Package 'constrainedKriging'

April 30, 2015

Version 0.2.4

Type Package

Title Constrained, Covariance-Matching Constrained and Universal Point or Block Kriging

Date 2015-04-23

Author Christoph Hofer

Maintainer Christoph Hofer <christoph.hofer@zhaw.ch>

Description Provides functions for efficient computations of nonlinear spatial predictions with local change of support. This package supplies functions for tow-dimensional spatial interpolation by constrained, covariance-matching constrained and universal (external drift) kriging for points or block of any shape for data with a nonstationary mean function and an isotropic weakly stationary variogram. The linear spatial interpolation methods, constrained and covariance-matching constrained kriging, provide approximately unbiased prediction for nonlinear target values under change of support. This package extends the range of geostatistical tools available in R and provides a veritable alternative to conditional simulation for nonlinear spatial prediction problems with local change of support.

License GPL (>= 2)

LazyLoad yes

LazyData yes

Depends R (>= 2.9.0), sp(>= 0.9-60), spatialCovariance(>= 0.6-4), methods

Imports rgeos(>= 0.2-17), RandomFields(>= 1.3.41),

Suggests spdep(>= 0.5-43)

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-04-30 23:59:28

R topics documented:

`constrainedKriging-package`

constrainedKriging a package for nonlinear spatial predictions with local change of support

---

Description

The constrainedKriging package provides functions for tow-dimensional spatial interpolation by constrained, covariance-matching constrained and universal (external drift) kriging for points or block of any shape for data with a nonstationary mean function and an isotropic weakly stationary variogram. The linear spatial interpolation methods, constrained and covariance-matching constrained kriging, provide approximately unbiased prediction for nonlinear target values under change of support.

In principle, the package provides two user functions, preCKrige and CKrige to calculate spatial prediction in two steps:

1. Call of `preCKrige` to calculate the variance-covariance matrices for defined sets of points or polygons (blocks).

2. Call of `CKrige` by using the output of preCKrige to calculate the spatial interpolation by one of the three kriging methods.

Details

The constrained kriging predictor proposed by Cressie (1993) and the covariance-matching constrained kriging predictor proposed by Aldworth and Cressie (2003) are linear in the data like the universal kriging predictor. However, the constrained kriging predictor satisfies in addition to the unbiasedness constraint of universal kriging a second constraint that matches the variances of the predictions to the variances of the prediction target (either a point value or a block mean).

The covariance-matching constrained kriging predictor matches for a set of blocks both the variances and covariances of predictions and prediction targets (points or block means) and is the extended version of the constrained kriging predictor.

Like constrained kriging, covariance-matching constrained kriging is less biased than universal kriging for nonlinear predictions and exactly unbiased if the spatial variable is Gaussian.

The formulas of the three kriging predictors are given below:

- universal kriging predictor for 1 block:

$$\widehat{Y}_{\mathrm{UK}}(B) = \mathbf{x}(B)'\widehat{\beta}_{\mathrm{GLS}} + \mathbf{c}'\mathbf{\Sigma}(\mathbf{Z} - \mathbf{X}\widehat{\beta}_{\mathrm{GLS}}),$$

- constrained kriging predictor for 1 block:

$$\widehat{Y}_{\mathrm{CK}}(B) = \mathbf{x}(B)'\widehat{\beta}_{\mathrm{GLS}} + K\mathbf{c}'\mathbf{\Sigma}(\mathbf{Z} - \mathbf{X}\widehat{\beta}_{\mathrm{GLS}}),$$

- covariance-matching constrained kriging predictor for m blocks:

$$\widehat{\mathbf{Y}}_{\mathrm{CMCK}} = \mathbf{X}_m\widehat{\beta}_{\mathrm{GLS}} + \mathbf{K}'\mathbf{C}'\mathbf{\Sigma}(\mathbf{Z} - \mathbf{X}\widehat{\beta}_{\mathrm{GLS}}),$$

where $\mathbf{Z} = c(Z(s_1), \ldots, Z(s_n))'$ is the vector with the data; $\mathbf{s} = (x, y)'$ indicates a location in the survey domain and $Y(B)$ is the block mean value of the block area B; $\mathbf{X} = (\mathbf{x}_1(B), \ldots, \mathbf{x}_n(B))'$ is the design matrix of the data and $\mathbf{X}_m$ is the design matrix of the target blocks; $\widehat{\boldsymbol{\beta}}_{\mathrm{GLS}}$ is the vector with the generalised least square estimate of the linear regression coefficients; $\mathbf{C} = (\mathbf{c}_1, \ldots, \mathbf{c}_m)$ is a (n,m)-matrix that contains the covariances between the m prediction targets (point or blocks) and the n data points; $\mathbf{\Sigma}$ is the covariance matrix of the data; the scalar

$$K = (\mathrm{Var}[Y(B)] - \mathrm{Var}[\mathbf{x}(B)\widehat{\boldsymbol{\beta}}_{\mathrm{GLS}}])^{\frac{1}{2}} / (\mathrm{Var}[\widehat{Y}_{UK}(B)] - \mathbf{Var}[\mathbf{x}(B)\widehat{\boldsymbol{\beta}}_{\mathrm{GLS}}])^{\frac{1}{2}} = (P/Q)^{\frac{1}{2}}$$

and the (m, m)-matrix

$$\mathbf{K} = \mathbf{Q}_1^{-1}\mathbf{P}_\mathbf{1},$$

where the (m,m)-matrix

$$\mathbf{P}_1 = (\mathrm{Cov}[\mathbf{Y}, \mathbf{Y}'] - \mathrm{Cov}[\mathbf{X}_m\widehat{\boldsymbol{\beta}}_{\mathrm{GLS}}, (\mathbf{X}_m\widehat{\boldsymbol{\beta}}_{\mathrm{GLS}})'])^{\frac{1}{2}}$$

and the (m, m)-matrix

$$\mathbf{Q}_1 = (\mathrm{Cov}[\mathbf{Y}_{\mathrm{UK}}, \mathbf{Y}'_{\mathrm{UK}}] - \mathrm{Cov}[\mathbf{X}_m\widehat{\boldsymbol{\beta}}_{\mathrm{GLS}}, (\mathbf{X}_m\widehat{\boldsymbol{\beta}}_{\mathrm{GLS}})'])^{\frac{1}{2}}$$

The mean square prediction error (MSEP) of the three predictors are:

- MSPE of the universal kriging predictor for 1 block:

$$MSPE[\widehat{Y}_{UK}(B)] = \mathrm{Cov}[(\widehat{Y}_{UK}(B) - Y(B))(\widehat{Y}_{UK}(B) - Y(B))']$$

- MSPE of the constrained kriging predictor for 1 block:

$$MSPE[\widehat{\mathbf{Y}}_{CK}] = MSPE[\widehat{\mathbf{Y}}_{UK}] + (P^{\frac{1}{2}} + Q^{\frac{1}{2}})^2,$$

- MSPE of the covariance-matching constrained kriging predictor for m blocks:

$$MSPE[\widehat{\mathbf{Y}}_{CMCK}] = MSPE[\widehat{\mathbf{Y}}_{UK}] + (\mathbf{P}_1 - \mathbf{Q}_1)(\mathbf{P}_1 - \mathbf{Q}_1).$$

Author(s)

Christoph Hofer <christoph.hofer@allumni.ethz.ch>

References

Aldworth, J. and Cressie, N. (2003). Prediction of nonlinear spatial functionals. *Journal of Statistical Planning and Inference*, **112**, 3–41

Cressie, N. (1993). Aggregation in geostatistical problems. In A. Soares, editor, *Geostatistics Troia 92*, **1**, pages 25–36, Dordrecht. Kluwer Academic Publishers.

Hofer, C. and Papritz, A. (2010). Predicting threshold exceedance by local block means in soil pollution surveys. *Mathematical Geosciences*. **42**, 631-656, doi: 10.1007/s11004-010-9287-4

Hofer, C. and Papritz, A. (in preparation). constrainedKriging: an R-package for customary, constrained and covariance-matching constrained point or block kriging. *Computers & Geosciences*.

`CKrige` *Spatial interpolation (2D) by constrained, covariance-matching constrained and universal, global Point or Block Kriging*

Description

Function for constrained, covariance-matching constrained and universal (external drift kriging) point or block (of any shape) kriging in a global neighbourhood and for isotropic covariance models.

Usage

```
CKrige( formula, data, locations, object, ...)

## S4 method for signature 'formula,data.frame,formula,preCKrigePolygons'
CKrige(formula, data, locations, object, method = 2, ex.out = F)

## S4 method for signature 'formula,data.frame,formula,preCKrigePoints'
CKrige(formula, data, locations, object, method = 2, ex.out = F)
```

Arguments

`formula`	formula of the linear regression model in the form `response ~ terms of covariates`, for ordinary kriging use the formula `response ~ 1`
`data`	a data frame with the values of the covariates, the names of the covariates used in the formula object must match the column names of `data`.
`locations`	a `formula` object that describe the coordinates of the data locations (e.g `~ x+y`)
`object`	either an object of the class "`preCKrigePolygons`" for block kriging or of the class "`preCKrigePoints`" for point kriging. In general the output object of the `preCKrige` function.
`...`	two further arguments to control the spatial interpolation method and the output
`method`	numeric value to choose the kriging method `1` universal (external drift), `method = 2` constrained, `method = 3` covariance-matching constrained kriging. By default, `method = 2`
`ex.out`	logical value, if `ex.out` is set `TRUE` `CKrige` returns an extended output with additional information, see details for more informations, by default `ex.out = F`

Details

The CKrige function depends always on a preCKrige output object that contains the parameter of the isotropic covariance model as well as the covariates of the prediction targets.

Value

By default, CKrige returns an object of the class SpatialPointsDataFrame or SpatialPolygonsDataFrame depending whether the input object for the object argument is of the class "preCKrigePoints" or "preCKrigePolygons".

The data frame of the returned object contains the following columns independent of the selected kriging method:

prediction	numeric vector with the kriging prediction of the chosen method
prediction.se	numeric vector with the root mean square error (kriging standard error)

The data frame contains 3 additional columns with constrained kriging parameters, if the argument method = 2 of the CKrige function:

sqrt.P	numeric vector with sqrt(Var[target point or block] - Var[fitted values])
sqrt.Q	numeric vector with sqrt(Var[universal kriging predictor] - Var[fitted values])
K	numeric vector with sqrt.P / sqrt.Q

The data frame contains 3 additional columns with covariance-matching constrained kriging parameters, if the argument method = 3 of the CKrige function:

P1.11	numeric vector, first element of the matrix P1 = (Cov[target point or block] - Cov[fitted values])^(1/2)
Q1.11	numeric vector, first element of the matrix Q1 = (Cov[universal kriging predictor] - Cov[fitted values])^(1/2)
K.11	numeric vector, first element of the matrix K = O1^-1P1[1,1]

The CKrige function returns a list with the following components if the argument ex.out = T and the argument method is either 1 or 2:

object	either an object of the class SpatialPolygonsDataFrame or SpatialPointsDataFrame as described above
krig.method	numeric scalar, number of the chosen kriging method 1, 2 or 3.
parameter	list with 2 components. First component beta.coef is the vector with the Generalized Least Square coefficients of the linear regression and the second component cov.beta contains the covariance matrix of the Generalized Least Square coefficients.

sk.weights	if argument method = 1 or method = 2 sk.weights is a matrix with the simple kriging weights. The ith column contains the simple kriging weights of the ith prediction target object. If the argument method = 3 the list component sk.weights is a list. Each list component contains the matrix with the simple kriging weights of the prediction target and its defined neighbours.
inv.Sigma	matrix, inverse covariance matrix of the data
residuals	numeric vector with the Generalized Least Square residuals of the linear regression.

The list of the extended output contains the additional component CMCK.par if the argument method = 3. The CMCK.par component is a list of lists with CMCK parameters, in particular P1 list of the P1 matrices, Q1 list of the Q1 matrices and K list of the K matrices.

Note

print and summary methods for the different CKrige output objects are available.

Author(s)

Christoph Hofer, <christoph.hofer@alumni.ethz.ch>

References

See main help page of the **constrainedKriging** package.

See Also

preCKrige

Examples

```
## Not run:
# load data
data(meuse,meuse.blocks)

# approximation of block variance
# pixel area = 75m x 75m
# exponential covariance function with measurement error = 0, nugget = 0.05,
# part. sill =  0.15 and range parameter = 192.5


preCK=preCKrige(newdata=meuse.blocks,model=
    covmodel("exponential",0,0.05,0.15,192.5),pwidth=75,pheight=75)

# block prediction by constrained kriging on the log scale

CK=CKrige(formula=log(zinc)~sqrt(dist),data=meuse,
locations=~x+y,object=preCK,ex.out=TRUE)

# backtransformation to the original scale for the CK prediction
```

```
beta=CK$parameter$beta.coef
M=meuse.blocks@data$M
c1 <-  0.2
c2  <- beta[2]^2 * meuse.blocks@data$M
CK$object@data$Zn=exp(CK$object@data$prediction
+ 0.5*(0.2+beta[2]^2*M-unlist(preCK@covmat)))

# U: upper limits of the relative 95
# U multiplied by the predictions CK$object@data$Zn gives
# the upper limits of the 95

CK$object@data$U=exp(CK$object@data$prediction
    +1.96*CK$object@data$prediction.se) /CK$object@data$Zn

# plots with spplot, generic function in the sp package
# the plot shows the constrained kriging predictions on
# the orginal scale
# function ck.colors(n) create a rainbow-like  color vector

breaks <- seq(0, 1850, by = 185)
spplot(CK$object,zcol="Zn",at=breaks,col.regions=ck.colors(10),
colorkey=list(labels=list(at=breaks,labels=breaks)))

# plot of the factor to get the upper bound of the
97.5

breaks=seq(1,3.2,by=0.2)
spplot(CK$object,zcol="U",at=breaks,col.regions=ck.colors(11),
colorkey=list(labels=list(at=breaks,labels=breaks)))

## End(Not run)
```

`covmodel` *Create isotropic covariance model*

Description

Function to generate isotropic covariance models, or add an isotropic covariance model to an existing isotropic model

Usage

```
covmodel(modelname, mev, nugget,variance, scale,
parameter, add.covmodel)

## S3 method for class 'covmodel'
print(x, ...)
```

Arguments

modelname	character vector, name of the covariance model, e.g. "exponential", "spherical", "gauss". A call of covmodel() without a function argument displays a table with all available models and their parameters. Check the CovarianceFct in the **RandomFields** package for detailed information about the covariance functions.
mev	numeric value, variance of the measurement error
nugget	numeric value, variance of microstructure white noise process (range smaller than the data support)
variance	numeric value, partial sill of the variogram model
scale	numeric value, scale parameter of the variogram model
parameter	numeric vector of covariance parameters, missing for some model like nugget, spherical or gauss or
add.covmodel	object of the class covmodel that is added to the covariance model defined by modelname (see examples)
x	a covariance model generated by covmodel
...	further printing arguments

Value

an object of the class covmodel that define a covariance model.

Note

The names and parametrisation of the covariance model originate from the CovarianceFct in the **RandomFields** package. The values of the arguments mev, nugget, variance and scale are by default = 0.

Please, be aware that you only can generate spatial isotropic covariance models, Time-Space models or so called (hypermodels) are not implemented.

Author(s)

Christoph Hofer <christoph.hofer@alumni.ethz.ch>

Examples

```
## Not run:
# table with all available covariance models and their
# parameters
covmodel()

# exponential model without a measurement error and without a nugget,
# partial sill = 10, scale  parameter = 15
covmodel(modelname = "exponential", variance = 10, scale = 15)

# exponential model with a measurement error ( mev = 0.5) and a
# nugget (nugget = 2.1), exponential partial  sill (variance = 10)
# and scale parameter = 15
```

```
covmodel(modelname = "exponential", mev  = 0.5, nugget = 2.1,
variance = 10, scale = 15)

## End(Not run)
```

`meuse.blocks` *Meuse block*

Description

`meuse.blocks` is an object of the class `SpatialPolygonsDataFrame` that contains the coordinates of 259 artificially defined by putting a grid with 150 m mesh width over the the flood plain area of the river Meuse, near the village Stein. The 259 x 2 data frame contains the covariate `dist` and the attribute `M` for each block.

Usage

```
data(meuse.blocks)
```

Format

The object contains the following slots:

polygons an object of the class `SpatialPolygons` that contains the coordinates of the 259 blocks

data a 259 x 2 data frame contains:

- **dist** mean Euclidean distance of the blocks from the river, normalized to the interval [0;1]
- **M** the (spatial) variance of the mean distance between points, uniformly distributed within the blocks, and the river

Author(s)

Christoph Hofer <christoph.hofer@alumni.ethz.ch>

See Also

preCKrige and CKrige

Examples

```
## Not run:
data(meuse.blocks)
summary(meuse.blocks)
### show the shape of the 259 blocks
plot(meuse.blocks)
### plot maps of the covariate \code{dist} and attribute \code{M}
spplot(meuse.blocks)

## End(Not run)
```

plot.preCKrigePolygons *Plotting polygon configurations and their approximated areas*

Description

Plotting method for objects of the class `preCKrige.polygons`. The plot show the block neighbourhood configuration for one polygon (block) of a `preCKrige.polygons` object as well as its area approximation by the pixels.

Usage

```
## S3 method for class 'preCKrigePolygons'
plot(x, index, ...)
```

Arguments

`x`	object of the class `preCKrigePolygons`. In general the output object of a `preCKrige` function call.
`index`	numeric value, list index of the desired polygon (block) of the polygon list `x@polygons`
`...`	further plotting parameters

Author(s)

Christoph Hofer <christoph.hofer@alumni.ethz.ch>

See Also

`preCKrige`

Examples

```
## Not run:
### load data
data(meuse,meuse.blocks)

### plot blocks
plot(meuse.blocks)

### compute the approximated block variance of each block in
### meuse.blocks without the definition of neighbours blocks (default)
preCK_1  <- preCKrige(newdata = meuse.blocks,
    model = covmodel("exponential", 0.05, 0.15, scale = 192.5),
    pwidth = 75, pheight = 75)

### plot block approximation of block 59
plot(preCK_1, 59)
```

```
### define neighbours
if(require(spdep))
{
neighbours <- poly2nb(meuse.blocks)
class(neighbours)
### neighbours should be an object of the class "list"
class(neighbours) <- "list"
### compute the approximated block variance-covariance matrices of each block in
### meuse.blocks without the defined block neighbours
preCK_2 <- preCKrige(newdata = meuse.blocks, neighbours = neighbours,
    model = covmodel("exponential", 0.05, 0.15, scale = 192.5),
    pwidth = 75, pheight = 75)

### plot block approximation of block 59 and its
### block neighbours
plot(preCK_1, 59)
}
if(!require(spdep))
{
cat("Please, install the package spdep to excute this example.\n")
}


## End(Not run)
```

---

preCKrige *Spatial Variance-Covariance Matrices for Points and Polygons of any Shape*

Description

The function preCKrige provides (approximated) spatial variance-covariance matrices for user defined sets of polygons (blocks) of any shape or points for two-dimensional isotropic random fields. The polygon (block) areas of a set of polygons (polygon neighbourhood configuration) are approximated by pixels and the block-block covariances are approximated by averaging the pixel covariances of the approximated polygon (block) areas.

The returned object of preCKrige is, in general, needed by CKrige for spatial point or block interpolation by constrained, covariance-matching constrained or universal (external drift) kriging.

Usage

```
preCKrige( newdata, neighbours, model, ... )

## S4 method for signature 'SpatialPoints,ANY,covmodel'
preCKrige(newdata, neighbours, model)

## S4 method for signature 'SpatialPointsDataFrame,ANY,covmodel'
```

```
preCKrige(newdata, neighbours, model)

## S4 method for signature 'SpatialPolygons,ANY,covmodel'
preCKrige(newdata, neighbours, model,
pwidth = 0, pheight = 0, napp = 1)

## S4 method for signature 'SpatialPolygonsDataFrame,ANY,covmodel'
preCKrige(newdata, neighbours, model,
pwidth = 0, pheight = 0, napp = 1)
```

Arguments

newdata	either an object of the classes "SpatialPoints" or "SpatialPointsDataFrame" that contains the point coordinates and if necessary additional point information (covariates) stored in the data.frame of the SpatialPointsDataFrame object, or an object of the classes "SpatialPolygons" or "SpatialPolygonsDataFrame" with the coordinates of the polygons (block) and if necessary additional polygon information (covariates) stored in the data frame of the SpatialPointsDataFrame object.
neighbours	list of n integer vectors, where n = number of points if newdata is an object of the classes "SpatialPoints" or "SpatialPointsDataFrame" or n = number of polygons (blocks) if newdata is an object of the classes "SpatialPolygons" or "SpatialPolygonsDataFrame". The ith list component define the neighbours of the ith point or polygon (block) in newdata. If newdata is an object of the classes "SpatialPolygons" or "SpatialPolygonsDataFrame" each list component contains the list indices of the neighbour polygons of the polygon list newdata@polygons. If newdata is an object of the classes "SpatialPoints" or "SpatialPointsDataFrame" each list component contains the row indices of the neighbour points from the point-coordinates matrix. The ith list component is set to integer(0) if the ith polygon or point have no (defined) neighbours. By default, the points or polygons have no neighbours. See the second example where the function poly2nb of the **spedp** package is used to build such a neighbours list.
model	object of class "covmodel". The object contains the parameter of the isotropice covariance function, generated by the function covmodel.
...	further arguments if newdata is of class "SpatialPolygons" or "SpatialPolygonsDataFrame"
pwidth	positive numeric scalar, defines the width of the pixels which are used to approximate the polygon (block) area.
pheight	positive numeric scalar, defines the height of the pixels which are used to approximate the polygon (block) area.
napp	positive integer scalar. napp > 1 reduces the block-block variance-covariance approximation error. By default, napp = 1. See details.

Details

If the objects for newdata are of the classes "SpatialPolygons" or "SpatialPolygonsDataFrame", preCKrige searches the polygon (block) neighbourhood configuration (defined by the argument neighbours) with the largest bounding box and generates a pixel gird with the same area as the largest bounding box. Subsequently, the covariance matrix of this pixels is calculated by using the **spatialCovariance** package and the polygon (block) areas of each polygon neighbourhood configuration are approximated by the pixels. Finally, the block-block covariances are approximated by averaging the pixel covariances of the approximated polygon (block) areas.

By default, napp = 1 means that the approximation of the block-block variance-covariance matrix for each polygon neighbourhood configuration is based on one specific polygon (block) area approximation defined by one grid of pixels. If napp > 1 the approximation of the block-block variance-covariance matrix for one polygon neighbourhood configuration is based on the mean of napp approximated block-block variance-covariance matrices in order to reduce the approximation error. Each of the napp block-block variance-covariance approximations are based on a new, randomly shifted pixel gird which results each time in a new block area approximation. Large values of the argument napp increases the computation time.

There is a plot method plot.preCKrigePolygons for preCKrige output objects of the class "preCKrige.polygons" to visually control the polygon (block) area approximation by the pixels.

Value

preCKrige returns a S4 object, either of class "preCKrigePolygons" if the object for the argument newdata is of the class "SpatialPolygons" or "SpatialPolygonsDataFrame" or an S4 object of the class "preCKrigePoints" if the object for the argument newdata is of the class "SpatialPoints" or "SpatialPointsDataFrame" (see preCKrigePolygons or preCKrigePoints for the general structures of these two classes).

Notation:

n = total number of polygons or points
i = 1, . . . , n
m = 1 + number of (defined) neighbours of the ith polygon
rpix = number of pixel grid rows
cpix = number of pixel grid columns
npix = rpix * cpix

An object of the class "preCKrigePoints" contains the following slots:

covmat	list, length = n, of point-point covariance matrices.
posindex	list of numeric vectors, length = n, ith list component contains a vector with the row indices of the m-1 neighbours of the ith point.
model	object of the class "covmodel" with the parameter of the used covariance function.
data	is the data frame of the SpatialPointsDataFrame object. This data frame is used to build the target points design matrix for spatial interpolation by the CKrige function. data is empty, dim(data) = (0,0), if newdata is an object of the class "SpatialPoints".

coords matrix, dim = (n, 2) with the coordinates of the points.

An object of the class "preCKrigePolygons" contains the following slots (values are accessible by the @ operator):

covmat list of matrices, length = n, ith list component contains the approximated block-block covariance matrix of the ith polygon and its neighbourhood configuration.

se.covmat list of matrices, length = n, ith list component contains a matrix with the standard errors of the approximated block-block covariances of the ith polygon and its neighbourhood configuration, values are NaN if preCKrige argument napp = 1. See details.

pixconfig list of lists, length = n, ith list component contains at list with the information about the pixel used for the covariance approximation of the ith polygon and its neighbours. The list components of pixconfig are described below.

pixcovmat matrix, dim = (npix, npix), covariance matrix of the pixels.

model object of the class "covmodel" with the parameter of the used covariance function.

data is the data frame of the SpatialPolygonsDataFrame object. This data frame is used to build the target polygons (blocks) design matrix for spatial interpolation by the CKrige function. data is empty, dim(data) = (0, 0), if newdata is an object of the class "SpatialPolygons".

polygons SpatialPolygons object. A list (length = n) with the polygons of the newdata object.

The slot pixconfig is a list, length(pixconfig) = n, of lists, each of length = 10, with information about the pixel grid used to approxiamte the polygon (block) areas. The ith list of pixconfig contains the following 10 components:

pixcenter is a matrix, dim = (m,2) with the coordinates of the pixels centroids of the pixel grid for the ith polygon neighbourhood configuration. The dimension of the matrix is 2napp

rowwidth preCKrige input argument pheight.

colwidth preCKrige input argument pwidth

nrows positive numeric scalar, number of rows of the pixel grid.

ncols positive numeric scalar, number of columns of the pixel grid.

no.pix.in.poly is a numeric vector, length = m , each number indicates by how many pixels a polygon of the ith polygon configuratioin is approximated.

sa.polygons logical vector, length = m, TRUE means that the ith polygon is treated as point and FALSE means that the polygon is approximated by pixels. See section Note for more details.

polygon.centroids
is a matrix, dim = (m,2) with the coordinates of the polygon centroids from the ith polygon neighbourhood configuration.

`posindex` is a numeric vector, length = m, that contains the list indices of the ith polygon and its neighbours defined in the ith list component of the `neighbours` argument of the `preCKrige` function.

`pix.in.poly` is a binary matrix, dim = (npix, m). `pix.in.poly[k,l] = 1` indicate that the centroid of the kth pixel lies in the area of the lth polygon and `pix.in.poly[k,l] = 0` indicate that the kth pixel centroid does not lie in the area of the lth polygon.

Note

A polygon (block) is treated as point if the polygon area is smaller than the (defined) pixel area or if all pixel centroid of the generated pixel grid lie outside the polygon (block) area. If a pixel centroid lies inside a polygon that has a smaller area than a pixel, the pixel is allocated to the polygon (block) by which it share the largest area.

The point-point covariances are calculated via the `CovarianceFct` of the **RandomFields** package and the point-block covariances are calculated in C.

Author(s)

Christoph Hofer, <christoph.hofer@alumni.ethz.ch>

References

See main help page of the **constrainedKriging** package.

See Also

`CKrige`

Examples

```
## Not run:
### first example
### load data

data(meuse,meuse.blocks)

### plot blocks
plot(meuse.blocks)

### compute the approximated block variance of each block in  meuse.blocks
### without the definition of neighbours blocks (default) for an exponential
### covariance function without a measurement error, a nugget  = 0.15 (micro
### scale white noise process) and a scale parameter = 192.5
preCK_1  <- preCKrige(newdata = meuse.blocks, model = covmodel(modelname =
              "exponential", mev = 0, nugget = 0.05, variance = 0.15,
      scale = 192.5), pwidth = 75, pheight = 75)

### plot block approximation of block 59
plot(preCK_1, 59)
```

```
### second example
### define neighbours by using the poly2nb function
### of the spdep package
if(require(spdep))
{
neighbours <- poly2nb(meuse.blocks)
class(neighbours)
### neighbours should be an object of the class "list"
class(neighbours) <- "list"
### compute the approximated block variance-covariance
### matrices of each block in meuse.blocks without the
### defined block neighbours
preCK_2 <- preCKrige(newdata = meuse.blocks, neighbours = neighbours,
    model = covmodel("exponential", 0.05, 0.15, scale = 192.5),
    pwidth = 75, pheight = 75)

### plot block approximation of block 59 and its
### block neighbours
plot(preCK_2, 59)
}
if(!require(spdep))
{
cat("Please, install the package spdep to execute this example.\n")
}

## End(Not run)
```

`preCKrigePoints-class` *Class "preCKrigePoints"*

Description

Class of objects that are generated by `preCKrige` if the attribute `newdata` is of the class `SpatialPoints` or `SpatialPointsDataFrame`. This class has a `show` and `summary` method.

Objects from the Class

Objects can be created by calls of the form `new("preCKrigePoints", ...)`.

Slots

`covmat`: Object of class `"list"`

`posindex`: Object of class `"list"`

`model`: Object of class `"list"`

`data`: Object of class `"data.frame"`

`coords`: Object of class `"matrix"`

Methods

CKrige `signature(formula = "formula", data = "data.frame", locations = "formula", object = "preCKrige`
...

Author(s)

Christoph Hofer <christoph.hofer@alumni.ethz.ch>

See Also

preCKrige, preCKrigePolygons

Examples

```
showClass("preCKrigePoints")
```

---

preCKrigePolygons-class

Class "preCKrigePolygons" ~~~

Description

Class of objects that are generated by preCKrige if the attribute newdata is of the class SpatialPolygons or SpatialPolygonsDataFrame. This class has a show and summary method.

Objects from the Class

Objects can be created by calls of the form new("preCKrigePolygons", ...).

Slots

covmat: Object of class "list"

se.covmat: Object of class "list"

pixconfig: Object of class "list"

pixcovmat: Object of class "matrix"

model: Object of class "list" ~~

data: Object of class "data.frame"

polygons: Object of class "list"

Methods

CKrige `signature(formula = "formula", data = "data.frame", locations = "formula", object = "preCKrige`
...

Author(s)

Christoph Hofer

References

–

See Also

preCKrige, preCKrigePoints

Examples

```
showClass("preCKrigePolygons")
```

Index