Package: TDAstats (via r-universe)

November 6, 2024

Type Package

Title Pipeline for Topological Data Analysis

Version 0.4.1

Description A comprehensive toolset for any useR conducting topological data analysis, specifically via the calculation of persistent homology in a Vietoris-Rips complex. The tools this package currently provides can be conveniently split into three main sections: (1) calculating persistent homology; (2) conducting statistical inference on persistent homology calculations; (3) visualizing persistent homology and statistical inference. The published form of TDAstats can be found in Wadhwa et al. (2018) <doi:10.21105/joss.00860>. For a general background on computing persistent homology for topological data analysis, see Otter et al. (2017) <doi:10.1140/epjds/s13688-017-0109-5>. To learn more about how the permutation test is used for nonparametric statistical inference in topological data analysis, read Robinson & Turner (2017) <doi:10.1007/s41468-017-0008-7>. To learn more about how TDAstats calculates persistent homology, you can visit the GitHub repository for Ripser, the software that works behind the scenes at <https://github.com/Ripser/ripser>. This package has been published as Wadhwa et al. (2018) <doi:10.21105/joss.00860>.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.3)

Imports ggplot2 (>= 2.2.1), Rcpp (>= 0.12.15)

URL https://rrrlw.github.io/TDAstats

BugReports https://github.com/rrrlw/TDAstats/issues

LinkingTo Rcpp

RoxygenNote 6.1.0

SystemRequirements C++11

Suggests testthat (>= 2.0.0), knitr, rmarkdown, covr

VignetteBuilder knitr

Repository https://rrrlw.r-universe.dev

RemoteUrl https://github.com/rrrlw/tdastats

RemoteRef HEAD

RemoteSha 777d70f6fccd61e55c77702f058efc9359f53e64

Contents

`calculate_homology` *Calculate Persistent Homology of a Point Cloud*

Description

Calculates the persistent homology of a point cloud, as represented by a Vietoris-Rips complex. This function is an R wrapper for Ulrich Bauer's Ripser C++ library for calculating persistent homology. For more information on the C++ library, see <https://github.com/Ripser/ripser>.

Usage

```
calculate_homology(mat, dim = 1, threshold = -1, p = 2L,
  format = "cloud", standardize = FALSE, return_df = FALSE)
```

Arguments

`mat`	numeric matrix containing point cloud or distance matrix
`dim`	maximum dimension of features to calculate
`threshold`	maximum diameter for computation of Vietoris-Rips complexes
`p`	number of the prime field Z/pZ to compute the homology over

format	format of ‘mat‘, either "cloud" for point cloud or "distmat" for distance matrix
standardize	boolean determining whether point cloud size should be standardized
return_df	defaults to ‘FALSE‘, returning a matrix; if ‘TRUE‘, returns a data frame

Details

The ‘mat‘ parameter should be a numeric matrix with each row corresponding to a single point, and each column corresponding to a single dimension. Thus, if ‘mat‘ has 50 rows and 5 columns, it represents a point cloud with 50 points in 5 dimensions. The ‘dim‘ parameter should be a positive integer. Alternatively, the ‘mat‘ parameter could be a distance matrix (upper triangular half is ignored); note: ‘format‘ should be specified as "ldm".

Value

3-column matrix or data frame, with each row representing a TDA feature

Examples

```
# create a 2-d point cloud of a circle (100 points)
num.pts <- 100
rand.angle <- runif(num.pts, 0, 2*pi)
pt.cloud <- cbind(cos(rand.angle), sin(rand.angle))

# calculate persistent homology (num.pts by 3 numeric matrix)
pers.hom <- calculate_homology(pt.cloud)
```

---

circle2d *2-dimensional point cloud of a unit circle*

Description

A dataset containing the Cartesian coordinates of 100 points uniformly distributed on the circumference of a unit circle.

Usage

```
circle2d
```

Format

A matrix with 100 rows and 2 columns: the x- and y-coordinates

Source

https://github.com/rrrlw/TDAstats/blob/master/data-raw/circle2d.R

`id_significant` *Identify Significant Features in Persistent Homology*

Description

An empirical method (bootstrap) to differentiate between features that constitute signal versus noise based on the magnitude of their persistence relative to one another. Note: you must have at least 5 features of a given dimension to use this function.

Usage

```
id_significant(features, dim = 1, reps = 100, cutoff = 0.975)
```

Arguments

`features`	3xn data frame of features; the first column must be dimension, the second birth, and the third death
`dim`	dimension of features of interest
`reps`	number of replicates
`cutoff`	percentile cutoff past which features are considered significant

Examples

```
# get dataset (noisy circle) and calculate persistent homology
angles <- runif(100, 0, 2 * pi)
x <- cos(angles) + rnorm(100, mean = 0, sd = 0.1)
y <- sin(angles) + rnorm(100, mean = 0, sd = 0.1)
annulus <- cbind(x, y)
phom <- calculate_homology(annulus)

# find threshold of significance
# expecting 1 significant feature of dimension 1 (Betti-1 = 1 for annulus)
thresh <- id_significant(features = as.data.frame(phom),
                         dim = 1,
                         reps = 500,
                         cutoff = 0.975)

# generate flat persistence diagram
# every feature higher than `thresh` is significant
plot_persist(phom, flat = TRUE)
```

`permutation_test` *Statistical Inference for Topological Data Analysis*

Description

Conducts a permutation test for nonparametric statistical inference of persistent homology in topological data analysis.

Usage

```
permutation_test(data1, data2, iterations, exponent = 1, update = 0,
  ...)
```

Arguments

`data1`	first dataset
`data2`	second dataset
`iterations`	number of iterations for distribution in permutation test
`exponent`	parameter 'p' that returns Wasserstein-p metric
`update`	if greater than zero, will print a message every 'update' iterations
`...`	arguments for 'calculate_homology' used for each permutation; this includes the 'format', 'dim', and 'threshold' parameters

Details

The persistent homology of two point clouds are compared with the Wasserstein metric (where Wasserstein-1 is also known as the Earth Mover's Distance). However, the magnitude of the metric for a single pair of point clouds is meaningless without a reference distribution. This function uses a permutation test (permuting the points between the two clouds) as a nonparametric hypothesis test for statistical inference.

For more details on permutation tests for statistical inference in topological data analysis, see Robinson A, Turner K. Hypothesis testing for topological data analysis. J Appl Comput Topology. 2017; 1(2): 241-261.<doi:10.1007/s41468-017-0008-7>

Value

list containing results of permutation test

phom.dist *Calculate Distance between Homology Matrices*

Description

Calculates the distance between two matrices containing persistent homology features, usually as returned by the ‘calculate_homology‘ function.

Usage

```
phom.dist(phom1, phom2, limit.num = 0)
```

Arguments

phom1	3-by-n numeric matrix containing persistent homology for first dataset
phom2	3-by-n numeric matrix containing persistent homology for second dataset
limit.num	limit comparison to only top ‘limit.num‘ features in each dimension

Details

Note that the absolute value of this measure of distance is not meaningful without a null distribution or at least another value for relative comparison (e.g. finding most similar pair within a triplet).

Value

distance vector (1 element per dimension) between ‘phom1‘ and ‘phom2‘

plot_barcode *Plot Persistent Homology as Topological Barcode*

Description

Plots a feature matrix as a topological barcode. See ‘plot_persist‘ for an alternate visualization method of persistent homology.

Usage

```
plot_barcode(feature.matrix)
```

Arguments

feature.matrix	nx3 matrix representing persistent homology features

Details

The ‘feature.matrix‘ parameter should be a numeric matrix with each row corresponding to a single feature. It should have 3 columns corresponding to feature dimension (col 1), feature birth (col 2), and feature death (col 3). The first column should be filled with integers, and the next two columns should be filled with numeric values. The output from the ‘calculate_homology‘ function in this package will be a valid value for the ‘feature.matrix‘ parameter.

This function uses the ggplot2 framework to generate persistence diagrams. For details, see: Wickham H (2009). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag: New York, NY.

Value

ggplot instance representing topological barcode

Examples

```
# create a 2-d point cloud of a circle (100 points)
num.pts <- 100
rand.angle <- runif(num.pts, 0, 2*pi)
pt.cloud <- cbind(cos(rand.angle), sin(rand.angle))

# calculate persistent homology (num.pts by 3 numeric matrix)
pers.hom <- calculate_homology(pt.cloud)

# plot calculated homology features as persistence diagram
plot_barcode(pers.hom)
```

---

plot_persist *Plot Persistent Homology as Persistence Diagram*

Description

Plots a feature matrix as a persistence diagram. See ‘plot_barcode‘ for an alternate visualization method of persistent homology.

Usage

```
plot_persist(feature.matrix, flat = FALSE, cutoff = 0)
```

Arguments

feature.matrix	nx3 matrix representing persistent homology features
flat	default FALSE; if TRUE, plots flat persistent homology instead
cutoff	threshold for significant features; line added as marker on plot

Details

The 'feature.matrix' parameter should be a numeric matrix with each row corresponding to a single feature. It should have 3 columns corresponding to feature dimension (col 1), feature birth (col 2), and feature death (col 3). The first column should be filled with integers, and the next two columns should be filled with numeric values. The output from the 'calculate_homology' function in this package will be a valid value for the 'feature.matrix' parameter.

This function uses the ggplot2 framework to generate persistence diagrams. For details, see: Wickham H (2009, ISBN:9780387981413). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag: New York, NY.

Value

ggplot instance representing persistence diagram

Examples

```
# create a 2-d point cloud of a circle (100 points)
num.pts <- 100
rand.angle <- runif(num.pts, 0, 2*pi)
pt.cloud <- cbind(cos(rand.angle), sin(rand.angle))

# calculate persistent homology (num.pts by 3 numeric matrix)
pers.hom <- calculate_homology(pt.cloud)

# plot calculated homology features as persistence diagram
plot_persist(pers.hom)
```

---

sphere3d *3-dimensional point cloud of a unit sphere*

Description

A dataset containing the Cartesian coordinates of 100 points uniformly distributed on the surface of a unit sphere.

Usage

```
sphere3d
```

Format

A matrix with 100 rows and 3 columns: the x-, y-, and z-coordinates

Source

https://github.com/rrrlw/TDAstats/blob/master/data-raw/sphere3d.R

TDAstats — *Statistical Inference for Persistent Homology in Topological Data Analysis*

Description

This package aims to be a comprehensive toolset for any useR conducting topological data analysis, specifically via the calculation of persistent homology in a Vietoris-Rips complex. The tools this package currently provides can be conveniently split into three main sections: (1) calculating persistent homology; (2) conducting statistical inference on persistent homology calculations; (3) visualizing persistent homology and statistical inference.

unif2d — *2-dimensional point cloud of a unit square*

Description

A dataset containing the Cartesian coordinates of 100 points uniformly distributed within a unit square.

Usage

```
unif2d
```

Format

A matrix with 100 rows and 2 columns: the x- and y-coordinates

Source

https://github.com/rrrlw/TDAstats/blob/master/data-raw/unif2d.R

unif3d — *3-dimensional point cloud of a unit cube*

Description

A dataset containing the Cartesian coordinates of 100 points uniformly distributed within a unit cube.

Usage

```
unif3d
```

Format

A matrix with 100 rows and 3 columns: the x-, y-, and z-coordinates

Source

https://github.com/rrrlw/TDAstats/blob/master/data-raw/unif3d.R

Index