Package 'SpatPCA'

January 9, 2020

Type Package

Title Regularized Principal Component Analysis for Spatial Data

Version 1.2.0.1

Date 2020-01-08

URL https://github.com/egpivo/SpatPCA

BugReports https://github.com/egpivo/SpatPCA/issues

Description Provide regularized principal component analysis incorporating smoothness, sparseness and orthogonality of eigenfunctions
by using the alternating direction method of multipliers algorithm (Wang and Huang, 2017, <DOI:10.1080/10618600.2016.1157483>). The
method can be applied to either regularly or irregularly spaced data, including 1D, 2D, and 3D.

License GPL-3

Imports Rcpp, RcppParallel

LinkingTo Rcpp, RcppArmadillo, RcppParallel

SystemRequirements C++11

Author Wen-Ting Wang, Hsin-Cheng Huang

Maintainer Wen-Ting Wang <egpivo@gmail.com>

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-01-09 14:00:05 UTC

R topics documented:

SpatPCA-package *Regularized Principal Component Analysis for Spatial Data*

Description

A new regularization approach to estimate the leading spatial patterns via smoothness and sparseness penalties, and spatial predctions for spatial data that may be irregularly located in space (including 1D, 2D and 3D), and obtain the spatial prediction at the designated locations.

Details

Package:	SpatPCA
Type:	Package
Version:	1.2.0.1
Date:	2020-01-08
License:	GPL version 3

Author(s)

Wen-Ting Wang <egpivo@gmail.com> and Hsin-Cheng Huang <hchuang@stat.sinica.edu.tw>

spatpca *Regularized PCA for spatial data*

Description

Produce spatial dominant patterns and spatial predictions at the designated locations according to the specified tuning parameters or the selected tuning parameters by the M-fold cross-validation.

Usage

```
spatpca(x, Y, M = 5, K = NULL, K.select = ifelse(is.null(K), TRUE, FALSE),
tau1 = NULL, tau2 = NULL, gamma = NULL, x_new = NULL, center = FALSE,
plot.cv = FALSE, maxit = 100, thr = 1e-04)
```

Arguments

x	Location matrix ($p \times d$). Each row is a location. d is the dimension of locations
Y	Data matrix ($n \times p$) stores the values at p locations with sample size n.
K	Optional user-supplied number of eigenfunctions; default is NULL. If K is NULL or K.select is TRUE, K is selected automatically.

K.select	If TRUE, K is selected automatically; otherwise, K.select is set to be user-supplied K. Default depends on user-supplied K.
tau1	Optional user-supplied numeric vector of a nonnegative smoothness parameter sequence. If NULL, 10 tau1 values in a range are used.
tau2	Optional user-supplied numeric vector of a nonnegative sparseness parameter sequence. If NULL, none of tau2 is used.
gamma	Optional user-supplied numeric vector of a nonnegative tuning parameter sequence. If NULL, 10 values in a range are used.
x_new	New location matrix. If NULL, it is x.
M	Optional number of folds; default is 5.
center	If TRUE, center the columns of Y. Default is FALSE.
plot.cv	If TRUE, plot the cv values. Default is FALSE.
maxit	Maximum number of iterations. Default value is 100.
thr	Threshold for convergence. Default value is 10^{-4}.

Details

An ADMM form of the proposed objective function is written as

$$\min_{\boldsymbol{\Phi}} \|\boldsymbol{Y} - \boldsymbol{Y}\boldsymbol{\Phi}\boldsymbol{\Phi}'\|_F^2 + \tau_1 \text{tr}(\boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{\Phi}) + \tau_2 \sum_{k=1}^{K}\sum_{j=1}^{p} |\phi_{jk}|,$$

subject to $\boldsymbol{\Phi}^T\boldsymbol{\Phi} = \boldsymbol{I}_K$, where $\boldsymbol{Y}$ is a data matrix, $\boldsymbol{\Omega}$ is a smoothness matrix, and $\boldsymbol{\Phi} = \{\phi_{jk}\}$.

Value

eigenfn	Estimated eigenfunctions at the new locations, x_new.
Yhat	Prediction of Y at the new locations, x_new.
stau1	Selected tau1.
stau2	Selected tau2.
sgamma	Selected gamma.
cv1	cv socres for tau1.
cv2	cv socres for tau2.
cv3	cv socres for gamma.
tau1	Sequence of tau1-values used in the process.
tau2	Sequence of tau2-values used in the process.
gamma	Sequence of gamma-values used in the process.
Yc	If center is TRUE, Yc is the centered Y; else, Yc is equal to Y.

Author(s)

Wen-Ting Wang and Hsin-Cheng Huang

References

Wang, W.-T. and Huang, H.-C. (2017). Regularized principal component analysis for spatial data. *Journal of Computational and Graphical Statistics* **26** 14-25.

Examples

```
###### 1D: regular locations
x_1D <- as.matrix(seq(-5, 5, length = 50))
Phi_1D <- exp(-x_1D^2) / norm(exp(-x_1D^2), "F")
set.seed(1234)
Y_1D <- rnorm(n = 100, sd = 3) %*% t(Phi_1D) + matrix(rnorm(n = 100 * 50), 100, 50)
cv_1D <- spatpca(x = x_1D, Y = Y_1D)
plot(x_1D, cv_1D$eigenfn[, 1], type = "l", main = "1st eigenfunction")
lines(x_1D, svd(Y_1D)$v[, 1], col = "red")
legend("topleft", c("SpatPCA", "PCA"), lty = 1:1, col = 1:2)

## Not run:
### 1D: artificial irregular locations
rm_loc <- sample(1:50, 20)
x_1Drm <- x_1D[-rm_loc]
Y_1Drm <- Y_1D[,-rm_loc]
x_1Dnew <- as.matrix(seq(-5, 5, length = 100))
cv_1D <- spatpca(x = x_1Drm, Y = Y_1Drm, tau2 = 1:100, x_new = x_1Dnew)
plot(x_1Dnew, cv_1D$eigenfn, type = "l", main = "eigenfunction")
plot(cv_1D$Yhat[, 50], xlab = "n", ylab = "Yhat", type = "l",
    main = paste("prediction at x = ", x_1Dnew[50]))

###### 2D: Daily 8-hour ozone averages for sites in the Midwest (USA)
library(fields)
library(pracma)
data(ozone2)
x <- ozone2$lon.lat
Y <- ozone2$y
date <- as.Date(ozone2$date, format = "%y%m%d")
rmna <- !colSums(is.na(Y))
YY <- matrix(Y[, rmna], nrow = nrow(Y))
YY <- detrend(YY, "linear")
xx <- x[rmna, ]
cv <- spatpca(x = xx, Y = YY)
quilt.plot(xx, cv$eigenfn[,1])
map("state", xlim = range(xx[, 1]), ylim = range(xx[, 2]), add = T)
map.text("state", xlim = range(xx[, 1]), ylim = range(xx[, 2]), cex = 2, add = T)
plot(date, YY %*% cv$eigenfn[,1], type = "l", ylab = "1st Principal Component")

### new loactions
new_p = 200
x_lon <- seq(min(xx[, 1]), max(xx[, 1]), length = new_p)
x_lat <- seq(min(xx[, 2]), max(xx[, 2]), length = new_p)
xx_new <- as.matrix(expand.grid(x = x_lon, y = x_lat))
eof <- spatpca(x = xx, Y = YY, K = cv$Khat, tau1 = cv$stau1, tau2 = cv$stau2, x_new = xx_new)
quilt.plot(xx_new, eof$eigenfn[,1], nx = new_p, ny = new_p, xlab = "lon.", ylab = "lat.")
map("state", xlim = range(x_lon), ylim = range(x_lat), add = T)
```

```
map.text("state", xlim = range(x_lon), ylim = range(x_lat), cex = 2, add = T)

###### 3D: regular locations
x <- y <- z <- as.matrix(seq(-5, 5, length = 10))
d <- expand.grid(x, y, z)
Phi_3D <- exp(-d[, 1]^2 - d[, 2]^2 - d[, 3]^2) / norm(exp(-d[, 1]^2 - d[, 2]^2 - d[, 3]^2), "F")
Y_3D <- rnorm(n = 1000, sd = 3) %*% t(Phi) + matrix(rnorm(n = 100 * 10^3), 100, 10^3)
cv_3D <- spatpca(x = d, Y = Y_3D, tau2 = seq(0, 1000, length = 10))

library(plot3D)
library(RColorBrewer)
cols <- colorRampPalette(brewer.pal(9, "Blues"))(10)
isosurf3D(x, y, z, colvar = array(cv_3D$eigenfn[, 1], c(p, p, p)),
            level= seq(min(cv_3D$eigenfn[, 1]), max(cv_3D$eigenfn[, 1]), length = 10),
            ticktype = "detailed",
            colkey = list(side = 1),
            col = cols)
## End(Not run)
```

Index