

# Package ‘RCSL’

July 24, 2025

**Type** Package

**Title** Rank Constrained Similarity Learning for single cell RNA sequencing data

**Version** 1.16.0

**Date** 2021-04-01

**Maintainer** Qinglin Mei <meiqinglinkf@163.com>

**Description** A novel clustering algorithm and toolkit RCSL (Rank Constrained Similarity Learning) to accurately identify various cell types using scRNA-seq data from a complex tissue. RCSL considers both local similarity and global similarity among the cells to discern the subtle differences among cells of the same type as well as larger differences among cells of different types. RCSL uses Spearman’s rank correlations of a cell’s expression vector with those of other cells to measure its global similarity, and adaptively learns neighbour representation of a cell as its local similarity. The overall similarity of a cell to other cells is a linear combination of its global similarity and local similarity.

**URL** <https://github.com/QinglinMei/RCSL>

**Depends** R (>= 4.1)

**License** Artistic-2.0

**VignetteBuilder** knitr

**biocViews** SingleCell, Software, Clustering, DimensionReduction, RNASeq, Visualization, Sequencing

**Suggests** testthat, knitr, BiocStyle, rmarkdown, mclust, tidyverse, tinytex

**Imports** RcppAnnoy, igraph, NbClust, Rtsne, ggplot2(>= 3.4.0), methods, pracma, umap, grDevices, graphics, stats, Rcpp (>= 0.11.0), MatrixGenerics, SingleCellExperiment

**LazyData** TRUE

**Encoding** UTF-8

**RoxygenNote** 7.3.1

**NeedsCompilation** no

**Author** Qinglin Mei [cre, aut],  
Guojun Li [fnd],  
Zhengchang Su [fnd]  
**git\_url** https://git.bioconductor.org/packages/RCSL  
**git\_branch** RELEASE\_3\_21  
**git\_last\_commit** 37f108a  
**git\_last\_commit\_date** 2025-04-15  
**Repository** Bioconductor 3.21  
**Date/Publication** 2025-07-23

Contents

|                              |           |
|------------------------------|-----------|
| ann . . . . .                | 2         |
| BDSM . . . . .               | 3         |
| EProjSimplexdiag . . . . .   | 4         |
| EstClusters . . . . .        | 4         |
| EucDist . . . . .            | 5         |
| GenesFilter . . . . .        | 5         |
| getLineage . . . . .         | 6         |
| NeigRepresent . . . . .      | 6         |
| PlotMST . . . . .            | 7         |
| PlotPseudoTime . . . . .     | 8         |
| PlotTrajectory . . . . .     | 9         |
| RCSL . . . . .               | 10        |
| SimS . . . . .               | 11        |
| TrajectoryAnalysis . . . . . | 12        |
| yan . . . . .                | 14        |
| <b>Index</b>                 | <b>15</b> |

---

|     |                                                              |
|-----|--------------------------------------------------------------|
| ann | <i>Cell type annotations of ‘yan’ datasets by Yan et al.</i> |
|-----|--------------------------------------------------------------|

---

Description

Cell type annotations of ‘yan’ datasets by Yan et al.

Usage

ann

Format

An object of class data.frame with 90 rows and 1 columns.

**Source**

<http://dx.doi.org/10.1038/nsmb.2660>

Each row corresponds to one cell of 'yan' dataset

---

|      |                                                                                                                                                                                    |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BDSM | <i>Calculate the bolock-diagnal matrix <math>B</math> <math>\min_B \geq 0, B * I = I, F' * F = I \  B - A \ _1 + r * \  B \ ^2 + 2 * \lambda * \text{trace}(F' * L * F)</math></i> |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

---

**Description**

Calculate the bolock-diagnal matrix  $B$   $\min_B \geq 0, B * I = I, F' * F = I \| B - A \|_1 + r * \| B \|^2 + 2 * \lambda * \text{trace}(F' * L * F)$

**Usage**

BDSM(S, C)

**Arguments**

S                      the calculated initial similarity matrix S

C                      the estimated number of clusters C

**Value**

B block-diagonal matrix

y clustering results

**Examples**

```
gfData <- GenesFilter(yan)
res_SimS <- SimS(gfData)
C <- EstClusters(res_SimS$drData, res_SimS$S)
BDSM(res_SimS$S, C)
```

---

|                  |                                                                                                           |
|------------------|-----------------------------------------------------------------------------------------------------------|
| EProjSimplexdiag | <i>Solve the problem: <math>\min 1/2 * x' * L * x - x' * d</math> s.t. <math>x \geq 0, 1'x = 1</math></i> |
|------------------|-----------------------------------------------------------------------------------------------------------|

---

**Description**

Solve the problem:  $\min 1/2 * x' * L * x - x' * d$  s.t.  $x \geq 0, 1'x = 1$

**Usage**

```
EProjSimplexdiag(d, l)
```

**Arguments**

|   |                  |
|---|------------------|
| d | matrix or vector |
| l | matrix or vector |

**Value**

x

---

|             |                                                                 |
|-------------|-----------------------------------------------------------------|
| EstClusters | <i>Estimate the optimal number of clusters C for clustering</i> |
|-------------|-----------------------------------------------------------------|

---

**Description**

Estimate the optimal number of clusters C for clustering

**Usage**

```
EstClusters(drData, S)
```

**Arguments**

|        |                                                |
|--------|------------------------------------------------|
| drData | gene expression matrix after PCA processing    |
| S      | the calculated similarity matrix S from "SimS" |

**Value**

C the estimated number of clusters

**Examples**

```
gfData <- GenesFilter(yan)
res_SimS <- SimS(gfData)
EstClusters(res_SimS$drData, res_SimS$S)
```

---

|         |                                                                               |
|---------|-------------------------------------------------------------------------------|
| EucDist | <i>Solve the problem: <math>\ A-B\ ^2 = \ A\ ^2 + \ B\ ^2 - 2*A'*B</math></i> |
|---------|-------------------------------------------------------------------------------|

---

**Description**

Solve the problem:  $\|A-B\|^2 = \|A\|^2 + \|B\|^2 - 2*A'*B$

**Usage**

EucDist(A, B)

**Arguments**

|   |                  |
|---|------------------|
| A | matrix or vector |
| B | matrix or vector |

**Value**

d matrix or vector

---

|             |                                                                               |
|-------------|-------------------------------------------------------------------------------|
| GenesFilter | <i>Perform the step of gene filtering to normalizaed gene expression data</i> |
|-------------|-------------------------------------------------------------------------------|

---

**Description**

Perform the step of gene filtering to normalizaed gene expression data

**Usage**

GenesFilter(data, gfRatio = 0.025)

**Arguments**

|         |                                       |
|---------|---------------------------------------|
| data    | the normalized gene expression matrix |
| gfRatio | the ratio of genes filtering          |

**Value**

the gene expression matrix after genes filtering gfData

**Examples**

```
data(yan)
GenesFilter(yan)
```

---

|            |                                                                                                   |
|------------|---------------------------------------------------------------------------------------------------|
| getLineage | <i>Infer the development lineage based on the clustering results from RCSL and the pseudotime</i> |
|------------|---------------------------------------------------------------------------------------------------|

---

### Description

Infer the development lineage based on the clustering results from RCSL and the pseudotime

### Usage

```
getLineage(drData, clustRes, pseudoTime, simMeasure = "kendall")
```

### Arguments

|            |                                                                              |
|------------|------------------------------------------------------------------------------|
| drData     | preprocessed gene expression data (each column represent a cell)             |
| clustRes   | the clustering results identified by RCSL                                    |
| pseudoTime | inferred by PlotPseudoTime() using the similarity matrix S and starting cell |
| simMeasure | the calculation method of measuring the cluster centers' similarity          |

### Value

lineage the cell lineages connected all the cluster centers based on the clustering results from RCSL

### Examples

```
gfData <- GenesFilter(yan)
TrueLabel <- ann$cell_type1
res_SimS <- SimS(gfData)
C <- EstClusters(res_SimS$drData, res_SimS$S)
res_BDSM <- BDSM(res_SimS$S, C)
Pseudo <- PlotPseudoTime(res_SimS$S, TrueLabel, startPoint=1)
getLineage(res_SimS$drData, res_BDSM$y, Pseudo$pseudoTime)
```

---

|               |                                                                                                     |
|---------------|-----------------------------------------------------------------------------------------------------|
| NeigRepresent | <i>Calculate the neighbor representation of cells to the low-dimensional gene expression matrix</i> |
|---------------|-----------------------------------------------------------------------------------------------------|

---

### Description

Calculate the neighbor representation of cells to the low-dimensional gene expression matrix

**Usage**

```
NeigRepresent(
  drData,
  NN.method = "KNN",
  Dis.method = "Euclidean",
  LSH.TreeNum = 30,
  LSH.Dim = 500,
  LSH.Dis = "angular",
  neiRatio = 0.65
)
```

**Arguments**

|             |                                                            |
|-------------|------------------------------------------------------------|
| drData      | gene expression matrix after dimensionality reduced by PCA |
| NN.method   | the method of finding neighbors                            |
| Dis.method  | the distance metric in finding neighbors                   |
| LSH.TreeNum | the tree number of LSH                                     |
| LSH.Dim     | the dimension in LSH                                       |
| LSH.Dis     | the distance metric in LSH                                 |
| neiRatio    | ratio of the number of selected                            |

**Value**

the similarity matrix measured by neighbor representation NR

**Examples**

```
gfData <- GenesFilter(yan)
res_SimS <- SimS(gfData)
NeigRepresent(res_SimS$drData)
```

---

PlotMST

---

*Plot the visualization of constructed Minimum Spanning Tree based on the clustering results of RCSL*


---

**Description**

Plot the visualization of constructed Minimum Spanning Tree based on the clustering results of RCSL

**Usage**

```
PlotMST(
  drData,
  clustRes,
  TrueLabel,
  dataName = "",
  fontSize = 12,
  VisualMethod = "umap"
)
```

**Arguments**

|                           |                                                              |
|---------------------------|--------------------------------------------------------------|
| <code>drData</code>       | preprocessed gene expression data                            |
| <code>clustRes</code>     | the clustering results identified by RCSL                    |
| <code>TrueLabel</code>    | the real cell types to color the dots in plot                |
| <code>dataName</code>     | the name of the data that will be showed in the plot         |
| <code>fontSize</code>     | the font size of the plot                                    |
| <code>VisualMethod</code> | the method for 2D visualization including UMAP,t-SNE and PCA |

**Value**

MSTPlot ggplot object of the visualization of constructed MST

**Examples**

```
gfData <- GenesFilter(yan)
TrueLabel <- ann$cell_type1
res_SimS <- SimS(gfData)
C <- EstClusters(res_SimS$drData,res_SimS$S)
res_BDSM <- BDSM(res_SimS$S,C)
PlotMST(res_SimS$drData,res_BDSM$y,TrueLabel)
```

---

|                |                                                                                                                                            |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| PlotPseudoTime | <i>Infer the pseudo-temporal ordering between the cell types using the distance from a cell type to the predefined starting cell type.</i> |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------|

---

**Description**

Infer the pseudo-temporal ordering between the cell types using the distance from a cell type to the predefined starting cell type.

**Usage**

```
PlotPseudoTime(
  S,
  TrueLabel,
  startPoint,
  fontSize = 12,
  dataName = "",
  sim = TRUE
)
```

**Arguments**

|                         |                                                                  |
|-------------------------|------------------------------------------------------------------|
| <code>S</code>          | the similarity matrix calculated by <code>SimS()</code> function |
| <code>TrueLabel</code>  | the real cell types used to indicate the vertical axis           |
| <code>startPoint</code> | the position of the starting cell in the matrix                  |
| <code>fontSize</code>   | the font size of the plot                                        |
| <code>dataName</code>   | the name of the data that will be showed in the plot             |
| <code>sim</code>        | indicate the input data is similarity matrix or not              |

**Value**

`PseudoTime`

`PseudoTimePlot` ggplot object of the pseudo-temporal ordering of cells

**Examples**

```
gfData <- GenesFilter(yan)
TrueLabel <- ann$cell_type1
res_SimS <- SimS(gfData)
PlotPseudoTime(res_SimS$S, TrueLabel, startPoint=1)
```

---

|                |                                                                                       |
|----------------|---------------------------------------------------------------------------------------|
| PlotTrajectory | <i>Infer the developmental trajectories based on the clustering results from RCSL</i> |
|----------------|---------------------------------------------------------------------------------------|

---

**Description**

Infer the developmental trajectories based on the clustering results from RCSL

**Usage**

```
PlotTrajectory(
  gfData,
  clustRes,
  TrueLabel,
  lineage,
  fontSize = 12,
  dataName = "",
  VisualMethod = "umap"
)
```

**Arguments**

|              |                                                                  |
|--------------|------------------------------------------------------------------|
| gfData       | preprocessed gene expression data (each column represent a cell) |
| clustRes     | the clustering results identified by RCSL                        |
| TrueLabel    | the real cell types                                              |
| lineage      | the lineage obtained by getLineage()                             |
| fontSize     | the size of font in the plot                                     |
| dataName     | the name of the data that will be showed in the plot             |
| VisualMethod | the display method of 2-D visualization                          |

**Value**

TrajectoryPlot ggplot object of the inferred developmental trajectories

**Examples**

```
gfData <- GenesFilter(yan)
TrueLabel <- ann$cell_type1
res_SimS <- SimS(gfData)
C <- EstClusters(res_SimS$drData, res_SimS$S)
res_BDSM <- BDSM(res_SimS$S, C)
Pseudo <- PlotPseudoTime(res_SimS$S, TrueLabel, startPoint=1)
Linea <- getLineage(res_SimS$drData, res_BDSM$y, Pseudo$pseudoTime)
PlotTrajectory(gfData, res_BDSM$y, TrueLabel, lineage=Linea)
```

---

RCSL

---

*Perform the RCSL program*


---

**Description**

Perform the RCSL program

Usage

```
RCSL(  
  data,  
  GF = TRUE,  
  gfRatio = 0.025,  
  pcRatio = 0.95,  
  NN.method = "KNN",  
  Dis.method = "Euclidean",  
  neiRatio = 0.65  
)
```

Arguments

|            |                                                                   |
|------------|-------------------------------------------------------------------|
| data       | normalizaed gene expression matrix(each column represents a cell) |
| GF         | should I need the gene filter step?                               |
| gfRatio    | the ratio of the gene filter                                      |
| pcRatio    | the ratio between the variance of the                             |
| NN.method  | the method of finding neighbors                                   |
| Dis.method | the distance metric in finding neighbors                          |
| neiRatio   | ratio of the number of selected                                   |

Value

- gfData gene expression matrix after genes filtering
- B block-diagonal matrix
- C estimated number of clusters
- y clustering results

Examples

```
data(yan)  
data <- log2(yan+1)  
RCSL(yan[,1:20])
```

---

|      |                                                |
|------|------------------------------------------------|
| SimS | <i>Calculate the initial similarity matrix</i> |
|------|------------------------------------------------|

---

Description

Calculate the initial similarity matrix

Usage

```
SimS(  
  data,  
  pcRatio = 0.95,  
  gamma = 0.8,  
  NN.method = "KNN",  
  Dis.method = "Euclidean",  
  LSH.TreeNum = 30,  
  LSH.Dim = 1000,  
  LSH.Dis = "angular",  
  neiRatio = 0.65  
)
```

Arguments

|             |                                              |
|-------------|----------------------------------------------|
| data        | gene expression matrix after genes filtering |
| pcRatio     | the ratio between the variance of the        |
| gamma       | the ratio of the global simialrity           |
| NN.method   | the method of finding neighbors              |
| Dis.method  | the distance metric in finding neighbors     |
| LSH.TreeNum | the tree number of LSH                       |
| LSH.Dim     | the dimension in LSH                         |
| LSH.Dis     | the distance metric in LSH                   |
| neiRatio    | ratio of the number of selected              |

Value

initial similarity matrix S  
gene expression matrix after PCA processing drData

Examples

```
gfData <- GenesFilter(yan)  
SimS(gfData)
```

---

|                    |                            |
|--------------------|----------------------------|
| TrajectoryAnalysis | <i>Trajectory analysis</i> |
|--------------------|----------------------------|

---

Description

Trajectory analysis

**Usage**

```
TrajectoryAnalysis(
  gfData,
  drData,
  S,
  clustRes,
  fontSize = 12,
  TrueLabel,
  startPoint,
  dataName = "",
  sim = TRUE,
  simMeasure = "kendall",
  VisualMethod = "umap"
)
```

**Arguments**

|              |                                                                     |
|--------------|---------------------------------------------------------------------|
| gfData       | preprocessed gene expression data (each column represent a cell)    |
| drData       | preprocessed gene expression data (each column represent a cell)    |
| S            | the similarity matrix calculated by SimS() function                 |
| clustRes     | the clustering results identified by RCSL                           |
| fontSize     | the size of font in the plot                                        |
| TrueLabel    | the real cell types used to indicate the vertical axis              |
| startPoint   | the position of the starting cell in the matrix                     |
| dataName     | the name of the data that will be showed in the plot                |
| sim          | indicate the input data is simialrity matrix or not                 |
| simMeasure   | the calculation method of measuring the cluster centers' similarity |
| VisualMethod | the display method of 2-D visualization                             |

**Value**

PseudoTimePlot, MSTPlot, TrajectoryPlot

**Examples**

```
gfData <- GenesFilter(yan)
TrueLabel <- ann$cell_type1
res_SimS <- SimS(gfData)
C <- EstClusters(res_SimS$drData, res_SimS$S)
res_BDSM <- BDSM(res_SimS$S, C)
TrajectoryAnalysis(gfData, res_SimS$drData, res_SimS$S, res_BDSM$y,
  TrueLabel=TrueLabel, startPoint=1)
```

---

yan

*A public scRNA-seq dataset by Yan et al.*

---

**Description**

A public scRNA-seq dataset by Yan et al.

**Usage**

yan

**Format**

An object of class `data.frame` with 20214 rows and 90 columns.

**Source**

<http://dx.doi.org/10.1038/nsmb.2660>

Columns represent cells, rows represent genes expression values.

# Index

## \* datasets

ann, [2](#)  
yan, [14](#)

ann, [2](#)

BDSM, [3](#)

EProjSimplexdiag, [4](#)

EstClusters, [4](#)

EucDist, [5](#)

GenesFilter, [5](#)

getLineage, [6](#)

NeigRepresent, [6](#)

PlotMST, [7](#)

PlotPseudoTime, [8](#)

PlotTrajectory, [9](#)

RCSL, [10](#)

SimS, [11](#)

TrajectoryAnalysis, [12](#)

yan, [14](#)