Package ‘BPST’

April 15, 2019

Type Package

Title Bivariate Spline over Triangulation

Version 0.1.0

Author@R c(person(``Lily'', ``Wang'', role = ``aut'', email = ``lilywang@iastate.edu''), person(``Min-Jun'', ``Lai'', role = ``aut'', email = ``mjlai@uga.edu''), person(``Guannan'', ``Wang'', role = c(``trl'',``cre''), email = ``gwang01@wm.edu''), person(``Myungjin'', ``Kim'', role = ``trl''), person(``Xinyi'', ``Li'', role = ``trl''), person(``Jingru'', ``Mu'', role = ``trl''), person(``Jue'', ``Wang'', role = ``trl''), person(``Yueying'', ``Wang'', role = ``trl''), person(``Shan'', ``Yu'', role = ``trl''))

Author Lily Wang [aut],
Ming-Jun Lai [aut],
GuanNan Wang [trl, cre],
Myungjin Kim, [trl],
Xinyi Li, [trl],
Jingru Mu, [trl],
Jue Wang, [trl],
Yueying Wang, [trl],
Shan Yu, [trl]

Maintainer GuanNan Wang <gwang01@wm.edu>

Description Spline smoothing via bivariate spline over triangulation.

License GPL (>= 2)

Imports Rcpp (>= 1.0.0)

Depends Matrix, pracma

LinkingTo Rcpp

RoxygenNote 6.1.1

R topics documented:

basis 2
cv.BPST 3
data.BPST 5
fit.BPST 6
inVT 8
plot.BPST 9

predict.BPST 9
Tr1 10
Tr2 11
V1 11
V2 12

Index **13**

`basis` *Bivariate Spline Basis Function*

Description

This function generates the basis for bivariate spline over triangulation.

Usage

```
basis(V, Tr, d = 5, r = 1, Z, Hmtx = TRUE, Kmtx = TRUE,
  QR = TRUE, TA = TRUE)
```

Arguments

`V`	The `N` by two matrix of vertices of a triangulation, where `N` is the number of vertices. Each row is the coordinates for a vertex.
`Tr`	The triangulation matrix of dimention `nT` by three, where `nT` is the number of triangles in the triangulation. Each row is the indices of vertices in `V`.
`d`	The degree of piecewise polynomials – default is 5, and usually `d` is greater than one. -1 represents piecewise constant.
`r`	The smoothness parameter – default is 1, and $0 \le$ `r` $<$ `d`.
`Z`	The cooridinates of dimension `n` by two. Each row is the coordinates of a point.
`Hmtx`	The indicator of whether the smoothness matrix `H` need to be generated – default is `TRUE`.
`Kmtx`	The indicator of whether the energy matrix `K` need to be generated – default is `TRUE`.
`QR`	The indicator of whether a QR decomposition need to be performed on the smoothness matrix – default is `TRUE`.
`TA`	The indicator of whether the area of the triangles need to be calculated – default is `TRUE`.

Details

This R program is modified based on the Matlab program written by Ming-Jun Lai from the University of Georgia and Li Wang from the Iowa State University.

Value

A list of vectors and matrice, including:

- `B` — The spline basis function of dimension `n` by `nT*{(d+1)(d+2)/2}`, where `n` is the number of observationed points, `nT` is the number of triangles in the given triangulation, and `d` is the degree of the spline. If some points do not fall in the triangulation, the generation of the spline basis will not take those points into consideration.
- `Ind.inside` — A vector contains the indexes of all the points which are inside the triangulation.
- `H` — The smoothness matrix.
- `Q2` — The Q2 matrix after QR decomposition of the smoothness matrix `H`.
- `K` — The thin-plate energy function.
- `tria.all` — The area of each triangle within the given triangulation.

Examples

```
# example 1
xx=c(-0.25,0.75,0.25,1.25)
yy=c(-0.25,0.25,0.75,1.25)
Z=cbind(xx,yy)
d=4; r=1;
V0=rbind(c(0,0),c(1,0),c(1,1),c(0,1))
Tr0=rbind(c(1,2,3),c(1,3,4))
basis(V0,Tr0,d,r,Z)
```

---

cv.BPST *Cross-validation using Bivariate Penalized Spline over Triangulation*

Description

This function implements k-fold cross-validation via bivariate penlized spline over triangulation, and returns the mean squared prediction error.

Usage

```
cv.BPST(Y, Z, V, Tr, d = 5, r = 1, lambda = 10^seq(-6, 6, by = 0.5),
  nfold = 10, Hmtx = TRUE, Kmtx = TRUE, QR = TRUE, TA = TRUE)
```

Arguments

- `Y` — The response variable observed over the domain.
- `Z` — The cooridinates of dimension `n` by two. Each row is the coordinates of a point.

V	The N by two matrix of vertices of a triangulation, where N is the number of vertices. Each row is the coordinates for a vertex.
Tr	The triangulation matrix of dimention nT by three, where nT is the number of triangles in the triangulation. Each row is the indices of vertices in V.
d	The degree of piecewise polynomials – default is 5, and usually d is greater than one. -1 represents piecewise constant.
r	The smoothness parameter – default is 1, and $0 \leq$ r $<$ d.
lambda	The tuning parameter – default is $10^(-6, -5.5, -5, \ldots, 5, 5.5, 6)$.
Hmtx	The indicator of whether the smoothness matrix H need to be generated – default is TRUE.
Kmtx	The indicator of whether the energy matrix K need to be generated – default is TRUE.
QR	The indicator of whether a QR decomposition need to be performed on the smoothness matrix – default is TRUE.
TA	The indicator of whether the area of the triangles need to be calculated – default is TRUE.
nfolds	The number of folds – default is 10. Although nfold can be as large as the sample size (leave-one-out CV), it is not recommended for large datasets. Smallest value allowable for nfolds is 3.

Details

This R program is modified based on the Matlab program written by Ming-Jun Lai from the University of Georgia and Li Wang from the Iowa State University.

Value

lamc	The tuning parameter selected by k-fold cross validation (CV).
mspe	The mean squared prediction error calculated by k-fold cross validation (CV).

Examples

```
# Triangulation
# Option 1;
data(V1); data(Tr1); d=5; r=1; V=V1; Tr=Tr1;
# Option 2
# data(V2); data(Tr2); d=5; r=1; V=V2; Tr=Tr2;
d=-1; r=1;
# Grid Points
n1.grid=101; n2.grid=101; n.grid=n1.grid*n2.grid;
u.grid=seq(0,1,length.out=n1.grid)
v.grid=seq(0,1,length.out=n2.grid)
```

```
uu.grid=rep(u.grid,each=n2.grid)
vv.grid=rep(v.grid,times=n1.grid)
Z.grid=as.matrix(cbind(uu.grid,vv.grid))
func=1; sigma=0.1;
gridpoints=data.BPST(Z.grid,V,Tr,func,sigma,2019)
Y.grid=gridpoints$Y; mu.grid=gridpoints$mu;
ind=gridpoints$ind; ind.grid=(1:n.grid)[ind==1];
# Simulation parameters
n=2000;
ind.sam=sort(sample(ind.grid,n))
Y=as.matrix(gridpoints$Y[ind.sam]); Z=as.matrix(gridpoints$Z[ind.sam,]);
cv.BPST(Y,Z,V,Tr,d,r,lambda=10^seq(-6,6,by=0.5),nfold=10)
```

`data.BPST` *Generate testing dataset for bivariate spline smoothing.*

Description

This function generate the testing dataset for bivariate spline smoothing.

Usage

```
data.BPST(Z, V, Tr, func = 1, sigma = 0.1, iter = 2019)
```

Arguments

`Z`	The cooridinates of dimension `n` by two. Each row is the coordinates of a point.
`V`	The `N` by two matrix of vertices of a triangulation, where `N` is the number of vertices. Each row is the coordinates for a vertex.
`Tr`	The triangulation matrix of dimention `nT` by three, where `nT` is the number of triangles in the triangulation. Each row is the indices of vertices in `V`.
`func`	The choice of test function – default is 1. Possible choices include 1,2,3,4,5,6,7,8.
`sigma`	The standard deviation of the white noise – default is 0.1.

Details

This R program is modified based on the Matlab program written by Ming-Jun Lai from the University of Georgia and Li Wang from the Iowa State University.

Value

A list of vectors and matrice, including:

`Y`	The response variable.
`mu`	The mean function.
`Z`	The coordinates.
`ind`	A vector contains the indicators whether the point is inside the given triangulation.

fit.BPST *Model Fitting using Bivariate Penalized Spline over Triangulation*

Description

This function conducts the model fitting via bivariate penlized spline over triangulation.

Usage

```
fit.BPST(Y, Z, V, Tr, d = 5, r = 1, lambda = 10^seq(-6, 6, by = 0.5),
  Hmtx = TRUE, Kmtx = TRUE, QR = TRUE, TA = TRUE)
```

Arguments

Y	The response variable observed over the domain.
Z	The cooridinates of dimension n by two. Each row is the coordinates of a point.
V	The N by two matrix of vertices of a triangulation, where N is the number of vertices. Each row is the coordinates for a vertex.
Tr	The triangulation matrix of dimention nT by three, where nT is the number of triangles in the triangulation. Each row is the indices of vertices in V.
d	The degree of piecewise polynomials – default is 5, and usually d is greater than one. -1 represents piecewise constant.
r	The smoothness parameter – default is 1, and $0 \leq$ r $<$ d.
lambda	The tuning parameter – default is $10^(- 6, -5.5, -5, \ldots, 5, 5.5, 6)$.
Hmtx	The indicator of whether the smoothness matrix H need to be generated – default is TRUE.
Kmtx	The indicator of whether the energy matrix K need to be generated – default is TRUE.
QR	The indicator of whether a QR decomposition need to be performed on the smoothness matrix – default is TRUE.
TA	The indicator of whether the area of the triangles need to be calculated – default is TRUE.

Details

This R program is modified based on the Matlab program written by Ming-Jun Lai from the University of Georgia and Li Wang from the Iowa State University.

Value

A list of vectors and matrice, including:

- `gamma_hat` The estimated spline coefficients.
- `lamc` The tuning parameter selected by Generalized Cross Validation (GCV).
- `B` The spline basis function of dimension `n` by `nT*{(d+1)(d+2)/2}`, where `n` is the number of observationed points, `nT` is the number of triangles in the given triangulation, and `d` is the degree of the spline. The length of points means the length of ordering indices of observation points. If some points do not fall in the triangulation, the generation of the spline basis will not take those points into consideration.
- `Ind.inside` A vector contains the indexes of all the points which are inside the triangulation.
- `H` The smoothness matrix.
- `Q2` The Q2 matrix after QR decomposition of the smoothness matrix `H`.
- `K` The thin-plate energy function.
- `tria.all` The area of each triangle within the given triangulation.

Examples

```
# Triangulation
# Option 1;
# data(V1); data(Tr1); d=5; r=1; V=V1; Tr=Tr1;
# Option 2
data(V2); data(Tr2); d=5; r=1; V=V2; Tr=Tr2;
d=5; r=1;
# Grid Points
n1.grid=101; n2.grid=101; n.grid=n1.grid*n2.grid;
u.grid=seq(0,1,length.out=n1.grid)
v.grid=seq(0,1,length.out=n2.grid)
uu.grid=rep(u.grid,each=n2.grid)
vv.grid=rep(v.grid,times=n1.grid)
Z.grid=as.matrix(cbind(uu.grid,vv.grid))
func=1; sigma=0.1;
gridpoints=data.BPST(Z.grid,V,Tr,func,sigma,2019)
Y.grid=gridpoints$Y; mu.grid=gridpoints$mu;
ind=gridpoints$ind; ind.grid=(1:n.grid)[ind==1];
# Simulation parameters
n=2000;
ind.sam=sort(sample(ind.grid,n))
Y=as.matrix(gridpoints$Y[ind.sam]); Z=as.matrix(gridpoints$Z[ind.sam,]);
mfit=fit.BPST(Y,Z,V,Tr,d,r,lambda=10^seq(-6,6,by=0.5))
rmse=sqrt(mean((Y-mfit$Yhat)^2,na.rm=TRUE))
mpred=predict(mfit,Z.grid)
rmspe=sqrt(mean((Y.grid-mpred$Ypred)^2,na.rm=TRUE))
cat("rmse =",rmse,"and rmspe =",rmspe,"\n")
plot(mfit,Z.grid)
```

inVT *Decide whether a point is inside of a given triangulation.*

Description

This function is used to decided whether a point is inside of a given triangulation.

Usage

```
inVT(V0, Tr0, xx, yy)
```

Arguments

V0	The N by two matrix of vertices of a triangulation, where N is the number of vertices. Each row is the coordinates for a vertex.
Tr0	The triangulation matrix of dimention nT by three, where nT is the number of triangles in the triangulation. Each row is the indices of vertices in V.
xx	The x-cooridinate of points of dimension n by one.
yy	The y-cooridinate of points of dimension n by one.

Details

This R program is modified based on the Matlab program written by Ming-Jun Lai from the University of Georgia and Li Wang from the Iowa State University.

Value

A list of vectors, including:

ind	A vector of dimension n by one matrix that lists whether the points are inside of a given triangulation. 0 – represents outside the triangulation, while 1 – represents inside the triangulation.
ind.inside	A vector contains the indexes of all the points which are inside the triangulation.

Examples

```
xx=c(-0.25,0.75,0.25,1.25)
yy=c(-0.25,0.25,0.75,1.25)
V0=rbind(c(0,0),c(1,0),c(1,1),c(0,1))
Tr0=rbind(c(1,2,3),c(1,3,4))
inVT(V0,Tr0,xx,yy)
```

`plot.BPST` *Produces the contour plot for the estimated surface of a fitted "BPST" object.*

Description

This function produces the contour map for the estimated surface of a fitted "BPST" object.

Usage

```
## S3 method for class 'BPST'
plot(mfit, Zgrid = NULL)
```

Arguments

`mfit`	Fitted “BPST" object.
`Zgrid`	The grid points used to construct the contour plot.

Details

This R program is modified based on the Matlab program written by Ming-Jun Lai from the University of Georgia and Li Wang from the Iowa State University.

Value

None

`predict.BPST` *Make predictions from a fitted BPST object.*

Description

This function is used to make predictions of a fitted BPST object.

Usage

```
## S3 method for class 'BPST'
predict(mfit, Zpred = NULL)
```

Arguments

`mfit`	Fitted “BPST" object.
`Zpred`	The cooridinates for prediction – default is the observed coordinates, `Z`.

Details

This R program is modified based on the Matlab program written by Ming-Jun Lai from the University of Georgia and Li Wang from the Iowa State University.

Value

A vector of predicted values is returned.

Examples

```
# Triangulation
# Option 1;
# data(V1); data(Tr1); d=5; r=1; V=V1; Tr=Tr1;
# Option 2
data(V2); data(Tr2); d=5; r=1; V=V2; Tr=Tr2;
d=5; r=1;
# Grid Points
n1.grid=101; n2.grid=101; n.grid=n1.grid*n2.grid;
u.grid=seq(0,1,length.out=n1.grid)
v.grid=seq(0,1,length.out=n2.grid)
uu.grid=rep(u.grid,each=n2.grid)
vv.grid=rep(v.grid,times=n1.grid)
Z.grid=as.matrix(cbind(uu.grid,vv.grid))
func=1; sigma=0.1;
gridpoints=data.BPST(Z.grid,V,Tr,func,sigma,2019)
Y.grid=gridpoints$Y; mu.grid=gridpoints$mu;
ind=gridpoints$ind; ind.grid=(1:n.grid)[ind==1];
# Simulation parameters
n=2000;
ind.sam=sort(sample(ind.grid,n))
Y=as.matrix(gridpoints$Y[ind.sam]); Z=as.matrix(gridpoints$Z[ind.sam,]);
mfit=fit.BPST(Y,Z,V,Tr,d,r,lambda=10^seq(-6,6,by=0.5))
rmse=sqrt(mean((Y-mfit$Yhat)^2,na.rm=TRUE))
mpred=predict(mfit,Z.grid)
rmspe=sqrt(mean((Y.grid-mpred$Ypred)^2,na.rm=TRUE))
cat("rmse =",rmse,"and rmspe =",rmspe,"\n")
```

Tr1 *Example of triangulation of a rectangular domain.*

Description

A matrix of a triangulation over a rectangular domain.

Usage

```
data('Tr1')
```

Format

`Tr1` is a matrix of a triangulation over a rectangular domain.

References

This example is generated from R package “Triangulation".

`Tr2` *Example of triangulation of an irregular domain.*

Description

A matrix of a triangulation over an irregular domain.

Usage

```
data('Tr2')
```

Format

`Tr2` is a matrix of a triangulation over an irregular domain.

References

This example is generated from R package “Triangulation".

`V1` *Example of vertices of a rectangular domain.*

Description

A matrix of coordinates of the vertices over a rectangular domain.

Usage

```
data('V1')
```

Format

`V1` is a matrix of coordinates of the vertices over a rectangular domain.

References

This example is generated from R package “Triangulation".

`V2` *Example of vertices of an irregular domain.*

Description

A matrix of coordinates of the vertices over an irregular domain.

Usage

```
data('V2')
```

Format

`V2` is a matrix of coordinates of the vertices over an irregular domain.

References

This example is generated from R package “Triangulation".

Index

∗Topic **datasets**
    Tr1, 10
    Tr2, 11
    V1, 11
    V2, 12

basis, 2

cv.BPST, 3

data.BPST, 5

fit.BPST, 6

inVT, 8

plot.BPST, 9
predict.BPST, 9

Tr1, 10
Tr2, 11

V1, 11
V2, 12