Package ‘roots’

July 11, 2017

Title Reconstructing Ordered Ontogenic Trajectories
Version 1.0
Description A set of tools to reconstruct ordered ontogenic trajectories from single cell RNAseq data.
Depends R (>= 3.0)
Imports animation (>= 2.4), rARPACK (>= 0.11-0), igraph (>= 1.0.0)
License GPL (>= 2)
Encoding UTF-8
LazyData true
RoxygenNote 6.0.1
Suggests knitr, rmarkdown
VignetteBuilder knitr
NeedsCompilation no
Author Wajid Jawaid [aut, cre]
Maintainer Wajid Jawaid <wjRT1@camNacNuk^>
Repository CRAN
Date/Publication 2017-07-11 16:52:20 UTC

R topics documented:

  animPlot ............................................................ 2
  animPlotGif .......................................................... 3
  applyGaussianKernelwithVariableSigma ................................ 4
  bgGeneNorm .......................................................... 5
  calculateVariableSigmas .............................................. 6
  colGrad .............................................................. 7
  diffuseMat .......................................................... 7
  diffuseProj .......................................................... 9
  fastDist ............................................................ 10
  filterGenes ......................................................... 11
  findLouvain ......................................................... 12
Description

Animation plot

Usage

```r
animPlot(data, ccm, delay = 0.1, darken = 1, lwd = 1, c.cex = 1,
main = "", ...)```

Arguments

- `data` : Dimensionality reduction plot
- `ccm` : Dataframe of indices and momentum
- `delay` : Delay between frames in seconds
- `darken` : Passed to colGrad() function
- `lwd` : Line width
- `c.cex` : Size of points.
- `main` : Plot title
- `...` : Passed to plot() function

Details

Animation plot Generates plot in base R that gradually updates giving the impression of an animation

Value

Generates plot

Author(s)

Wajid Jawaid

Examples

```r
## Not run:
x <- animPlot(x, ccm)

## End(Not run)```
**animPlotGif**  

*Generates a GIF animating*

---

**Description**

Generates a GIF animation

**Usage**

```r
animPlotGif(data, ccm, delay = 0.1, darken = 1, lwd = 1, c.cex = 1, 
main = "", gif = "animation", img.name = "tempPlot", plot.par = NULL, 
point.col = "#333333", arrowLength = 0.1, ...)
```

**Arguments**

- **data**: Reduced dimensionality map to be used for visualisation
- **ccm**: Dataframe of indices and momentums
- **delay**: Delay between frames in seconds
- **darken**: Passed to colGrad() function
- **lwd**: Line width
- **c.cex**: Size of points.
- **main**: Title
- **gif**: Name of movie
- **img.name**: Name of temporary image files generated
- **plot.par**: Passed to R base par() function
- **point.col**: Colour of background points
- **arrowLength**: Modify length of arrow
- **...**: Passed to plot() function

**Details**

Generates a GIF animation

**Value**

Produces an animated GIF with given file name

**Author(s)**

Wajid Jawaid
applyGaussianKernelWithVariableSigma

Apply Gaussian Kernel using Laleh Haghverdi’s variable sigma

Description

Apply Gaussian Kernel using Laleh Haghverdi’s variable sigma

Usage

applyGaussianKernelWithVariableSigma(d2, rsigmas, csigmas = NULL)

Arguments

d2
rsigmas
csigmas

Squared distance metric
Sigmas for cells in the rows
Sigmas for cells in the columns

Details

Apply Gaussian Kernel using Laleh Haghverdi’s variable sigma

Value

Returns matrix of same size as d2.

Author(s)

Wajid Jawaid

Examples

## Not run:
d <- applyGaussianKernelWithVariableSigma(dist, sigmas)
## End(Not run)
bgGeneNorm

Normalise by background gene set

Description
Normalise by background gene set

Usage
bgGeneNorm(x, threshold = 0.05)

Arguments
x Matrix to be normalised with cells in rows and genes in columns
threshold Default 0.05. The threshold below which a gene is deemed background

Details
Normalise by background gene set. Find background genes that are expressed at a lower percentage of the total library size per cell than 'threshold' parameter. These genes are used to calculate a normalisation factor.

Value
Returns a normalised matrix of same dimensions as 'x'

Author(s)
Wajid Jawaid

Examples
## Not run:
normGenes <- bgGeneNorm(x)

## End(Not run)
calculateVariableSigmas

*Calculates sigmas for a distance matrix*

**Description**

Calculates sigmas for a distance matrix

**Usage**

`calculateVariableSigmas(d, knn)`

**Arguments**

- `d` Square distance matrix with 0 diagonal
- `knn` Number of nearest neighbours to use for calculation

**Details**

Calculates sigmas for a distance matrix Using Laleh Haghverdi’s method

**Value**

Returns a vector of sigmas

**Author(s)**

wj241

**Examples**

```r
## Not run:
sigmas <- calculateVariableSigmas(dist, 5)

## End(Not run)
```
colGrad

Generates a smooth colour gradient

Description
Generates a smooth colour gradient

Usage
colGrad(x, darken = 1)

Arguments
- `x` Number of colours required
- `darken` Multiplication factor. Must be less than 1. Smaller the darker.

Details
Generates a smooth colour gradient Goes from red to red/green to green to green/blue to blue to blu/red

Value
Returns vector of RGB colours

Author(s)
Wajid Jawaid

Examples

```r
gradientColors <- colGrad(10)
```

diffuseMat

Generic diffusion function

Description
Generic diffusion function using automated individualised sigma calculation

Usage
diffuseMat(data, ndims = 20, nsig = 5, removeFirst = TRUE,
    useARPACK = TRUE, distfun = NULL, sigmas = NULL, sqdistmat = NULL)
Arguments

- **data**: Matrix of data with genes in rows and cells in columns.
- **ndims**: Number of dimensions to return
- **nsig**: For automatic sigma calculation
- **removefirst**: Default TRUE. Removes the first eigenvector
- **useARPACK**: Default TRUE. Uses Arnoldi algorithm for eigenvector calculations
- **distfun**: A different distance function that returns the **squared** distance
- **sigmas**: Manually provide sigma
- **sqdistmat**: **Squared** distance matrix. Give your own squared distance matrix.

Details

Generic diffusion function using automated individualised sigma calculation.

A Gaussian kernel is applied to the chosen distance metric producing an \(n \times n\) square unnormalised symmetric transition matrix, \(A\). Let \(D\) be an \(n \times n\) diagonal matrix with row(column) sums of \(A\) as entries. The density corrected transition matrix will now be:

\[
D^{-1}AD^{-1}
\]

and can be normalised:

\[
B^{-1}D^{-1}AD^{-1}
\]

where \(B\) is an \(n \times n\) diagonal matrix with row sums of the density corrected transition matrix as entries. The eigen decomposition of this matrix can be simplified by solving the symmetric system:

\[
B^{-\frac{1}{2}}D^{-1}AD^{-1}B^{-\frac{1}{2}}R = R\lambda'
\]

where \(R'\) is a matrix of the right eigenvectors that solve the system and \(\lambda'\) is the corresponding eigenvalue diagonal matrix. Now the solution of:

\[
B^{-1}D^{-1}AD^{-1}R = R\lambda
\]

in terms of \(R'\) and \(B^{-\frac{1}{2}}\) is:

\[
B^{-1}D^{-1}AD^{-1}B^{-\frac{1}{2}}R' = B^{-\frac{1}{2}}R'\lambda'
\]

and

\[
R = B^{-\frac{1}{2}}R'
\]

This \(R\) without the first eigenvector is returned as the diffusion map.

Value

List output containing:
values  Eigenvalues, excluding the first eigenvalue, which should always be 1.

vectors  Matrix of eigen vectors in columns, first eigen vector removed.

nconv  Number of eigen vectors/values that converged.

niter  Iterations taken for Arnoldi algorithm to converge.

nops  Number of operations.

val0  1st eigen value - should be 1. If not be suspicious!

vec0  1st eigen vector - should be $n^{-\frac{1}{2}}$, where n is the number of cells/samples.

usedARPACK  Predicates use of ARPACK for spectral decomposition.

distfun  Function used to calculate the squared distance.

nn  Number of nearest neighbours used for calculating sigmas.

d2  Matrix of squared distances, returned from distfun.

sigmas  Vector of sigmas. Same length as number of cells if individual sigmas were calculated, otherwise a scalar if was supplied.

gaussian  Unnormalised transition matrix after applying Gaussian.

markov  Normalised gaussian matrix.

densityCorrected  Matrix after applying density correction to markov.

Author(s)

Wajid Jawaid

References


Examples

```r
## Not run:
xx <- diffuseMat(x)

## End(Not run)
```

diffuseProj

Predicts diffusion map projection from new data points

Description

Predicts diffusion map projection from new data points

Usage

diffuseProj(dm, x, data, distfun)
Arguments

- `dm`: Output from `diffuseMat2` function
- `x`: Matrix of new data points. Features in rows and cells in columns.
- `data`: Original data used to generate diffusion map
- `distfun`: A distance function that takes new data as first parameter and previous data as second variable returning a squared distance measure, with each sample in the rows and distance to previous data points in columns, e.g. function(x, y) \((1 - \text{cor}(x, y))^2\).

Details

Predicts diffusion map projection from new data points

Value

Returns a matrix with projected diffusion components.

Author(s)

Wajid Jawaid

Examples

```r
## Not run: 
y <- diffuseProj(xx, newData, oldData, function(z) (1-cor(z))^2)
## End(Not run)
```

---

**fastDist**  
*Fast vectorised Euclidean distance calculator*

Description

Fast vectorised Euclidean distance calculator

Usage

`fastDist(x, squared = FALSE)`

Arguments

- `x`: Matrix with vectors in columns.
- `squared`: Will not perform the square root, i.e. will return the squared ‘L2-norm’.

Details

Calculates Euclidean distances between vectors arranged as columns in a matrix.
filterGenes

Value

Returns a matrix of pairwise distances

Author(s)

Wajid Jawaid

Examples

```r
## Not run:
dist <- fastDist(x)
## End(Not run)
```

---

filterGenes  Filter genes

Description

Filter genes

Usage

```r
filterGenes(x, mu = 0.01, cv = 2, fano = FALSE)
```

Arguments

- `x`  Matrix to be normalised with cells in rows and genes in columns
- `mu`  Mean threshold
- `cv`  Coefficient of variation or Fano factor threshold.
- `fano`  Default TRUE. Predicate treat CV as Fano factor or CV

Details

Filter genes Filter genes by mean and either coefficient of variation, cv or Fano factor.

Value

Returns a filtered matrix with same number of cells but fewer genes than `x`

Author(s)

Wajid Jawaid
findLouvain

## Examples

```r
## Not run:
exressionGenesFiltered <- filterGenes(x)

## End(Not run)
```

### Description

Louvain clustering on transition matrix

### Usage

```r
findLouvain(mkv)
```

### Arguments

- `mkv` : Transition matrix

### Details

Louvain clustering on transition matrix

### Value

Returns a list with graph, dataframe and community object

### Author(s)

Wajid Jawaid

### Examples

```r
## Not run:
xx <- findLouvain(mkv)
xx$c11

## End(Not run)
```
**Find next cell function**

**Description**
Find next cell function

**Usage**
```
fnc(rdmap, tm, curInd, mom = NULL, momAdj = 0.5, w1 = exp(1), w2 = 1, varEst = 10)
```

**Arguments**
- `rdmap`: reduced dimensionality matrix with cells in rows and dims in columns
- `tm`: Transition matrix
- `curInd`: Current state on `tm`
- `mom`: Current momentum vector
- `momAdj`: Weighting to adjust momentum. From 0-1. Lower numbers make smaller adjustment to momentum vector.
- `w1`: Parameter - Base used for modifying of `tm` probs.
- `w2`: Parameter - Multiplification factor used for modifying `tm` probs.
- `varEst`: Number of alternatives to sample for estimating variance.

**Details**
Find next cell function. Transition probabilities are modified by calculating the cosine of the angle between the current momentum vector and the vector on the `rdmap` required for each transition. The transition probability is adjusted by multiplying by \( w1^{w2 \cdot \text{cosine_angle}} \) and then normalising.

**Value**
Returns index of new cell and new momentum vector

**Author(s)**
Wajid Jawaid

**Examples**
```
## Not run:
nextCell <- fnc(rdmap, tm, curInd)

## End(Not run)
```
Description
Return a plausible developmental journey

Usage
getTraj(rdmap, tm, sourceCellInds, terminalCellsInd = NULL, momAdj = 0.5, 
      w1 = exp(1), w2 = 1, simLen = 50, sim.seed = NULL, varEst = 10)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rdmap</td>
<td>reduced dimensionality matrix with cells in rows and dims in columns</td>
</tr>
<tr>
<td>tm</td>
<td>Transition matrix</td>
</tr>
<tr>
<td>sourceCellInds</td>
<td>Starting cell indices</td>
</tr>
<tr>
<td>terminalCellsInd</td>
<td>Terminal cell indices</td>
</tr>
<tr>
<td>momAdj</td>
<td>Weighting to adjust momentum. From 0-1. Lower numbers make smaller adjustment to momentum vector.</td>
</tr>
<tr>
<td>w1</td>
<td>Parameter - Base used for modifying of tm probs.</td>
</tr>
<tr>
<td>w2</td>
<td>Parameter - Multiplification factor used for modifying tm probs.</td>
</tr>
<tr>
<td>simLen</td>
<td>Maximum number of allowable transitions</td>
</tr>
<tr>
<td>sim.seed</td>
<td>Random seed for reproducibility</td>
</tr>
<tr>
<td>varEst</td>
<td>Number of alternatives to sample for estimating variance.</td>
</tr>
</tbody>
</table>

Details
Return a plausible developmental journey

Value
Returns a data.frame of ordered indices and momentums

Author(s)
Wajid Jawaid

Examples
```R
## Not run:
traj <- getTraj(rdmap, tm, startCells, terminalCells)
## End(Not run)
```
goggles

View single cell dataset

Description

View single cell dataset

Usage

goggles(x, pcaDims = 90, nsig = 5, dmat = NULL, mkv = NULL,
plotDims = 2, kernSq = 2, ...)

Arguments

  x            Matrix with cells in rows and gene in columns
  pcaDims      Number of PCA dimensions to keep for distance measure
  nsig         Number of significant neighbours to keep for Gaussian kernel
  dmat         Optional. Give your own distance matrix
  mkv          Optional. Give your own markov matrix.
  plotDims     Default 2. Number of dimensions to plot
  kernSq       Factor to tighten kernel - operates on sigmas.
  ...          Additonal parameters not currently in use

Details

View single cell dataset

Value

A list of l, dimensionality reduced data.frame; clust, returned from louvainClust(); adj, Sparse, pruned adjacency matrix; dmat, distance matrix; pca, PCA reduced matrix. sparse, diagnostics on adj prior to applying sparseMarkov().

Author(s)

Wajid Jawaid

Examples

## Not run:
xx <- goggles(x)
plot(xx$l)

## End(Not run)
sparseMarkov

Make markov matrix sparse

Usage

sparseMarkov(mkv, knn)

Arguments

mkv Markov matrix
knn Number of nearest neighbours. See above.

Details

Make markov matrix sparse Choose knn as the maximum number of similar cells are likely to exist in your dataset.

Value

Markovian sparse matrix.

Author(s)

Wajid Jawaid
Index

animPlot, 2
animPlotGif, 3
applyGaussianKernelwithVariableSigma, 4

bgGeneNorm, 5
calculateVariableSigmas, 6
colGrad, 7
diffuseMat, 7
diffuseProj, 9

fastDist, 10
filterGenes, 11
findLouvain, 12
fnc, 13

getTraj, 14
goggles, 15

sparseMarkov, 16