

Package ‘persistence’

September 7, 2026

Type Package

Title Optimal Graph Partition using the Persistence

Version 1.0.1

Description Calculate the optimal vertex partition of a graph using the persistence as objective function.
These subroutines have been used in Avellone et al.
<[doi:10.1007/s10288-023-00559-z](https://doi.org/10.1007/s10288-023-00559-z)> and Avellone et al.
<[doi:10.1016/j.ins.2025.123032](https://doi.org/10.1016/j.ins.2025.123032)>.
This package is deprecated and has been superseded by the 'scalednap' package, which provides the same functionality and additional features; new and existing users should install 'scalednap' instead.

License GPL (>= 2)

Encoding UTF-8

SystemRequirements C++20

Suggests igraph

NeedsCompilation yes

Collate 'persistence-exports.R' 'cluster_milano.R'
'global_persistence.R' 'local_persistence.R' 'zzz.R'

Config/roxygen2/version 8.0.0

Author Alessandro Avellone [aut, cre],
Paolo Bartesaghi [aut],
Stefano Benati [aut],
Rosanna Grassi [aut]

Maintainer Alessandro Avellone <alessandro.avellone@unimib.it>

Repository CRAN

Date/Publication 2026-09-07 16:30:02 UTC

Contents

persistence-package	2
cluster_milano	2
global_persistence	4
local_persistence	5

Index

7

persistence-package *Persistence (deprecated)*

Description

Given a non-oriented graph, calculates the optimal vertex partition using persistence as the objective function.

This package is deprecated and has been superseded by the **scalednap** package, a strict superset that provides the same functions and additional functionality. Please install **scalednap** instead (`install.packages("scalednap")`); the same algorithm is also available for Python (`pip install scalednap`).

Details

See manual entries.

Author(s)

Maintainer: Alessandro Avellone <alessandro.avellone@unimib.it>

Authors:

- Alessandro Avellone <alessandro.avellone@unimib.it>
- Paolo Bartesaghi <paolo.bartesaghi@unimi.it>
- Stefano Benati <stefano.benati@unitn.it>
- Rosanna Grassi <rosanna.grassi@unimib.it>

cluster_milano *cluster_milano*

Description

Calculates the vertex partition with maximum global null-adjusted persistence. This function is polymorphic: it automatically detects the input type and accepts either a vertex vector (accompanied by an edge list) or directly an igraph object.

Usage

```
cluster_milano(x, ...)

## Default S3 method:
cluster_milano(
  x,
  edge_list,
  weights = NULL,
  membership = NULL,
  H0 = TRUE,
  seed = NULL,
  tol = NULL,
  max_level = 0L,
  ...
)

## S3 method for class 'igraph'
cluster_milano(
  x,
  membership = NULL,
  H0 = TRUE,
  seed = NULL,
  tol = NULL,
  max_level = 0L,
  ...
)
```

Arguments

<code>x</code>	An integer or character vector representing the graph vertices, OR an object of class <code>igraph</code> .
<code>...</code>	Additional arguments passed to specific methods (e.g., <code>edge_list</code> , <code>weights</code> , <code>tol</code> , <code>max_level</code> , etc.).
<code>edge_list</code>	Integer matrix with two columns representing the graph edge list.
<code>weights</code>	Numeric vector of positive edge weights. If <code>NULL</code> , all weights default to 1.
<code>membership</code>	Integer vector representing the starting partition: $x_i = k$ if i in C_k . If <code>NULL</code> , each vertex starts in its own cluster.
<code>H0</code>	Logical value. Default is <code>TRUE</code> . If <code>TRUE</code> , returns the null-adjusted persistence. If <code>FALSE</code> , returns the persistence probability.
<code>seed</code>	Non-negative integer seed for the random number generator. If <code>NULL</code> , an internal default is used.
<code>tol</code>	Optional numeric tolerance for the stopping criterion. If <code>NULL</code> (default), an adaptive threshold is calculated dynamically in C++.
<code>max_level</code>	Optional integer representing the maximum number of aggregation levels. If <code>0</code> (default) or <code>NULL</code> , the algorithm runs until convergence.

Value

A list with three elements:

membership The optimal vertex partition.

score The measure value of the optimal partition.

seed The seed used to generate random numbers.

Examples

```
library(persistence)

# --- EXAMPLE 1: Standard input (vectors and matrices) ---
edg <- c(1, 2, 1, 3, 1, 4, 2, 3, 3, 4, 4, 5, 5, 6, 5, 7, 6, 7)
edge_list <- matrix(edg, ncol = 2, byrow = TRUE)
vertex <- c(1, 2, 3, 4, 5, 6, 7)
cluster_milano(x = vertex, edge_list = edge_list)

# --- EXAMPLE 2: igraph input ---
if (requireNamespace("igraph", quietly = TRUE)) {
  g <- igraph::make_ring(10)
  cluster_milano(g)
}
```

global_persistence	<i>global_persistence</i>
--------------------	---------------------------

Description

Given a partition of the graph vertices, calculates the global persistence as the sum of the local persistences of the individual clusters. Persistence can be either null-adjusted or probability-based. This function is polymorphic: it automatically detects the input type and accepts either a vertex vector (accompanied by an edge list) or directly an igraph object.

Usage

```
global_persistence(x, ...)

## Default S3 method:
global_persistence(x, edge_list, weights = NULL, membership, H0 = 0, ...)

## S3 method for class 'igraph'
global_persistence(x, membership, H0 = 0, ...)
```

Arguments

<code>x</code>	An integer or character vector representing the graph vertices, OR an object of class <code>igraph</code> .
<code>...</code>	Additional arguments passed to specific methods (e.g., <code>edge_list</code> , <code>weights</code> , <code>membership</code> , <code>H0</code>).
<code>edge_list</code>	Integer matrix with two columns representing the graph edge list.
<code>weights</code>	Numeric vector of edge weights. If <code>NULL</code> , all weights default to 1.
<code>membership</code>	Integer vector of vertex cluster assignments: $x_i = k$ if i in C_k .
<code>H0</code>	Optional numeric value in $[0, 1]$, or <code>NULL</code> . Default is 0.

Value

A list with two elements:

score The global persistence of the partition.

clusters_value The local persistence of each cluster. A value of `NaN` indicates that cluster C_k is empty in the input `membership`.

Examples

```
library(persistence)

# --- EXAMPLE 1: Standard input (vectors and matrices) ---
edg <- c(1, 2, 1, 3, 1, 4, 2, 3, 3, 4, 4, 5, 5, 6, 5, 7, 6, 7)
edge_list <- matrix(edg, ncol = 2, byrow = TRUE)
vertex <- c(1, 2, 3, 4, 5, 6, 7)
mem <- c(1, 1, 1, 1, 2, 2, 2)
global_persistence(x = vertex, edge_list = edge_list, membership = mem)

# --- EXAMPLE 2: igraph input ---
if (requireNamespace("igraph", quietly = TRUE)) {
  g <- igraph::make_ring(10)
  mem <- c(rep(1, 5), rep(2, 5))
  global_persistence(g, membership = mem)
}
```

<code>local_persistence</code>	<i>local_persistence</i>
--------------------------------	--------------------------

Description

Given the incidence vector of a vertex subset, calculates either the persistence probability or the null-adjusted persistence of cluster C . This function is polymorphic: it automatically detects the input type and accepts either a vertex vector (accompanied by an edge list) or directly an `igraph` object.

Usage

```

local_persistence(x, ...)

## Default S3 method:
local_persistence(x, edge_list, weights = NULL, cluster, H0 = 0, ...)

## S3 method for class 'igraph'
local_persistence(x, cluster, H0 = 0, ...)

```

Arguments

<code>x</code>	An integer or character vector representing the graph vertices, OR an object of class <code>igraph</code> .
<code>...</code>	Additional arguments passed to specific methods (e.g., <code>edge_list</code> , <code>weights</code> , <code>cluster</code> , <code>H0</code>).
<code>edge_list</code>	Integer matrix with two columns representing the graph edge list.
<code>weights</code>	Numeric vector of edge weights. If <code>NULL</code> , all weights default to 1.
<code>cluster</code>	Binary incidence vector of the cluster: $x_i = 1$ if i in C , 0 otherwise.
<code>H0</code>	Optional numeric value in $[0, 1]$, or <code>NULL</code> . Default is 0.

Value

Numeric scalar: the persistence probability when `H0 = NULL`, the null-adjusted persistence when `H0 = 0`, or the null-adjusted persistence density when `H0` is in $(0, 1]$.

Examples

```

library(persistence)

# --- EXAMPLE 1: Standard input (vectors and matrices) ---
edg <- c(1, 2, 1, 3, 1, 4, 2, 3, 3, 4, 4, 5, 5, 6, 5, 7, 6, 7)
edge_list <- matrix(edg, ncol = 2, byrow = TRUE)
vertex <- c(1, 2, 3, 4, 5, 6, 7)
cluster_bin <- c(1, 1, 1, 1, 0, 0, 0)
local_persistence(x = vertex, edge_list = edge_list, cluster = cluster_bin)

# --- EXAMPLE 2: igraph input ---
if (requireNamespace("igraph", quietly = TRUE)) {
  g <- igraph::make_ring(10)
  cluster_bin <- c(rep(1, 5), rep(0, 5))
  local_persistence(g, cluster = cluster_bin)
}

```

Index

`cluster_milano`, [2](#)

`global_persistence`, [4](#)

`local_persistence`, [5](#)

`persistence (persistence-package)`, [2](#)

`persistence-package`, [2](#)