Package ‘FuseSOM’

April 3, 2024

Title A Correlation Based Multiview Self Organizing Maps Clustering For IMC Datasets

Version 1.4.0

Description A correlation-based multiview self-organizing map for the characterization of cell types in highly multiplexed in situ imaging cytometry assays (FuseSOM) is a tool for unsupervised clustering. FuseSOM is robust and achieves high accuracy by combining a Self Organizing Map architecture and a Multiview integration of correlation based metrics. This allows FuseSOM to cluster highly multiplexed in situ imaging cytometry assays.

License GPL-2

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.2

Imports psych, FCPS, analogue, coop, pheatmap, ggplotify, fastcluster, fpc, ggplot2, stringr, ggpubr, proxy, cluster, diptest, methods, SummarizedExperiment, stats, S4Vectors

LazyData false

BuildResaveData false

Depends R (>= 4.2.0)

Suggests knitr, BiocStyle, rmarkdown, SingleCellExperiment

VignetteBuilder knitr

BugReports https://github.com/ecool50/FuseSOM/issues

LinkingTo Rcpp

biocViews SingleCell, CellBasedAssays, Clustering, Spatial

git_url https://git.bioconductor.org/packages/FuseSOM

git_branch RELEASE_3_18

git_last_commit ab1cc41

git_last_commit_date 2023-10-24

Repository Bioconductor 3.18
Function to do arsinh normalization

Description
Function to do arsinh normalization

Usage
.arsinhNnorm(x, cofactor = 5)

Arguments
x  A numeric or complex vector
cofactor  Cofactor of the vector. Default is 5.

Value
Arsinh normalized vector.
.computeElbow

A function to compute the elbow point given a set of points

Description

A function to compute the elbow point given a set of points

Usage

.computerElbow(vals)

Arguments

vals Values to compute the elbow point of.

Value

A integer indicating the elbow point of vals.

.minmaxNorm

Function to do min max normalization

Description

Function to do min max normalization

Usage

.minmaxNorm(x)

Arguments

x Matrix to min max normalize.

Value

Max normalized version of x
.runDiscriminant

Description

Function to do percentile normalization

Usage

.percentileNorm(x)

Arguments

x 

Matrix to percentile normalise.

Value

percentile normalized version of x

.runDiscriminant

 Discriminant cluster estimator

Description

Function to estimate the number of clusters using discriminant analysis parts of this function is based on the sigclust2 package by Patrick Kimes see https://github.com/pkimes/sigclust2

Usage

.runDiscriminant(distMat, minClusterSize, alpha = 0.001)

Arguments

distMat 

A distance matrix

minClusterSize 

The minimum cluster size

alpha 

a value between 0 and 1 specifying the desired level of cutoff

Value

Optimal number of clusters
Description

Creates uniformly distributed data of same dimensionality as input data this function was obtained from the Stab package

Usage

.uniformData(data)

Arguments

data A data matrix.

Value

Uniform random noise with dim(data)

clusterPrototypes Cluster prototypes

Description

Cluster the prototypes from the Self Organizing Map Clustering is done using hierarchical clustering with the average linkage function

Usage

clusterPrototypes(somModel, numClusters = NULL)

Arguments

somModel the self organizing map
numClusters the number of clusters to generate

Value

the cluster labels
computeGridSize

Estimate the optimal grid size

Description

The function finds the eigenvalues of the sample covariance matrix. It will then return the number of significant eigenvalues according to the Tracy-Widom test. The function is based on the estKW function from the SC3 package.

Usage

computeGridSize(dataset)

Arguments

dataset The optimal grid size.

Value

the optimal grid size.

Author(s)

Elijah Wille ewil3501@uni.sydney.edu.au

Examples

data("risom_dat")
risomMarkers <- c("CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD"
)
prototypes <- generatePrototypes(risom_dat[, risomMarkers])
clusters <- clusterPrototypes(prototypes, 23)

data("risom_dat")
risomMarkers <- c("CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD"
)
computeGridSize(risom_dat[, risomMarkers])
estimateNumCluster  

Estimate number of clusters

Description

A function for estimating the number of clusters using various methods available are: Discriminant, Distance (Gap, Silhouette, Slope, Jump, and Within Cluster Distance,) and Instability

Usage

estimateNumCluster(data, method = c("Discriminant", "Distance"), kSeq = 2:20)

Arguments

data  
the SOM object generated by generatePrototypes(), or an object of class SingleCellExperiment or SpatialExperiment.

method  
one of Discriminant, Distance, Stability. By default, everything is run

kSeq  
a sequence of the number of clusters to try. Default is 2:20 clusters

Value

A list containing the cluster estimations if a dataframe or matrix is provided

A SingleCellExperiment with a cluster estimate in it's metadata if a SingleCellExperiment or SpatialExperiment object is provided

Author(s)

Elijah WIllie ewil3501@uni.sydney.edu.au

Examples

data("risom_dat")
risomMarkers <- c("CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD"
)
res <- runFuseSOM(risom_dat, markers = risomMarkers, numClusters = 23)
res.est.k <- estimateNumCluster(res$model, kSeq = 2:25)
FuseSOM

Description

FuseSOM provides a pipeline for the clustering of highly multiplexed in situ imaging cytometry assays. This pipeline uses the Self Organizing Map architecture coupled with Multiview hierarchical clustering. We also provide functions for normalisation and estimation of the number of clusters.

Details

The FuseSOM package provides three categories of important functions: foo, bar and baz.

generatePrototypes

Generate a Self Organizing Map

Description

A self organizing map of the marker intensities is generated and the prototypes are returned. The grid size is determined automatically.

Usage

generatePrototypes(data, verbose = FALSE, size = NULL)

Arguments

data the marker intensities
verbose should the progress be printed out
size The optimal grid size for the Self Organizing Map

Value

the self organizing map object

Examples

data("risom_dat")
risomMarkers <- c("CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD")
generatePrototypes(risom_dat[, risomMarkers])
markerHeatmap

Generate expression heatmap

Description

A function for generating a heat map of marker expression across clusters

Usage

markerHeatmap(
  data,
  markers = NULL,
  clusters = NULL,
  threshold = 2,
  clusterMarkers = FALSE,
  fontSize = 14
)

Arguments

data a matrix or dataframe where the rows are samples and columns are markers
markers a list of markers of interest. If not provided, all columns will be used
clusters a vector of cluster labels
threshold the value to threshold the marker expression at
clusterMarkers should the rows(markers) of the heatmap be clustered
fontSize the size of the text on the heatmap

Value

a heatmap with the markers in the rows and clusters in the columns

Author(s)

Elijah Willie ewil3501@uni.sydney.edu.au

Examples

data("risom_dat")
risomMarkers <- c("CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD"
)
res <- runFuseSOM(risom_dat, markers = risomMarkers, numClusters = 23)
p.heat <- markerHeatmap(risom_dat, risomMarkers, clusters = res$clusters)
normalizeData

**Normalize Marker Intensities**

**Description**

The matrix of intensities is normalised based on one of four different methods: these methods include Percentile, zscore, arsinh and minmax.

**Usage**

```r
normalizeData(data, markers, method = "none", cofactor = 5)
```

**Arguments**

- `data`: the raw intensity scores.
- `markers`: the markers of interest.
- `method`: the normalisation method.
- `cofactor`: the cofactor for arsinh normalisation.

**Value**

normalised matrix.

**Author(s)**

Elijah Willie ewil3501@uni.sydney.edu.au

**Examples**

```r
data("risom_dat")
risomMarkers <- c("CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD")
normalizeData(risom_dat[, risomMarkers])
```

---

normalizeData

**Normalize Marker Intensities**

**Description**

The matrix of intensities is normalised based on one of four different methods: these methods include Percentile, zscore, arsinh and minmax.
normalizeData(data, markers, method = "none", cofactor = 5)

Arguments

- **data**: the raw intensity scores.
- **markers**: the markers of interest.
- **method**: the normalization method.
- **cofactor**: the cofactor for arsinh normalization.

Value

- normalised matrix.

Author(s)

Elijah Willie ewil3501@uni.sydney.edu.au

Examples

```r
data("risom_dat")
risomMarkers <- c("CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD")
normaliseData(risom_dat[, risomMarkers])
```

optiPlot

Generate elbow plots

Description

A function generating the elbow plot for the optimal number of clusters returned by the estimateNumCluster() function. Methods available are: Gap, Silhouette, Slope, Jump, and Within Cluster Distance (WCD).

Usage

optiPlot(data, method = "jump")

Arguments

- **data**: a Self Organizing Map object generated by generatePrototypes(), or an object of class SingleCellExperiment or SpatialExperiment.
- **method**: one of 'jump', 'slope', 'wcd', 'gap', or 'silhouette'.
an elbow plot object where the optimal number of clusters is marked

Author(s)

Elijah Willie ewil3501@uni.sydney.edu.au

Examples

data("risom_dat")
risomMarkers <- c(
  "CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD"
)
res <- runFuseSOM(risom_dat, markers = risomMarkers, numClusters = 23)
resEstK <- estimateNumCluster(res$model, kSeq = 2:25)
p <- optiPlot(resEstK, method = "jump")

Description

IMC Breast Cancer Data Data from A spatial atlas of breast cancer progression using MIBI-TOF and tissue transcriptomics

Usage

data(risom_dat)

Format

An object of class "data.frame".

Source

Mendeley Data, https://data.mendeley.com/datasets/d87vg86zd8/3

References

T. Risom, et al. Transition to invasive breast cancer is associated with progressive changes in the structure and composition of tumor stroma Cell, 185 (2022), pp. 299-310 (ScienceDirect)
runFuseSOM

A wrapper function to run the FuseSOM algorithm

Description

This function accepts a matrix, dataframe or a SingleCellExperiment object. For matrices and dataframes, it is assumed that markers are the columns and samples rows.

Usage

runFuseSOM(
  data,
  markers = NULL,
  numClusters = NULL,
  assay = NULL,
  clusterCol = "clusters",
  size = NULL,
  verbose = FALSE
)

Arguments

data: a matrix, dataframe, SingleCellExperiment or SpatialExperiment object.
markers: the markers of interest. If this is not provided, all columns will be used.
numClusters: the number of clusters to be generated from the data.
assay: the assay of interest if SingleCellExperiment object is used.
clusterCol: the name of the column to store the clusters in.
size: the size of the square grid. eg for a 10X10 grid, size = 10.
verbose: should the generation of the Self Organising Map be printed?

Value

A list containing the SOM model and the cluster labels if a dataframe or matrix is provided.
A SingleCellExperiment object with labels in coldata, and SOM model in metadata if a SingleCell-Experiment or SpatialExperiment object is provided.

Author(s)

Elijah Wllie ewil3501@uni.sydney.edu.au
Examples

data("risom_dat")
risomMarkers <- c("CD45", "SMA", "CK7", "CK5", "VIM", "CD31", "PanKRT", "ECAD")
res <- runFuseSOM(
  risom_dat,
  markers = risomMarkers, numClusters = 23, size = 8
)

---

somInitPca.default these functions were obtained from https://rdrr.io/rforge/yasomi/ with some major modifications

Description

these functions were obtained from https://rdrr.io/rforge/yasomi/ with some major modifications

Usage

## Default S3 method:
somInitPca(data, somGrid, weights, with.princomp = FALSE, ...)

Arguments

data The data to which the SOM will be fitted, a matrix or data frame of observations (which should be scaled)
somGrid A somgrid object
weights Optional weights for the data points
with.princomp Switch specifying whether the princomp should be used instead of the prcomp for computing the principal components when no weights are given (see details)
... not used

Value

A list containing: prototype, a matrix containing appropriate initial prototypes, and data.pca the results of the PCA conducted on the data
Index

* datasets
  - risom_dat, 12
  - arsinhNnorm, 2
  - computeElbow, 3
  - minmaxNorm, 3
  - percentileNorm, 4
  - runDiscriminant, 4
  - uniformData, 5
  - clusterPrototypes, 5
  - computeGridSize, 6
  - diabetesData(risom_dat), 12
  - estimateNumCluster, 7
  - FuseSOM, 8
  - generatePrototypes, 8
  - markerHeatmap, 9
  - normaliseData, 10
  - normalizeData, 10
  - optiPlot, 11
  - risom_dat, 12
  - runFuseSOM, 13
  - somInitPca.default, 14