# Package 'pterrace'

October 12, 2017

**Type** Package

**Title** Persistence Terrace for Topological Data Analysis

**Version** 1.0

**Author** Chul Moon

**Maintainer** Chul Moon <chulmoon@uga.edu>

**Description** Plots the summary graphic called the persistence terrace for topological inference. It also provides the aid tool called the terrace area plot for determining the number of significant topological features.

**Encoding** UTF-8

**License** GPL-2

**RoxygenNote** 6.0.1

**Imports** TDA, doParallel, foreach, plotly, viridis

## R topics documented:

---

computept       *Computes a persistence terrace of point cloud data.*

---

#### Description

Computes a persistence terrace of point cloud data for given smoothing parameters. For the detailed explanation of the parameters data, sp, fun, lim, by, lib, sublevel, see the R package TDA. A parallel option is available to reduce computation time.

## Usage

```
computept(data, sp, fun = kde, lim = NULL, by = NULL,
  maxdimension = max(NCOL(data)), lib = "Dionysus", sublevel = FALSE,
  par = FALSE, ncore = NULL)
```

## Arguments

| | |
|---|---|
| data | a matrix of n by d, a point cloud of n points in d dimensions. |
| sp | a vector of smoothing parameter values to be used in the persistence terrace. |
| fun | a smoothing function. The default is the Gaussian kernel density estimator. |
| lim | a 2 by d matrix specifying the range of space to compute. The default value is NULL. |
| by | a numeric value or vector that specifies the grid. The default value is NULL. |
| maxdimension | a integer of specifying the maximum dimension of persistence terrace to compute. |
| lib | a library to compute persistent homology. The default is "Dionysus". |
| sublevel | a logical value selecting whether persistent homology is computed to super-level sets (FALSE) or sub-level sets (TRUE). The default value is FALSE. |
| par | if TRUE, the user can compute the persistence terrace in parallel using multiple cores. The default value is FALSE. |
| ncore | an integer selecting the number of cores to use when parallel computing option is selected. The default value is NULL. |

## Value

The function `computept` returns a list of the computed persistence terraces up to given dimension. For each dimension, the computed persistence terrace includes the following components:

| | |
|---|---|
| x | The vector of the smoothing parameters used in computation. The x-axis of the persistence terrace. |
| y | The vector of filtration where the Betti number changes. The y-axis of the persistence terrace. |
| z | The matrix of Betti numbers. |

## Examples

```
# load three circle data
data(three_circle_dat)

# input variables
Xlim <- c(-4,12)
Ylim <- c(-4,9)
lim <- cbind(Xlim, Ylim)
by <- 0.1
spseq <- seq(0.01,1.5,length.out = 25)

# compute persistence terrace
threecirclept <- computept(three_circle_dat,sp=spseq,lim=lim,by=by)
# compute persistence terrace with parallel option
threecirclept <- computept(three_circle_dat,sp=spseq,lim=lim,by=by,par=TRUE)
```

---

| four_feature_dat | *Point cloud of four features; square, circle, equilateral and isosceles triangles* |

---

**Description**

A dataset containing 2,400 points of four features on the two-dimensional space: 400 points from a a square, 800 points from a circle and 1,200 points from two triangles.

**Usage**

```
data(four_feature_dat)
```

**Format**

A matrix of 2,400 rows and 2 columns

**Examples**

```
# load four feature dat
data(four_feature_dat)

# input variables
Xlim <- c(-7,5)
Ylim <- c(-5,6)
lim <- cbind(Xlim, Ylim)
by <- 0.1

# compute barcodes
four_feature_pt <- computept(four_feature_dat,sp=seq(0.01,0.6,length.out = 50),lim=lim,by=by,par=TRUE)

## draw area plot
terracearea(four_feature_pt,dimension=1,maxheight=20)
## draw persistence terrace
plotpt(four_feature_pt,cmax=6,dimension=1)
```

---

| muscle_fiber_dat | *Point cloud sampled from the muscle tissue cross-sectional image* |

---

**Description**

A dataset containing 6,500 points: 5,000 points sampled from the muscle fiber image and 1,500 points sampled from the boundary lines of the image.

**Usage**

```
data(muscle_fiber_dat)
```

**Format**

A matrix of 6,500 rows and 2 columns

### Examples

```
# load muscle fiber data
data(muscle_fiber_dat)

# input variables
Xlim <- c(-50,350)
Ylim <- c(-50,250)
lim <- cbind(Xlim, Ylim)
by=2

# compute barcodes
muscle_fiber_pt=computebars(muscle_fiber_dat,sp=seq(2,40,length.out = 100),lim=lim,by=by,par=TRUE)

# draw terrace area plot
terracearea(muscle_fiber_pt,dimension=1,maxheight=20)
# draw persistence terrace
plotpt(muscle_fiber_pt,cmax=12,dimension=1)
```

---

plotpt                        *Draws a persistence terrace.*

---

### Description

Draws a persistence terrace.

### Usage

```
plotpt(xyz, dimension, cmax = 10, satellite = T)
```

### Arguments

| | |
|---|---|
| xyz | a list of computed persistence terraces returned by the function computept. |
| dimension | an integer selecting the dimension of the persistence terrace to plot. |
| cmax | an integer specifying the maximum number of colors to be used. The default value is 10. |
| satellite | a logical variable to select a viewpoint. If FALSE, plots the overall view of the persistence terrace. The default value is TRUE. |

### Value

The function plotpt returns the persistence terrace graphic.

### Examples

```
# load three circle data
data(three_circle_dat)

# input variables
Xlim <- c(-4,12)
Ylim <- c(-4,9)
lim <- cbind(Xlim, Ylim)
by <- 0.1
```

```
spseq <- seq(0.01,1.5,length.out = 25)

# compute persistence terrace
threecirclept <- computept(three_circle_dat,sp=spseq,lim=lim,by=by)

# draw persistence terrace, satellite view
plotpt(threecirclept,dimension=1)
# draw persistence terrace, overall view
plotpt(threecirclept,dimension=1,satellite=F)
```

---

terracearea                 *Draws a terrace area plot for a persistence terrace.*

---

### Description

Draws a terrace area plot for a persistence terrace.

### Usage

```
terracearea(xyz, dimension, maxheight = NULL)
```

### Arguments

| | |
|---|---|
| xyz | a list of computed persistence terraces returned by the function computept. |
| dimension | an integer selecting the dimension of the persistence terrace to plot the terrace area plot. |
| maxheight | an integer specifying the maximum heights to show. The default value is NULL. |

### Value

The function `terracearea` returns the terrace area plot.

### Examples

```
# load three circle data
data(three_circle_dat)

# input variables
Xlim <- c(-4,12)
Ylim <- c(-4,9)
lim <- cbind(Xlim, Ylim)
by <- 0.1
spseq <- seq(0.01,1.5,length.out = 25)

# compute persistence terrace
threecirclept <- computept(three_circle_dat,sp=spseq,lim=lim,by=by)

# draw terrace area plot
terracearea(threecirclept,dimension=1)
```

---

three_circle_dat                    *Point cloud of three circles of different size and point density*

---

### Description

A dataset containing 600 randomly generated points: 200 points from each circle.

### Usage

```
data(three_circle_dat)
```

### Format

A matrix of 600 rows and 2 columns

### Examples

```
# load three circle data
data(three_circle_dat)

# input variables
Xlim <- c(-4,12)
Ylim <- c(-4,9)
lim <- cbind(Xlim, Ylim)
by <- 0.1
spseq <- seq(0.01,1.5,length.out = 50)

# compute persistence terrace with parallel option
three_circle_pt <- computept(three_circle_dat,sp=spseq,lim=lim,by=by,par=TRUE)

# draw terrace area plot
terracearea(three_circle_pt,dimension=1)
# draw persistence terrace, satellite view
plotpt(three_circle_pt,dimension=1)
# draw persistence terrace, overall view
plotpt(three_circle_pt,dimension=1,satellite=F)
```

---

two_circle_density_dat

*Point cloud of two circles of same size but different point density*

---

### Description

A dataset containing 500 randomly generated points: 100 points from low point density circle and 400 from high point density circle.

### Usage

```
data(two_circle_density_dat)
```

**Format**

A matrix of 500 rows and 2 columns

**Examples**

```
# load data
data(two_circle_density_dat)

# input variables
Xlim <- c(-3,6)
Ylim <- c(-3,3)
lim <- cbind(Xlim, Ylim)
by <- 0.1
spseq <- seq(0.01,1,length.out=25)

# compute persistence terrace with parallel option
two_circle_density_pt <- computept(two_circle_density_dat,sp=spseq,lim=lim,by=by,par=TRUE)

# draw persistence terrace, satellite view
plotpt(two_circle_density_pt,dimension=1)
```

---

two_circle_noise_added_dat

*Point cloud of two circles of different size and density with noise*

---

**Description**

A dataset containing 350 randomly generated points.

**Usage**

```
data(two_circle_noise_added_dat)
```

**Format**

A matrix of 350 rows and 2 columns

**Examples**

```
# load data
data(two_circle_noise_added_dat)

# input variables
Xlim <- c(-3,4)
Ylim <- c(-3,3)
lim <- cbind(Xlim, Ylim)
by <- 0.05
spseq <- seq(0.01,0.7,length.out=100)

# compute persistence terrace with parallel option
two_circle_noise_pt <- computept(two_circle_noise_added_dat,sp=spseq,lim=lim,by=by,par=TRUE)

# draw persistence terrace, satellite view
plotpt(two_circle_noise_pt,dimension=1)
```

---

two_circle_size_dat *Point cloud of two circles of same point density but different size*

---

### Description

A dataset containing 1000 randomly generated points: 200 points from a radius one circle and 800 points from a radius four circle.

### Usage

```
data(two_circle_size_dat)
```

### Format

A matrix of 1000 rows and 2 columns

### Examples

```
# load data
data(two_circle_size_dat)

# input variables
Xlim <- c(-4,13)
Ylim <- c(-7,7)
lim <- cbind(Xlim, Ylim)
by <- 0.1
spseq <- seq(0.01,3,length.out = 25)

# compute persistence terrace with parallel option
two_circle_size_pt <- computept(two_circle_size_dat,sp=spseq,lim=lim,by=by,par=TRUE)

# draw persistence terrace, satellite view
plotpt(two_circle_size_pt,dimension=1)
```

# Index