Package ‘TDAvec’

October 31, 2022

Type  Package
Title  Vector Summaries of Persistence Diagrams
Version  0.1.3
Description  Tools for computing various vector summaries of persistence diagrams studied in Topological Data Analysis. For improved computational efficiency, all code for the vector summaries is written in 'C++' using the 'Rcpp' package.
License  GPL (>= 2)
Encoding  UTF-8
Imports  Rcpp (>= 1.0.9), TDA, microbenchmark
LinkingTo  Rcpp
Suggests  knitr
VignetteBuilder  knitr
NeedsCompilation  yes
RoxygenNote  7.2.1
Author  Umar Islambekov [aut],
        Alexey Luchinsky [aut, cre],
        Hasani Pathirana [ctb]
Maintainer  Alexey Luchinsky <aluchi@bsu.edu>
Repository  CRAN
Date/Publication  2022-10-31 16:00:23 UTC

R topics documented:

  computeECC ................................................................. 2
  computeNL ................................................................. 3
  computePES ................................................................. 4
  computePI ................................................................. 5
  computePL ................................................................. 7
  computePS ................................................................. 8
  computeVAB ............................................................... 10
  computeVPB ............................................................... 11
computeECC

**A Vector Summary of the Euler Characteristic Curve**

**Description**

Vectorizes the Euler characteristic curve

\[ \chi(t) = \sum_{k=0}^{d} (-1)^k \beta_k(t), \]

where \( \beta_0, \beta_1, \ldots, \beta_d \) are the Betti curves corresponding to persistence diagrams \( D_0, D_1, \ldots, D_d \) of dimensions 0, 1, \ldots, \( d \) respectively, all computed from the same filtration.

**Usage**

computeECC(D, maxhomDim, scaleSeq)

**Arguments**

- \( D \): matrix with three columns containing the dimension, birth and death values respectively
- \( \text{maxhomDim} \): maximum homological dimension considered (0 for \( H_0 \), 1 for \( H_1 \), etc.)
- \( \text{scaleSeq} \): numeric vector of increasing scale values used for vectorization

**Value**

A numeric vector whose elements are the average values of the Euler characteristic curve computed between each pair of consecutive scale points of \( \text{scaleSeq} = \{t_1, t_2, \ldots, t_n\} \):

\[
\left( \frac{1}{\Delta t_1} \int_{t_1}^{t_2} \chi(t) dt, \frac{1}{\Delta t_2} \int_{t_2}^{t_3} \chi(t) dt, \ldots, \frac{1}{\Delta t_{n-1}} \int_{t_{n-1}}^{t_n} \chi(t) dt \right),
\]

where \( \Delta t_k = t_{k+1} - t_k \)

**Author(s)**

Umar Islambekov

**References**

computeNL

Examples

N <- 100
set.seed(123)
# sample N points uniformly from unit circle and add Gaussian noise
X <- TDA::circleUnif(N,r=1) + rnorm(2*N,mean = 0,sd = 0.2)

# compute a persistence diagram using the Rips filtration built on top of X
D <- TDA::ripsDiag(X,maxdimension = 1,maxscale = 2)$diagram

scaleSeq = seq(0,2,length.out=11) # sequence of scale values

# compute ECC
computeECC(D,maxhomDim=1,scaleSeq)

---

computeNL

A Vector Summary of the Normalized Life Curve

Description

For a given persistence diagram \( D = \{(b_i, d_i)\}_{i=1}^N \), computeNL() vectorizes the normalized life (NL) curve

\[
sl(t) = \frac{1}{L} \sum_{i=1}^{N} (d_i - b_i) \mathbf{1}_{[b_i, d_i)}(t),
\]

where \( L = \sum_{i=1}^{N} (d_i - b_i) \). Points of \( D \) with infinite death value are ignored.

Usage

computeNL(D, homDim, scaleSeq)

Arguments

D matrix with three columns containing the dimension, birth and death values respectively

homDim homological dimension (0 for \( H_0 \), 1 for \( H_1 \), etc.)

scaleSeq numeric vector of increasing scale values used for vectorization

Value

A numeric vector whose elements are the average values of the persistent entropy summary function computed between each pair of consecutive scale points of \( \text{scaleSeq}=\{t_1, t_2, \ldots, t_n\} \):

\[
\left( \frac{1}{\Delta t_1} \int_{t_1}^{t_2} sl(t)dt, \frac{1}{\Delta t_2} \int_{t_2}^{t_3} sl(t)dt, \ldots, \frac{1}{\Delta t_{n-1}} \int_{t_{n-1}}^{t_n} sl(t)dt \right),
\]

where \( \Delta t_k = t_{k+1} - t_k \).
References


Examples

```r
N <- 100
set.seed(123)
# sample N points uniformly from unit circle and add Gaussian noise
X <- TDA::circleUnif(N=r=1) + rnorm(2*N,mean = 0,sd = 0.2)

# compute a persistence diagram using the Rips filtration built on top of X
D <- TDA::ripsDiag(X,maxdimension = 1,maxscale = 2)$diagram

scaleSeq = seq(0,2,length.out=11) # sequence of scale values

# compute NL for homological dimension H_0
computeNL(D,homDim=0,scaleSeq)

# compute NL for homological dimension H_1
computeNL(D,homDim=1,scaleSeq)
```

computePES

A Vector Summary of the Persistent Entropy Summary Function

Description

For a given persistence diagram $D = \{(b_i, d_i)\}_{i=1}^N$, computePES() vectorizes the persistent entropy summary (PES) function

$$S(t) = -\sum_{i=1}^N \frac{l_i}{L} \log_2 \left( \frac{l_i}{L} \right) \mathbb{1}_{[b_i, d_i]}(t),$$

where $l_i = d_i - b_i$ and $L = \sum_{i=1}^N l_i$. Points of $D$ with infinite death value are ignored

Usage

`computePES(D, homDim, scaleSeq)`

Arguments

- `D`       matrix with three columns containing the dimension, birth and death values respectively
- `homDim`  homological dimension (0 for $H_0$, 1 for $H_1$, etc.)
- `scaleSeq` numeric vector of increasing scale values used for vectorization
computePI

A Vector Summary of the Persistence Surface

Value
A numeric vector whose elements are the average values of the persistent entropy summary function computed between each pair of consecutive scale points of scaleSeq={t_1, t_2, ..., t_n}:

\[
\left( \frac{1}{\Delta t_1} \int_{t_1}^{t_2} S(t)dt, \frac{1}{\Delta t_2} \int_{t_2}^{t_3} S(t)dt, \ldots, \frac{1}{\Delta t_{n-1}} \int_{t_{n-1}}^{t_n} S(t)dt \right),
\]

where \(\Delta t_k = t_{k+1} - t_k\).

Author(s)
Umar Islambekov

References

Examples
```r
N <- 100
set.seed(123)
# sample N points uniformly from unit circle and add Gaussian noise
X <- TDA::circleUnif(N, r=1) + rnorm(2*N, mean = 0, sd = 0.2)
# compute a persistence diagram using the Rips filtration built on top of X
D <- TDA::ripsDiag(X, maxdimension = 1, maxscale = 2)$diagram

scaleSeq = seq(0, 2, length.out=11) # sequence of scale values
# compute PES for homological dimension H_0
computePES(D, homDim=0, scaleSeq)

# compute PES for homological dimension H_1
computePES(D, homDim=1, scaleSeq)
```

Description
For a given persistence diagram \(D = \{(b_i, p_i)\}_{i=1}^N\), `computePI()` computes the persistence image (PI) - a vector summary of the persistence surface:

\[
\rho(x, y) = \sum_{i=1}^{N} f(b_i, p_i) \phi(b_i, p_i)(x, y),
\]
where $\phi(b_i, p_i)(x, y)$ is the Gaussian distribution with mean $(b_i, p_i)$ and covariance matrix $\sigma^2 I_{2 \times 2}$ and

$$f(b, p) = w(p) = \begin{cases} 0 & y \leq 0 \\ p/p_{\text{max}} & 0 < p < p_{\text{max}} \\ 1 & y \geq p_{\text{max}} \end{cases}$$

is the weighting function with $p_{\text{max}}$ being the maximum persistence value among all persistence diagrams considered in the experiment. Points of $D$ with infinite persistence value are ignored.

**Usage**

```r
computePI(D, homDim, xSeq, ySeq, sigma)
```

**Arguments**

- **D**
  - matrix with three columns containing the dimension, birth and persistence values respectively
- **homDim**
  - homological dimension (0 for $H_0$, 1 for $H_1$, etc.)
- **xSeq**
  - numeric vector of increasing $x$ (birth) values used for vectorization
- **ySeq**
  - numeric vector of increasing $y$ (persistence) values used for vectorization
- **sigma**
  - standard deviation of the Gaussian

**Value**

A numeric vector whose elements are the average values of the persistence surface computed over each cell of the two-dimensional grid constructed from $xSeq=\{x_1, x_2, \ldots, x_n\}$ and $ySeq=\{y_1, y_2, \ldots, y_m\}$:

$$\left(\frac{1}{\Delta x_1 \Delta y_1} \int_{[x_1, x_2] \times [y_1, y_2]} \rho(x, y)dA, \ldots, \frac{1}{\Delta x_{n-1} \Delta y_{m-1}} \int_{[x_{n-1}, x_n] \times [y_{m-1}, y_m]} \rho(x, y)dA\right),$$

where $dA = dx dy$, $\Delta x_k = x_{k+1} - x_k$ and $\Delta y_j = y_{j+1} - y_j$.

**Author(s)**

Umar Islambekov

**References**


**Examples**

```r
N <- 100
set.seed(123)
# sample N points uniformly from unit circle and add Gaussian noise
X <- TDA::circleUnif(N, r=1) + rnorm(2*N, mean = 0, sd = 0.2)
# compute a persistence diagram using the Rips filtration built on top of X
```
computePL

A Vector Summary of the Persistence Landscape Function

Description

Vectorizes the persistence landscape (PL) function constructed from a given persistence diagram. The $k$th order landscape function of a persistence diagram $D = \{(b_i, d_i)\}_{i=1}^N$ is defined as

$$\lambda_k(t) = k\max_{1 \leq i \leq N} \Lambda_i(t), \quad k \in \mathbb{N},$$

where $k\max$ returns the $k$th largest value and

$$\Lambda_i(t) = \begin{cases} t - b_i & t \in [b_i, \frac{b_i + d_i}{2}] \\ d_i - t & t \in \left(\frac{b_i + d_i}{2}, d_i\right] \\ 0 & \text{otherwise} \end{cases}$$

Usage

```
computePL(D, homDim, scaleSeq, k=1)
```

Arguments

- **D**: matrix with three columns containing the dimension, birth and death values respectively
- **homDim**: homological dimension ($0$ for $H_0$, $1$ for $H_1$, etc.)
- **scaleSeq**: numeric vector of increasing scale values used for vectorization
- **k**: order of landscape function. By default, $k=1$
Value

A numeric vector whose elements are the values of the \( k \)th order landscape function evaluated at each point of \( \text{scaleSeq} = \{ t_1, t_2, \ldots, t_n \} \):

\[
(\lambda_k(t_1), \lambda_k(t_2), \ldots, \lambda_k(t_n))
\]

Author(s)

Umar Islambekov

References


Examples

\[
\begin{align*}
N & \leftarrow 100 \\
\text{set.seed}(123) \\
& \# \text{ sample } N \text{ points uniformly from unit circle and add Gaussian noise} \\
X & \leftarrow \text{TDA::circleUnif}(N, r=1) + \text{rnorm}(2*N, \text{mean} = 0, \text{sd} = 0.2) \\
& \# \text{ compute a persistence diagram using the Rips filtration built on top of } X \\
D & \leftarrow \text{TDA::ripsDiag}(X, \text{maxdimension} = 1, \text{maxscale} = 2) \$\text{diagram} \\
\text{scaleSeq} & = \text{seq}(0, 2, \text{length.out}=11) \# \text{ sequence of scale values} \\
& \# \text{ compute persistence landscape (PL) for homological dimension } H_0 \text{ with order of landscape } k=1 \\
\text{computePL}(D, \text{homDim}=0, \text{scaleSeq}, k=1) \\
& \# \text{ compute persistence landscape (PL) for homological dimension } H_1 \text{ with order of landscape } k=1 \\
\text{computePL}(D, \text{homDim}=1, \text{scaleSeq}, k=1)
\end{align*}
\]

computePS

A Vector Summary of the Persistence Silhouette Function

Description

Vectorizes the persistence silhouette (PS) function constructed from a given persistence diagram. The \( p \)th power silhouette function of a persistence diagram \( D = \{(b_i, d_i)\}_{i=1}^N \) is defined as

\[
\phi_p(t) = \frac{\sum_{i=1}^N |d_i - b_i|^p \Lambda_i(t)}{\sum_{i=1}^N |d_i - b_i|^p},
\]
where
\[ \Lambda_i(t) = \begin{cases} 
  t - b_i & t \in [b_i, b_i + d_i] \\
  d_i - t & t \in (b_i + \frac{d_i}{2}, d_i] \\
  0 & \text{otherwise}
\end{cases} \]

Points of \( D \) with infinite death value are ignored

**Usage**

```r
computePS(D, homDim, scaleSeq, p=1)
```

**Arguments**

- `D` matrix with three columns containing the dimension, birth and death values respectively
- `homDim` homological dimension (0 for \( H_0 \), 1 for \( H_1 \), etc.)
- `scaleSeq` numeric vector of increasing scale values used for vectorization
- `p` power of the weights for the silhouette function. By default, \( p=1 \)

**Value**

A numeric vector whose elements are the average values of the \( p \)th power silhouette function computed between each pair of consecutive scale points of \( \text{scaleSeq} = \{ t_1, t_2, \ldots, t_n \} \):

\[
\left( \frac{1}{\Delta t_1} \int_{t_1}^{t_2} \phi_p(t) dt, \frac{1}{\Delta t_2} \int_{t_2}^{t_3} \phi_p(t) dt, \ldots, \frac{1}{\Delta t_{n-1}} \int_{t_{n-1}}^{t_n} \phi_p(t) dt \right),
\]

where \( \Delta t_k = t_{k+1} - t_k \)

**Author(s)**

Umar Islambekov

**References**


**Examples**

```r
N <- 100
set.seed(123)
# sample N points uniformly from unit circle and add Gaussian noise
X <- TDA::circleUnif(N,r=1) + rnorm(2*N,mean = 0,sd = 0.2)
# compute a persistence diagram using the Rips filtration built on top of X
D <- TDA::ripsDiag(X,maxdimension = 1,maxscale = 2)$diagram
scaleSeq = seq(0,2,length.out=11) # sequence of scale values
```
computeVAB

A Vector Summary of the Betti Curve

Description
For a given persistence diagram $D = \{(b_i, d_i)\}_{i=1}^N$, computeVAB() vectorizes the Betti Curve

$$\beta(t) = \sum_{i=1}^N w(b_i, d_i) 1_{[b_i, d_i)}(t),$$

where the weight function $w(b, d) \equiv 1$

Usage
```
computeVAB(D, homDim, scaleSeq)
```

Arguments
- `D` : matrix with three columns containing the dimension, birth and death values respectively
- `homDim` : homological dimension (0 for $H_0$, 1 for $H_1$, etc.)
- `scaleSeq` : numeric vector of increasing scale values used for vectorization

Value
A numeric vector whose elements are the average values of the Betti curve computed between each pair of consecutive scale points of `scaleSeq`=$\{t_1, t_2, \ldots, t_n\}$:

$$\left( \frac{1}{\Delta t_1} \int_{t_1}^{t_2} \beta(t)dt, \frac{1}{\Delta t_2} \int_{t_2}^{t_3} \beta(t)dt, \ldots, \frac{1}{\Delta t_{n-1}} \int_{t_{n-1}}^{t_n} \beta(t)dt \right),$$

where $\Delta t_k = t_{k+1} - t_k$

Author(s)
Umar Islambekov, Hasani Pathirana

References
Examples

\begin{verbatim}
N <- 100
set.seed(123)
# sample N points uniformly from unit circle and add Gaussian noise
X <- TDA::circleUnif(N,r=1) + rnorm(2*N,mean = 0,sd = 0.2)

# compute a persistence diagram using the Rips filtration built on top of X
D <- TDA::ripsDiag(X,maxdimension = 1,maxscale = 2)$diagram

scaleSeq = seq(0,2,length.out=11) # sequence of scale values

# compute vector of averaged Bettis (VAB) for homological dimension H_0
computeVAB(D,homDim=0,scaleSeq)

# compute vector of averaged Bettis (VAB) for homological dimension H_1
computeVAB(D,homDim=1,scaleSeq)
\end{verbatim}

computeVPB

A Vector Summary of the Persistence Block

Description

For a given persistence diagram \( D = \{(b_i, p_i)\}_{i=1}^N \), computeVPB() vectorizes the persistence block

\[
f(x, y) = \sum_{i=1}^{N} 1_{E(b_i, p_i)}(x, y),
\]

where \( E(b_i, p_i) = [b_i - \frac{\lambda_i}{2}, b_i + \frac{\lambda_i}{2}] \times [p_i - \frac{\lambda_i}{2}, p_i + \frac{\lambda_i}{2}] \) and \( \lambda_i = 2\tau p_i \) with \( \tau \in (0, 1] \). Points of \( D \) with infinite persistence value are ignored.

Usage

computeVPB(D, homDim, xSeq, ySeq, tau)

Arguments

- \( D \): matrix with three columns containing the dimension, birth and persistence values respectively.
- \( \text{homDim} \): homological dimension (0 for \( H_0 \), 1 for \( H_1 \), etc.).
- \( \text{xSeq} \): numeric vector of increasing x (birth) values used for vectorization.
- \( \text{ySeq} \): numeric vector of increasing y (persistence) values used for vectorization.
- \( \text{tau} \): parameter (between 0 and 1) controlling block size. By default, \( \text{tau}=0.3 \).
Value

A numeric vector whose elements are the weighted averages of the persistence block computed over each cell of the two-dimensional grid constructed from \( x_{\text{Seq}} = \{x_1, x_2, \ldots, x_n\} \) and \( y_{\text{Seq}} = \{y_1, y_2, \ldots, y_m\} \):

\[
\frac{1}{\Delta x_1 \Delta y_1} \int_{[x_1, x_2] \times [y_1, y_2]} f(x, y) wdA, \ldots, \frac{1}{\Delta x_{n-1} \Delta y_{m-1}} \int_{[x_{n-1}, x_n] \times [y_{m-1}, y_m]} f(x, y) wdA,
\]

where \( wdA = (x + y) dx dy \), \( \Delta x_k = x_{k+1} - x_k \) and \( \Delta y_j = y_{j+1} - y_j \).

Author(s)

Umar Islambekov, Aleksei Luchinsky

References


Examples

```r
N <- 100
set.seed(123)
# sample N points uniformly from unit circle and add Gaussian noise
X <- TDA::circleUnif(N,r=1) + rnorm(2*N,mean = 0,sd = 0.2)

# compute a persistence diagram using the Rips filtration built on top of X
D <- TDA::ripsDiag(X,maxdimension = 1,maxscale = 2)$diagram

# switch from the birth-death to the birth-persistence coordinates
D[,3] <- D[,3] - D[,2]
colnames(D)[3] <- "Persistence"

# construct one-dimensional grid of scale values
ySeqH0 <- unique(quantile(D[1,]==0,3],probs = seq(0,1,by=0.2)))
tau <- 0.3 # parameter in [0,1] which controls the size of blocks around each point of the diagram
# compute VPB for homological dimension H_0
computeVPB(D,homDim = 0,xSeq=NA,ySeqH0,tau)

xSeqH1 <- unique(quantile(D[1,]==1,2],probs = seq(0,1,by=0.2)))
ySeqH1 <- unique(quantile(D[1,]==1,3],probs = seq(0,1,by=0.2)))
# compute VPB for homological dimension H_1
computeVPB(D,homDim = 1,xSeqH1,ySeqH1,tau)
```
Index

computeECC, 2
computeNL, 3
computePES, 4
computePI, 5
computePL, 7
computePS, 8
computeVAB, 10
computeVPB, 11