

# Package ‘SpecDetec’

May 7, 2026

**Title** Change Points Detection with Spectral Clustering

**Version** 1.0.0

**Description** Calculate change point based on spectral clustering with the option to automatically calculate the number of clusters if this information is not available.

**Depends** R (>= 3.4.0)

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**Imports** stats, abind

**RoxygenNote** 6.1.0

**NeedsCompilation** no

**Author** Luis Uzai [aut, cre]

**Maintainer** Luis Uzai <uzai\_ff@hotmail.com>

**Repository** CRAN

**Date/Publication** 2018-10-19 14:20:03 UTC

## Contents

calculateAffinityMatrix . . . . .	2
clusterEstimateNumber . . . . .	3
convertToMatrixTimeSeries . . . . .	3
DEVICE1 . . . . .	4
DEVICE2 . . . . .	4
DEVICE3 . . . . .	5
DEVICE4 . . . . .	5
DEVICE5 . . . . .	6
DEVICE6 . . . . .	6
FTIR1 . . . . .	7
FTIR2 . . . . .	7
FTIR3 . . . . .	8
FTIR4 . . . . .	8

FTIR5 . . . . .	9
FTIR6 . . . . .	9
gaussianKernel . . . . .	10
generateEigenvectorMatrix . . . . .	10
generateSimilarityMatrix . . . . .	11
getClusterFact . . . . .	12
getClusterProd . . . . .	12
getSpectralClusters . . . . .	13
Spec . . . . .	14

<b>Index</b>	<b>16</b>
--------------	-----------

---

calculateAffinityMatrix

*Calculate the affinity matrix based on the similarity matrix*

---

## Description

Calculate the affinity matrix based on the similarity matrix

## Usage

```
calculateAffinityMatrix(similarityMatrix, neighborsNumber = 2)
```

## Arguments

similarityMatrix

Matrix of similarity between all points in the time series

neighborsNumber

Number of neighbors to consider affinity between nodes

## Details

Calculate the affinity matrix based on the similarity matrix. If the number of neighbors is equal to or greater than the similarity matrix then the similarity and affinity matrix are equal.

## Value

Affinity matrix based on the similarity matrix

## Author(s)

Luis Gustavo Uzai

---

`clusterEstimateNumber`*Estimate the number of possible clusters*

---

**Description**

Adaptation of the bartlett method of the speccalt package to estimate the number of clusters in the context of spectral clustering to detect change points

**Usage**

```
clusterEstimateNumber(eigenvectorValues, tolerance, maxClusterNumber)
```

**Arguments**

<code>eigenvectorValues</code>	Eigenvector matrix based on the affinity matrix
<code>tolerance</code>	approximation to consider valid clusters
<code>maxClusterNumber</code>	maximum number of calculable clusters

**Details**

Adaptation of the bartlett method of the speccalt package to estimate the number of clusters in the context of spectral clustering to detect change points

**Value**

An estimated number of clusters

**Author(s)**

Luis Gustavo Uzai

---

`convertToMatrixTimeSeries`*Converts the time series to position and value matrix*

---

**Description**

Converts the time series to position and value matrix

**Usage**

```
convertToMatrixTimeSeries(data)
```

**Arguments**

data                    List of values corresponding to the time series

**Details**

Gets a list of values of any size and creates a key and value array of all positions

**Value**

The key matrix and value of the time series.

**Author(s)**

Luis Gustavo Uzai

---

DEVICE1	<i>DEVICE1</i>
---------	----------------

---

**Description**

Derivation of RefrigerationDevices of the UCR Time Series Classification Repository These problems were taken from data recorded as part of government sponsored study called Powering the Nation. The intention was to collect behavioural data about how consumers use electricity within the home to help reduce the UK's carbon footprint.

**Usage**

DEVICE1

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

DEVICE2	<i>DEVICE2</i>
---------	----------------

---

**Description**

Derivation of RefrigerationDevices of the UCR Time Series Classification Repository These problems were taken from data recorded as part of government sponsored study called Powering the Nation. The intention was to collect behavioural data about how consumers use electricity within the home to help reduce the UK's carbon footprint.

**Usage**

DEVICE2

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

DEVICE3

*DEVICE3*

---

**Description**

Derivation of RefrigerationDevices of the UCR Time Series Classification Repository These problems were taken from data recorded as part of government sponsored study called Powering the Nation. The intention was to collect behavioural data about how consumers use electricity within the home to help reduce the UK's carbon footprint.

**Usage**

DEVICE3

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

DEVICE4

*DEVICE4*

---

**Description**

Derivation of RefrigerationDevices of the UCR Time Series Classification Repository These problems were taken from data recorded as part of government sponsored study called Powering the Nation. The intention was to collect behavioural data about how consumers use electricity within the home to help reduce the UK's carbon footprint.

**Usage**

DEVICE4

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

DEVICE5

*DEVICE5*

---

**Description**

Derivation of RefrigerationDevices of the UCR Time Series Classification Repository These problems were taken from data recorded as part of government sponsored study called Powering the Nation. The intention was to collect behavioural data about how consumers use electricity within the home to help reduce the UK's carbon footprint.

**Usage**

DEVICE5

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

DEVICE6

*DEVICE6*

---

**Description**

Derivation of RefrigerationDevices of the UCR Time Series Classification Repository These problems were taken from data recorded as part of government sponsored study called Powering the Nation. The intention was to collect behavioural data about how consumers use electricity within the home to help reduce the UK's carbon footprint.

**Usage**

DEVICE6

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

FTIR1

*FTIR1*

---

**Description**

Derivation of Meat of the UCR Time Series Classification Repository Food spectrographs are used in chemometrics to classify food types, a task that has obvious applications in food safety and quality assurance. The classes are chicken, pork and turkey.

**Usage**

FTIR1

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

FTIR2

*FTIR2*

---

**Description**

Derivation of Meat of the UCR Time Series Classification Repository Food spectrographs are used in chemometrics to classify food types, a task that has obvious applications in food safety and quality assurance. The classes are chicken, pork and turkey.

**Usage**

FTIR2

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

FTIR3

*FTIR3*

---

**Description**

Derivation of Meat of the UCR Time Series Classification Repository Food spectrographs are used in chemometrics to classify food types, a task that has obvious applications in food safety and quality assurance. The classes are chicken, pork and turkey.

**Usage**

FTIR3

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

FTIR4

*FTIR4*

---

**Description**

Derivation of Meat of the UCR Time Series Classification Repository Food spectrographs are used in chemometrics to classify food types, a task that has obvious applications in food safety and quality assurance. The classes are chicken, pork and turkey.

**Usage**

FTIR4

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

FTIR5

*FTIR5*

---

**Description**

Derivation of Meat of the UCR Time Series Classification Repository Food spectrographs are used in chemometrics to classify food types, a task that has obvious applications in food safety and quality assurance. The classes are chicken, pork and turkey.

**Usage**

FTIR5

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

FTIR6

*FTIR6*

---

**Description**

Derivation of Meat of the UCR Time Series Classification Repository Food spectrographs are used in chemometrics to classify food types, a task that has obvious applications in food safety and quality assurance. The classes are chicken, pork and turkey.

**Usage**

FTIR6

**Format**

The format is: Value Class 1.063400 1 -0.953410 1 ... -0.596090 2 ...

---

gaussianKernel      *Calculate Gaussian Kernel*

---

**Description**

Measure of similarity between two points represented by x1 and x2

**Usage**

```
gaussianKernel(x1, x2, alpha = 1)
```

**Arguments**

x1	first valor to compute
x2	second valor to compute
alpha	Alpha Measure

**Details**

Measure of similarity between two points represented by x1 and x2

**Value**

Measure of similarity between two points.

**Author(s)**

Luis Gustavo Uzai

---

generateEigenvectorMatrix  
*Calculate the eigenvector of the affinity matrix*

---

**Description**

Calculate the eigenvector of the affinity matrix

**Usage**

```
generateEigenvectorMatrix(affinityMatrix)
```

**Arguments**

affinityMatrix	Affinity matrix based on the similarity matrix based on key and value matrix of the time series
----------------	---

**Details**

Calculates the laplacian matrix based on the affinity matrix and calculates the auto values of the graph with the eigen function

**Value**

Eigenvector matrix based on the affinity matrix

**Author(s)**

Luis Gustavo Uzai

---

`generateSimilarityMatrix`  
*Calculate Similarity Matrix*

---

**Description**

Use some similarity measure to calculate the similarity matrix

**Usage**

```
generateSimilarityMatrix(data, similarityMeasure)
```

**Arguments**

`data`                    Key and value matrix of a time series  
`similarityMeasure`        Measure of similarity between two points represented by x1 and x2

**Details**

Use some similarity measure to calculate the similarity matrix

**Value**

Matrix of similarity calculated from the key and value matrix.

**Author(s)**

Luis Gustavo Uzai

---

getClusterFact	<i>Get the Factor of the cluster position in relation to the matrix of eigenvectors</i>
----------------	---

---

**Description**

Get the Factor of the cluster position in relation to the matrix of eigenvectors

**Usage**

```
getClusterFact(eigenvectorValues, eigenvectorLengthLessOne, clusterNumber,
reverseClusterNumber)
```

**Arguments**

eigenvectorValues	Eigenvector matrix based on the affinity matrix
eigenvectorLengthLessOne	the eigenvector matrix size minus 1
clusterNumber	the cluster position number being tested
reverseClusterNumber	the number of the inverse position of the cluster being tested

**Details**

Gets the factor of the value and its opposite in relation to the matrix of the eigenvectors

**Value**

Factor of the cluster position in relation to the matrix of eigenvectors

**Author(s)**

Luis Gustavo Uzai

---

getClusterProd	<i>Get the Product of the cluster position in relation to the matrix of eigenvectors</i>
----------------	--

---

**Description**

Get the Product of the cluster position in relation to the matrix of eigenvectors

**Usage**

```
getClusterProd(eigenvectorValues, eigenvectorLengthLessOne, clusterNumber,
reverseClusterNumber)
```

**Arguments**

- eigenvectorValues  
Eigenvector matrix based on the affinity matrix
- eigenvectorLengthLessOne  
the eigenvector matrix size minus 1
- clusterNumber the cluster position number being tested
- reverseClusterNumber  
the number of the inverse position of the cluster being tested

**Details**

Gets the product of the value and its opposite in relation to the matrix of the eigenvectors

**Value**

Product of the cluster position in relation to the matrix of eigenvectors

**Author(s)**

Luis Gustavo Uzai

---

*getSpectralClusters*     *Clustering with the smallest eigenvectors from eigenvector Matrix*

---

**Description**

Clustering with the smallest eigenvectors from eigenvector Matrix

**Usage**

```
getSpectralClusters(eigenvectorMatrix, numberOfClusters = 2)
```

**Arguments**

- eigenvectorMatrix  
Eigenvector matrix based on the affinity matrix
- numberOfClusters  
maximum number of clusters for prediction

**Details**

Modified standard function present in kernlab to perform clustering with graph spectrum using standard version of K-Means

**Value**

K-Means Cluster Object

**Author(s)**

Luis Gustavo Uzai

Spec

*Calculate change points with spectral cluster***Description**

Calculate change point based on spectral clustering you have the option to automatically calculate the number of clusters if this information is not available

**Usage**

```
Spec(data, neighborsNumber = 5, tolerance = 0.01,
      maxNumberOfChangePoints = 19, estimationChangePointsNumber = NULL)
```

**Arguments**

data	List of values corresponding to the time series
neighborsNumber	Number of neighbors to consider affinity between nodes
tolerance	approximation to consider valid clusters, used only for calculation of forecast of change points, default 0.01
maxNumberOfChangePoints	maximum number of clusters for prediction : default 19
estimationChangePointsNumber	predicted number of change points in the series, if null, is automatically calculated: default null

**Details**

Calculate change point based on spectral clustering you have the option to automatically calculate the number of clusters if this information is not available. It uses the Gaussian Kernel for the calculation of affinity matrix and Kmeans for the spectral cluster, however, several other options can be used and the package must be customized to better suit the use.

**Value**

Numerical array with the position of the change points in the time series

**Author(s)**

Luis Gustavo Uzai

**Examples**

```
data <- DEVICE1[, 1]
realChangePoints <- c(which(diff(DEVICE1$Class) != 0))
calculateChangePoints <- Spec(data, neighborsNumber = 6,
                             tolerance = 0.005, estimationChangePointsNumber = 2)
minValue <- -99999
maxValue <- 99999
plot(data, type = "l", xlab = "x", ylab = "y")
for (r in 1:length(realChangePoints)) {
  lines(x = c(realChangePoints[r], realChangePoints[r]),
        y = c(minValue, maxValue), lwd = 2, col = "red")
}
for (n in 1:length(calculateChangePoints)) {
  lines(x = c(calculateChangePoints[n], calculateChangePoints[n]),
        y = c(minValue, maxValue), lwd = 2, col = "blue")
}
```

# Index

## \* datasets

DEVICE1, 4  
DEVICE2, 4  
DEVICE3, 5  
DEVICE4, 5  
DEVICE5, 6  
DEVICE6, 6  
FTIR1, 7  
FTIR2, 7  
FTIR3, 8  
FTIR4, 8  
FTIR5, 9  
FTIR6, 9

calculateAffinityMatrix, 2  
clusterEstimateNumber, 3  
convertToMatrixTimeSeries, 3

DEVICE1, 4  
DEVICE2, 4  
DEVICE3, 5  
DEVICE4, 5  
DEVICE5, 6  
DEVICE6, 6

FTIR1, 7  
FTIR2, 7  
FTIR3, 8  
FTIR4, 8  
FTIR5, 9  
FTIR6, 9

gaussianKernel, 10  
generateEigenvectorMatrix, 10  
generateSimilarityMatrix, 11  
getClusterFact, 12  
getClusterProd, 12  
getSpectralClusters, 13

Spec, 14