Package 'Silhouette'

July 30, 2025

Type Package

Title Proximity Measure Based Diagnostics for Standard, Soft, and Multi-Way Clustering

Version 0.9.4

Language en-US

Maintainer Shrikrishna Bhat K <skbhat.in@gmail.com>

Description Quantifies clustering quality by measuring both cohesion within clusters and separation between clusters. Implements advanced silhouette width computations for diverse clustering structures, including: simplified silhouette (Van der Laan et al., 2003) <doi:10.1080/0094965031000136012>, Probability of Alternative Cluster normalization methods (Raymaekers & Rousseeuw, 2022) <doi:10.1080/10618600.2022.2050249>, fuzzy clustering and silhouette diagnostics using membership probabilities (Campello & Hruschka, 2006; Bhat & Kiruthika, 2024) <doi:10.1016/j.fss.2006.07.006>, <doi:10.1080/23737484.2024.2408534>, and mul
way clustering extensions such as block and tensor clustering (Schepers et al., 2008; Bhat & Kiruthika, 2025) <doi:10.1007/s00357-008-9005-9>, <doi:10.21203/rs.3.rs-6973596/v1>. Provides tools for computation and visualization (Rousseeuw, 1987) <doi:10.1016/0377-0427(87)90125-7> to support robust and reproducible cluster diagnostics across standard, soft, and multi-way clustering settings.

License GPL-2

URL https://kskbhat.github.io/Silhouette/

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, ggplot2, ggpubr, methods

Suggests knitr, rmarkdown, testthat (>= 3.0.0), ppclust, blockcluster, cluster, factoextra, proxy

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Shrikrishna Bhat K [aut, cre, cph] (ORCID: <https://orcid.org/0009-0000-6180-5783>),
Kiruthika C [aut] (ORCID: <https://orcid.org/0000-0001-9655-702X>)

Repository CRAN

Date/Publication 2025-07-30 08:20:02 UTC

Contents

`extSilhouette` *Calculate Extended Silhouette Width for Multi-Way Clustering*

Description

Computes an extended silhouette width for multi-way clustering (e.g., biclustering, triclustering, or n-mode tensor clustering) by combining silhouette widths from a list of Silhouette objects, each representing one mode of clustering. The extended silhouette width is the weighted average of the average silhouette widths from each mode, weighted by the number of observations in each mode's silhouette analysis. The output is an object of class `extSilhouette`.

Usage

```
extSilhouette(sil_list, dim_names = NULL, print.summary = FALSE)
```

Arguments

`sil_list`	A list of objects of class `"Silhouette"`, typically the output of `Silhouette` or `softSilhouette`, where each object represents the silhouette analysis for one mode of multi-way clustering (e.g., rows, columns, or other dimensions in biclustering or tensor clustering).
`dim_names`	An optional character vector of dimension names (e.g., `c("Rows", "Columns")`). If `NULL`, defaults to `"Mode 1"`, `"Mode 2"`, etc.
`print.summary`	Logical; if `TRUE`, prints a summary of the extended silhouette width and dimension table. Default is `FALSE`.

Details

The extended silhouette width is computed as:

$$ExS = \frac{\sum(n_i \cdot w_i)}{\sum n_i}$$

where n_i is the number of observations in mode i (derived from `nrow(x$widths)`), and w_i is the average silhouette width for that mode (from `x$avg.width`). Each `Silhouette` object in `sil_list` must contain a non-empty `widths` data frame and a numeric `avg.width` value. Modes with zero observations ($n_i = 0$) are not allowed, as they would result in an undefined weighted average. For consistency make sure all `Silhouette` objects derived from same `method` and arguments.

Value

A list of class `"extSilhouette"` with the following components:

ext_sil_width A numeric scalar representing the extended silhouette width.

dim_table A data frame with columns `dimension` (e.g., "Mode 1", "Mode 2"), `n_obs` (number of observations), and `avg_sil_width` (average silhouette width for each mode).

References

Schepers, J., Ceulemans, E., & Van Mechelen, I. (2008). Selecting among multi-mode partitioning models of different complexities: A comparison of four model selection criteria. *Journal of Classification*, 25(1), 67–85. doi:10.1007/s0035700890059

Bhat Kapu, S., & Kiruthika, C. (2025). Block Probabilistic Distance Clustering: A Unified Framework and Evaluation. PREPRINT (Version 1) available at Research Square. doi:10.21203/rs.3.rs-6973596/v1

See Also

`Silhouette`, `softSilhouette`

Examples

```
# Example using iris dataset with two modes
data(iris)

if (requireNamespace("blockcluster", quietly = TRUE)) {
  library(blockcluster)
  result <- coclusterContinuous(
    as.matrix(iris[, -5]),
    nbcocluster = c(3, 2)
  )
} else {
  message("Install 'blockcluster': install.packages('blockcluster')")
}

if (requireNamespace("blockcluster", quietly = TRUE)) {
  sil_mode1 <- softSilhouette(
    prob_matrix = result@rowposteriorprob,
    method = "pac")
  sil_mode2 <- softSilhouette(
    prob_matrix = result@colposteriorprob,
    method = "pac"
    )

  # Extended silhouette
  ext_sil <- extSilhouette(list(sil_mode1, sil_mode2),print.summary = TRUE)
}
```

plotSilhouette *Plot Silhouette Analysis Results*

Description

Creates a silhouette plot for visualizing the silhouette widths of clustering results, with bars colored by cluster and an optional summary of cluster statistics in legend.

Usage

```
plotSilhouette(x, label = FALSE, summary.legend = TRUE, grayscale = FALSE, ...)
```

Arguments

`x`	An object of class `"Silhouette"`, typically the output of the `Silhouette` and `softSilhouette` function. Also supports objects classes `eclust`, `hcut`, `pam`, `clara`, `fanny`, or `silhouette` from **cluster**, **factoextra** packages. For these classes, explicitly call `plotSilhouette()` to generate the plot.
`label`	Logical; if `TRUE`, the x-axis is labeled with observation row indices from the input data and titled "Row Index". Defaults to `FALSE`.
`summary.legend`	Logical; if `TRUE`, prints a summary of average silhouette widths and sizes for each cluster in legend ("Cluster (Size): Width"). If `FALSE`, the legend shows only cluster numbers. Defaults to `TRUE`.
`grayscale`	Logical; if `TRUE`, the plot uses a grayscale color palette for clusters. If `FALSE`, uses the default or specified color palette. Defaults to `FALSE`.
`...`	Additional arguments passed to `ggpar` for customizing the plot (e.g., `palette`, `legend`).

Details

The Silhouette plot displays the silhouette width (`sil_width`) for each observation, grouped by cluster, with bars sorted by cluster and descending silhouette width. The `summary.legend` option adds cluster sizes and average silhouette widths to the legend.

This function replica of S3 method for objects of class `"Silhouette"`, typically produced by the `Silhouette` or `softSilhouette` functions in this package. It also supports objects of the following classes, with silhouette information extracted from their respective component:

- `"eclust"`: Produced by `eclust` from the **factoextra** package.
- `"hcut"`: Produced by `hcut` from the **factoextra** package.
- `"pam"`: Produced by `pam` from the **cluster** package.
- `"clara"`: Produced by `clara` from the **cluster** package.

- `"fanny"`: Produced by `fanny` from the **cluster** package.
- `"silhouette"`: Produced by `silhouette` from the **cluster** package.

For these classes (`"eclust"`, `"hcut"`, `"pam"`, `"clara"`, `"fanny"`, `"silhouette"`), users should explicitly call `plotSilhouette()` (e.g., `plotSilhouette(pam_result)`) to ensure the correct method is used, as the generic `plot()` may not dispatch to this function for these objects.

Value

A `ggplot2` object representing the Silhouette plot.

References

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20, 53–65. doi:10.1016/0377-0427(87)901257

See Also

`Silhouette`, `softSilhouette`, `eclust`, `hcut`, `pam`, `clara`, `fanny`, `silhouette`

Examples

```
data(iris)

# Crisp Silhouette with k-means
out <- kmeans(iris[, -5], 3)
if (requireNamespace("proxy", quietly = TRUE)) {
  library(proxy)
  dist <- dist(iris[, -5], out$centers)
  plot(Silhouette(dist))
}

#' # Fuzzy Silhouette with ppclust::fcm
if (requireNamespace("ppclust", quietly = TRUE)) {
  library(ppclust)
  out_fuzzy <- Silhouette(
    prox_matrix = "d",
    proximity_type = "dissimilarity",
    prob_matrix = "u",
    clust_fun = ppclust::fcm,
    x = iris[, 1:4],
    centers = 3,
    sort = TRUE
  )
  plot(out_fuzzy, summary.legend = FALSE, grayscale = TRUE)
} else {
  message("Install 'ppclust': install.packages('ppclust')")
}

# Silhouette plot for pam clustering
if (requireNamespace("cluster", quietly = TRUE)) {
  library(cluster)
  pam_result <- pam(iris[, 1:4], k = 3)
  plotSilhouette(pam_result)
}

# Silhouette plot for clara clustering
if (requireNamespace("cluster", quietly = TRUE)) {
  clara_result <- clara(iris[, 1:4], k = 3)
  plotSilhouette(clara_result)
}

# Silhouette plot for fanny clustering
if (requireNamespace("cluster", quietly = TRUE)) {
  fanny_result <- fanny(iris[, 1:4], k = 3)
  plotSilhouette(fanny_result)
}

# Example using base silhouette() object
if (requireNamespace("cluster", quietly = TRUE)) {
  sil <- silhouette(pam_result)
  plotSilhouette(sil)
}

# Silhouette plot for eclust clustering
if (requireNamespace("factoextra", quietly = TRUE)) {
  library(factoextra)
  eclust_result <- eclust(iris[, 1:4], "kmeans", k = 3, graph = FALSE)
  plotSilhouette(eclust_result)
}

# Silhouette plot for hcut clustering
if (requireNamespace("factoextra", quietly = TRUE)) {
  hcut_result <- hcut(iris[, 1:4], k = 3)
  plotSilhouette(hcut_result)
}
```

Silhouette *Calculate Silhouette Widths, Summary, and Plot for Clustering Results*

Description

Computes the silhouette width for each observation based on clustering results, measuring how similar an observation is to its own cluster compared to nearest neighbor cluster. The silhouette width ranges from -1 to 1, where higher values indicate better cluster cohesion and separation.

Usage

```
Silhouette(
  prox_matrix,
  proximity_type = c("dissimilarity", "similarity"),
  method = c("medoid", "pac"),
  prob_matrix = NULL,
  a = 2,
  sort = FALSE,
  print.summary = FALSE,
  clust_fun = NULL,
  ...
)

## S3 method for class 'Silhouette'
plot(x, label = FALSE, summary.legend = TRUE, grayscale = FALSE, ...)

## S3 method for class 'Silhouette'
summary(object, print.summary = TRUE, ...)
```

Arguments

`prox_matrix`	A numeric matrix where rows represent observations and columns represent proximity measures (e.g., distances or similarities) to clusters. Typically, this is a membership or dissimilarity matrix from clustering results. If `clust_fun` is provided, `prox_matrix` should be the name of the matrix component as a string (e.g., if `clust_fun` = `fcm` from **ppclust** package the `prox_matrix` = `"d"`).
`proximity_type`	Character string specifying the type of proximity measure in `prox_matrix`. Options are `"similarity"` (higher values indicate closer proximity) or `"dissimilarity"` (lower values indicate closer proximity). Defaults to `"dissimilarity"`.
`method`	Character string specifying the silhouette calculation method. Options are `"pac"` (Probability of Alternative Cluster) or `"medoid"`. Defaults to `"medoid"`.
`prob_matrix`	A numeric matrix where rows represent observations and columns represent cluster membership probabilities, depending on `prob_type`). If `clust_fun` is provided, `prob_matrix` should be the name of the matrix component as a string (e.g., `"u"` for `fcm`). When not `NULL`, fuzzy silhouette width is calculated. Defaults to `NULL` for crisp silhouette.
`a`	Numeric value controlling the fuzzifier or weight scaling in fuzzy silhouette averaging. Higher values increase the emphasis on strong membership differences. Must be positive. Defaults to `2`.
`sort`	Logical; if `TRUE`, sorts the output `widths` data frame by cluster and descending silhouette width. Defaults to `FALSE`.
`print.summary`	Logical; if `TRUE`, prints a summary table of average silhouette widths and sizes for each cluster. Defaults to `TRUE`.
`clust_fun`	Optional S3 or S4 function object or function as character string specifying a clustering function that produces the proximity measure matrix. For example, `fcm` or `"fcm"`. If provided, `prox_matrix` must be the name of the matrix component in the clustering output (e.g., `"d"` for `fcm` when `proximity_type` = `"dissimilarity"`). Defaults to `NULL`.
`...`	Additional arguments passed to `clust_fun`, such as `x,centers` for `fcm`.

`x`	An object of class `"Silhouette"`, typically the output of the `Silhouette` and `softSilhouette` function. Also supports objects classes `eclust`, `hcut`, `pam`, `clara`, `fanny`, or `silhouette` from **cluster**, **factoextra** packages. For these classes, explicitly call `plotSilhouette()` to generate the plot.
`label`	Logical; if `TRUE`, the x-axis is labeled with observation row indices from the input data and titled "Row Index". Defaults to `FALSE`.
`summary.legend`	Logical; if `TRUE`, prints a summary of average silhouette widths and sizes for each cluster in legend ("Cluster (Size): Width"). If `FALSE`, the legend shows only cluster numbers. Defaults to `TRUE`.
`grayscale`	Logical; if `TRUE`, the plot uses a grayscale color palette for clusters. If `FALSE`, uses the default or specified color palette. Defaults to `FALSE`.
`object`	An object of class `"Silhouette"`, typically the output of the `Silhouette` and `softSilhouette` function.

Details

The `Silhouette` function implements the Simplified Silhouette method introduced by Van der Laan, Pollard, & Bryan (2003), which adapts and generalizes the classic silhouette method of Rousseeuw (1987).

Clustering quality is evaluated using a proximity matrix, denoted as $\Delta = [\delta_{ik}]_{n\times K}$ for dissimilarity measures or $\Delta' = [\delta'_{ik}]_{n\times K}$ for similarity measures. Here, $i = 1, \ldots, n$ indexes observations, and $k = 1, \ldots, K$ indexes clusters. δ_{ik} represents the dissimilarity (e.g., distance) between observation i and cluster k, while δ'_{ik} represents similarity values.

The silhouette width $S(x_i)$ for observation i depends on the proximity type:

For **dissimilarity** measures:

$$S(x_i) = \frac{\min_{k'\neq k} \delta_{ik'} - \delta_{ik}}{N(x_i)}$$

For **similarity** measures:

$$S(x_i) = \frac{\delta'_{ik} - \max_{k'\neq k} \delta'_{ik'}}{N(x_i)}$$

where $N(x_i)$ is a normalizing factor defined by the method.

Choice of method: The normalizer $N(x_i)$ is selected according to the `method` argument. The method names reference their origins but may be used with any proximity matrix, not exclusively certain clustering algorithms:

- For `medoid` (Van der Laan et al., 2003):
  - Dissimilarity: $\max(\delta_{ik}, \min_{k'\neq k} \delta_{ik'})$
  - Similarity: $\max(\delta'_{ik}, \max_{k'\neq k} \delta'_{ik'})$
- For `pac` (Raymaekers & Rousseeuw, 2022):
  - Dissimilarity: $\delta_{ik} + \min_{k'\neq k} \delta_{ik'}$
  - Similarity: $\delta'_{ik} + \max_{k'\neq k} \delta'_{ik'}$

Note: The `"medoid"` and `"pac"` options reflect the normalization formula—not a requirement to use the PAM algorithm or posterior/ensemble methods—and are general scoring approaches. These methods can be applied to any suitable proximity matrix, including proximity, similarity, or dissimilarity matrices derived from **classification algorithms**. This flexibility means silhouette indices may be computed to assess group separation when clusters or groups are formed from classification-derived proximities, not only from unsupervised clustering.

If `prob_matrix` is `NULL`, the **crisp silhouette index** (CS) is:

$$CS = \frac{1}{n} \sum_{i=1}^{n} S(x_i)$$

summarizing overall clustering quality.

If `prob_matrix` is provided, denoted as $\Gamma = [\gamma_{ik}]_{n \times K}$, with γ_{ik} representing the probability of observation i belonging to cluster k, the **fuzzy silhouette index** (FS) is used:

$$FS = \frac{\sum_{i=1}^{n} \left(\gamma_{ik} - \max_{k' \neq k} \gamma_{ik'} \right)^{\alpha} S(x_i)}{\sum_{i=1}^{n} \left(\gamma_{ik} - \max_{k' \neq k} \gamma_{ik'} \right)^{\alpha}}$$

where α (the `a` argument) controls the emphasis on confident assignments.

Value

A data frame of class `"Silhouette"` containing cluster assignments, nearest neighbor clusters, silhouette widths for each observation, and weights (for fuzzy clustering). The object includes the following attributes:

proximity_type The proximity type used (`"similarity"` or `"dissimilarity"`).

method The silhouette calculation method used (`"medoid"` or `"pac"`).

Further, `summary` returns a list containing:

- `clus.avg.widths`: A named numeric vector of average silhouette widths per cluster.
- `avg.width`: The overall average silhouette width.
- `sil.sum`: A data frame with columns `cluster`, `size`, and `avg.sil.width` summarizing cluster sizes and average silhouette widths.

References

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20, 53–65. doi:10.1016/0377-0427(87)901257

Van der Laan, M., Pollard, K., & Bryan, J. (2003). A new partitioning around medoids algorithm. *Journal of Statistical Computation and Simulation*, 73(8), 575–584. doi:10.1080/0094965031000136012

Campello, R. J., & Hruschka, E. R. (2006). A fuzzy extension of the silhouette width criterion for cluster analysis. *Fuzzy Sets and Systems*, 157(21), 2858–2875. doi:10.1016/j.fss.2006.07.006

Raymaekers, J., & Rousseeuw, P. J. (2022). Silhouettes and quasi residual plots for neural nets and tree-based classifiers. *Journal of Computational and Graphical Statistics*, 31(4), 1332–1343. doi:10.1080/10618600.2022.2050249

Bhat Kapu, S., & Kiruthika. (2024). Some density-based silhouette diagnostics for soft clustering algorithms. Communications in Statistics: Case Studies, Data Analysis and Applications, 10(3-4), 221-238. doi:10.1080/23737484.2024.2408534

See Also

softSilhouette, plotSilhouette

Examples

```
# Standard silhouette with k-means on iris dataset
data(iris)
# Crisp Silhouette with k-means
out <- kmeans(iris[, -5], 3)
if (requireNamespace("ppclust", quietly = TRUE)) {
  library(proxy)
  dist <- proxy::dist(iris[, -5], out$centers)
  silh_out <- Silhouette(dist,print.summary = TRUE)
  plot(silh_out)
} else {
  message("Install 'ppclust': install.packages('ppclust')")
}

# Scree plot for optimal clusters (2 to 7)
if (requireNamespace("ppclust", quietly = TRUE)) {
  library(ppclust)
  avg_sil_width <- numeric(6)
  for (k in 2:7) {
    out <- Silhouette(
      prox_matrix = "d",
      proximity_type = "dissimilarity",
      prob_matrix = "u",
      clust_fun = ppclust::fcm,
      x = iris[, 1:4],
      centers = k,
      sort = TRUE
    )
    # Compute average silhouette width from widths
    avg_sil_width[k - 1] <- summary(out, print.summary = FALSE)$avg.width
  }
  plot(avg_sil_width,
    type = "o",
    ylab = "Overall Silhouette Width",
    xlab = "Number of Clusters",
    main = "Scree Plot"
  )
} else {
  message("Install 'ppclust': install.packages('ppclust')")
}
```

---

softSilhouette *Calculate Silhouette Width for Soft Clustering Algorithms*

Description

Computes silhouette widths for soft clustering results by interpreting cluster membership probabilities (or their transformations) as proximity measures. Although originally designed for evaluating clustering quality within a method, this adaptation allows heuristic comparison across soft clustering algorithms using average silhouette widths.

Usage

```
softSilhouette(
  prob_matrix,
  prob_type = c("pp", "nlpp", "pd"),
  method = c("pac", "medoid"),
  average = c("crisp", "fuzzy"),
  a = 2,
  print.summary = FALSE,
  clust_fun = NULL,
  ...
)
```

Arguments

`prob_matrix`	A numeric matrix where rows represent observations and columns represent cluster membership probabilities (or transformed probabilities, depending on `prob_type`). If `clust_fun` is provided, `prob_matrix` should be the name of the matrix component as a string (e.g., `"u"` for `fcm`).
`prob_type`	Character string specifying the type transformation of membership matrix considered as proximity matrix in `prob_matrix`. Options are: `"pp"` Posterior probabilities $[\gamma_{ik}]_{n\times K}$ (non-negative, typically summing to 1 per row), treated as similarities; `"nlpp"` Negative log of posterior probabilities $[-\ln\gamma_{ik}]_{n\times K}$ (non-positive), treated as dissimilarities.; `"pd"` Probability distribution $[\gamma_{ik}/\pi_k]_{n\times K}$ (normalized posterior probabilities relative to cluster proportions π_k), treated as similarities. Defaults to `"pp"`.
`method`	Character string specifying the silhouette calculation method. Options are `"pac"` (Probability of Alternative Cluster) or `"medoid"`. Defaults to `"pac"`.
`average`	Character string specifying the type of average silhouette width calculation. Options are `"crisp"` (simple average) or `"fuzzy"` (weighted average based on membership differences). Defaults to `"crisp"`.
`a`	Numeric value controlling the fuzzifier or weight scaling in fuzzy silhouette averaging. Higher values increase the emphasis on strong membership differences. Must be positive. Defaults to `2`.
`print.summary`	Logical; if `TRUE`, prints a summary table of average silhouette widths and sizes for each cluster. Defaults to `FALSE`.

`clust_fun`	Optional S3 or S4 function object or function as character string specifying a clustering function that produces the proximity measure matrix. For example, `fcm` or `"fcm"`. If provided, `prox_matrix` must be the name of the matrix component in the clustering output (e.g., `"d"` for `fcm` when `proximity_type = "dissimilarity"`). Defaults to `NULL`.
`...`	Additional arguments passed to `clust_fun`, such as `x,centers` for `fcm`.

Details

Although the silhouette method was originally developed for evaluating clustering structure within a single result, this implementation allows leveraging cluster membership probabilities from soft clustering methods to construct proximity-based silhouettes. These silhouette widths can be compared heuristically across different algorithms to assess clustering quality.

See doi:10.1080/23737484.2024.2408534 for more details.

Value

A data frame of class `"Silhouette"` containing cluster assignments, nearest neighbor clusters, silhouette widths for each observation, and weights (for fuzzy clustering). The object includes the following attributes:

proximity_type The proximity type used (`"similarity"` or `"dissimilarity"`).

method The silhouette calculation method used (`"medoid"` or `"pac"`).

References

Raymaekers, J., & Rousseeuw, P. J. (2022). Silhouettes and quasi residual plots for neural nets and tree-based classifiers. *Journal of Computational and Graphical Statistics*, 31(4), 1332–1343. doi:10.1080/10618600.2022.2050249

Bhat Kapu, S., & Kiruthika. (2024). Some density-based silhouette diagnostics for soft clustering algorithms. Communications in Statistics: Case Studies, Data Analysis and Applications, 10(3-4), 221-238. doi:10.1080/23737484.2024.2408534

See Also

`Silhouette`,`plotSilhouette`

Examples

```
# Compare two soft clustering algorithms using softSilhouett
# Example: FCM vs. FCM2 on iris data, using average silhouette width as a criterion
data(iris)
if (requireNamespace("ppclust", quietly = TRUE)) {
  fcm_result <- ppclust::fcm(iris[, 1:4], 3)
  out_fcm <- softSilhouette(prob_matrix = fcm_result$u,print.summary = TRUE)
  plot(out_fcm)
  sfcm <- summary(out_fcm, print.summary = FALSE)
} else {
  message("Install 'ppclust' to run this example: install.packages('ppclust')")
}
```

```
if (requireNamespace("ppclust", quietly = TRUE)) {
  fcm2_result <- ppclust::fcm2(iris[, 1:4], 3)
  out_fcm2 <- softSilhouette(prob_matrix = fcm2_result$u,print.summary = TRUE)
  plot(out_fcm2)
  sfcm2 <- summary(out_fcm2, print.summary = FALSE)
} else {
  message("Install 'ppclust' to run this example: install.packages('ppclust')")
}
# Compare average silhouette widths of fcm and fcm2
if (requireNamespace("ppclust", quietly = TRUE)) {
  cat("FCM average silhouette width:", sfcm$avg.width, "\n")
  cat("FCM2 average silhouette width:", sfcm2$avg.width, "\n")
}
```

Index