Package 'pflamelet'

December 17, 2020

Type Package

Title Persistence Flamelets: Computation and Inferential Tools

Version 0.1.1

Author Tullia Padellini

Maintainer Tullia Padellini <t.padellini@imperial.ac.uk>

Description Computes the Persistence Flamelets, a statistical tool for exploring the Topological Invariants of Scale-Space families introduced in Padellini and Brutti (2017) ``Persistence Flamelets: multiscale Persistent Homology for kernel density exploration" <arXiv:1709.07097>.
Flamelets can be built from either sub/super level sets of arbitrary functions or from a precomputed list of Persistence Diagrams.
In addition, this package provides functions to compute confidence bands for a Flamelet via bootstrap, assess significance of each feature, clean a Flamelet from topological noise, perform a two sample permutation test for groups of Flamelets and it also implements a topological heuristic for bandwidth selection for kernel density estimators.

License GPL-3

Imports TDA, plotly, viridis, pbapply, graphics, stats, abind

Suggests eegkit, dplyr

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-12-17 09:30:02 UTC

R topics documented:

`build.flamelet` *Persistence Flamelet*

Description

Computes the Persistence Flamelet from a list of Persistence Diagrams. If the input is a list of data points observed at different scales, at each resolution, Persistence Diagrams are built for the sub/superlevel set of an arbitrary function computed on X, and then used to compute the Flamelet.

Usage

```
build.flamelet(
  X,
  base.type = "landscape",
  base.param = 1,
  dimension = 1,
  tseq,
  diag.fun = distFct,
  sublevel = TRUE,
  h.grid = NULL,
  lim = NULL,
  by = NULL,
  scale = TRUE,
  precomputed.diagram = FALSE,
  info.message = FALSE,
  band = FALSE,
  B = 10,
  alpha = 0.95
)
```

Arguments

`X`	a list of m persistence diagrams representing the scale-space family at different resolutions, or a list of m matrix/data.frame containing the pointcloud at different resolutions. If `X` is a n-by-d matrix or a data.frame containing a d-dimensional pointcloud, this function computes the Flamelet on the `diag.fun`.
`base.type`	a string specifying whether the Flamelet is built from Persistence Landscapes ("landscape") or Persistence Silhouettes ("silhouettes").
`base.param`	the order k of the Flamelet (if `base.type=="landscape"`) or the power p of the Flamelet (if `base.type=="silhouette"`).

dimension	the topological dimension of the flamelet (0 for connected components, 1 for loops, ...).
tseq	a vector of dimension k containing the values at which the Flamelet function is evaluated for a fixed scale level.
diag.fun	the function whose sub/super-level set define the persistent homology groups. Corresponds to the argument FUN of the gridDiag function in the TDA package.
sublevel	a logical indicating whether the Persistent Homology should be computed on sub or superlevel set of the function given as diag.fun.
h.grid	vector of dimension m containing bandwidths for the KDE, representing the scale parameter of the Flamelet.
lim	2-by-d matrix, where the i-th column contains the range of the grid over which the function specified in diag.fun is computed for the i-th variable.
by	a scalar (or a vector if different values are selected for each dimension). specifying spaces between elements on the grid whose outernmost element are defined by lim.
scale	a logical indicating whether or not the Persistence Diagrams have to be scaled to be in the same range (needed only for visualization purposes).
precomputed.diagram	a logical indicating whether the Persistence Diagrams have to be computed or are given as input in the form of a list of diagrams.
info.message	a logical denoting if progress messages should be printed.
band	a logical indicating whether or not each Persistence Diagram should be cleaned by means of Bootstrap Bands (only possible when the Persistence Diagrams are computed within the function and are not given as an input).
B	number of bootstrap repetitions needed to compute the confidence band over Persistence Diagrams.
alpha	the confidence level of the bootstrap confidence bands.

Value

a k-by-m matrix containing the Persistence Flamelet.

References

T. Padellini and P. Brutti (2017) Persistence Flamelets: multiscale Persistent Homology for kernel density exploration https://arxiv.org/abs/1709.07097

Examples

```
library(TDA)
xx = rbind(circleUnif(50, 1), circleUnif(50, 1.5) + 3)
Xlim = c(-1, 5);  Ylim = c(-1, 5);  by = 0.05
lim = cbind(Xlim, Ylim)
foo.flamelet = build.flamelet(X = xx, h.grid = seq(0.01, 1, length.out = 40),
base.type = "landscape", dimension = 1,base.param = 1, lim = lim, by = by,
                            tseq = seq(0, .75, length.out = 500))
```

`flamelet.band` *Bootstrap Band for Persistence Flamelet*

Description

Computes a bootstrap band around the 0 of the Persistence Flamelet. If the input is a list of data points observed at different scales, at each resolution, Persistence Diagrams are built for the sub/superlevel set of an arbitrary function computed on X, and then used to compute the Flamelet.

Usage

```
flamelet.band(
  X,
  B,
  alpha,
  base.type = "landscape",
  base.param = 1,
  dimension = 1,
  tseq,
  diag.fun = distFct,
  sublevel = TRUE,
  h.grid = NULL,
  lim = NULL,
  by = NULL
)
```

Arguments

`X`	a list of m matrix/data.frame containing the pointcloud at different resolutions. If `X` is a n-by-d matrix or a data.frame containing a d-dimensional pointcloud, this function computes the Flamelet on the corresponding `diag.fun`.
`B`	number of bootstrap repetitions needed to compute the confidence band over Persistence Diagrams.
`alpha`	the confidence level of the bootstrap confidence bands.
`base.type`	a string specifying whether the Flamelet is built from Persistence Landscapes ("landscape") or Persistence Silhouettes ("silhouettes").
`base.param`	the order k of the Flamelet (if base.type=="landscape") or the power p of the Flamelet (if base.type=="silhouette").
`dimension`	the topological dimension of the flamelet (0 for connected components, 1 for loops, ...)
`tseq`	a vector of values at which the Flamelet function is evaluated for a fixed scale level.
`diag.fun`	the function whose sub/super-level set define the persistent homology groups. Corresponds to the argument `FUN` of the `gridDiag` function in the TDA package.

sublevel	a logical indicating whether the Persistent Homology should be computed on sub or superlevel set of the function given as diag.fun.
h.grid	vector of bandwidths for the KDE, representing the scale parameter of the Flamelet.
lim	2-by-d matrix, where the i-th column contains the range of the grid over which the function specified in diag.fun is computed for the i-th variable.
by	a scalar (or a vector if different values are selected for each dimension). specifying spaces between elements on the grid whose outernmost element are defined by lim.

Value

The quantile of level alpha necessary to build the confidence band. More details can be found in Padellini (2017).

References

T. Padellini and P. Brutti (2017) Persistence Flamelets: multiscale Persistent Homology for kernel density exploration https://arxiv.org/abs/1709.07097

Examples

```
library(TDA)
xx = rbind(circleUnif(50, 1), circleUnif(50, 1.5) + 3)
Xlim = c(-1, 5);  Ylim = c(-1, 5);  by = 0.05
lim = cbind(Xlim, Ylim)
foo.band = flamelet.band(X = xx, B = 10, alpha = 0.95,
                  tseq = seq(0, .75, length.out = 500), diag.fun = kde,
                  h.grid = seq(0.01, 1, length.out = 40), lim = lim, by = by)
```

---

flamelet.clean *Clean Persistence Flamelet*

Description

Remove all the values of the Persistence Flamelet which are not significantly different than 0

Usage

```
flamelet.clean(flamelet, band)
```

Arguments

flamelet	a kxm matrix corresponding to the Persistence Flamelet.
band	a scalar representing the confidence band.

Value

the value of the bandwidth corresponding to the most persistent feature of the Flamelet.

References

T. Padellini and P. Brutti (2017) Persistence Flamelets: multiscale Persistent Homology for kernel density exploration https://arxiv.org/abs/1709.07097

Examples

```
library(TDA)
xx = rbind(circleUnif(50, 1), circleUnif(50, 1.5) + 3)
Xlim = c(-1, 5);  Ylim = c(-1, 5);  by = 0.05
lim = cbind(Xlim, Ylim)
foo.flamelet = build.flamelet(X = xx, h.grid = seq(0.01, 1, length.out = 40),
base.type = "landscape", dimension = 1,base.param = 1, lim = lim, by = by,
                            tseq = seq(0, .75, length.out = 500))
foo.band = flamelet.band(X = xx, B = 10, alpha = 0.95,
                   tseq = seq(0, .75, length.out = 500), diag.fun = kde,
                   h.grid = seq(0.01, 1, length.out = 40), lim = lim, by = by)
new.flamelet = flamelet.clean(foo.flamelet, foo.band)
```

---

flamelet.plot *Plot Persistence Flamelet*

Description

Plot the Persistence Flamelet in its original 3-dimensional form, or in a 2-dimensional projection.

Usage

```
flamelet.plot(
  flamelet,
  scale.param,
  tseq,
  flat = FALSE,
  scale.name = "Bandwidth",
  band = NULL
)
```

Arguments

flamelet	a kxm matrix corresponding to the Persistence Flamelet.
scale.param	a vector of lenght m corresponding to the values of the scale parameter at which the Flamelet has been evaluated.

tseq	a vector of length k containing of values at which the Flamelet function is evaluated for a fixed scale level.
flat	a logical denoting whether the plot should be a 2-d projection of the Flamelet (TRUE) or a 3-d object (FALSE).
scale.name	name of the scale parameter.
band	scalar representing the confidence band for Persistence Flamelet. Only available when flat = FALSE.

References

T. Padellini and P. Brutti (2017) Persistence Flamelets: multiscale Persistent Homology for kernel density exploration https://arxiv.org/abs/1709.07097

Examples

```
library(TDA)
xx = rbind(circleUnif(50, 1), circleUnif(50, 1.5) + 3)
Xlim = c(-1, 5);  Ylim = c(-1, 5);  by = 0.05
lim = cbind(Xlim, Ylim)
foo.flamelet = build.flamelet(X = xx, h.grid = seq(0.01, 1, length.out = 40),
base.type = "landscape", dimension = 1,base.param = 1, lim = lim, by = by,
                            tseq = seq(0, .75, length.out = 500))
flamelet.plot(foo.flamelet, scale.param = seq(0.01, 1, length.out = 40),
             tseq = seq(0, .75, length.out = 500) )


foo.band = flamelet.band(X = xx, B = 10, alpha = 0.05,
                   tseq = seq(0, .75, length.out = 500), diag.fun = kde,
                   h.grid = seq(0.01, 1, length.out = 40), lim = lim, by = by)

flamelet.plot(foo.flamelet, scale.param = seq(0.01, 1, length.out = 40),
             tseq = seq(0, .75, length.out = 500), band = foo.band)
```

---

mpbandwidth *Maximum Persistence Bandwidth*

Description

Select the bandwidth that maximises the persistence of the scale-family, highlighting the most prominent topological feature of the KDE.

Usage

```
mpbandwidth(flamelet, scale.param)
```

Arguments

flamelet	a kxm matrix corresponding to the Persistence Flamelet.
scale.param	a vector of lenght m corresponding to the values of the scale parameter at which the Flamelet X has been evaluated.

Value

the value of the bandwidth corresponding to the most persistent feature of the Flamelet.

References

T. Padellini and P. Brutti (2017) Persistence Flamelets: multiscale Persistent Homology for kernel density exploration https://arxiv.org/abs/1709.07097

Examples

```
library(TDA)
xx = rbind(circleUnif(50, 1), circleUnif(50, 1.5) + 3)
Xlim = c(-1, 5);  Ylim = c(-1, 5);  by = 0.05
lim = cbind(Xlim, Ylim)
foo.flamelet = build.flamelet(X = xx, h.grid = seq(0.01, 1, length.out = 40),
base.type = "landscape", dimension = 1,base.param = 1, lim = lim, by = by,
                            tseq = seq(0, .75, length.out = 500))
mpband(flamelet = foo.flamelet, scale.param = seq(0.01, 1, length.out = 40))
```

---

permutation.test *Two-sample permutation test*

Description

Performs a permutation test to assess whether two samples of Persistence Flamelets are likely be random draw from the same distribution or if they come from different generating mechanisms, with p-value computed by means of bootstrap.

Usage

```
permutation.test(sample1, sample2, n.rep = NULL, seed = NULL)
```

Arguments

sample1	a k x m x n1 array corresponding to the Persistence Flamelet for the n1 subjects in sample 1
sample2	a k x m x n2 array corresponding to the Persistence Flamelet for the n2 subjects in sample 2
n.rep	an integer representing the number of bootstrap iterations. If NULL only the test statistics on the observed samples is returned
seed	an integer specifying a seed for the random shuffling

Value

a bootstrapped p-value

References

T. Padellini and P. Brutti (2017) Persistence Flamelets: multiscale Persistent Homology for kernel density exploration https://arxiv.org/abs/1709.07097

Examples

```
library(eegkit)
library(dplyr)

# import data from the eegkit package
data("eegdata")   # electroencephalography data
data("eegcoord")  # location of the electrodes
# add eeg channel name as variable and select only the 2d projection of the electrode location
eegcoord <- mutate(eegcoord, channel = rownames(eegcoord)) %>% select(channel, xproj, yproj)

# as EEG recordings are extremely unstable, they are typically averaged across repetitions
# here we average them across the 5 trials from each subject
eegmean <- eegdata %>% group_by(channel, time, subject) %>% summarise(mean = mean(voltage))
dim(eegmean) # 64 channels x 256 time points x 20 subjects

subjects <- unique(eegdata$subject)
# subjects 1:10 are alcoholic, 11:20 are control

# eegmean2 <- tapply(eegdata$voltage, list(eegdata$channel, eegdata$time, eegdata$subject), mean)
# Start by computing the list of Persistence Diagrams needed to build the flamelet for each subject
diag.list <- list()
t0 <- Sys.time()
for (sbj in subjects){

 # select signal for one subject and then remove channels for which there are no coordinates
  sbj.data = dplyr::filter(eegmean, subject == sbj, !(channel %in% c("X", "Y", "nd")  ))

  # add 2d projection of electrodes location
  sbj.data = left_join(sbj.data, eegcoord, by = "channel")

  # scale data
  sbj.data[, c(4:6)] = scale(sbj.data[,c(4:6)])


  # dsucc.List = list()

  diag.list.sbj = lapply(unique(sbj.data$time), function(time.idx){
    time.idx.data = filter(sbj.data, time == time.idx) %>% ungroup %>%
                    select(mean, xproj, yproj)
    time.idx.diag = ripsDiag(time.idx.data, maxdimension = 1, maxscale = 5,
                             library = "GUDHI", printProgress = F)
    return(time.idx.diag$diagram)
```

```
  })

  diag.list[[which(sbj== subjects)]] = diag.list.sbj

  print(paste("subject ", which(sbj == subjects), " of 20"))


}
t1 <- Sys.time()-t0
t1 # will take less than 5 minutes

tseq <- seq(0, 5, length = 500) # consider 5 as it is the
# same value as maxscale (hence largest possible persistence)

p_silh0 <- sapply(diag.list, FUN = build.flamelet,
base.type = "silhouette", dimension = 0,
tseq = tseq, precomputed.diagram = TRUE, simplify = 'array')

prova = permutation_test(p_sih0[,,1:10], p_silh0[,,11:20],  n.rep = 10, seed = 1)
```

---

pflamelet *pflamelet: Persistence Flamelets*

Description

Computes the Persistence Flamelets, a statistical tool for exploring the Topological Invariants of Scale-Space families introduced in Padellini and Brutti (2017) "Persistence Flamelets: multiscale Persistent Homology for kernel density exploration" <arXiv:1709.07097>.Flamelets can be built from either sub/super level sets of arbitrary functions or from a precomputed list of Persistence Diagrams. In addition, this package provides functions to compute confidence bands for a Flamelet via bootstrap, assess significance of each feature, clean a Flamelet from topological noise, perform a two sample permutation test for groups of Flamelets and it also implements a topological heuristic for bandwidth selection for kernel density estimators.

Index