Package ‘Spectrum’

January 20, 2025

Title Fast Adaptive Spectral Clustering for Single and Multi-View Data

Version 1.1

Author Christopher R John, David Watson

Maintainer Christopher R John <chris.r.john86@gmail.com>

Description A self-tuning spectral clustering method for single or multi-view data. 'Spectrum' uses a new type of adaptive density aware kernel that strengthens connections in the graph based on common nearest neighbours. It uses a tensor product graph data integration and diffusion procedure to integrate different data sources and reduce noise. 'Spectrum' uses either the eigengap or multimodality gap heuristics to determine the number of clusters. The method is sufficiently flexible so that a wide range of Gaussian and non-Gaussian structures can be clustered with automatic selection of K.

Depends R (>= 3.5.0)

License AGPL-3

Encoding UTF-8

LazyData true

Imports ggplot2, ClusterR, Rfast, diptest

Suggests knitr

VignetteBuilder knitr

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-02-10 08:40:02 UTC

Contents

`blobs` *8 blob like structures*

Description

A simulated dataset of 8 Gaussian blobs. Simulated using the 'clusterlab' CRAN package.

Usage

```
blobs
```

Format

A data frame with 10 rows and 800 variables

`brain` *A brain cancer dataset*

Description

A dataset containing The Cancer Genome Atlas expression data. From this publication https://tcga-data.nci.nih.gov/docs/publications/lgggbm_2016/. The first data frame is a 5133X150 RNA-seq data matrix, the second is a 262X150 miRNA-seq data matrix, the third is 45X150 protein array data matrix. The data was all pre-normalised then subject to log transform.

Usage

```
brain
```

Format

A list of data frames

Source

https://gdac.broadinstitute.org/

`circles` *Three concentric circles*

Description

Simulated data using the ’clusterSim’ CRAN package.

Usage

```
circles
```

Format

A data frame with 2 rows and 540 variables

`cluster_similarity` *cluster_similarity: cluster a similarity matrix using the Ng method*

Description

This function performs clustering of a similarity matrix following the method of Ng or of Melia. We recommend using the Ng method with GMM to cluster the eigenvectors instead of k-means.

Usage

```
cluster_similarity(A2, k = k, clusteralg = "GMM", specalg = "Ng")
```

Arguments

`A2`	Data frame or matrix: a similarity matrix
`k`	Numerical value: the number of clusters
`clusteralg`	Character value: GMM or km clustering algorithm (suggested=GMM)
`specalg`	Character value: Ng or Melia variant of spectral clustering (default=Ng)

Value

A numeric vector of cluster assignments

References

Ng, Andrew Y., Michael I. Jordan, and Yair Weiss. "On spectral clustering: Analysis and an algorithm." Advances in neural information processing systems. 2002.

Meila, Marina, et al. "Spectral Clustering: a Tutorial for the 2010's." Handbook of Cluster Analysis. CRC Press, 2016. 1-23.

Examples

```
ng_similarity <- cluster_similarity(missl[[1]],k=8)
```

CNN_kernel *CNN_kernel: fast adaptive density-aware kernel*

Description

CNN_kernel: fast adaptive density-aware kernel

Usage

```
CNN_kernel(mat, NN = 3, NN2 = 7)
```

Arguments

`mat` Matrix: matrix should have samples as columns and rows as features

`NN` Numerical value: the number of nearest neighbours to use when calculating local sigma

`NN2` Numerical value: the number of nearest neighbours to use when calculating common nearest neighbours

Value

A kernel matrix

Examples

```
CNN_kern <- CNN_kernel(blobs[,1:50])
```

`estimate_k` *estimate_k: estimate K using the eigengap or multimodality gap heuristics*

Description

This function will try to estimate K given a similarity matrix. Generally the maximum eigengap is preferred, but on some data examining the distribution of the eigenvectors as in the multimodality gap heuristic may be beneficial.

Usage

```
estimate_k(A2, maxk = 10, showplots = TRUE)
```

Arguments

`A2`	Data frame or matrix: a similarity matrix
`maxk`	Numerical value: maximum number of K to be considered
`showplots`	Character value: whether to show the plot on the screen

Value

A data frame containing the eigenvalues and dip-test statistics of the eigenvectors of the graph Laplacian

Examples

```
k_test <- estimate_k(missl[[1]])
```

`harmonise_ids` *harmonise_ids: works on a list of similarity matrices to add entries of NA where there are missing observations between views*

Description

Simply adds a column and row of NA with the missing ID for data imputation. The similarity matrix requires row and column IDs present for this to work.

Usage

```
harmonise_ids(l)
```

Arguments

`l`	A list of similarity matrices: those to be harmonised.

Value

A list of harmonised similarity matrices.

Examples

```
h_test <- harmonise_ids(missl)
```

integrate_similarity_matrices

integrate_similarity_matrices: integrate similarity matrices using a tensor product graph linear combination and diffusion technique

Description

Given a list of similarity matrices this function will integrate them running the Shu algorithm, also can reduce noise if the input is a list consisting of a single matrix.

Usage

```
integrate_similarity_matrices(kernellist, KNNs_p = 10,
  diffusion_iters = 4, method = "TPG")
```

Arguments

`kernellist`	A list of similarity matrices: those to be integrated
`KNNs_p`	Numerical value: number of nearest neighbours for KNN graph (default=10, suggested=10-20)
`diffusion_iters`	Numerical value: number of iterations for graph diffusion (default=4, suggested=2-6)
`method`	Character: either TPG (see reference below) or mean (default=TPG)

Value

An integrated similarity matrix

References

Shu, Le, and Longin Jan Latecki. "Integration of single-view graphs with diffusion of tensor product graphs for multi-view spectral clustering." Asian Conference on Machine Learning. 2016.

Examples

```
i_test <- integrate_similarity_matrices(misslfilled,method='mean')
```

kernel_pca *kernel_pca: A kernel pca function*

Description

kernel_pca: A kernel pca function

Usage

```
kernel_pca(datam, labels = FALSE, axistextsize = 18,
  legendtextsize = 18, dotsize = 3, similarity = TRUE)
```

Arguments

`datam`	Dataframe or matrix: a data frame with samples as columns, rows as features, or a kernel matrix
`labels`	Factor: to label the plot with colours
`axistextsize`	Numerical value: axis text size
`legendtextsize`	Numerical value: legend text size
`dotsize`	Numerical value: dot size
`similarity`	Logical flag: whether the input is a similarity matrix or not

Value

A kernel PCA plot

Examples

```
ex_kernel_pca <- kernel_pca(blobs[,1:50], similarity=FALSE)
```

mean_imputation *mean_imputation: mean imputation function for multi-view spectral clustering with missing data*

Description

Works on a list of similarity matrices to impute missing values using the mean from the other views.

Usage

```
mean_imputation(l)
```

Arguments

`l`	A list of data frames: all those to be included in the imputation.

Value

A list of completed data frames.

Examples

```
m_test <- mean_imputation(misslfilled)
```

missl *A list of the blob data as similarity matrices with a missing entry in one*

Description

Two copies of a simulated dataset of 8 Gaussian blobs in a list converted to a similarity matrix, but one has a missing observation.

Usage

```
missl
```

Format

A list of two data frames

misslfilled *A list of the blob data as similarity matrices with a missing entry in one filled with NAs*

Description

Two copies of a simulated dataset of 8 Gaussian blobs in a list converted to a similarity matrix, but one has a missing observation filled with NAs.

Usage

```
misslfilled
```

Format

A list of two data frames

ng_kernel *ng_kernel: Kernel from the Ng spectral clustering algorithm*

Description

This is the kernel from the Ng spectral clustering algorithm. It takes a global sigma which requires tuning for new datasets in most cases. It is possible to use the sigma_finder function to find a sigma for a dataset. Sigma is assumed to be squared already.

Usage

```
ng_kernel(data, sigma = 0.1)
```

Arguments

data	Data frame or matrix: with points as columns, features as rows
sigma	Numerical value: a global sigma that controls the drop off in affinity

Value

A similarity matrix of the input data

References

Ng, Andrew Y., Michael I. Jordan, and Yair Weiss. "On spectral clustering: Analysis and an algorithm." Advances in neural information processing systems. 2002.

Examples

```
ng_similarity <- ng_kernel(brain[[1]])
```

pca *pca: A pca function*

Description

pca: A pca function

Usage

```
pca(mydata, labels = FALSE, dotsize = 3, axistextsize = 18,
  legendtextsize = 18)
```

Arguments

`mydata`	Data frame or matrix: matrix or data frame with samples as columns, features as rows
`labels`	Factor: to label the plot with colours
`dotsize`	Numerical value: dot size
`axistextsize`	Numerical value: axis text size
`legendtextsize`	Numerical value: legend text size

Value

A pca plot object

Examples

```
ex_pca <- pca(blobs[,1:50])
```

`rbfkernel_b` *rbfkernel_b: fast self-tuning kernel*

Description

rbfkernel_b: fast self-tuning kernel

Usage

```
rbfkernel_b(mat, K = 3, sigma = 1)
```

Arguments

`mat`	Matrix: matrix should have samples as columns and rows as features
`K`	Numerical value: the number of nearest neighbours to use when calculating local sigma
`sigma`	Numerical value: a global sigma, usually left to 1 which has no effect

Value

A kernel matrix

Examples

```
stsc_kern <- rbfkernel_b(blobs[,1:50])
```

`sigma_finder` *sigma_finder: heuristic to find sigma for the Ng kernel*

Description

This is a heuristic to find the sigma for the kernel from the Ng spectral clustering algorithm. It returns a global sigma. It uses the mean K nearest neighbour distances of all samples to determine sigma.

Usage

```
sigma_finder(mat, NN = 3)
```

Arguments

`mat` Data frame or matrix: with points as columns, features as rows

`NN` Numerical value: the number of nearest neighbours to use (default=3)

Value

A global sigma

Examples

```
sig <- sigma_finder(blobs)
```

`Spectrum` *Spectrum: Fast Adaptive Spectral Clustering for Single and Multi-view Data*

Description

Spectrum is a self-tuning spectral clustering method for single or multi-view data. Spectrum uses a new type of adaptive density-aware kernel that strengthens connections between points that share common nearest neighbours in the graph. For integrating multi-view data and reducing noise a tensor product graph data integration and diffusion procedure is used. Spectrum analyses eigenvector variance or distribution to determine the number of clusters. Spectrum is well suited for a wide range of data, including both Gaussian and non-Gaussian structures.

Usage

```
Spectrum(data, method = 1, silent = FALSE, showres = TRUE,
  diffusion = TRUE, kerneltype = c("density", "stsc"), maxk = 10,
  NN = 3, NN2 = 7, showpca = FALSE, frac = 2, thresh = 7,
  fontsize = 18, dotsize = 3, tunekernel = FALSE,
  clusteralg = "GMM", FASP = FALSE, FASPk = NULL, fixk = NULL,
  krangemax = 10, runrange = FALSE, diffusion_iters = 4,
  KNNs_p = 10, missing = FALSE)
```

Arguments

`data` Data frame or list of data frames: contains the data with points to cluster as columns and rows as features. For multi-view data a list of dataframes is to be supplied with the samples in the same order.

`method` Numerical value: 1 = default eigengap method (Gaussian clusters), 2 = multimodality gap method (Gaussian/ non-Gaussian clusters), 3 = no automatic method (see fixk param)

`silent` Logical flag: whether to turn off messages

`showres` Logical flag: whether to show the results on the screen

`diffusion` Logical flag: whether to perform graph diffusion to reduce noise (default=TRUE)

`kerneltype` Character string: 'density' (default) = adaptive density aware kernel, 'stsc' = Zelnik-Manor self-tuning kernel

`maxk` Numerical value: the maximum number of expected clusters (default=10). This is data dependent, do not set excessively high.

`NN` Numerical value: kernel param, the number of nearest neighbours to use sigma parameters (default=3)

`NN2` Numerical value: kernel param, the number of nearest neighbours to use for the common nearest neigbours (default = 7)

`showpca` Logical flag: whether to show pca when running on one view

`frac` Numerical value: optk search param, fraction to find the last substantial drop (multimodality gap method param)

`thresh` Numerical value: optk search param, how many points ahead to keep searching (multimodality gap method param)

`fontsize` Numerical value: controls font size of the ggplot2 plots

`dotsize` Numerical value: controls the dot size of the ggplot2 plots

`tunekernel` Logical flag: whether to tune the kernel, only applies for method 2 (default=FALSE)

`clusteralg` Character string: clustering algorithm for eigenvector matrix (GMM or km)

`FASP` Logical flag: whether to use Fast Approximate Spectral Clustering (for v. high sample numbers)

`FASPk` Numerical value: the number of centroids to compute when doing FASP

`fixk` Numerical value: if we are just performing spectral clustering without automatic selection of K, set this parameter and method to 3

`krangemax` Numerical value: the maximum K value to iterate towards when running a range of K

`runrange` Logical flag: whether to run a range of K or not (default=FALSE), puts Kth results into Kth element of list

`diffusion_iters`
Numerical value: number of diffusion iterations for the graph (default=4)

`KNNs_p` Numerical value: number of KNNs when making KNN graph (default=10, suggested=10-20)

`missing` Logical flag: whether to impute missing data in multi-view analysis (default=FALSE)

Value

A list, containing: 1) cluster assignments, in the same order as input data columns 2) eigenvector analysis results (either eigenvalues or dip test statistics) 3) optimal K 4) final similarity matrix 5) eigenvectors and eigenvalues of graph Laplacian

Examples

```
res <- Spectrum(brain[[1]][,1:50])
```

spirals *Two spirals wrapped around one another*

Description

Simulated data using the ’mlbench’ CRAN package.

Usage

```
spirals
```

Format

A data frame with 2 rows and 180 variables

Index