

Package ‘somhca’

February 7, 2026

Type Package

Title Self-Organising Maps Coupled with Hierarchical Cluster Analysis

Version 0.3.0

Description Implements self-organising maps combined with hierarchical cluster analysis (SOM-HCA) for clustering and visualization of high-dimensional data. The package includes functions to estimate the optimal map size based on various quality measures and to generate a model using the selected dimensions. It also performs hierarchical clustering on the map nodes to group similar units. Documentation about the SOM-HCA method is provided in Pastorelli et al. (2024) [doi:10.1002/xrs.3388](https://doi.org/10.1002/xrs.3388).

License MIT + file LICENSE

Encoding UTF-8

Imports kohonen, aweSOM, dplyr, RColorBrewer, grDevices, stats, utils, maptree, fpc

RoxygenNote 7.3.3

NeedsCompilation no

Author Gianluca Pastorelli [aut, cre] (ORCID: <https://orcid.org/0000-0001-6926-1952>)

Maintainer Gianluca Pastorelli <gianluca.pastorelli@gmail.com>

Repository CRAN

Date/Publication 2026-02-07 14:10:02 UTC

Contents

clusterSOM	2
finalSOM	3
generatePlot	4
getClusterData	5
loadMatrix	5
optimalSOM	7

Index	8
--------------	----------

clusterSOM

*Perform Clustering on SOM Nodes***Description**

Groups similar nodes of a SOM using hierarchical clustering. By default, the optimal number of clusters is determined automatically using the KGS penalty function, but the user can also specify a fixed number of clusters.

Usage

```
clusterSOM(
  model,
  n_clusters = NULL,
  validity_indices = TRUE,
  plot_result = TRUE,
  input = NULL
)
```

Arguments

<code>model</code>	A trained SOM model object.
<code>n_clusters</code>	Optional integer. If provided, specifies the number of clusters to cut the SOM dendrogram into. If NULL (default), the optimal number of clusters is determined automatically using the KGS penalty function.
<code>validity_indices</code>	A logical value indicating whether to compute and print popular clustering validity indices (Silhouette, Dunn, CH, Pearson Gamma). Default is 'TRUE'.
<code>plot_result</code>	A logical value indicating whether to plot the clustering result. Default is 'TRUE'.
<code>input</code>	An optional input specifying either: File Path A string specifying the path to a CSV file. In-Memory Data A data frame or matrix containing numeric data. If provided, clusters are assigned to the observations in the original dataset, and the updated data is stored in a package environment as 'DataAndClusters'.

Value

Invisibly returns 'NULL'. If 'plot_result = TRUE', a plot of the clusters on the SOM grid is produced. If 'input' is provided, the clustered dataset is stored in the package environment and can be retrieved with 'getClusterData()'.

Examples

```
# Create a toy matrix with 9 columns and 100 rows
data <- matrix(rnorm(900), ncol = 9, nrow = 100) # 900 random numbers, 100 rows, 9 columns

# Run the finalSOM function with the mock data
model <- finalSOM(data, dimension = 6, iterations = 700)

# Example 1: Perform clustering using the mock model
clusterSOM(model, plot_result = TRUE)

# Example 2: Assign SOM-based clusters to an in-memory data frame
df <- data.frame(
  ID = paste0("Sample", 1:100), # Character column for row headings
  matrix(rnorm(900), ncol = 9, nrow = 100) # Numeric data
)
clusterSOM(model, plot_result = FALSE, input = df)
getClusterData()

# Example 3: Load toy data from a CSV file, perform clustering, and retrieve the clustered dataset
file_path <- system.file("extdata", "toy_data.csv", package = "somhca")
clusterSOM(model, plot_result = FALSE, input = file_path)
getClusterData()
```

finalSOM

Train Final SOM Model

Description

Re-trains the SOM using a specified optimal grid size and number of iterations.

Usage

```
finalSOM(data, dimension, iterations, chunk = 100)
```

Arguments

data	A preprocessed data matrix containing the input data for SOM training.
dimension	An integer specifying the dimension of the square SOM grid (e.g., 5 results in a 5x5 grid).
iterations	An integer defining the number of iterations for training the SOM model. Use a large value, e.g., 500 or higher, for improved training (an error message could suggest that reducing the number of iterations might be necessary). For larger grids, more iterations may be required to ensure convergence. Reducing iterations may speed training but risk under-trained neurons.
chunk	An integer specifying the number of iterations per training block. The SOM will be trained in chunks of this many iterations, with a progress message printed after each block. This helps notify the user that the function is running and not frozen. Default is 100. Larger values reduce the frequency of messages; smaller values provide more frequent updates but may slightly slow execution.

Value

A trained SOM model object.

Examples

```
# Create a toy matrix with 9 columns and 100 rows
data <- matrix(rnorm(900), ncol = 9, nrow = 100) # 900 random numbers, 100 rows, 9 columns

# Run the finalSOM function with the mock data
myFinalSOM <- finalSOM(data, dimension = 6, iterations = 700)
```

generatePlot

Generate SOM Visualization Plots

Description

Creates various types of plots to visualize and evaluate the trained SOM model.

Usage

```
generatePlot(model, plot_type, data = NULL)
```

Arguments

model	A trained SOM model object.
plot_type	An integer specifying the type of plot to generate. Options are: 1 Training progress plot (changes during training). 2 Node count plot (number of samples mapped to each node) for assessing map quality. 3 U-matrix plot (visualizing similarities between neighboring nodes). 4 Weight vector plot (patterns in the distributions of variables). 5 Kohonen heatmaps for all variables in the dataset (distribution of single variables across the map).
data	A preprocessed data matrix containing the input data. Required only for 'plot_type = 5'.

Value

A plot or a series of plots is generated and displayed based on the specified type.

Examples

```
# Create a toy matrix with 9 columns and 100 rows
data <- matrix(rnorm(900), ncol = 9, nrow = 100) # 900 random numbers, 100 rows, 9 columns

# Assign column names to the data matrix
colnames(data) <- paste("Var", 1:ncol(data), sep = "_")

# Run the finalSOM function with the mock data
model <- finalSOM(data, dimension = 6, iterations = 700)

# Generate plots using the mock model
generatePlot(model, plot_type = 2)
generatePlot(model, plot_type = 5, data)
```

getClusterData	<i>Retrieve Clustered Data</i>
----------------	--------------------------------

Description

Access the dataset with cluster assignments stored by ‘clusterSOM’.

Usage

```
getClusterData()
```

Value

A data frame with the clustered dataset.

loadMatrix	<i>Load Data and Convert to a Matrix</i>
------------	--

Description

Loads data from a CSV file or an in-memory object (data frame or matrix), optionally removes specified columns, and applies specified normalization methods before converting the data to a matrix. In the original dataset, rows represent observations (e.g., samples), columns represent variables (e.g., features), and all cells (except for column headers and, if applicable, row headers) must only contain numeric values.

Usage

```
loadMatrix(
  input,
  remove_columns = NULL,
  remove_row_headings = FALSE,
  scaling = "no"
)
```

Arguments

input	A string specifying the path to the CSV file, or an in-memory object (data frame or matrix).
remove_columns	Optional integer, character, or vector of either. If specified, removes the columns of the dataset indicated by position or name. This is useful, for example, when the first column contains non-numeric identifiers (e.g., sample names) that should be excluded from the analysis. Default is 'NULL'.
remove_row_headings	Deprecated, use 'remove_columns = 1' instead. A logical value. If 'TRUE', removes the first column of the dataset. Default is 'FALSE'.
scaling	A string specifying the scaling method. Options are: "no" No scaling is applied (default). "simpleFeature" Each column is divided by its maximum value. "minMax" Each column is scaled to range [0, 1]. "zScore" Each column is Z-score standardized.

Value

A matrix with the processed data.

Examples

```
# Example 1: Load toy data from a CSV file
file_path <- system.file("extdata", "toy_data.csv", package = "somhca")

# Run the loadMatrix function with the mock data
myMatrix <- loadMatrix(file_path, remove_columns = 1, scaling = "minMax")

# Example 2: Load from a toy data frame
df <- data.frame(
  ID = paste0("Sample", 1:100), # Character column for row headings
  matrix(rnorm(900), nrow = 100, ncol = 9) # Numeric data
)

# Run the loadMatrix function with the mock data
myMatrix <- loadMatrix(df, remove_columns = 1, scaling = "zScore")

# Example 3: Load from a toy matrix
mat <- matrix(rnorm(900), nrow = 100, ncol = 9) # Numeric data

# Run the loadMatrix function with the mock data
myMatrix <- loadMatrix(mat, scaling = "simpleFeature")
```

optimalSOM

*Estimate Optimal SOM Grid Size***Description**

Computes the optimal grid size for training a SOM using various quality measures and heuristic approaches.

Usage

```
optimalSOM(data, method = "A", increments = 1, iterations = NULL)
```

Arguments

<code>data</code>	A preprocessed data matrix containing the input data for SOM training.
<code>method</code>	A character string indicating the method for estimating the maximum grid dimension. Options are: "A" Uses the heuristic formula by Vesanto et al. (default). "B" Applies an alternative heuristic approach. numeric Manually specified maximum dimension.
<code>increments</code>	An integer specifying the step size for increasing grid dimensions (default is 1). For example, set increments to 2 or 5 to increment the grid size by 2 or 5 rows/columns at each step. Smaller increments lead to more granular searches but may increase computation time; larger increments risk errors if they exceed the estimated maximum SOM grid dimensions.
<code>iterations</code>	Optional integer. Number of iterations ('rlen') for SOM training. If set to NULL (default), the function automatically calculates a sensible number of iterations based on the dataset size. If you want to override this, you can provide a numeric value. A lower value, such as less than 500, helps reduce computation time. If the process takes too long or an error occurs, try reducing the number of iterations for quicker results.

Value

A data frame summarizing optimal SOM grid dimensions and suggested iterations for each quality measure. Use these results to select the most suitable grid size for your SOM.

Examples

```
# Create a toy matrix with 9 columns and 100 rows
data <- matrix(rnorm(900), ncol = 9, nrow = 100) # 900 random numbers, 100 rows, 9 columns

# Run the optimalSOM function with the mock data
myOptimalSOM <- optimalSOM(data, method = "A", increments = 2, iterations = 300)
```

Index

clusterSOM, [2](#)

finalSOM, [3](#)

generatePlot, [4](#)

getClusterData, [5](#)

loadMatrix, [5](#)

optimalSOM, [7](#)