Package ‘RPDE’

June 26, 2015

Version 0.9-1

Date 2015-06-18

Title Regression with Partial Differential Equations penalization

Author Eardi Lila [aut, cre],Laura Sangalli [aut],Jim Ransay [aut]

Maintainer Eardi Lila <eardi.lila@mail.polimi.it>

Depends R (>= 3.0.0), RTriangle, rgl

Suggests MASS

Description

The package implements a model for the analysis of data distributed over irregularly shaped spatial domains with complex boundaries, strong concavities and interior holes. Moreover the estimation of surfaces and spatial fields can take into account a prior knowledge on the phenomenon under study. The prior knowledge included in the model derives from physics, physiology or mechanics of the problem at hand, and is formalized in terms of a partial differential equation governing the phenomenon behavior. An accurate surface estimation is achieved by means of piecewise linear and quadratic finite elements.

License CC BY-NC-SA 3.0

NeedsCompilation yes

R topics documented:

create.FEM.basis *Creates a Finite Element Method basis*

Description

Sets up a finite element basis for the analysis of spatial data. It requires a TRIMESH2D object as input. The finite elements used for functional data analysis are first or second order Lagrangian elements. These are triangles covering a region and the basis system is piecewise polinomials (linear or quadratic). There is a basis function associated with each node in the system. When ORDER = 1 the basis system is piecewise linear and the nodes are the vertices of the triangles. When ORDER = 2 the basis system is piecewise quadratic and the nodes are points that are etiher the vertices of the triangles or midpoints of edges of triangles.

Usage

```
create.FEM.basis(mesh, order, CPP_CODE = FALSE)
```

Arguments

`mesh` A `TRIMESH2D` object; see `create.MESH.2D`.

`order` Order of elements, which may be either 1 or 2. The order of the element must be less or equal respect to the order of the mesh.

`CPP_CODE` if `TRUE` avoids the computation of some additional elements, not necessary if the functions working with the FEM basis are called with the flag `CPP_CODE=TRUE`

Value

A object of class `FEM`. This contains the mesh along with the following variables:

`order` Order of elements.

`nbasis` The number of basis.

`J` The area of each triangle of the basis.

`transf` A matrix such that J[i,,] is the 2-by-2 tranformation matrix such that transforms the nodes of the reference triangle in the node of the i-th triangle.

`metric` A matrix such that metric[i,,] is the 2-by-2 tranformation matrix equal to `J[i,,]^{-1}*J[i,,]^{-T}`

Examples

```
## Creates an object TRIMESH2D with a concavity and second order nodes
mesh<-create.MESH.2D(nodes=rbind(c(0, 0), c(0, 1), c(0.5, 0.5), c(1, 1), c(1, 0)),
                    segments=rbind(c(1, 2), c(2, 3), c(3, 4), c(4, 5), c(5, 1)), order=1)
## Plot it
plot(mesh)

# Creates the basis object
basisobj = create.FEM.basis(mesh, order=1)
```

`create.MESH.2D` *Creates a TRIMESH2D object*

Description

Creates a mesh object starting from a domain described by the input parameters. Whether only the nodes are specified the mesh is constructed on the convex hull. This function is based on the Triangle library (`http://www.cs.cmu.edu/~quake/triangle.html`). The triangulation is a constrained conforming Delaunay triangulation in which additional vertices, called Steiner points, can be inserted into segments to improved the quality of the triangulation.

Usage

```
create.MESH.2D(nodes, nodesmarkers = NA, nodesattributes = NA, segments = NA, segmentsmarkers = NA, hc
```

Arguments

`nodes` A n-by-2 matrix of nodes to be used as triangles' vertices.

`nodesmarkers` A vector of length n specifying whether the node is on the border or not.

`nodesattributes` A n-by-#attributes matrix specifying a set of attributes for each node.

`segments` A matrix where each row is composed by two nodes indexes, specifying which segments need to be preserved in the triangulation. Commonly used for specifying the border.

`segmentsmarkers` A matrix where the i-th row specifies a set of attributes of i-th segments.

`holes` A matrix where the i-th row specifies the x and y coordinate of an hole.

`triangles` A 3 or 6 column matrix, respectively for 1st and 2nd order meshes, in which each row contains indices in `P` of vertices. Useful to create a TRIMESH object when a triangulation has already been performed with an external software.

`order` A numeric that can assume values `1,2`, that specifies the order of the elements of the traingulation

`verbosity` A numeric that can assume values `0,1,2`, that specifies the output verbosity in the triangulation process.

Value

An object of class `TRIMESH2D`. This contains an updated list of the input parameters

Examples

```
## Creates an object TRIMESH2D on the convex hull of the specified nodes
mesh<-create.MESH.2D(nodes=rbind(c(0, 0), c(0, 1), c(0.5, 0.5), c(1, 1), c(1, 0)))
## Plot it
plot(mesh)

## Creates an object TRIMESH2D with a concavity and second order nodes
mesh<-create.MESH.2D(nodes=rbind(c(0, 0), c(0, 1), c(0.5, 0.5), c(1, 1), c(1, 0)),
                     segments=rbind(c(1, 2), c(2, 3), c(3, 4), c(4, 5), c(5, 1)))
## Plot it
plot(mesh)

## Creates an object TRIMESH2D with second order nodes starting from a first order triangulation
## specified by nodes and triangles
mesh<-create.MESH.2D(nodes=rbind(c(0, 0), c(0, 1), c(0.5, 0.5), c(1, 1), c(1, 0)),
                     triangles=rbind(c(2,1,3), c(3,5,4), c(5,3,1)), order = 2)
## Plot it
plot(mesh)
```

MeuseBorder *Border of Meuse river data set*

Description

This data set gives the descriptions of the domain's border of the Meuse river data set

Usage

```
data(MeuseBorder)
```

Format

This data frame contains the following columns:

- **V1** Index of the starting point of the border segments
- **V2** Index of the ending point of the border segments

Examples

```
data(MeuseBorder)
summary(MeuseBorder)
```

MeuseData *Meuse river data set – original, full data set*

Description

This data set gives locations and top soil heavy metal concentrations (ppm), along with a number of soil and landscape variables, collected in a flood plain of the river Meuse, near the village Stein. Heavy metal concentrations are bulk sampled from an area of approximately 15 m x 15 m.

Usage

```
data(MeuseData)
```

Format

This data frame contains the following columns:

sample sample number

x a numeric vector; x-coordinate (m) in RDM (Dutch topographical map coordinates)

y a numeric vector; y-coordinate (m) in RDM (Dutch topographical map coordinates)

cadmium topsoil cadmium concentration, ppm.; note that zero cadmium values in the original data set have been shifted to 0.2 (half the lowest non-zero value)

copper topsoil copper concentration, ppm.

lead topsoil lead concentration, ppm.

zinc topsoil zinc concentration, ppm.

elev relative elevation

om organic matter, as percentage

ffreq flooding frequency class

soil soil type

lime lime class

landuse landuse class

dist.m distance to river Meuse (metres), as obtained during the field survey

in.pit logical; indicates whether this is a sample taken in a pit

in.meuse155 logical; indicates whether the sample is part of the `meuse` (i.e., filtered) data set; in addition to the samples in a pit, an sample (139) with outlying zinc content was removed

in.BMcD logical; indicates whether the sample is used as part of the subset of 98 points in the various interpolation examples of Burrough \& McDonnell

Note

`sample` refers to original sample number. Eight samples were left out because they were not indicative for the metal content of the soil. They were taken in an old pit. One sample contains an outlying zinc value, which was also discarded for the meuse (155) data set.

Author(s)

The actual field data were collected by Ruud van Rijn and Mathieu Rikken; data compiled for R by Edzer Pebesma

References

P.A. Burrough, R.A. McDonnell, 1998. Principles of Geographical Information Systems. Oxford University Press.

http://www.gstat.org/

Examples

```
data(MeuseData)
summary(MeuseData)
```

refine.MESH.2D *Refine the triangulation*

Description

Refines a TRIMESH2D object following the constrained imposed with the input parameters. This function is based on the Triangle library (http://www.cs.cmu.edu/~quake/triangle.html). The triangulation is a constrained conforming Delaunay triangulation in which additional vertices, called Steiner points, can be inserted into segments to improved the quality of the triangulation.

Usage

```
refine.MESH.2D(mesh, minimum_angle = NA, maximum_area = NA, delaunay = FALSE, verbosity = 0)
```

Arguments

`mesh`	A TRIMESH2D object created through `create.MESH.2D`.
`minimum_angle`	A numeric specifying the minimum angle of the triangles in the ouput triangulation.
`maximum_area`	A numeric specifying the maximum area of the triangles in the ouput triangulation.
`delaunay`	If `TRUE` the output triangulation is a Delaunay triangulation
`verbosity`	A numeric that can assume values `0,1,2`, that specifies the output verbosity in the triangulation process.

Value

An object of class `TRIMESH2D`.

Examples

```
## Creates an object TRIMESH2D with a concavity and second order nodes
mesh_coarse<-create.MESH.2D(nodes=rbind(c(0, 0), c(0, 1), c(0.5, 0.5), c(1, 1), c(1, 0)),
                     segments=rbind(c(1, 2), c(2, 3), c(3, 4), c(4, 5), c(5, 1)))
## Plot it
plot(mesh_coarse)
## Refines the the triangulation in specified in the \code{mesh_coarse} object
mesh<-refine.MESH.2D(mesh_coarse,maximum_area = 0.005, delaunay = TRUE)
## Plot the refined mesh
plot(mesh)
```

`smooth.FEM.basis` *Compute a solution for a Spatial Spline problem*

Description

Compute a solution for a Spatial Spline problem followinf the model in: Sangalli, Ramsay, Ramsay (2013).

Usage

```
smooth.FEM.basis(locations = NULL, observations, basisobj, lambda,
                 covariates = NULL, BC = NULL, GCV = TRUE,
                 CPP_CODE = TRUE)
```

Arguments

`observations` A vector specifying the observed values on the domain. The locations of the observations can be specified with the `locations` argument, otherwise the locations are intented to be the corresponding nodes of the mesh. `NA` values are admissible to indicate the missing value on the corresponding node.

`locations` A 2 column matrix where each row specifies the coordinates of the corresponding observation.

`basisobj` An onbject of type FEM; See `create.FEM.basis`.

`lambda` A scalar smoothing parameter.

`covariates` A design matrix where each row represents the covariates associated to each row.

`BC` A list with two vectors: `Indices`, a vector with the indices for the border points to apply a Dirichlet Border Condition;`Values` a vector with the values that the the nodes specified in `Indices` must assume.

`GCV` If `TRUE` computes the trace of the smoothing matrix, the estimate of the error's variance and the Generalized Cross Validation parameter, for value of `lambda`.

`CPP_CODE` if `TRUE` avoids the computation of some additional elements, not necessary if the functions working with the FEM basis are called with the flag `CPP_CODE=TRUE`

Value

A list with the following variables:

FELSPLOBJ
: A FOBJ object of the FEM type defined by the coefficients vector resulting from smoothing.

LAPLACEFD
: A FOBJ object of the FEM type for the value of the Laplace operator

DOF
: If GCV is TRUE, a vector with the trace of the smoothing matrix for each lambda.

sigma
: If GCV is TRUE, a vector with the estimate of the standard deviation of the error for each lambda.

GCV
: If GCV is TRUE, a vector with the GCV index for each lambda.

Examples

```
data(meuseall)
## Creates an object TRIMESH2D with a concavity and second order nodes
mesh<-create.MESH.2D(nodes=rbind(c(0, 0), c(0, 1), c(0.5, 0.5), c(1, 1), c(1, 0)),
                    segments=rbind(c(1, 2), c(2, 3), c(3, 4), c(4, 5), c(5, 1)), order=1)
## Plot it
plot(mesh)

# Creates the basis object
basisobj = create.FEM.basis(mesh, order=1)
```

---

smooth.FEM.PDE.basis *Compute a solution for a Spatial Regression with PDE Penalization*

Description

Compute a solution for a for a Spatial Regression with PDE Penalization model in Azzimonti et al (2013).

Usage

```
smooth.FEM.PDE.basis(locations = NULL, observations, basisobj,
                  lambda, PDE_parameters, covariates = NULL, BC = NULL, GCV = TRUE,
                      CPP_CODE = TRUE)
```

Arguments

observations
: A vector specifying the observed values on the domain. The locations of the observations can be specified with the locations argument, otherwise the locations are intented to be the corresponding nodes of the mesh. NA values are admissible to indicate the missing value on the corresponding node.

locations
: A 2 column matrix where each row specifies the coordinates of the corresponding observation.

basisobj
: An onbject of type FEM; See create.FEM.basis.

`lambda` A scalar smoothing parameter.

`PDE_parameters` A list containing the parameters of the penalizing PDE, with: `K` a 2-by-2 matrix indicating the diffusion coefficient matrix, `beta` a vector of length 2 with the coefficients of the advection coefficients and `c` a numeric indicating the reaction coefficient.

`covariates` A design matrix where each row represents the covariates associated to each row.

`BC` A list with two vectors: `Indices`, a vector with the indices for the border points to apply a Dirichlet Border Condition;`Values` a vector with the values that the the nodes specified in `Indices` must assume.

`GCV` If `TRUE` computes the trace of the smoothing matrix, the estimate of the error's variance and the Generalized Cross Validation parameter, for value of `lambda`.

`CPP_CODE` if `TRUE` avoids the computation of some additional elements, not necessary if the functions working with the FEM basis are called with the flag `CPP_CODE=TRUE`

Value

A list with the following variables:

`FELSPLOBJ` A FOBJ object of the FEM type defined by the coefficients vector resulting from smoothing.

`LAPLACEFD` A FOBJ object of the FEM type for the value of the Laplace operator

`DOF` If GCV is `TRUE`, a vector with the trace of the smoothing matrix for each lambda.

`sigma` If GCV is `TRUE`, a vector with the estimate of the standard deviation of the error for each lambda.

`GCV` If GCV is `TRUE`, a vector with the GCV index for each lambda.

Examples

```
## Creates an object TRIMESH2D with a concavity and second order nodes
mesh<-create.MESH.2D(nodes=rbind(c(0, 0), c(0, 1), c(0.5, 0.5), c(1, 1), c(1, 0)),
                     segments=rbind(c(1, 2), c(2, 3), c(3, 4), c(4, 5), c(5, 1)), order=1)
## Plot it
plot(mesh)

# Creates the basis object
basisobj = create.FEM.basis(mesh, order=1)
```

`smooth.FEM.PDE.SV.basis` *Compute a solution for a Spatial Spline problem*

Description

Compute a solution for a Spatial Spline problem followinf the model in: Sangalli, Ramsay, Ramsay (2013).

Usage

```
smooth.FEM.PDE.SV.basis(locations = NULL, observations, basisobj, lambda, PDE_parameters, covariates
```

Arguments

`observations` A vector specifying the observed values on the domain. The locations of the observations can be specified with the `locations` argument, otherwise the locations are intented to be the corresponding nodes of the mesh. `NA` values are admissible to indicate the missing value on the corresponding node.

`locations` A 2 column matrix where each row specifies the coordinates of the corresponding observation.

`basisobj` An onbject of type FEM; See `create.FEM.basis`.

`lambda` A scalar smoothing parameter.

`PDE_parameters` A list containing the space varying parameters of the penalizing PDE, with: `K` a function that for each point of the domain specified as a vector of length 2, returns a 2-by-2 matrix indicating the diffusion coefficient matrix, `beta` a function that for each point of the domain specified as a vector of length 2, returns a vector of length 2 indicating the advection coefficients and `c` a function that for each point of the domain specified as a vector of length 2, returns a numeric indicating the reacttion coefficient.

`covariates` A design matrix where each row represents the covariates associated to each row.

`BC` A list with two vectors: `Indices`, a vector with the indices for the border points to apply a Dirichlet Border Condition;`Values` a vector with the values that the the nodes specified in `Indices` must assume.

`GCV` If `TRUE` computes the trace of the smoothing matrix, the estimate of the error's variance and the Generalized Cross Validation parameter, for value of `lambda`.

`CPP_CODE` if `TRUE` avoids the computation of some additional elements, not necessary if the functions working with the FEM basis are called with the flag `CPP_CODE=TRUE`

Value

A list with the following variables:

`FELSPLOBJ` A FOBJ object of the FEM type defined by the coefficients vector resulting from smoothing.

`LAPLACEFD` A FOBJ object of the FEM type for the value of the Laplace operator

`DOF` If GCV is `TRUE`, a vector with the trace of the smoothing matrix for each lambda.

`sigma` If GCV is `TRUE`, a vector with the estimate of the standard deviation of the error for each lambda.

`GCV` If GCV is `TRUE`, a vector with the GCV index for each lambda.

Examples

```
## Creates an object TRIMESH2D with a concavity and second order nodes
mesh<-create.MESH.2D(nodes=rbind(c(0, 0), c(0, 1), c(0.5, 0.5), c(1, 1), c(1, 0)),
                    segments=rbind(c(1, 2), c(2, 3), c(3, 4), c(4, 5), c(5, 1)), order=1)
## Plot it
plot(mesh)

# Creates the basis object
basisobj = create.FEM.basis(mesh, order=1)
```

Index