Package 'GPFDA'

September 11, 2023

Type Package

Title Gaussian Process for Functional Data Analysis

Version 3.1.3

Date 2023-09-10

Author Jian Qing Shi, Yafeng Cheng, Evandro Konzen

Maintainer Evandro Konzen <gpfda.r@gmail.com>

Description Functionalities for modelling functional data with multidimensional inputs, multivariate functional data, and non-separable and/or non-stationary covariance structure of function-valued processes. In addition, there are functionalities for functional regression models where the mean function depends on scalar and/or functional covariates and the covariance structure depends on functional covariates. The development version of the package can be found on <https://github.com/gpfda/GPFDA-dev>.

License GPL-3

Depends R (>= 3.6)

Imports Rcpp, splines, mgcv, fields, interp, stats, graphics, grDevices, fda, fda.usc

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.2.3

Encoding UTF-8

Suggests MASS, mvtnorm, knitr, rmarkdown

VignetteBuilder knitr

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-09-10 22:22:39 UTC

R topics documented:

`calcScaleDistMats` *Calculate matrices for NSGP covariance function*

Description

Calculates matrices 'ScaleMat' and 'DistMat', which are used to obtain NSGP covariance matrices

Usage

```
calcScaleDistMats(A_List, coords)
```

Arguments

`A_List`	List of anisotropy matrices
`coords`	Matrix of input coordinates (covariates)

Value

A list of ScaleMat and DistMat matrices

Examples

```
## See examples in vignette:
# vignette("nsgpr", package = "GPFDA")
```

---

covMat *Calculate a covariance matrix*

Description

Evaluates one of the following covariance functions at input vectors t and t':

- Powered exponential
- Rational quadratic
- Matern
- Linear

Usage

```
cov.pow.ex(hyper, input, inputNew = NULL, gamma = 2)

cov.rat.qu(hyper, input, inputNew = NULL)

cov.matern(hyper, input, inputNew = NULL, nu)

cov.linear(hyper, input, inputNew = NULL)
```

Arguments

hyper	The hyperparameters. It must be a list with certain names. See details.
input	The covariate t. It must be either a matrix, where each column represents a covariate, or a vector if there is only one covariate.
inputNew	The covariate t'. It also must be a vector or a matrix. If NULL (default), 'inputNew' will be set to be equal to 'input' and the function will return a squared, symmetric covariance matrix.
gamma	Power parameter used in powered exponential kernel function. It must be 0<gamma<=2. Default to 2, which gives the squared exponential covariance function.
nu	Smoothness parameter of the Matern class. It must be a positive value.

Details

The names for the hyperparameters should be:

- "pow.ex.v" and "pow.ex.w" (powered exponential);
- "rat.qu.v", "rat.qu.w" and "rat.qu.a" (rational quadratic);
- "matern.v" and "matern.w" (Matern);
- "linear.i" and "linear.a" (linear);
- "vv" (Gaussian white noise).

Value

A covariance matrix

References

Shi, J. Q., and Choi, T. (2011), "Gaussian Process Regression Analysis for Functional input", CRC Press.

`D2` *Second derivative of the likelihood*

Description

Calculate the second derivative of the likelihood function with respect to one of the hyperparameters, given the first and second derivative of the kernel with respect to that hyperparameter.

Usage

```
D2(d1, d2, inv.Q, Alpha.Q)
```

Arguments

`d1`	First derivative of the kernel function with respect to the required hyperparameter.
`d2`	Second derivative of the kernel function with respect to the required hyperparameter.
`inv.Q`	Inverse of covariance matrix Q.
`Alpha.Q`	This is alpha * alpha'- invQ, where invQ is the inverse of the covariance matrix Q, and alpha = invQ * Y, where Y is the response.

Details

The function calculates the second derivative of the log-likelihood, using the first and second derivative of the kernel functions.

Value

A number.

References

Shi, J. Q., and Choi, T. (2011), "Gaussian Process Regression Analysis for Functional Data", CRC Press.

Examples

```
## This function is used in the vignette 'co2':
# vignette("co2", package = "GPFDA")
```

dataExampleGPFR *Data simulated in the GPFR example*

Description

A list containing training and test data simulated from a functional regression model.

In the training set, there are M=20 independent realisations and the functional response and the functional covariate are observed on a grid of n=20 time points.

The test set includes a single realisation observed on a grid of n_new=60 time points.

Both training and test sets also have a scalar covariate.

Usage

```
dataExampleGPFR
```

Format

A list with seven elements:

tt A vector of length 50

response_train A (20 x 50) matrix

x_train A (20 x 50) matrix

scalar_train A (20 x 2) matrix

t_new A vector of length 60

response_new A vector of length 60

x_new A vector of length 60

scalar_new A (1 x 2) matrix

Details

Data used in the GPFR example, see vignette("gpfr").

dataExampleMGPR *Data simulated in the MGPR example*

Description

A list containing data simulated from a MGPR model.

The dataset contains 30 realisations from a trivariate process. Each of the three functions is observed on 250 time points on [0,1].

Usage

```
dataExampleMGPR
```

Format

A list with two elements:

input List of 3 numeric vectors, each one being the time points where the corresponding function is observed.

response List of 3 matrices containing the observed 250 datapoints. Each column is an independent realisation.

Details

Data used in the MGPR example, see vignette("mgpr").

distanceMatrix *Calculate generalised distances*

Description

Calculate the generalised distance between vectors t and t' using an anisotropy matrix A.

- `distMat` and `distMatSq` calculate:

$$[(t-t')^{p/2}]^T A (t-t')^{p/2}$$

- `distMatLinear` and `distMatLinearSq` calculate:

$$t^T A t'$$

Usage

```
distMat(input, inputNew, A, power)

distMatSq(input, A, power)

distMatLinear(input, inputNew, A)

distMatLinearSq(input, A)
```

Arguments

input	Vector of the input coordinate t
inputNew	Vector of the input coordinate t'
A	Anisotropy matrix A
power	Power value p

Details

The distMatSq and distMatLinearSq functions are used when input vectors t and t' are identical, returning a symmetric matrix.

When distMat and distMatSq functions are used in powered exponential kernels, power=1 gives the exponential kernel and power=2 gives the squared exponential one.

distMatLinear and distMatLinearSq functions are used in the linear covariance kernel.

Value

A matrix

---

gpfr *Gaussian process functional regression (GPFR) model*

Description

Use functional regression (FR) model for the mean structure and Gaussian Process (GP) for the covariance structure.

Let 'n' be the number of time points 't' of functional objects and 'nrep' the number of independent replications in the sample.

Usage

```
gpfr(
  response,
  time = NULL,
  uReg = NULL,
  fxReg = NULL,
  fyList = NULL,
  uCoefList = NULL,
  fxList = NULL,
  concurrent = TRUE,
  fxCoefList = NULL,
  gpReg = NULL,
  hyper = NULL,
  NewHyper = NULL,
  Cov = "pow.ex",
  gamma = 2,
  nu = 1.5,
  useGradient = T,
  rel.tol = 1e-10,
  trace.iter = 5,
  fitting = FALSE
)
```

Arguments

response	Response data. It can be an 'fd' object or a matrix with 'nrep' rows and 'n' columns.
time	Input 't' of functional objects. It is a numeric vector of length 'n'.
uReg	Scalar covariates for the FR model. It should be a matrix with 'nrep' rows.
fxReg	Functional covariates for the FR model. It can be a matrix with 'nrep' rows and 'n' columns, an 'fd' object, or a list of matrices or 'fd' objects.
fyList	A list to control the smoothing of response.
uCoefList	A list to control the smoothing of the regression coefficient function of the scalar covariates in the FR model.
fxList	A list to control the smoothing of functional covariates in the FR model.
concurrent	Logical. If TRUE (default), concurrent functional regression will be carried out; otherwise, the full functional regression will be carried out.
fxCoefList	A list to control the smoothing of the regression coefficient function of functional covariates in the functional concurrent model.
gpReg	Covariates in the GP model. It should be a matrix, a numeric vector, an 'fd' object, a list of matrices or a list of 'fd' objects.
hyper	Vector of initial hyperparameters. Default to NULL.
NewHyper	Vector of names of new hyperparameters from the customized kernel function.
Cov	Covariance function(s) to use. Options are: 'linear', 'pow.ex', 'rat.qu', and 'matern'. Default to 'power.ex'.

gamma	Power parameter used in powered exponential kernel function. It must be 0<gamma<=2.
nu	Smoothness parameter of the Matern class. It must be a positive value.
useGradient	Logical. If TRUE, first derivatives will be used in the optimization.
rel.tol	Relative tolerance passed to nlminb(). Default to be 1e-10.
trace.iter	Print the processing of iterations of optimization.
fitting	Logical. If TRUE, fitting is carried out. Default to FALSE.

Details

fyList is a list with the following items:

- time: a sequence of time points; default to be 100 points from 0 to 1.
- nbasis: number of basis functions used in smoothing, default to be less than or equal to 23.
- norder: order of the functional curves; default to be 6.
- bSpline: logical. If TRUE (default), B-splines basis is used; otherwise, Fourier basis is used.
- Pen: default to be c(0,0), meaning that the penalty is only applied to the second order derivative of the curve, with no penalty for the zero-th and first order derivatives of the curve.
- lambda: smoothing parameter for the penalty, default to be 1e-4.

fxList is similar to fyList. However, it is a list of lists to allow for different specifications for each functional covariate if there are multiple ones.

uCoefList and fxCoefList are similar to each other. Each one is expected to be a list of lists. If a list of one element is provided, then the items of this element are applied to each of the functional coefficients of scalar covariates and of functional covariates, respectively.

- rtime: range of time, default to be c(0,1).
- nbasis: nnumber of basis functions used in smoothing, default to be less than or equal to 19.
- norder: order of the functional curves; default to be 6.
- bSpline: logical. If TRUE (default), B-splines basis is used; otherwise, Fourier basis is used.
- Pen: default to be c(0,0).
- lambda: smoothing parameter for the penalty, default to be 1e4.
- bivar:logical. Used for non-concurrent models; if TRUE, bivariate basis will be used; if FALSE (default), normal basis will be used; see details in bifdPar.
- lambdas: smoothing parameter for the penalty of the additional basis, default to be 1.

Note that all items have default settings.

Value

A list containing:

hyper Estimated hyperparameters

I A vector of estimated standard deviation of hyperparameters

modellist List of FR models fitted before Gaussian process

CovFun Covariance function used

gamma Parameter 'gamma' used in Gaussian process with powered exponential kernel

nu Parameter 'nu' used in Gaussian process with Matern kernel

init_resp Raw response data

resid_resp Residual after the fitted values from FR models have been taken out

fitted Fitted values

fitted.sd Standard deviation of the fitted values

ModelType The type of the model applied in the function.

lTrain Training scalar covariates for the FR model

fTrain Training functional covariates for the FR model

mfTrainfd List of 'fd' objects from training data for FR model with functional covariates

gpTrain Training data for Gaussian Process

time Input time 't'

iuuL Inverse of covariance matrix for uReg

iuuF Inverse of covariance matrix for fxReg

fittedFM Fitted values from the FR model

fyList fyList object used

References

- Ramsay, J., and Silverman, B. W. (2006), "Functional Data Analysis", 2nd ed., Springer, New York.
- Shi, J. Q., and Choi, T. (2011), "Gaussian Process Regression Analysis for Functional Data", CRC Press.

Examples

```
## See examples in vignette:
# vignette("gpfr", package = "GPFDA")
```

`gpfrPredict` *Prediction of GPFR model*

Description

Make predictions for test input data based on the GPFR model learnt by the 'gpfr' function. Both Type I and Type II predictions can be made.

Usage

```
gpfrPredict(
  train,
  testInputGP,
  testTime = NULL,
  uReg = NULL,
  fxReg = NULL,
  gpReg = NULL,
  GPpredict = TRUE
)
```

Arguments

train	An object of class 'gpfr' obtained by the the 'gpfr' function.
testInputGP	Test input data for the GP prediction. It must be a numeric vector, a matrix or an 'fd' object.
testTime	Test time points for prediction. If NULL, default settings will be applied.
uReg	Scalar covariates data of a new batch for the FR model.
fxReg	Functional covariates data of a new batch for the FR model.
gpReg	Input data for the GP part used for Type I prediction. It must be a list of three items. The names of the items must be 'response', 'input', and 'time'. The item 'response' is the observed response for a new batch; 'input' is the observed functional covariates for a new batch,;'time' is the observed time for the previous two. If NULL (default), Type II prediction is carried out.
GPpredict	Logical. If TRUE (default), GPFR prediction is carried out; otherwise only predictions based on the FR model is carried out.

Details

If 'gpReg' is provided, then Type I prediction is made. Otherwise, Type II prediction is made.

Value

A list containing:

ypred.mean The mean values of the prediction.

ypred.sd The standard deviation of the predictions.

predictionType Prediction type if GPFR prediction is carried out.

train All items trained by 'gpfr'.

References

- Ramsay, J., and Silverman, B. W. (2006), "Functional Data Analysis", 2nd ed., Springer, New York.
- Shi, J. Q., and Choi, T. (2011), "Gaussian Process Regression Analysis for Functional Data", CRC Press.

Examples

```
## See examples in vignette:
# vignette("gpfr", package = "GPFDA")
```

---

gpr *Gaussian process regression (GPR) model*

Description

Gaussian process regression for a single or multiple independent realisations.

Usage

```
gpr(
  response,
  input,
  Cov = "pow.ex",
  m = NULL,
  hyper = NULL,
  NewHyper = NULL,
  meanModel = 0,
  mu = NULL,
  gamma = 2,
  nu = 1.5,
  useGradient = T,
  iter.max = 100,
  rel.tol = 8e-10,
  trace = 0,
  nInitCandidates = 1000
)
```

Arguments

response	Response data. It should be a matrix, where each column is a realisation. It can be a vector if there is only one realisation.
input	Input covariates. It must be either a matrix, where each column represents a covariate, or a vector if there is only one covariate.
Cov	Covariance function(s) to use. Options are: 'linear', 'pow.ex', 'rat.qu', and 'matern'. Default to 'power.ex'.
m	If Subset of Data is to be used, m denotes the subset size and cannot be larger than the total sample size. Default to NULL.
hyper	The hyperparameters. Default to NULL. If not NULL, then it must be a list with appropriate names.
NewHyper	Vector of names of the new hyperparameters of the customized kernel function. These names must have the format: xxxxxx.x, i.e. '6 digit' followed by 'a dot' followed by '1 digit'. This is required for both 'hyper' and 'NewHyper'

`meanModel` Type of mean function. It can be

0 Zero mean function

1 Constant mean function to be estimated

't' Linear model for the mean function

'avg' The average across replications is used as the mean function. This is only used if there are more than two realisations observed at the same input coordinate values.

Default to 0. If argument 'mu' is specified, then 'meanModel' will be set to 'userDefined'.

`mu` Mean function specified by the user. It must be a vector. Its length must be the same as the sample size, that is, nrow(response).

`gamma` Power parameter used in powered exponential kernel function. It must be 0<gamma<=2.

`nu` Smoothness parameter of the Matern class. It must be a positive value.

`useGradient` Logical. If TRUE, first derivatives will be used in the optimization.

`iter.max` Maximum number of iterations allowed. Default to 100. If 'rel.tol' is reduced, then the number of iterations needed will be less.

`rel.tol` Relative convergence tolerance. Default to 8e-10. Smaller rel.tol means higher accuracy and more time to converge.

`trace` The value of the objective function and the parameters is printed every trace'th iteration. Defaults to 0 which indicates no trace information is to be printed.

`nInitCandidates`

Number of initial hyperparameter vectors. The optimization starts with the best.

Details

The most important function of the package. It fits the GPR model and stores everything necessary for prediction. The optimization used in the function is 'nlminb'. The names for the hyperparameters should be: "linear.a" for linear covariance function, "pow.ex.w", "pow.ex.v" for power exponential, "rat.qu.s", "rat.qu.a" for rational quadratic, "matern.w", "matern.v" for Matern, "vv" for variance of Gaussian white noise. All hyperparameters should be in one list.

Value

A list containing:

hyper Hyperparameters vector estimated from training data

var.hyper Variance of the estimated hyperparameters

fitted.mean Fitted values for the training data

fitted.sd Standard deviation of the fitted values for the training data

train.x Training covariates

train.y Training response

train.yOri Original training response

train.DataOri Original training covariates

idxSubset Index vector identifying which observations were selected if Subset of Data was used.

CovFun Covariance function type

gamma Parameter used in powered exponential covariance function

nu Parameter used in Matern covariance function

Q Covariance matrix

mean Mean function

meanModel Mean model used

meanLinearModel 'lm' object if mean is a linear regression. NULL otherwise.

conv An integer. 0 means converge; 1 otherwise.

hyper0 Starting point of the hyperparameters vector.

References

Shi, J. Q., and Choi, T. (2011), "Gaussian Process Regression Analysis for Functional Data", CRC Press.

Examples

```
## See examples in vignettes:

# vignette("gpr_ex1", package = "GPFDA")
# vignette("gpr_ex2", package = "GPFDA")
# vignette("co2", package = "GPFDA")
```

`gprPredict` *Prediction of GPR model*

Description

Prediction of GPR model

Usage

```
gprPredict(
  train = NULL,
  inputNew = NULL,
  noiseFreePred = F,
  hyper = NULL,
  input = NULL,
  Y = NULL,
  mSR = NULL,
  Cov = NULL,
  gamma = NULL,
  nu = NULL,
  meanModel = 0,
  mu = 0
)
```

Arguments

train	A 'gpr' object obtained from 'gpr' function. Default to NULL. If NULL, learning is done based on the other given arguments; otherwise, prediction is made based on the trained model of class gpr'.
inputNew	Test input covariates. It must be either a matrix, where each column represents a covariate, or a vector if there is only one covariate.
noiseFreePred	Logical. If TRUE, predictions will be noise-free.
hyper	The hyperparameters. Default to NULL. If not NULL, then it must be a list with appropriate names.
input	Input covariates. It must be either a matrix, where each column represents a covariate, or a vector if there is only one covariate.
Y	Training response. It should be a matrix, where each column is a realisation. It can be a vector if there is only one realisation.
mSR	Subset size m if Subset of Regressors method is used for prediction. It must be smaller than the total sample size.
Cov	Covariance function(s) to use. Options are: 'linear', 'pow.ex', 'rat.qu', and 'matern'. Default to 'power.ex'.
gamma	Power parameter used in powered exponential kernel function. It must be 0<gamma<=2.
nu	Smoothness parameter of the Matern class. It must be a positive value.
meanModel	Type of mean function. It can be **0** Zero mean function **1** Constant mean function to be estimated **'t'** Linear model for the mean function **'avg'** The average across replications is used as the mean function. This is only used if there are more than two realisations observed at the same input coordinate values. Default to 0. If argument 'mu' is specified, then 'meanModel' will be set to 'userDefined'.
mu	Mean function specified by the user. It must be a vector. Its length must be the same as the sample size, that is, nrow(response).

Value

A list containing

pred.mean Mean of predictions

pred.sd Standard deviation of predictions

newdata Test input data

noiseFreePred Logical. If TRUE, predictions are noise-free.

... Objects of 'gpr' class.

Examples

```
## See examples in vignettes:

# vignette("gpr_ex1", package = "GPFDA")
# vignette("gpr_ex2", package = "GPFDA")
# vignette("co2", package = "GPFDA")
```

`mat2fd` *Create an 'fd' object from a matrix*

Description

Easy setting up for creating an 'fd' object

Usage

```
mat2fd(mat, fdList = NULL)
```

Arguments

`mat`	Input data, should be a matrix with ncol time points and nrow replications or samples.
`fdList`	A list with following items:
	time Sequence of time points (default to be 100 points from 0 to 1).
	nbasis Number of basis functions used in smoothing, default to be less or equal to 23.
	norder Order of the functional curves default to be 6.
	bSpline Logical, if TRUE (default), b-Spline basis is used; otherwise, Fourier basis is used.
	Pen Default to be c(0,0), meaning that the penalty is on the second order derivative of the curve, since the weight for zero-th and first order derivatives of the curve are set to zero.
	lambda Smoothing parameter for the penalty. Default to be 1e-4.

Details

All items listed above have default values. If any item is required to change, add that item into the list; otherwise, leave it as NULL. For example, if one only wants to change the number of basis functions, do:

```
mat2fd(SomeMatrix,list(nbasis=21))
```

Value

An 'fd' object

References

Ramsay, J., and Silverman, B. W. (2006),

Examples

```
require(fda)
require(fda.usc)
nrep <- 20   # number of replications
n <- 100     # number of time points
input <- seq(-1, pi, length.out=n) # time points
ry <- rnorm(nrep, sd=10)
y <- matrix(NA, ncol=n, nrow=nrep)
for(i in 1:nrep)  y[i,] <- sin(2*input)*ry[i]

plot.fdata(fdata(y,input))

yfd <- mat2fd(y, list(lambda=0.01))
plot(yfd)

yfd <- mat2fd(y, list(lambda=0.00001))
plot(yfd)
```

---

mgpCovMat *Calculate a multivariate Gaussian processes covariance matrix given a vector of hyperparameters*

Description

Calculate a multivariate Gaussian processes covariance matrix given a vector of hyperparameters

Usage

```
mgpCovMat(Data, hp)
```

Arguments

Data	List of two elements: 'input' and 'response'. The element 'input' is a list of N vectors, where each vector represents the input covariate values for a particular output. The element 'response' is the corresponding list of N matrices (if there are multiple realisations) or vectors (for a single realisation) representing the response variables.
hp	Vector of hyperparameters

Value

Covariance matrix

References

Shi, J. Q., and Choi, T. (2011), "Gaussian Process Regression Analysis for Functional Data", CRC Press.

Examples

```
## See examples in vignette:
# vignette("mgpr", package = "GPFDA")
```

`mgpr` *Multivariate Gaussian process regression (MGPR) model*

Description

Multivariate Gaussian process regression where each of the N outputs is unidimensional. The multivariate output is allowed to have multiple independent realisations.

Usage

```
mgpr(Data, m = NULL, meanModel = 0, mu = NULL)
```

Arguments

`Data`	List of two elements: 'input' and 'response'. The element 'input' is a list of N vectors, where each vector represents the input covariate values for a particular output. The element 'response' is the corresponding list of N matrices (if there are multiple realisations) or vectors (for a single realisation) representing the response variables.
`m`	If Subset of Data is to be used in the estimation, m denotes the subset size. It cannot be larger than the total sample size. Default to NULL (Subsetting is not used).
`meanModel`	Type of mean function applied to all outputs. It can be **0** Zero mean function for each output. **1** Constant mean function to be estimated for each output. **'t'** Linear model for the mean function of each output. **'avg'** The average across replications is used as the mean function of each output. This can only be used if there are more than two realisations observed at the same input values. Default to 0. If argument 'mu' is specified, then 'meanModel' will be set to 'userDefined'.
`mu`	Vector of concatenated mean function values defined by the user. Default to NULL.

Value

A list containing:

fitted.mean Fitted values for the training data

fitted.sd Standard deviation of the fitted values for training data

N Number of response variables

X Original input variables

Y Original response

idx Index vector identifying to which output the elements of concatenated vectors correspond to.

Cov Covariance matrix

mean Concatenated mean function

meanModel Mean model used for each output

meanLinearModel 'lm' object for each output if the linear regression model is used for the mean functions. NULL otherwise.

References

Shi, J. Q., and Choi, T. (2011), "Gaussian Process Regression Analysis for Functional Data", CRC Press.

Examples

```
## See examples in vignette:
# vignette("mgpr", package = "GPFDA")
```

`mgprPredict` *Prediction of MGPR model*

Description

Prediction of MGPR model

Usage

```
mgprPredict(
  train,
  DataObs = NULL,
  DataNew,
  noiseFreePred = F,
  meanModel = NULL,
  mu = 0
)
```

Arguments

train	A 'mgpr' object obtained from 'mgpr' function. If NULL, predictions are made based on DataObs informed by the user.
DataObs	List of observed data. Default to NULL. If NULL, predictions are made based on the trained data (included in the object of class 'mgpr') used for learning.
DataNew	List of test input data.
noiseFreePred	Logical. If TRUE, predictions will be noise-free.
meanModel	Type of mean function applied to all outputs. It can be **0** Zero mean function for each output. **1** Constant mean function to be estimated for each output. **'t'** Linear model for the mean function of each output. **'avg'** The average across replications is used as the mean function of each output. This can only be used if there are more than two realisations observed at the same input values. Default to 0. If argument 'mu' is specified, then 'meanModel' will be set to 'userDefined'.
mu	Vector of concatenated mean function values defined by the user. Default to NULL.

Value

A list containing

pred.mean Mean of predictions for the test set.

pred.sd Standard deviation of predictions for the test set.

noiseFreePred Logical. If TRUE, predictions are noise-free.

Examples

```
## See examples in vignette:
# vignette("mgpr", package = "GPFDA")
```

---

nsgpCovMat *Calculate a NSGP covariance matrix given a vector of hyperparameters*

Description

Calculate a NSGP covariance matrix given a vector of hyperparameters

Usage

```
nsgpCovMat(
  hp,
  input,
  inputSubsetIdx = NULL,
  nBasis = 5,
  corrModel = corrModel,
  gamma = NULL,
  nu = NULL,
  cyclic = NULL,
  whichTau = NULL,
  calcCov = T
)
```

Arguments

hp	Vector of hyperparameters estimated by function nsgpr.
input	List of Q input variables (see Details).
inputSubsetIdx	A list identifying a subset of the input values to be used in the estimation (see Details).
nBasis	Number of B-spline basis functions in each coordinate direction along which parameters change.
corrModel	Correlation function specification used for g(.). It can be either "pow.ex" or "matern".
gamma	Power parameter used in powered exponential kernel function. It must be 0<gamma<=2.
nu	Smoothness parameter of the Matern class. It must be a positive value.
cyclic	Logical vector of dimension Q which defines which covariates are cyclic (periodic). For example, if basis functions should be cyclic only in the first coordinate direction, then cyclic=c(T,F). cyclic must have the same dimension of whichTau. If cyclic is TRUE for some coordinate direction, then cyclic B-spline functions will be used and the varying parameters (and their first two derivatives) will match at the boundaries of that coordinate direction.
whichTau	Logical vector of dimension Q identifying which input coordinates the parameters are function of. For example, if Q=2 and parameters change only with respect to the first coordinate, then we set whichTau=c(T,F).
calcCov	Logical. Calculate covariance matrix or not. If FALSE, time or spatially-varying parameters are still provided.

Value

A list containing

Cov Covariance matrix

vareps Noise variance

As_perTau List of varying anisotropy matrix over the input space

sig2_perTau Vector of signal variance over the input space

References

Konzen, E., Shi, J. Q. and Wang, Z. (2020) "Modeling Function-Valued Processes with Nonseparable and/or Nonstationary Covariance Structure" <arXiv:1903.09981>.

Examples

```
## See examples in vignette:
# vignette("nsgpr", package = "GPFDA")
```

`nsgpCovMatAsym` *Calculate an asymmetric NSGP covariance matrix*

Description

Calculate an asymmetric NSGP covariance matrix

Usage

```
nsgpCovMatAsym(
  hp,
  input,
  inputNew,
  nBasis = 5,
  corrModel = corrModel,
  gamma = NULL,
  nu = NULL,
  cyclic = NULL,
  whichTau = NULL
)
```

Arguments

`hp`	Vector of hyperparameters estimated by function nsgpr.
`input`	List of Q input variables (see Details).
`inputNew`	List of Q test set input variables.
`nBasis`	Number of B-spline basis functions in each coordinate direction along which parameters change.
`corrModel`	Correlation function specification used for g(.). It can be either "pow.ex" or "matern".
`gamma`	Power parameter used in powered exponential kernel function. It must be 0<gamma<=2.
`nu`	Smoothness parameter of the Matern class. It must be a positive value.
`cyclic`	Logical vector of dimension Q which defines which covariates are cyclic (periodic). For example, if basis functions should be cyclic only in the first coordinate direction, then cyclic=c(T,F). cyclic must have the same dimension of whichTau. If cyclic is TRUE for some coordinate direction, then cyclic B-spline functions will be used and the varying parameters (and their first two derivatives) will match at the boundaries of that coordinate direction.

whichTau — Logical vector of dimension Q identifying which input coordinates the parameters are function of. For example, if Q=2 and parameters change only with respect to the first coordinate, then we set whichTau=c(T,F).

Value

An asymmetric covariance matrix

References

Konzen, E., Shi, J. Q. and Wang, Z. (2020) "Modeling Function-Valued Processes with Nonseparable and/or Nonstationary Covariance Structure" <arXiv:1903.09981>.

---

nsgpr — *Estimation of a nonseparable and/or nonstationary covariance structure (NSGPR model)*

Description

Estimate the covariance structure of a zero-mean Gaussian Process with Q-dimensional input coordinates (covariates).

Multiple realisations for the response variable can be used, provided they are observed on the same grid of dimension n_1 x n_2 x ... x n_Q.

Let n = n_1 x n_2 x ... x n_Q and let nSamples be the number of realisations.

Usage

```
nsgpr(
  response,
  input,
  corrModel = "pow.ex",
  gamma = 2,
  nu = 1.5,
  whichTau = NULL,
  nBasis = 5,
  cyclic = NULL,
  unitSignalVariance = F,
  zeroNoiseVariance = F,
  sepCov = F,
  nInitCandidates = 300,
  absBounds = 6,
  inputSubsetIdx = NULL
)
```

Arguments

response	Response variable. This should be a (n x nSamples) matrix where each column is a realisation
input	List of Q input variables (see Details).
corrModel	Correlation function specification used for g(.). It can be either "pow.ex" or "matern".
gamma	Power parameter used in powered exponential kernel function. It must be 0<gamma<=2.
nu	Smoothness parameter of the Matern class. It must be a positive value.
whichTau	Logical vector of dimension Q identifying which input coordinates the parameters are function of. For example, if Q=2 and parameters change only with respect to the first coordinate, then we set whichTau=c(T,F).
nBasis	Number of B-spline basis functions in each coordinate direction along which parameters change.
cyclic	Logical vector of dimension Q which defines which covariates are cyclic (periodic). For example, if basis functions should be cyclic only in the first coordinate direction, then cyclic=c(T,F). cyclic must have the same dimension of whichTau. If cyclic is TRUE for some coordinate direction, then cyclic B-spline functions will be used and the varying parameters (and their first two derivatives) will match at the boundaries of that coordinate direction.
unitSignalVariance	Logical. TRUE if we assume realisations have variance 1. This is useful when we want to estimate an NSGP correlation function.
zeroNoiseVariance	Logical. TRUE if we assume the realisations are noise-free.
sepCov	Logical. TRUE only if we fix to zero all off-diagonal elements of the varying anisotropy matrix. Default to FALSE, allowing for a separable covariance function.
nInitCandidates	number of initial hyperparameter vectors which are used to evaluate the log-likelihood function at a first step. After evaluating the log-likelihood using these 'nInitCandidates' vectors, the optimisation via nlminb() begins with the best of these vectors.
absBounds	lower and upper boundaries for B-spline coefficients (if wanted).
inputSubsetIdx	A list identifying a subset of the input values to be used in the estimation (see Details).

Details

The input argument for Q=2 can be constructed as follows:

```
n1 <- 10
n2 <- 1000
input <- list()
input[[1]] <- seq(0,1,length.out = n1)
input[[2]] <- seq(0,1,length.out = n2)
```

If we want to use every third lattice point in the second input variable (using Subset of Data), then we can set

```
inputSubsetIdx <- list()
inputSubsetIdx[[1]] <- 1:n1
inputSubsetIdx[[2]] <- seq(1,n2, by=3)
```

Value

A list containing:

MLEsts Maximum likelihood estimates of B-spline coefficients and noise variance.

response Matrix of response.

inputMat Input coordinates in a matrix form

corrModel Correlation function specification used for g(.)

References

Konzen, E., Shi, J. Q. and Wang, Z. (2020) "Modeling Function-Valued Processes with Nonseparable and/or Nonstationary Covariance Structure" <arXiv:1903.09981>.

Examples

```
## See examples in vignette:
# vignette("nsgpr", package = "GPFDA")
```

`nsgprPredict` *Prediction of NSGPR model*

Description

Prediction of NSGPR model

Usage

```
nsgprPredict(
  hp,
  response,
  input,
  inputNew,
  noiseFreePred = F,
  nBasis = nBasis,
  corrModel = corrModel,
  gamma = gamma,
  nu = nu,
  cyclic = cyclic,
  whichTau = whichTau
)
```

Arguments

hp	Vector of hyperparameters estimated by function nsgpr.
response	Response variable. This should be a (n x nSamples) matrix where each column is a realisation
input	List of Q input variables (see Details).
inputNew	List of Q test set input variables.
noiseFreePred	Logical. If TRUE, predictions will be noise-free.
nBasis	Number of B-spline basis functions in each coordinate direction along which parameters change.
corrModel	Correlation function specification used for g(.). It can be either "pow.ex" or "matern".
gamma	Power parameter used in powered exponential kernel function. It must be 0<gamma<=2.
nu	Smoothness parameter of the Matern class. It must be a positive value.
cyclic	Logical vector of dimension Q which defines which covariates are cyclic (periodic). For example, if basis functions should be cyclic only in the first coordinate direction, then cyclic=c(T,F). cyclic must have the same dimension of whichTau. If cyclic is TRUE for some coordinate direction, then cyclic B-spline functions will be used and the varying parameters (and their first two derivatives) will match at the boundaries of that coordinate direction.
whichTau	Logical vector of dimension Q identifying which input coordinates the parameters are function of. For example, if Q=2 and parameters change only with respect to the first coordinate, then we set whichTau=c(T,F).

Value

A list containing

pred.mean Mean of predictions for the test set.

pred.sd Standard deviation of predictions for the test set.

noiseFreePred Logical. If TRUE, predictions are noise-free.

References

Konzen, E., Shi, J. Q. and Wang, Z. (2020) "Modeling Function-Valued Processes with Nonseparable and/or Nonstationary Covariance Structure" <arXiv:1903.09981>.

Examples

```
## See examples in vignette:
# vignette("nsgpr", package = "GPFDA")
```

plot.gpfr *Plot GPFR model for either training or prediction*

Description

Plot GPFR model for either training or prediction

Usage

```
## S3 method for class 'gpfr'
plot(
  x,
  type = c("raw", "meanFunction", "fitted", "prediction"),
  ylab = "y",
  xlab = "t",
  ylim = NULL,
  realisations = NULL,
  alpha = 0.05,
  colourTrain = 2,
  colourNew = 4,
  mar = c(4.5, 5.1, 2.2, 0.8),
  oma = c(0, 0, 1, 0),
  cex.lab = 1.5,
  cex.axis = 1,
  cex.main = 1.5,
  ...
)
```

Arguments

x	Plot GPFR for training or prediction from a given object of 'gpfr' class.
type	Required type of plots. Options are: 'raw', 'meanFunction', 'fitted' and 'prediction'.
ylab	Title for the y axis.
xlab	Title for the x axis.
ylim	Graphical parameter. If NULL (default), it is chosen automatically.
realisations	Index vector identifying which training realisations should be plotted. If NULL (default), all training realisations are plotted. For predictions, 'realisations' should be '0' if no training realisation is to be plotted.
alpha	Significance level used for 'fitted' or 'prediction'. Default is 0.05.
colourTrain	Colour for training realisations when 'type' is set to 'prediction' and 'realisations' is positive.
colourNew	Colour for predictive mean for the new curve when 'type' is set to 'prediction'.
mar	Graphical parameter passed to par().

oma	Graphical parameter passed to par().
cex.lab	Graphical parameter passed to par().
cex.axis	Graphical parameter passed to par().
cex.main	Graphical parameter passed to par().
...	Other graphical parameters passed to plot().

Value

A plot.

Examples

```
## See examples in vignette:
# vignette("gpfr", package = "GPFDA")
```

---

plot.gpr *Plot GPR model for either training or prediction*

Description

Plot Gaussian process for a given an object of class 'gpr'.

Usage

```
## S3 method for class 'gpr'
plot(
  x,
  fitted = F,
  col.no = 1,
  ylim = NULL,
  realisation = NULL,
  main = NULL,
  cex.points = NULL,
  lwd.points = NULL,
  pch = NULL,
  lwd = NULL,
  ...
)
```

Arguments

x	The 'gpr' object from either training or predicting of the Gaussian Process.
fitted	Logical. Plot fitted values or not. Default to FALSE. If FALSE, plot the predictions.
col.no	Column number of the input matrix. If the input matrix has more than one columns, than one of them will be used in the plot. Default to be the first one.

ylim	Range value for y-axis.
realisation	Integer identifying which realisation should be plotted (if there are multiple).
main	Title for the plot
cex.points	Graphical parameter
lwd.points	Graphical parameter
pch	Graphical parameter
lwd	Graphical parameter
...	Graphical parameters passed to plot().

Value

A plot

Examples

```
## See examples in vignette:
# vignette("gpr_ex1", package = "GPFDA")
```

---

plot.mgpr *Plot predictions of GPR model*

Description

Plot predictons of each element of the multivariate Gaussian Process for a given an object of class 'mgpr'.

Usage

```
## S3 method for class 'mgpr'
plot(
  x,
  DataObs,
  DataNew,
  realisation,
  alpha = 0.05,
  ylim = NULL,
  mfrow = NULL,
  cex = 2,
  mar = c(4.5, 7.1, 0.2, 0.8),
  oma = c(0, 0, 0, 0),
  cex.lab = 2,
  cex.axis = 1.5,
  ...
)
```

Arguments

x	An object of class 'mgpr'.
DataObs	List of observed data.
DataNew	List of test data.
realisation	Index identifying which realisation should be plotted.
alpha	Significance level used for MGPR predictions. Default is 0.05.
ylim	Range of y-axis.
mfrow	Graphical parameter.
cex	Graphical parameter.
mar	Graphical parameter passed to par().
oma	Graphical parameter passed to par().
cex.lab	Graphical parameter passed to par().
cex.axis	Graphical parameter passed to par().
...	Graphical parameters passed to plot().

Value

A plot showing predictions of each element of the multivariate process.

Examples

```
## See examples in vignette:
# vignette("mgpr", package = "GPFDA")
```

---

plotImage *Draw an image plot for a given two-dimensional input*

Description

Draw an image plot for a given two-dimensional input

Usage

```
plotImage(
  response,
  input,
  realisation = 1,
  n1,
  n2,
  main = " ",
  zlim = NULL,
  cex.axis = 1,
  cex.lab = 2.5,
```

```
  legend.cex.axis = 1,
  font.main = 2,
  cex.main = 2,
  legend.width = 2,
  mar = c(2.1, 2.1, 3.1, 6.1),
  oma = c(0, 1, 0, 0),
  nGrid = 200,
  enlarge_zlim = NULL
)
```

Arguments

response	Data to be plotted (e.g. matrix of predictions)
input	Matrix of two columns representing the input coordinates.
realisation	Integer identifying which realisation should be plotted (if there are multiple).
n1	Number of datapoints in the first coordinate direction
n2	Number of datapoints in the second coordinate direction
main	Title for the plot
zlim	Range of z-axis
cex.axis	Graphical parameter
cex.lab	Graphical parameter
legend.cex.axis	Graphical parameter
font.main	Graphical parameter
cex.main	Graphical parameter
legend.width	Graphical parameter
mar	Graphical parameter
oma	Graphical parameter
nGrid	Dimension of output grid in each coordinate direction
enlarge_zlim	Additional quantity to increase the range of zlim

Value

A plot

Examples

```
## See examples in vignette:
# vignette("gpr_ex2", package = "GPFDA")
```

`plotmgpCovFun` *Plot auto- or cross-covariance function of a multivariate Gaussian process*

Description

Plot auto- or cross-covariance function of a multivariate Gaussian process

Usage

```
plotmgpCovFun(
  type = "Cov",
  output,
  outputp,
  Data,
  hp,
  idx,
  ylim = NULL,
  xlim = NULL,
  mar = c(4.5, 5.1, 2.2, 0.8),
  oma = c(0, 0, 0, 0),
  cex.lab = 1.5,
  cex.axis = 1,
  cex.main = 1.5
)
```

Arguments

`type`	Logical. It can be either 'Cov' (for covariance function) or 'Cor' (for corresponding correlation function).
`output`	Integer identifying one element of the multivariate process.
`outputp`	Integer identifying one element of the multivariate process. If 'output' and 'outputp' are the same, the auto-covariance function will be plotted. Otherwise, the cross-covariance function between 'output' and 'outputp' will be plotted.
`Data`	List of two elements: 'input' and 'response'. The element 'input' is a list of N vectors, where each vector represents the input covariate values for a particular output. The element 'response' is the corresponding list of N matrices (if there are multiple realisations) or vectors (for a single realisation) representing the response variables.
`hp`	Vector of hyperparameters
`idx`	Index vector identifying to which output the elements of concatenated vectors correspond to.
`ylim`	Graphical parameter
`xlim`	Graphical parameter
`mar`	Graphical parameter passed to par().

oma	Graphical parameter passed to par().
cex.lab	Graphical parameter passed to par().
cex.axis	Graphical parameter passed to par().
cex.main	Graphical parameter passed to par().

Value

A plot

Examples

```
## See examples in vignette:
# vignette("mgpr", package = "GPFDA")
```

---

unscaledCorr *Calculate an unscaled NSGP correlation matrix*

Description

Calculate an unscaled NSGP correlation matrix

Usage

```
unscaledCorr(Dist.mat, corrModel, gamma = NULL, nu = NULL)
```

Arguments

Dist.mat	Distance matrix
corrModel	Correlation function specification used for g(.). It can be either "pow.ex" or "matern".
gamma	Power parameter used in powered exponential kernel function. It must be 0<gamma<=2.
nu	Smoothness parameter of the Matern class. It must be a positive value.

Value

A matrix

References

Konzen, E., Shi, J. Q. and Wang, Z. (2020) "Modeling Function-Valued Processes with Nonseparable and/or Nonstationary Covariance Structure" <arXiv:1903.09981>.

Examples

```
## See examples in vignette:
# vignette("nsgpr", package = "GPFDA")
```

Index