate_component, 31 * comfigure configuration_setting_generation, 12 * dataset, assemble. byDataset, 4 * dataset, assemble. byDataset, 4 * MeuroGenesis4, 24 * NeuroGenesis4, 24 * NeuroGenesis4, afterWrap, 25 * NeuroGenesis4, afterWrap, 25 * NeuroGenesis4. info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNaValue, 13 * for configuration_setting_generation, 12 * sepnen, sepn		
concatICA, 8 jointICA, 18 sepICA, 37 * Joint, jointICA, 18 jointICA, 18 * compNameAssign, 7 * CompNameAssign, 7 * compNameAssignSep, 8 * component rotate_component, 31 * configure rotate_component, 31 * configure rotate_component, 31 * configure rotate_component, 31 * configure configuration_setting_generation, 12 twoStageLCA, 47 twoStageLCA, rank, 48 * Linked linkedPcA, 23 * Marchenko-Pastur, marchenko-pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * component rotate_component, 3 * component rotate_component, 31 * configuration_setting_generation, 12 * expression sidMrap, 42 sidMrap, 42 sidWrapProjection, 43 * extractor sampleNameExtractor, 34 filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, sepICA, 37 sepNMF, 38 geneNameExtractor, 15	concatICA, 8 jointICA, 18 sepICA, 37 * Joint, jointICA, 18 jointICA, 18 jointICA, 18 jointICA, 18 jointICA, 18 jointICA, 18 * LOCA twoStageiLCA, 44 twoStageiLCA, 47 twoStagelCA, 47 twoStagelCA, 47 twoStagelCA, 48 * Linked linkedPCA, 23 * Marchenko-Pastur, marchenko-pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNaValue, 13 * NMF concatPCA, 11 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * sepICA, 37 sepIMF, 38 * Component rotate_component, 31 * component rotate_component, 3 * component rotate_component rotate_component, 3 * component rotate_component, 31 * configuration_setting_generation, 12 * sepression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepIMF, 38 * geneNameExtractor, 15		•
jointICA, 18 sepICA, 37 sopInt, Joint, Joint, JointCA, 18 **CompNameAssign, 7 compNameAssign, 7 compNameAssign, 7 compNameAssignSep, 8 component **LCA twoStageiLCA, 44 twoStageiLCA, 47 twoStagelCA, 47 twoStagelCA, ank, 48 **Linked linkedPCA, 23 **Marchenko-Pastur, marchenko-pastur_quantile, 23 **Multiple pveMultiple, 29 **NA, filterNAValue, 13 **NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 **PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 **PVE, pveMultiple, 29 **PVE, pveMultiple, 29 pveSep, 30 **Spike sepICA, 37 sepNMF, 38 **Exicate compNameAssign, 7 compNameAssign, 7 compNameAssign, 7 compNameAssign, 7 compNameAssign, 15 c	jointICA, 18 sepICA, 37 * Joint, jointICA, 18 * LCA twoStageiLCA, 44 twoStageiLCA, 47 twoStageLCA. rank, 45 twoStageLCA. rank, 48 * Linked linkedPCA, 23 * Marchenko-Pastur, marchenko-pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * SepRe geneNameExtractor, 15	concatICA, 8	
sepICA, 37 * Joint, jointICA, 18 * LCA * twoStageiLCA, 44 * twoStageiLCA. rank, 45 * twoStagelCA. arak, 48 * Linked * linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * SepNMF, 38 compNameAssign, 7 compNameAssign, 5 geneNameAssign, 15 geneNameExtractor, 15	* Joint, joint, joint, joint, last LCA * LCA * twoStageiLCA, 44 * twoStageiLCA, 47 * twoStageiLCA, 47 * twoStagelCA, 47 * Linked 1 inkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * analysis, sepNMF, 38 * component rotate_component, 31 * configuration_setting_generation, 12 * configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * filter configuration_setting_generation, 40 * expression sjdWrapProjection, 43 * filter configuration_setting_generation, 40 * expression sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 40 * expression sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 40 * expression sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 12 * gene, sepICA, 37 sepNMF, 38 * geneNameAssign, 15 geneNameExtractor, 15		_
* Joint compNameAssignSep, 8 jointICA, 18 * component rotate_component, 31 * LCA	* Joint, jointICA, 18 * LCA twoStageiLCA, 44 twoStageiLCA, 47 twoStagelCA, 77 twoStagelCA, 78 timed linked linkedPCA, 23 * Marchenko-pastur, marchenko-pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepNMF, 38 * compNameAssignSep, 8 * component rotate_component, 31 rotate_component, 31 rotate_component rotate_component, 31 rotate_component rotate_component, 31 rotate_component rotate_component, 31 rotate_component, 31 rotate_component rotate_component, 31 * configuration_setting_generation, 12 * sepne, geneNameExtractor, 15		
* LCA	* LCA	* Joint,	
* LCA twoStageiLCA, 44 twoStageiLCA, rank, 45 twoStageiLCA, rank, 48 * configure configuration_setting_generation, twoStagelCA.rank, 48 * Linked linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 * NE separate pveSep, 30 * Spike BEMA, 6 * sepNMF, 38 * configuration_setting_generation, 12 * sepNMF, 38 * for configuration_setting_generation, 40 * expression sigMrap, 42 sigMrapProjection, 43 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 40 * expression sigMrapProjection, 43 * for configuration_setting_generation, 40 * expression sigMrapProjection, 43 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 12 * sepneNameAssign, 15 geneNameAssign, 15 geneNameExtractor, 15	* LCA	jointICA, 18	* component
twoStagelCA.rank, 45 twoStagelCA, 47 twoStagelCA.rank, 48 * Linked linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 * Separate pveSep, 30 * Sepike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * configuration_setting_generation, 12 * configuration_setting_generation, 12 * configuration_setting_generation, 12 * sextractor simulated_data_generation, 40 * expression sidWrap, 42 sigdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 12 * expression sidWrap, 42 sidWrapProjection, 43 * extractor configuration_setting_generation, 12 * expression sidWrap, 42 sidWrapProjection, 43 * extractor configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 * geneNameAssign, 15 geneNameExtractor, 15	twoStagelLCA.rank, 45 twoStagelCA.rank, 48 *Linked linkedPCA, 23 *Marchenko-Pastur, marchenko_pastur_quantile, 23 *Nultiple pveMultiple, 29 *NA, filterNAValue, 13 *NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 *PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 *PVE, pveMultiple, 29 *PVE, pveMultiple, 29 *PVE, pveMultiple, 29 *Separate pveSep, 30 *Spike BEMA, 6 *analysis, sepICA, 37 sepNMF, 38 *configuration_setting_generation, 12 sepremation, 40 *configuration_setting_generation, 40 *configuration_setting_generation, 40 *expression sjdWrap, 42 sjdWrap, 42 sjdWrapProjection, 43 *for configuration_setting_generation, 40 *expression sjdWrap, 42 sjdWrapProjection, 43 *for configuration_setting_generation, 40 *expression sjdWrap, 42 sjdWrapProjection, 43 *for configuration_setting_generation, 12 *extractor sampleNameExtractor, 34 *for configuration_setting_generation, 12 *analysis, sepICA, 37 sepNMF, 38 geneNameAssign, 15 geneNameExtractor, 15	* LCA	_
twoStageLCA, 47 twoStageLCA.rank, 48 * Linked linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 * Separate pveSep, 30 * Spike BEMA, 6 * Linked * dataframe, frameToMatrix, 14 * dataset, assemble. byDataset, 4 * datasets * MutroGenesis 4, 24 NeuroGenesis 4, 24 NeuroGenesis 4. afterWrap, 25 NeuroGenesis 4. info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 40 * expression sjdWrap 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 * geneNameAssign, 15 geneNameExtractor, 15	twoStageLCA, 47 twoStageLCA.rank, 48 * Linked linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 * PVE, pveMultiple, 29 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * dataframe, frameToMatrix, 14 * dataset, assemble.byDataset, 4 * datasets * MutroGenesis 4, 24 NeuroGenesis 4, 24 NeuroGenesis 4. afterWrap, 25 NeuroGenesis 4. info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 12 * expression sjdWrap, 42 sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 12 * configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 * geneNameAssign, 15 geneNameExtractor, 15	twoStageiLCA, 44	* configure
<pre>twoStageLCA.rank, 48 * Linked</pre>	twoStageLCA.rank, 48 * Linked	twoStageiLCA.rank,45	configuration_setting_generation,
* Linked linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 12 * SpiNMF, 38 filterNavalue, 13 * dataset balanceData, 6 * dataset configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor configuration_setting_generation, 40 * spike spitch, 37 sepNMF, 38 geneNameAssign, 15 geneNameExtractor, 15	* Linked linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * Separate SepNMF, 38 * SepNMF, 38 * SepNMF, 38 * SepNAF	twoStageLCA, 47	12
linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 * PVE, pveMultiple, 29 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * dataset * dataset * heuroGenesis4.afterWrap, 25 NeuroGenesis4.info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 40 * expression simulated_data_generation, 40 * expression sigWrap, 42 sigWrapProjection, 43 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 40 * filter configuration_setting_generation, 40 * gene, geneNameAssign, 15 geneNameExtractor, 15	linkedPCA, 23 * Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * dataset, assemble.byDataset, 4 * datasets mtx, 24 NeuroGenesis4, 24 NeuroGenesis4.afterWrap, 25 NeuroGenesis4.info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 20 * data configuration_setting_generation, 40 * expression simulated_data_generation, 40 * expression sidWrap, 42 sidWrap, 42 sidWrap, 42 sidWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15	twoStageLCA.rank,48	* dataframe,
* Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepPMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * dataset NeuroGenesis4. 24 NeuroGenesis4. afterWrap, 25 NeuroGenesis4. info, 25 * dataset sept dataset sept data configuration_setting_generation, 25 * data configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 40 * configuration_setting_generation, 34 * gene, geneNameAssign, 15 geneNameExtractor, 15	* Marchenko-Pastur, marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * dataset NeuroGenesis4. afterWrap, 25 NeuroGenesis4. info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 12 * expression sjdWrap, 42 sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15	* Linked	frameToMatrix, 14
<pre>marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * Sepick sepICA, 37 sepNMF, 38 * dataset NeuroGenesis4. afterWrap, 25 NeuroGenesis4. info, 25 * dataset veuroGenesis4. info, 25 * dataset velataset configuration_setting_generation, 40 * data configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter * configuration_setting_generation, 12 * gene, gene, gene, gene, gene, gene, gene, geneNameAssign, 15 geneNameExtractor, 15</pre>	<pre>marchenko_pastur_quantile, 23 * Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * SepICA, 37 sepNMF, 38 * dataset NeuroGenesis4. afterWrap, 25 NeuroGenesis4. info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 12 * configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 12 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 * geneNameAssign, 15 geneNameExtractor, 15</pre>	linkedPCA, 23	* dataset,
* Multiple pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * SepICA, 37 sepNMF, 38 mtx, 24 NeuroGenesis4, 24 NeuroGenesis4. afterWrap, 25 NeuroGenesis4. info, 25 NeuroGenesis4. info, 25 NeuroGenesis4. afterWrap, 25 NeuroGenesis4. after Wrap, 25 NeuroGenesis4. after Wrap, 25 Neurogenesis4. after Wrap, 25 Neurosensella, 6 **anderson after Wrap, 24 **expection, 40 **expection,	* Multiple pveMultiple, 29 * NA, NeuroGenesis4, 24 * NA, FilterNAValue, 13 * NMF ConcatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA ConcatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * SepICA, 37 sepNMF, 38 mtx, 24 NeuroGenesis4, 24 NeuroGenesis4. afterWrap, 25 NeuroGenesis4. info, 25 NeuroGenesis4. info, 25 NeuroGenesis4. afterWrap, 25 NeuroGenesis4. after Wrap, 25 NeuroGenesis4. after Wrap, 25 NeuroSenson after	* Marchenko-Pastur,	assemble.byDataset,4
pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * SepICA, 37 sepNMF, 38 * NeuroGenesis4. afterWrap, 25 NeuroGenesis4. info, 25 NeuroGenesis4. afterWrap, 25 NeuroGenesis4. after Wrap, 25 NeuroSender after Wrap,	pveMultiple, 29 * NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * Expression BEMA, 6 * SepNMF, 38 * Red * NeuroGenesis4. afterWrap, 25 NeuroGenesis4. info, 25 * NeuroGenesis4. afterWrap, 25 NeuroGenesis4. after Wrap, 25 NeuroGenesis4. after Wrap, 25 NeuroSeponder * data* * configuration_setting_generation, 25 NeuroSeponder * sepPCA data* * sepP	<pre>marchenko_pastur_quantile, 23</pre>	* datasets
* NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike sepICA, 37 sepNMF, 38 NeuroGenesis4. info, 25 * data NeuroGenesis4. info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 40 * expression simulated_data_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for * spike sepICA, 37 sepICA, 37 sepNMF, 38 NeuroGenesis4. info, 25 * data configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15	* NA, filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike sepICA, 37 sepNMF, 38 * NeuroGenesis4.info, 25 * data NeuroGenesis4.info, 25 * dataset balanceData, 6 * data configuration_setting_generation, 40 * expression simulated_data_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 * gene, geneNameExtractor, 15	* Multiple	mtx, 24
filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike spike spicA, 37 sepNMF, 38 NeuroGenesis4.info, 25 * data configuration_setting_generation, 12 * expression sjdWrap, 42 sjdWrap, 42 sjdWrapProjection, 43 * extractor sjdWrapProjection, 43 * extractor sinulated_data_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 filter configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 sepenNameExtractor, 15	filterNAValue, 13 * NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * NeuroGenesis4.info, 25 * data balanceData, 6 * data configuration_setting_generation, 12 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 NeuroGenesis4.info, 25 * data configuration_setting_generation, 12 * gene, sepenenameAssign, 15 geneNameExtractor, 15	pveMultiple, 29	NeuroGenesis4, 24
* NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 sepNMF, 38 * Red configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 pveSep, 30 * for * Spike sepICA, 37 sepNMF, 38 * gene, sepICA, 37 sepNMF, 38 * geneNameAssign, 15 geneNameExtractor, 15	* NMF concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 sepNMF, 38 * Adata configuration_setting_generation, 40 * expression simulated_data_generation, 40 * expression sidWrap, 42 simulated_data_generation, 40 * expression sidWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 geneNameAssign, 15 geneNameExtractor, 15	* NA,	NeuroGenesis4.afterWrap, 25
concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 balanceData, 6 * data configuration_setting_generation, 12 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, sepICA, 37 sepNMF, 38 seplonameExtractor, 15	concatNMF, 9 jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 balanceData, 6 * data configuration_setting_generation, 12 * expression sidWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 12 * gene, gene, geneNameAssign, 15 geneNameExtractor, 15	filterNAValue, 13	NeuroGenesis4.info, 25
jointNMF, 20 projectNMF, 28 sepNMF, 38 *PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 *PVE, pveMultiple, 29 pveSep, 30 *Separate pveSep, 30 *Spike BEMA, 6 *Spike sepICA, 37 sepNMF, 38 * data configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 geneNameAssign, 15 geneNameExtractor, 15	jointNMF, 20 projectNMF, 28 sepNMF, 38 * PCA * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * SepICA, 37 sepNMF, 38 * data configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 geneNameAssign, 15 geneNameExtractor, 15	* NMF	* dataset
projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike pveSep, 30 * Spike BEMA, 6 * analysis, sepNMF, 38 configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 configuration_setting_generation, 12 * gene, sepICA, 37 sepNMF, 38 configuration_setting_generation, 15 geneNameAssign, 15 geneNameExtractor, 15	projectNMF, 28 sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * SepNMF, 38 configuration_setting_generation, 40 * expression sjdWrap, 42 sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 configuration_setting_generation, 15 geneNameAssign, 15 geneNameExtractor, 15	concatNMF, 9	balanceData, 6
sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Spike BEMA, 6 * SepNMF, 38 12 simulated_data_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * analysis, sepICA, 37 sepNMF, 38 12 simulated_data_generation, 40 * expression sjdWrapProjection, 43 * extractor configuration, 43 * for configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15	sepNMF, 38 * PCA concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * sepICA, 37 sepNMF, 38 12 simulated_data_generation, 40 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, sepICA, 37 sepNMF, 38 geneNameAssign, 15 geneNameExtractor, 15	jointNMF, 20	* data
* PCA simulated_data_generation, 40 concatPCA, 11 * expression jointPCA, 21 sigdWrap, 42 sepPCA, 39 sigdWrapProjection, 43 * PVE, extractor pveMultiple, 29 sampleNameExtractor, 34 pveSep, 30 * filter * Separate filterNAValue, 13 pveSep, 30 * for * Spike configuration_setting_generation, BEMA, 6 12 * analysis, sepICA, 37 geneNameAssign, 15 geneNameExtractor, 15	* PCA simulated_data_generation, 40 concatPCA, 11	projectNMF,28	configuration_setting_generation,
concatPCA, 11 jointPCA, 21 sepPCA, 39 sjdWrap, 42 sjdWrapProjection, 43 * PVE, pveMultiple, 29 pveSep, 30 * filter * Separate pveSep, 30 * for * Spike sepMA, 6 BEMA, 6 BEMA, 6 sepICA, 37 sepNMF, 38 * extractor sampleNameExtractor, 34 * for configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15	concatPCA, 11 jointPCA, 21 sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * expression sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, gene, geneNameAssign, 15 geneNameExtractor, 15	sepNMF, 38	12
jointPCA, 21 sepPCA, 39 sjdWrap, 42 sjdWrapProjection, 43 * PVE, pveMultiple, 29 sampleNameExtractor, 34 pveSep, 30 * filter * Separate pveSep, 30 * for * Spike configuration_setting_generation, BEMA, 6 12 * analysis, sepICA, 37 sepNMF, 38 sjdWrap, 42 sampleNameExtractor, 34 * extractor sampleNameExtractor, 34 * filter configuration_setting_generation, 12 * gene, sepICA, 37 sepNMF, 38 sepNMF, 38 speneNameAssign, 15 sepeneNameExtractor, 15	jointPCA, 21 sepPCA, 39 sjdWrap, 42 sjdWrapProjection, 43 * PVE, pveMultiple, 29 sampleNameExtractor, 34 pveSep, 30 * filter * Separate pveSep, 30 * for * Spike configuration_setting_generation, BEMA, 6 12 * analysis, sepICA, 37 sepNMF, 38 sjdWrap, 42 sjdWrapProjection, 43 * extractor sampleNameExtractor, 34 siglter configuration_setting_generation, 12 * gene, sepICA, 37 sepNMF, 38 sepNMF, 38	* PCA	${\sf simulated_data_generation}, 40$
sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * filter * Separate pveSep, 30 * for * Spike BEMA, 6 BEMA, 6 analysis, sepICA, 37 sepNMF, 38 sigdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, gene, geneNameAssign, 15 geneNameExtractor, 15	sepPCA, 39 * PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 sigdWrapProjection, 43 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15		-
* PVE, pveMultiple, 29 pveSep, 30 * filter * Separate pveSep, 30 * for * Spike BEMA, 6 BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15	* PVE, pveMultiple, 29 pveSep, 30 * Separate pveSep, 30 * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 * extractor sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15		
pveMultiple, 29 pveSep, 30 * filter * Separate pveSep, 30 * for * Spike BEMA, 6 BEMA, 6 sepICA, 37 sepNMF, 38 sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15	pveMultiple, 29 pveSep, 30 * filter * Separate pveSep, 30 * for * Spike BEMA, 6 * analysis, sepICA, 37 sepNMF, 38 sampleNameExtractor, 34 * filter filterNAValue, 13 * for configuration_setting_generation, 12 * gene, geneNameAssign, 15 geneNameExtractor, 15		sjdWrapProjection,43
pveSep, 30	<pre>pveSep, 30 * filter * Separate</pre>	•	
*Separate filterNAValue, 13 pveSep, 30 *for *Spike configuration_setting_generation, BEMA, 6 12 *analysis, sepICA, 37 sepNMF, 38 geneNameAssign, 15 geneNameExtractor, 15	* Separate filterNAValue, 13 pveSep, 30 * for * Spike configuration_setting_generation, BEMA, 6 12 * analysis, sepICA, 37 sepNMF, 38 geneNameAssign, 15 geneNameExtractor, 15		
pveSep, 30	pveSep, 30		
* Spike configuration_setting_generation, BEMA, 6 12 * analysis, sepICA, 37 geneNameAssign, 15 geneNameExtractor, 15	* Spike configuration_setting_generation, BEMA, 6 12 * analysis, sepICA, 37 geneNameAssign, 15 geneNameExtractor, 15	-	•
BEMA, 6 12 * analysis, * gene, sepICA, 37 geneNameAssign, 15 sepNMF, 38 geneNameExtractor, 15	BEMA, 6 12 * analysis, * gene, sepICA, 37 geneNameAssign, 15 sepNMF, 38 geneNameExtractor, 15		
* analysis, * gene, sepICA, 37 geneNameAssign, 15 sepNMF, 38 geneNameExtractor, 15	* analysis, * gene, sepICA, 37 geneNameAssign, 15 sepNMF, 38 geneNameExtractor, 15	-	
sepICA, 37 geneNameAssign, 15 sepNMF, 38 geneNameExtractor, 15	sepICA, 37 geneNameAssign, 15 sepNMF, 38 geneNameExtractor, 15		
sepNMF, 38 geneNameExtractor, 15	sepNMF, 38 geneNameExtractor, 15		_
·	· · ·		
	sepPCA, 39 * generation	·	· · · · · · · · · · · · · · · · · · ·
sepPCA, 39 * generation		sepPCA, 39	* generation

52 INDEX

configuration_setting_generation,	Procrustes, 26
12	* projection,
simulated_data_generation, 40	projectiLCA,27
* genes	projectNMF, 28
getMatch, 17	* quantile
* gene	marchenko_pastur_quantile, 23
geneScreen, 16	* random
sjdWrap, 42	score_generation, 36
sjdWrapProjection, 43	* rank,
* iLCA	twoStageiLCA.rank,45
projectiLCA, 27	twoStageLCA.rank, 48
* images	* rotation,
assemble.byComponent, 3	rotate_component, 31
assemble.byDataset,4	* sample,
SJDScorePlotter, 40	sampleNameAssign, 32
* independent	sampleNameAssignProj, 32
twoStageiLCA, 44	sampleNameAssignSep, 33
* indepentdent,	* score,
twoStageiLCA.rank, 45	scoreNameAssign, 34
* joint,	scoreNameAssignProj, 35
jointNMF, 20	scoreNameAssignSep, 35
jointPCA, 21	
projectNMF, 28	* score score_generation, 36
* matrix	
	* screen,
frameToMatrix, 14	geneScreen, 16
* name,	* separate
sampleNameExtractor, 34	sepICA, 37
* name	sepNMF, 38
compNameAssign, 7	sepPCA, 39
compNameAssignSep, 8	* sequential
datasetNameExtractor, 13	linkedPCA, 23
geneNameAssign, 15	* setting
geneNameExtractor, 15	configuration_setting_generation,
groupNameExtractor, 18	12
sampleNameAssign, 32	* shared
sampleNameAssignProj, 32	getMatch, 17
sampleNameAssignSep, 33	sjdWrap,42
scoreNameAssign, 34	sjdWrapProjection, 43
scoreNameAssignProj,35	* simulated
scoreNameAssignSep, 35	simulated_data_generation, 40
* normalize	* the
normalizeData, 26	configuration_setting_generation,
* numbers	12
BEMA, 6	* two-staged,
* of	twoStageiLCA,44
configuration_setting_generation,	twoStageiLCA.rank, 45
12	twoStageLCA, 47
* pairwise,	${\sf twoStageLCA.rank}, {\sf 48}$
concatICA, 8	* twoStageiLCA,
concatNMF, 9	projectiLCA,27
concatPCA, 11	* weighting
* procrustes	weightData, 50

INDEX 53

addSJDtoSeurat, 3	scoreNameAssignSep, 35
assemble.byComponent,3	ScreePlot_LCvsPC, 37
assemble.byDataset,4	sepICA, 37
	sepNMF, 38
balanceData, 6	sepPCA, 39
BEMA, 6	simulated_data_generation, 40
	SJDScorePlotter, 40
compNameAssign, 7	sjdWrap, 42
compNameAssignSep, 8	sjdWrapProjection, 43
concatICA, 8	Sjuwi apri o jection, 43
concatNMF, 9	twoStageiLCA, 44
concatPCA, 11	twoStageiLCA.rank, 45
configuration_setting_generation, 12	twoStageLCA, 47
55 18a. a 516556518_86 a 516, 12	_
datasetNameExtractor, 13	twoStageLCA.rank,48
,	umap_tsne_onLC, 49
filterNAValue, 13	ullap_tsile_offec, 49
frameToMatrix, 14	weightData, 50
	weightbata, 30
geneNameAssign, 15	
geneNameExtractor, 15	
geneScreen, 16	
getMatch, 17	
groupNameExtractor, 18	
Group Hame Extractor, 10	
jointICA, 18	
jointNMF, 20	
jointPCA, 21	
Joint 67, 21	
linkedPCA, 23	
,	
marchenko_pastur_quantile, 23	
mtx, 24	
NeuroGenesis4,24	
NeuroGenesis4.afterWrap,25	
NeuroGenesis4.info,25	
normalizeData, 26	
Procrustes, 26	
projectiLCA, 27	
projectNMF, 28	
pveMultiple, 29	
pveSep, 30	
p. 1000p, 20	
rebalanceData, 31	
rotate_component, 31	
r · · · · · · · · · · · · · · · · ·	
sampleNameAssign, 32	
sampleNameAssignProj, 32	
sampleNameAssignSep, 33	
sampleNameExtractor, 34	
score_generation, 36	
scoreNameAssign, 34	
scoreNameAssignProj, 35	
JOG CHAIRCAJJIANI I OJ, JJ	