Package 'GPCsign'

May 7, 2026

Title Gaussian Process Classification as Described in Bachoc et al. (2020)

Version 0.1.1

Description Parameter estimation and prediction of Gaussian Process Classifier models as described in Bachoc et al. (2020) <doi:10.1007/S10898-020-00920-0>. Important functions : gpcm(), predict.gpcm(), update.gpcm().

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

Imports DiceKriging, tmvtnorm, TruncatedNormal (>= 2.3), future.apply, future, methods, stats

Suggests DiceDesign, testthat

NeedsCompilation no

Author Morgane Menz [aut, cre],
Céline Helbert [aut],
Victor Picheny [aut],
François Bachoc [aut],
Naoual Serraji [ctb]

Maintainer Morgane Menz <morgane.menz@ifpen.fr>

Repository CRAN

Date/Publication 2025-02-27 14:10:02 UTC

Contents

gpcm *Fit and/or create a Gaussian Process Classification (GPC) model*

Description

gpcm is used to fit GPC models. When parameters are known, the function creates a model using given parameters. Otherwise, they are estimated by Maximum Likelihood. In both cases, the result is a gpcm object.

Usage

```
gpcm(f, Xf, covtype = "matern5_2", noise.var = 1e-6,
     coef.cov = NULL, coef.m = NULL, multistart = 1,
     seed = NULL, lower = NULL, upper = NULL, nsimu = 100,
     normalize = TRUE, X.mean = NULL, X.std = NULL, MeanTransform = NULL)
```

Arguments

f	a vector containing the binary observations (+/-1) corresponding to the class labels.
Xf	a matrix representing the design of experiments.
covtype	a character string specifying the covariance structure for the latent GP. Default is matern5_2.
noise.var	variance value standing for the homogeneous nugget effect. Default is 1e-6.
coef.cov	an optional vector containing the values for covariance parameters for the latent GP. (See below).
coef.m	an optional scalar corresponding to the mean value of the latent GP. If both coef.cov and coef.m are provided, no covariance parameter estimation is performed. If at least one of them is missing, both are estimated.
multistart	an optional integer indicating the number of initial points from which running the BFGS for covariance parameter optimization. The multiple optimizations will be performed in parallel provided that a parallel backend is registered (see package future)
seed	to fix the seed, default is NULL.
lower	(see below).
upper	lower, upper: bounds for the covariance parameters (scalars or vectors), if NULL they are set to 0.2 and 3, respectively.
nsimu	the number of samples of the latent process at observation points Xf to generate. Must be a non-null integer.
normalize	a logical parameter indicating whether to normalize the input matrix Xf. If TRUE, the matrix will be normalized using X.mean and X.std values if given; otherwise, the mean and standard deviation are computed and used for normalization.
X.mean	(see below).

X.std	optional vectors containing mean and standard deviation values for each column of the input matrix. If they are not provided, they are computed from the input matrix Xf.
MeanTransform	optional character string specifying a transform of the latent process mean coef.m. If positive (resp. negative), coef.m is constrained to be positive (resp. negative) by an exponential transform.

Details

The generation of the matrix of samples of the latent process Z_obs is done using Gibbs sampling. See rtmvnorm function in tmvtnorm package.

Value

An object of class gpcm. See gpcm-class.

Author(s)

Morgane MENZ, Céline HELBERT, Victor PICHENY, François BACHOC. Contributors: Naoual SERRAJI.

References

Bachoc, F., Helbert, C. & Picheny, V. Gaussian process optimization with failures: classification and convergence proof. *J Glob Optim* **78**, 483–506 (2020). doi:10.1007/s10898020009200.

Kotecha, J. H., Djuric, P. M. (1999). Gibbs Sampling Approach For Generation of Truncated Multivariate Gaussian Random Variables. *IEEE Computer Society*, 1757–1760.

Wilhelm, S. tmvtnorm: Truncated Multivariate Normal and Student t Distribution. R package version 1.6. https://CRAN.R-project.org/package=tmvtnorm.

Roustant, O., Ginsbourger, D. & Deville, Y. Contributors: Chevalier, C. , Richet, Y. DiceKriging: Kriging Methods for Computer Experiments. R package version 1.6.0. https://CRAN.R-project.org/package=DiceKriging.

Byrd, R. H., Lu, P., Nocedal, J. and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, **16**, 1190–1208. doi:10.1137/0916069.

Examples

```
# ----------------------------------
# A 1D example - sinusoidal function
# ----------------------------------
sinusoidal_function <- function(x) {
  sin(4 * pi * x)
}

# Design of Experiments Xf and the corresponding signs f
Xf <- as.matrix(c(0.07, 0.19, 0.42, 0.56, 0.81, 0.90))
f <- rep(1, length(Xf))
f[(sinusoidal_function(Xf) < 0)] <- -1
```

```
# GPC model
GPCmodel <- gpcm(f, Xf, multistart = 3)

# Graphics of predictions
x <- as.matrix(seq(0, 1, length.out = 101))
result <- predict(object = GPCmodel, newdata = x)
probabilities <- result$prob
index <- match(Xf, x)
plot(x, probabilities, pch = "-")
points(Xf[f == 1], probabilities[index[f == 1]], pch = 20, col = "blue")
points(Xf[f == -1], probabilities[index[f == -1]], pch = 20, col = "red")
abline(h = 0.5, lty = 2)
legend("topright",title = "DoE Xf",title.cex = 0.7, legend = c("+", "-"),
     col = c("blue", "red"), pch = 20)


# ----------------------------------
# A 2D example - Branin-Hoo function
# ----------------------------------

# 30-points DoE, and the corresponding response
d <- 2
nb_PX <- 30
require(DiceDesign)
X <- lhsDesign(nb_PX, d, seed = 123)$design
Xopt <- maximinSA_LHS(X, T0 = 10, c = 0.99, it = 10000)
x <- Xopt$design
require(DiceKriging)
fx <- apply(x, 1, branin)
f <- ifelse(fx < 14, -1, 1)
Xf <- as.matrix(x)

# Fit and create a GPC model without parallelisation
t0 <- proc.time()
GPCmodel <- gpcm(f, Xf, multistart = 3, seed = 123)
t1 = proc.time() - t0
cat(" time elapsed : ",t1[3])
print(GPCmodel)

# Graphics - Predict probabilities
ngrid <- 50
x.grid <- seq(0, 1., length.out = ngrid)
grid <- as.matrix(expand.grid(x.grid, x.grid))
probabilities <- predict(GPCmodel, newdata = grid, light.return = TRUE)
filled.contour(x.grid, x.grid, matrix(probabilities, ngrid, ngrid),
               color.palette = function(n) hcl.colors(n, "RdYlBu", rev = FALSE),
               main = "probabilities map",
               plot.axes = {
                 axis(1)
                 axis(2)
                points(Xf[f == 1, 1], Xf[f == 1, 2], col = "blue", pch = 21, bg = "blue")
                points(Xf[f == -1, 1], Xf[f == -1, 2], col = "red", pch = 21, bg = "red")
```

```
                }
)

# Fit and create a GPC model with parallelisation
## Use multisession futures
require(future)
plan(multisession)
t0 = proc.time()
GPCmodel2 <-  gpcm(f,Xf, multistart = 3, seed = 123 )
t1 = proc.time() - t0
cat(" time elapsed : ",t1[3])
print(GPCmodel2)
## Explicitly close multisession workers by switching plan
plan(sequential)
```

---

gpcm-class *Gaussian Process Classification (GPC) models class*

Description

S4 class for GPC models.

Slots

`d` Object of class `"integer"`. The spatial dimension.

`n` Object of class `"integer"`. The number of observations.

`X` Object of class `"matrix"`. The design of experiments.

`y` Object of class `"matrix"`. The vector of binary observations at design points (+/-1) corresponding to the class labels.

`X.std` Object of class `"numeric"`. The vector of standard deviation values of design points.

`X.mean` Object of class `"numeric"`. The vector of mean values of design points.

`call` Object of class `"language"`. User call reminder.

`coef.m` Object of class `"numeric"`. Mean coefficient of latent GP.

`coef.cov` Object of class `"numeric"`. Covariance coefficients of latent GP.

`covariance` Object of class `"covKernel"`. A DiceKriging object specifying the covariance structure.

`noise.flag` Object of class `"logical"`. Are the observations noisy?

`noise.var` Object of class `"numeric"`. Nugget effect.

`param.estim` Object of class `"logical"`. `TRUE` if at least one parameter is estimated, `FALSE` otherwise.

`lower` Object of class `"numeric"`. Lower bounds for covariance parameters estimation.

`upper` Object of class `"numeric"`. Upper bounds for covariance parameters estimation.

logLik Object of class "numeric". Value of the log-Likelihood at its optimum.

Z_obs Object of class "matrix". A nobs * nsimu matrix of samples of the latent process at design points.

l Object of class "numeric". Lower truncation points. Parameter to generate new Z_obs.

u Object of class "numeric". Upper truncation points. Parameter to generate new Z_obs.

K Object of class "matrix". Covariance matrix of design points. Parameter to generate new Z_obs

invK Object of class "matrix". The inverse of the matrix K whose Cholesky decomposition was given.

MeanTransform object of class "character". 'positive' if coef.m is constrained to be positive by an exponential transform, 'negative' if coef.m is constrained to be negative.

Objects from the Class

To create a gpcm object, use gpcm. See also this function for more details.

Author(s)

Morgane MENZ, Céline HELBERT, Victor PICHENY, François BACHOC. Contributors: Naoual SERRAJI.

See Also

gpcm for more details about slots and to create a gpcm object. {covStruct.create} in DiceKriging to construct a covariance structure.

---

logLikFunc *Compute logLikelihood*

Description

Computes and returns the log-likelihood value, the covariance matrix of latent process and covariance structure of a Gaussian Process Classification (GPC) model.

Usage

```
logLikFunc(par, f, Xf, covtype = "matern5_2", noise.var = 1e-6,
     seed = NULL, MeanTransform = NULL, return.all = FALSE)
```

Arguments

par	vector contains the coef.m and the log of coef.cov.
f	vector of binary observations (+/-1) corresponding to the class labels.
Xf	a matrix representing the design of experiments.
covtype	a character string specifying the covariance structure for the latent GP. Default is matern5_2.

noise.var	nugget effect. Default is 1e-6.
seed	to fix the seed, default is NULL.
MeanTransform	optional character string specifying a transform of the latent process mean coef.m. If positive (resp. negative), coef.m is constrained to be positive (resp. negative) by an exponential transform.
return.all	an optional boolean. If FALSE, only the log-likelihood is returned; if TRUE, K and cov.fun are also returned. Default is FALSE.

Value

logLik	the log-likelihood.
K	the covariance matrix of latent process.
cov.fun	a DiceKriging object specifying the covariance structure.

Author(s)

Morgane MENZ, Céline HELBERT, Victor PICHENY, François BACHOC. Contributors: Naoual SERRAJI.

References

Bachoc, F., Helbert, C. & Picheny, V. Gaussian process optimization with failures: classification and convergence proof. *J Glob Optim* **78**, 483–506 (2020). doi:10.1007/s10898020009200

Botev, Z., Belzile, L. TruncatedNormal: Truncated Multivariate Normal and Student Distributions. R package version 2.2.2 https://cran.r-project.org/package=TruncatedNormal

Botev, Z. I. (2017), *The normal law under linear restrictions:simulation and estimation via minimax tilting*, Journal of the Royal Statistical Society, Series B, **79** (1), pp. 1-24.

Roustant, O., Ginsbourger, D. & Deville, Y. Contributors: Chevalier, C. , Richet, Y. DiceKriging: Kriging Methods for Computer Experiments. R package version 1.6.0. https://CRAN.R-project.org/package=DiceKriging.

Examples

```
# ------------
# A 1D example
# ------------

# Design of Experiments Xf and the corresponding signs f
Xf <- as.matrix(c(0.08, 0.27, 0.42, 0.65, 0.78, 0.84))
f <- c(1, -1, -1, 1, -1, -1)

# loglikelihood and covariance matrix at Xf
par <- c(coef.cov = 0.1, coef.m = 0)
result <- logLikFunc(par = par, f = f, Xf = Xf, return.all = TRUE)
K <- result$K
logLik <- result$logLik
print(logLik)
```

predict *Predict class probability at newdata for a Gaussian Process Classification (GPC) model*

Description

Predicted probability of class 1. Optionally, conditional covariance based on a gpcm model and 95% quantiles of the probability of class 1 are returned.

Usage

```
## S3 method for class 'gpcm'
predict(object, newdata, nsimu = NULL,
light.return = FALSE, checkNames=FALSE, seed = NULL, ...)
```

Arguments

object	an object of class gpcm.
newdata	a vector, matrix of points to be predicted.
nsimu	an optional integer indicating whether to resample latent GP at observation points and how many samples are required. If NULL, current samples are used. Default is NULL.
light.return	an optional boolean. If TRUE, only prob is returned. Default is FALSE.
checkNames	an optional boolean. If TRUE, a consistency test is performed between the names of newdata and the names of the experimental design (contained in object@Xf). Default is FALSE.
seed	to fix the seed (used if nsimu is not NULL). Default is NULL.
...	no other argument for this method

Value

prob	the (averaged) probability of class 1 at newdata.
lower95, upper95	95% confidence bounds for the probability at newdata.
probs	a matrix of sample predicted probabilities.
Zsimu_var, Zsimu_mean	conditional variance vector and mean matrix of the latent GP at newdata.
cov	conditional covariance matrix at newdata.
c	an auxiliary matrix, containing all the covariances between newdata and design points Xf.
lambda	an auxiliary vector, product of the inverse covariance matrix invK returned by object and the unconditional covariance matrix c between newdata and design points Xf.
kz	an auxiliary matrix, corresponding to the unconditional covariance matrix at newdata.

Author(s)

Morgane MENZ, Céline HELBERT, Victor PICHENY, François BACHOC. Contributors: Naoual SERRAJI.

References

Bachoc, F., Helbert, C. & Picheny, V. Gaussian process optimization with failures: classification and convergence proof. *J Glob Optim* **78**, 483–506 (2020). doi:10.1007/s10898020009200.

Roustant, O., Ginsbourger, D. & Deville, Y. Contributors: Chevalier, C. , Richet, Y. DiceKriging: Kriging Methods for Computer Experiments. R package version 1.6.0. https://CRAN.R-project.org/package=DiceKriging.

Examples

```
# ----------------------------------
# A 2D example - Branin-Hoo function
# ----------------------------------

# 30-points DoE, and the corresponding response
d <- 2
nb_PX <- 30
require(DiceDesign)
X <- lhsDesign(nb_PX, d, seed = 123)$design
Xopt <- maximinSA_LHS(X, T0 = 10, c = 0.99, it = 1000)
x <- Xopt$design
require(DiceKriging)
fx <- apply(x, 1, branin)
s <- ifelse(fx < 14, -1, 1)
f <- s
Xf <- as.matrix(x)

# Bulding GPC model
GPCmodel <- gpcm(f = f, Xf = Xf, coef.m = -0.1, coef.cov=c(0.8,0.5))

# Graphics - Predict probabilities
ngrid <- 50
x.grid <- seq(0, 1., length.out = ngrid)
grid <- as.matrix(expand.grid(x.grid, x.grid))
probabilities <- predict(object = GPCmodel, newdata = grid)$prob
filled.contour(x.grid, x.grid, matrix(probabilities, ngrid, ngrid),
               color.palette = function(n) hcl.colors(n, "RdYlBu", rev = FALSE),
               main = "probabilities map",
               plot.axes = {
                 axis(1)
                 axis(2)
                points(Xf[f == 1, 1], Xf[f == 1, 2], col = "blue", pch = 21, bg = "blue")
                points(Xf[f == -1, 1], Xf[f == -1, 2], col = "red", pch = 21, bg = "red")
               }
)
```

show *Print values of a Gaussian Process Classification (GPC) model*

Description

Show method for gpcm object. Printing the main features of a GPC model.

Usage

```
show.gpcm(object)
```

Arguments

object	an object of class gpcm. See gpcm.

Value

returns an invisible 'NULL'

Author(s)

Morgane MENZ, Céline HELBERT, Victor PICHENY, François BACHOC. Contributors: Naoual SERRAJI.

See Also

gpcm()

Examples

```
## 20-points DoE, and the corresponding response
d <- 2
nb_PX <- 20
require(DiceDesign)
x <- lhsDesign(nb_PX, d, seed = 123)$design
require(DiceKriging)
fx <- apply(x, 1, branin)
f <- ifelse(fx < 14, -1, 1)
Xf <- as.matrix(x)

## GPC model
model <- gpcm(f, Xf, coef.m=0, coef.cov=c(0.5,0.5))

## print the result
show(model)
```

update *Update of a Gaussian Process Classification (GPC) model*

Description

Update a gpcm object when one or many new observations are added.

Usage

```
## S3 method for class 'gpcm'
update(object, newf, newXf, newX.alreadyExist,
 newnoise.var, covandmean.reestim=TRUE, multistart = 1, seed = NULL,
  lower = NULL, upper = NULL, nsimu = 100, normalize = TRUE, ...)
```

Arguments

object	an object of gpcm class.
newf	a vector corresponding to the new binary observations (+/-1) at newXf locations. These locations can be new locations or existing ones.
newXf	a matrix with object@d columns representing the locations to be updated. These locations can be new locations or existing ones.
newX.alreadyExist	Boolean: indicate whether the locations newXf are all news or not. Default: TRUE, corresponding to existing locations in newX.
newnoise.var	optional scalar, nugget effect at new observations.
covandmean.reestim	should the mean and covariance parameters be re-estimated? Default is TRUE.
multistart	an optional integer indicating the number of initial points from which running the BFGS for covariance parameter optimization. Default is 1.
seed	to fix the seed, default is NULL.
lower	(see below).
upper	lower, upper: bounds for the covariance parameters (scalars or vectors), if NULL they are set to 0.2 and 3, respectively.
nsimu	an integer indicating the number of samples of the latent GP at observation points Xf to generate. Must be a non-null integer. Default is 100.
normalize	a logical parameter indicating whether to normalize the input matrix Xf. If TRUE, the matrix will be normalized using X.mean and X.std values if given; otherwise, the mean and standard deviation are calculated and used for normalization.
...	no other argument for this method

Value

Updated gpcm object.

Author(s)

Morgane MENZ, Céline HELBERT, Victor PICHENY, François BACHOC. Contributors: Naoual SERRAJI.

References

Bachoc, F., Helbert, C. & Picheny, V. Gaussian process optimization with failures: classification and convergence proof. *J Glob Optim* **78**, 483–506 (2020). doi:10.1007/s10898020009200.

Kotecha, J. H., Djuric, P. M. (1999). Gibbs Sampling Approach For Generation of Truncated Multivariate Gaussian Random Variables. *IEEE Computer Society*, 1757–1760.

Wilhelm, S. tmvtnorm: Truncated Multivariate Normal and Student t Distribution. R package version 1.6. https://CRAN.R-project.org/package=tmvtnorm.

Roustant, O., Ginsbourger, D. & Deville, Y. Contributors: Chevalier, C. , Richet, Y. DiceKriging: Kriging Methods for Computer Experiments. R package version 1.6.0. https://CRAN.R-project.org/package=DiceKriging.

Byrd, R. H., Lu, P., Nocedal, J. and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, **16**, 1190–1208. doi:10.1137/0916069.

See Also

gpcm

Examples

```
# ----------------------------------
# A 1D example - sinusoidal function
# ----------------------------------

# Test function
sinusoidal_function <- function(x) {
  sin(4 * pi * x)}

# Desing of Experiment Xf and the corresponding sign f
Xf <- as.matrix(c(0.07, 0.19, 0.42, 0.56, 0.81, 0.90))
f <- rep(1,length(Xf)); f[(sinusoidal_function(Xf)<0)]<- -1

# Builidng a GPC model
GPCmodel1 <- gpcm(f = f, Xf = Xf, coef.m=0, coef.cov=0.26)
print(GPCmodel1)

# New points added to the gpcm object.
newXf <- as.matrix(c(0.1,0.5,0.7, 0.95))
newf <- rep(1,length(newXf)); newf[(sinusoidal_function(newXf)<0)]<- -1

# Updating GPC model
NewGPCmodel <- update(object = GPCmodel1, newf = newf, newXf = newXf)
print(NewGPCmodel)
```

Index