

# Package ‘netSmooth’

April 16, 2024

**Type** Package

**Title** Network smoothing for scRNAseq

**Version** 1.22.0

**Description** netSmooth is an R package for network smoothing of single cell RNA sequencing data. Using bio networks such as protein-protein interactions as priors for gene co-expression, netsmooth improves cell type identification from noisy, sparse scRNAseq data.

**biocViews** Network, GraphAndNetwork, SingleCell, RNASeq, GeneExpression, Sequencing, Transcriptomics, Normalization, Preprocessing, Clustering, DimensionReduction

**URL** <https://github.com/BIMSBbioinfo/netSmooth>

**BugReports** <https://github.com/BIMSBbioinfo/netSmooth/issues>

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.5), scater (>= 1.15.11), clusterExperiment (>= 2.1.6)

**Imports** entropy, SummarizedExperiment, SingleCellExperiment, Matrix, cluster, data.table, stats, methods, DelayedArray, HDF5Array (>= 1.15.13)

**Suggests** knitr, testthat, Rtsne, biomaRt, igraph, STRINGdb, NMI, pheatmap, ggplot2, BiocStyle, rmarkdown, BiocParallel, uwot

**VignetteBuilder** knitr

**RoxygenNote** 7.0.2

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

**git\_branch** RELEASE\_3\_18

**git\_last\_commit** d9163f4

**git\_last\_commit\_date** 2023-10-24

**Repository** Bioconductor 3.18

**Date/Publication** 2024-04-15

**Author** Jonathan Ronen [aut, cre],  
Altuna Akalin [aut]

**Maintainer** Jonathan Ronen <yablee@gmail.com>

## R topics documented:

calc2DEntropy	2
clusterExperimentWorkflow	3
clusterOne	4
dimReduce	4
human.ppi	5
l1NormalizeColumns	5
l1NormalizeRows	6
mouse.ppi	6
netSmooth,matrix-method	7
pickDimReduction,matrix-method	9
projectFromNetworkRecombine,matrix-method	10
projectOnNetwork,matrix-method	10
randomWalkByIterations	11
randomWalkByMatrixInv,matrix-method	12
randomWalkBySolve,matrix-method	12
robustClusters,SummarizedExperiment-method	13
scoreSmoothing	14
smallPPI	14
smallscRNAseq	15
smoothAndRecombine,matrix-method	15
<b>Index</b>	<b>17</b>

---

calc2DEntropy	<i>Calculate entropy in 2D data</i>
---------------	-------------------------------------

---

### Description

Calculate entropy in 2D data

### Usage

```
calc2DEntropy(x, numBins1 = 20, numBins2 = 20)
```

### Arguments

x	the 2D data to get entropy from
numBins1	the number of bins along the first dimension to discretize data into
numBins2	the number of bins along the second dimension to discretize data into

**Value**

The Shannon entropy in the 2D data x

---

clusterExperimentWorkflow

*Performs clustering workflow using 'clusterExperiment' functions*

---

**Description**

Performs clustering workflow using 'clusterExperiment' functions

**Usage**

```
clusterExperimentWorkflow(
  se,
  dimReduceFlavor = c("pca", "tsne", "dm", "umap"),
  cluster.ks = 5:10,
  cluster.function = "pam",
  nVarDims = c(100, 500, 1000),
  makeConsensusProportion = 0.7,
  makeConsensusMinSize = 4,
  runMergeClusters = TRUE,
  is.counts = TRUE,
  random.seed = 1
)
```

**Arguments**

se	SummarizedExperiment object
dimReduceFlavor	algorithm for reduced dimension embedding step
cluster.ks	range of Ks to cluster over
cluster.function	clustering algorithm to use for all clusterings
nVarDims	numbers of variable genes to perform clusterings over
makeConsensusProportion	proportion of times samples need to be co-clustered for co-clustering step
makeConsensusMinSize	minimum cluster size
runMergeClusters	logical: merge similar clusters
is.counts	logical: is data counts
random.seed	passed to clusterExperiment. set to NULL in order to not set a random seed.

**Value**

cluster assignments

---

clusterOne	<i>Run one clustering using kmeans o PAM</i>
------------	--

---

**Description**

Run one clustering using kmeans o PAM

**Usage**

```
clusterOne(x, algorithm = c("kmeans", "pam"), k = 5)
```

**Value**

kmeans or PAM cluster assignments

---

dimReduce	<i>Get lower dimension embedding</i>
-----------	--------------------------------------

---

**Description**

Get lower dimension embedding

**Usage**

```
dimReduce(
  x,
  flavor = c("pca", "tsne", "umap"),
  k = 2,
  is.counts = TRUE,
  ntop = 500
)
```

**Arguments**

x	gene expresion matrix [GENES x SAMPLES]
flavor	the algorithm to use to obtain the dimensionality reduction must be in c('pca', 'tsne', 'umap')
k	the number of dimensions in the reduced dimension representation
is.counts	logical: is 'x' counts data
ntop	number of most variable genes to use for dimensionality reduction

**Value**

reduced dimensionality representation

---

human.ppi	<i>Human Protein-Protein interaction graph</i>
-----------	--

---

**Description**

An adjacency matrix of the 10 percent highest confidence interactions between human proteins on STRINGdb.

**Usage**

```
human.ppi
```

**Format**

A square matrix where  $A_{ij}=1$  if gene  $i$  interacts with gene  $j$

**Details**

See the script in `'system.file(package="netSmooth", "data-raw", "make_ppi_from_string.R")'` for full details of how this object was made.

**Source**

<http://www.string-db.org/>

---

l1NormalizeColumns	<i>Column-normalize a sparse, symmetric matrix (using the l1 norm) so that each column sums to 1.</i>
--------------------	---

---

**Description**

Column-normalize a sparse, symmetric matrix (using the l1 norm) so that each column sums to 1.

**Usage**

```
l1NormalizeColumns(A)
```

**Arguments**

A                    matrix

**Value**

column-normalized sparse matrix object

`l1NormalizeRows`      *Row-normalize a sparse, symmetric matrix (using the l1 norm) so that each row sums to 1.*

---

**Description**

Row-normalize a sparse, symmetric matrix (using the l1 norm) so that each row sums to 1.

**Usage**

```
l1NormalizeRows(A)
```

**Arguments**

A                      matrix

**Value**

row-normalized sparse matrix object

---

`mouse.ppi`              *Mouse Protein-Protein interaction graph*

---

**Description**

An adjacency matrix of the 10 percent highest confidence interactions between mouse proteins on STRINGdb.

**Usage**

```
mouse.ppi
```

**Format**

A square matrix where  $A_{ij}=1$  if gene  $i$  interacts with gene  $j$

**Details**

See the script in `'system.file(package="netSmooth", "data-raw", "make_ppi_from_string.R")'` for full details of how this object was made.

**Source**

<http://www.string-db.org/>

---

`netSmooth,matrix-method`*Perform network smoothing of gene expression or other omics data*

---

**Description**

Perform network smoothing of gene expression or other omics data

**Usage**

```
## S4 method for signature 'matrix'
netSmooth(
  x,
  adjMatrix,
  alpha = "auto",
  normalizeAdjMatrix = c("rows", "columns"),
  autoAlphaMethod = c("robustness", "entropy"),
  autoAlphaRange = 0.1 * (seq_len(9)),
  autoAlphaDimReduceFlavor = "auto",
  is.counts = TRUE,
  bpparam = BiocParallel::SerialParam(),
  ...
)

## S4 method for signature 'SummarizedExperiment'
netSmooth(x, ...)

## S4 method for signature 'SingleCellExperiment'
netSmooth(x, ...)

## S4 method for signature 'Matrix'
netSmooth(
  x,
  adjMatrix,
  alpha = "auto",
  normalizeAdjMatrix = c("rows", "columns"),
  autoAlphaMethod = c("robustness", "entropy"),
  autoAlphaRange = 0.1 * (seq_len(9)),
  autoAlphaDimReduceFlavor = "auto",
  is.counts = TRUE,
  bpparam = BiocParallel::SerialParam(),
  ...
)

## S4 method for signature 'DelayedMatrix'
netSmooth(
  x,
```

```

adjMatrix,
alpha = "auto",
normalizeAdjMatrix = c("rows", "columns"),
autoAlphaMethod = c("robustness", "entropy"),
autoAlphaRange = 0.1 * (seq_len(9)),
autoAlphaDimReduceFlavor = "auto",
is.counts = TRUE,
bpparam = BiocParallel::SerialParam(),
filepath = NULL,
...
)

```

### Arguments

x	matrix or SummarizedExperiment
adjMatrix	adjacency matrix of gene network to use
alpha	numeric in [0,1] or 'auto'. if 'auto', the optimal value for alpha will be automatically chosen among the values specified in 'autoAlphaRange', using the strategy specified in 'autoAlphaMethod'
normalizeAdjMatrix	how to normalize the adjacency matrix possible values are 'rows' (in-degree) and 'columns' (out-degree)
autoAlphaMethod	if 'robustness', pick alpha that gives the highest proportion of samples in robust clusters if 'entropy', pick alpha that gives highest Shannon entropy in 2D PCA embedding
autoAlphaRange	if 'alpha=optimal', search these values for the best alpha
autoAlphaDimReduceFlavor	algorithm for dimensionality reduction that will be used to pick the optimal value for alpha. Either the 2D embedding to calculate the Shannon entropy for (if 'autoAlphaMethod=entropy'), or the dimensionality reduction algorithm to be used in robust clustering (if 'autoAlphaMethod=robustness')
is.counts	logical: is the assay count data
bpparam	instance of bpparam, for parallel computation with the 'alpha=auto' option. See the BiocParallel manual.
...	arguments passed on to 'robustClusters' if using the robustness criterion for optimizing alpha
filepath	String: Path to location where hdf5 output file is supposed to be saved. Will be ignored when regular matrices or SummarizedExperiment are used as input.

### Value

network-smoothed gene expression matrix or SummarizedExperiment object



**Examples**

```
x <- matrix(rnbinom(12000, size=1, prob = .1), ncol=60)
rownames(x) <- paste0('gene', seq_len(dim(x)[1]))

adj_matrix <- matrix(as.numeric(rnorm(200*200)>.8), ncol=200)
rownames(adj_matrix) <- colnames(adj_matrix) <- paste0('gene', seq_len(dim(x)[1]))
x.smoothed <- netSmooth(x, adj_matrix, alpha=0.5)
```

---

```
pickDimReduction,matrix-method
```

*Pick the dimensionality reduction method for a dataset that gives the 2D embedding with the highest entropy*

---

**Description**

Pick the dimensionality reduction method for a dataset that gives the 2D embedding with the highest entropy

**Usage**

```
## S4 method for signature 'matrix'
pickDimReduction(x, flavors = c("pca", "tsne", "umap"), is.counts = TRUE)

## S4 method for signature 'SummarizedExperiment'
pickDimReduction(x)

## S4 method for signature 'Matrix'
pickDimReduction(x, flavors = c("pca", "tsne", "umap"), is.counts = TRUE)

## S4 method for signature 'DelayedMatrix'
pickDimReduction(x, flavors = c("pca", "tsne", "umap"), is.counts = TRUE)
```

**Arguments**

x	matrix or SummarizedExperiment object [GENES x SAMPLES]
flavors	list of dimensionality reduction algorithms to try. Currently the options are "pca", "tsne" and "umap"
is.counts	logical: is exprs count data

**Value**

name of dimensionality reduction method that gives the highest 2d entropy

**Examples**

```
x <- matrix(rnbinom(60000, size=1, prob = .1), ncol=100)
pickDimReduction(x)
```

---

```
projectFromNetworkRecombine,matrix-method
```

*Combine gene expression from smoothed space (that of the network) with the expression of genes that were not smoothed (not present in network)*

---

### Description

Combine gene expression from smoothed space (that of the network) with the expression of genes that were not smoothed (not present in network)

### Usage

```
## S4 method for signature 'matrix'
projectFromNetworkRecombine(original_expression, smoothed_expression)
```

### Arguments

```
original_expression
    the non-smoothed expression

smoothed_expression
    the smoothed gene expression, in the space of the genes defined by the network

filepath
    String: Path to location where hdf5 output file is supposed to be saved. Will be ignored when regular matrices or SummarizedExperiment are used as input.
```

### Value

a matrix in the dimensions of original\_expression, where values that are present in smoothed\_expression are copied from there.

---

```
projectOnNetwork,matrix-method
```

*Project the gene expression matrix onto a lower space of the genes defined in the smoothing network*

---

### Description

Project the gene expression matrix onto a lower space of the genes defined in the smoothing network

### Usage

```
## S4 method for signature 'matrix'
projectOnNetwork(gene_expression, new_features, missing.value = 0)
```

**Arguments**

gene_expression	gene expression matrix
new_features	the genes in the network, on which to project the gene expression matrix
missing.value	value to assign to genes that are in network, but missing from gene expression matrix

**Value**

the gene expression matrix projected onto the gene space defined by new\_features

---

randomWalkByIterations

*Smooth data on graph by computing iterations*

---

**Description**

Smooth data on graph by computing iterations

**Usage**

```
randomWalkByIterations(  
  f0,  
  adjMatrix,  
  alpha,  
  normalizeAdjMatrix = c("rows", "columns"),  
  tol = 1e-06,  
  max.iter = 100  
)
```

**Arguments**

f0	initial data matrix [NxM]
adjMatrix	adjacency matrix of graph to network smooth on will be column-normalized.
alpha	smoothing coefficient (1 - restart probability of random walk)
tol	the tolerance (stopping criterion)
max.iter	the maximum number of iterations before terminating

**Value**

network-smoothed gene expression

---

```
randomWalkByMatrixInv,matrix-method
```

*Smooth data on graph by computing the closed-form steady state distribution of the random walk with restarts process.*

---

### Description

The closed-form solution is given by  $f_{ss} = (1 - \alpha) * (I - \alpha * A)^{-1} * f_0$  and is computed by matrix inversion in this function.

### Usage

```
## S4 method for signature 'matrix'
randomWalkByMatrixInv(
  f0,
  adjMatrix,
  alpha,
  normalizeAdjMatrix = c("rows", "columns")
)
```

### Arguments

f0	initial data matrix [NxM]
adjMatrix	adjacency matrix of graph to network smooth on will be column-normalized.
alpha	smoothing coefficient (1 - restart probability of random walk)

### Value

network-smoothed gene expression

---

```
randomWalkBySolve,matrix-method
```

*Smooth data on graph by solving the linear equation  $(I - \alpha * A) * E_{sm} = E * (1 - \alpha)$*

---

### Description

Smooth data on graph by solving the linear equation  $(I - \alpha * A) * E_{sm} = E * (1 - \alpha)$

### Usage

```
## S4 method for signature 'matrix'
randomWalkBySolve(E, A, alpha, normalizeAjdMatrix = c("rows", "columns"))
```

**Arguments**

E	initial data matrix [NxM]
A	adjacency matrix of graph to network smooth on will be column-normalized.
alpha	smoothing coefficient (1 - restart probability of random walk)

**Value**

network-smoothed gene expression

---

robustClusters, SummarizedExperiment-method

*Perform robust clustering on dataset, and calculate the proportion of samples in robust clusters*

---

**Description**

Perform robust clustering on dataset, and calculate the proportion of samples in robust clusters

**Usage**

```
## S4 method for signature 'SummarizedExperiment'
robustClusters(x, dimReduceFlavor = "auto", is.counts = TRUE, ...)
```

```
## S4 method for signature 'matrix'
robustClusters(x, ...)
```

**Arguments**

x	matrix or SummarizedExperiment object
dimReduceFlavor	algorithm for dimensionality reduction step of clustering procedure. May be 'pca', 'tsne', 'dm', 'umap' or 'auto', which uses shannon entropy to pick the algorithm.
is.counts	logical: is the data counts
...	arguments passed on to 'clusterExperimentWorkflow'

**Value**

list(clusters, proportion.robust)

**Examples**

```
data("smallscRNAseq")
robustClusters(smallscRNAseq, dimReduceFlavor='pca')
```

---

scoreSmoothing	<i>Calculate a score for a smoothing result, for picking the best alpha value</i>
----------------	---

---

**Description**

Calculate a score for a smoothing result, for picking the best alpha value

**Usage**

```
scoreSmoothing(x, method = c("entropy", "robustness"), is.counts = TRUE, ...)
```

**Arguments**

x	the network-smoothed expression matrix
method	the scoring method. 'entropy' calculates shannon entropy in a 2D PCA of the data. 'robustness' performs robsut clustering and reports the proportion of samples in robust clusters

**Value**

the score

---

smallPPI	<i>A small human Protein-Protein interaction graph for use in examples.</i>
----------	---

---

**Description**

Contains a synthetic PPI of human genes.

**Usage**

```
smallPPI
```

**Format**

An object of class `matrix` with 611 rows and 611 columns.

---

`smallscRNAseq`*A small single cell RNA-seq dataset for use in examples.*

---

**Description**

Contains scRNAseq profiles of human blastomeres.

**Usage**

```
smallscRNAseq
```

**Format**

```
SingleCellExperiment
```

**Source**

```
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE44183
```

---

`smoothAndRecombine, matrix-method`*Perform network smoothing on network when the network genes and the experiment genes aren't exactly the same.*

---

**Description**

The gene network might be defined only on a subset of genes that are measured in any experiment. Further, an experiment might not measure all genes that are present in the network. This function projects the experiment data onto the gene space defined by the network prior to smoothing. Then, it projects the smoothed data back into the original dimensions.

**Usage**

```
## S4 method for signature 'matrix'
smoothAndRecombine(
  gene_expression,
  adj_matrix,
  alpha,
  smoothing.function = randomWalkBySolve,
  normalizeAdjMatrix = c("rows", "columns")
)
```

**Arguments**

<code>gene_expression</code>	gene expression data to be smoothed [N_genes x M_samples]
<code>adj_matrix</code>	adjacency matrix of network to perform smoothing over. Will be column-normalized. Rownames and colnames should be genes.
<code>alpha</code>	network smoothing parameter (1 - restart probability in random walk model).
<code>smoothing.function</code>	must be a function that takes in data, adjacency matrix, and alpha. Will be used to perform the actual smoothing.
<code>normalizeAdjMatrix</code>	which dimension (rows or columns) should the adjacency matrix be normalized by. rows corresponds to in-degree, columns to out-degree.
<code>filepath</code>	String: Path to location where hdf5 output file is supposed to be saved. Will be ignored when regular matrices or SummarizedExperiment are used as input.

**Value**

matrix with network-smoothed gene expression data. Genes that are not present in smoothing network will retain original values.



# Index

- \* **datasets**
  - human.ppi, [5](#)
  - mouse.ppi, [6](#)
  - smallPPI, [14](#)
  - smallscRNAseq, [15](#)
- \* **internal**
  - calc2DEntropy, [2](#)
  - clusterExperimentWorkflow, [3](#)
  - clusterOne, [4](#)
  - dimReduce, [4](#)
  - l1NormalizeColumns, [5](#)
  - l1NormalizeRows, [6](#)
  - projectFromNetworkRecombine, matrix-method, [10](#)
  - projectOnNetwork, matrix-method, [10](#)
  - randomWalkByIterations, [11](#)
  - randomWalkByMatrixInv, matrix-method, [12](#)
  - randomWalkBySolve, matrix-method, [12](#)
  - scoreSmoothing, [14](#)
  - smoothAndRecombine, matrix-method, [15](#)
- calc2DEntropy, [2](#)
- clusterExperimentWorkflow, [3](#)
- clusterOne, [4](#)
- dimReduce, [4](#)
- human.ppi, [5](#)
- l1NormalizeColumns, [5](#)
- l1NormalizeRows, [6](#)
- mouse.ppi, [6](#)
- netSmooth (netSmooth, matrix-method), [7](#)
- netSmooth, DelayedMatrix-method (netSmooth, matrix-method), [7](#)
- netSmooth, Matrix-method (netSmooth, matrix-method), [7](#)
- netSmooth, matrix-method, [7](#)
- netSmooth, SingleCellExperiment-method (netSmooth, matrix-method), [7](#)
- netSmooth, SummarizedExperiment-method (netSmooth, matrix-method), [7](#)
- pickDimReduction (pickDimReduction, matrix-method), [9](#)
- pickDimReduction, DelayedMatrix-method (pickDimReduction, matrix-method), [9](#)
- pickDimReduction, Matrix-method (pickDimReduction, matrix-method), [9](#)
- pickDimReduction, matrix-method, [9](#)
- pickDimReduction, SummarizedExperiment-method (pickDimReduction, matrix-method), [9](#)
- projectFromNetworkRecombine, matrix-method, [10](#)
- projectOnNetwork (projectOnNetwork, matrix-method), [10](#)
- projectOnNetwork, matrix-method, [10](#)
- randomWalkByIterations, [11](#)
- randomWalkByMatrixInv, matrix-method, [12](#)
- randomWalkBySolve, matrix-method, [12](#)
- robustClusters (robustClusters, SummarizedExperiment-method), [13](#)
- robustClusters, matrix-method (robustClusters, SummarizedExperiment-method), [13](#)
- robustClusters, SummarizedExperiment-method, [13](#)

scoreSmoothing, [14](#)  
smallPPI, [14](#)  
smallscRNAseq, [15](#)  
smoothAndRecombine, matrix-method, [15](#)