Package: BayesGPfit (via r-universe)

November 5, 2024

Title Fast Bayesian Gaussian Process Regression Fitting

Version 1.1.0

Description Bayesian inferences on nonparametric regression via Gaussian Processes with a modified exponential square kernel using a basis expansion approach.

License GPL-2

Encoding UTF-8

LazyData true

Imports stats

Depends lattice

RoxygenNote 7.2.0

Repository https://kangjian2016.r-universe.dev

RemoteUrl https://github.com/kangjian2016/bayesgpfit

RemoteRef HEAD

RemoteSha 8cb69a5b8c5d09319a40920a3dd2eee836b59722

Contents

GP.Bayes.fit *Regular Bayesian fitting of Gaussian process regression on regular grid points with the modified exponential sqaured kernel.*

Description

Regular Bayesian fitting of Gaussian process regression on regular grid points with the modified exponential sqaured kernel.

Usage

```
GP.Bayes.fit(
  y,
  x,
  poly_degree = 60,
  a = 0.01,
  b = 20,
  num_results = 500L,
  iters_between_results = 2L,
  burn_in = 500L,
  a_sigma = 0.01,
  b_sigma = 0.01,
  a_zeta = 0.01,
  b_zeta = 0.01,
  center = NULL,
  scale = NULL,
  max_range = NULL,
  progress_bar = FALSE
)
```

Arguments

y	A vector of real numbers as the observations for the reponse variable.
x	A matrix of real numbers as grid points where rows are observations and columns are coordinates.
poly_degree	A integer number to specify the highest degree of Hermite polynomials. The default value is 10L.
a	A positive real number to specify the concentration parameter in the standard modified exponential squared kernel. The larger value the more the GP concentrates around the center. The default value is 0.01.
b	A positive real number to specify the smoothness parameter in the standard modified exponential squared kernel. The smaller value the smoother the GP is. The default value is 1.0.
num_results	An integer number to specify the number of posterior samples to save over MCMC iterations.

`iters_between_results`
: An integer number to specify the number of iterations to skip between two saved iterations.

`burn_in`
: An integer number to specify the burn-in number. The default value is 500L.

`a_sigma`
: A real number for the shape parameter in the Gamma prior of sigma2. The default value is 0.01.

`b_sigma`
: A real number for the rate parameter in the Gamma prior of sigma2. The default value is 0.01.

`a_zeta`
: A real number for the shape parameter in the Gamma prior of zeta. The default value is 0.01.

`b_zeta`
: A real number for the rate parameter in the Gamma prior of zeta. The default value is 0.01.

`center`
: A vector of real numbers specifying the centroid parameters in the modified exponential squared kernel. The default value is NULL and set to the center of the grid points: apply(x,2,mean).

`scale`
: A vector of positive numbers specifying the scale parameters in the modified exponential squared kernel. The default value is NULL and set to values such that grid points in a range of (-max_range,max_range) in each dimension.

`max_range`
: A positive real number indicating the maximum range of the grid points to specify the scale parameter. The default value is NULL and set to 6.

`progress_bar`
: A logical value to indicate whether a progress bar will be shown.

Value

A list of variables including the model fitting results

f A vector of real numbers for the posterior mean of the fitted curve.

x A matrix of real numbers for the grid points where rows are observations and columns are coordinates.

work_x A matrix of real numbers for the standardized grid points for the model fitting. It has the same dimension as "x".

sigma2 A real number for the posterior mean of the variance parameter of random errors.

tau2 A real number for the posterior mean of the variance parameter for the Gaussian process prior.

theta A vector of real numbers for the posterior mean of the basis coefficients for the Gaussian process.

Xmat A matrix real numbers for the basis functions evaluated at the standardized grid points (work_x), where rows are observations and columns are the basis functions

grid_size A real scalar for the grid size

center A vector of real numbers for the centroid parameters in the modified exponential squared kernel.

scale A vector of positive numbers for the scale parameters in the modified exponential squared kernel.

max_range A positive real number indicating the maximum range of the grid points to specify the scale parameter.

poly_degree An integer number to specify the highest degree of Hermite polynomials.

a A positive real number to specify the concentration parameter in the standard modified exponential squared kernel.

b A positive real number to specify the smoothness parameter in the standard modified exponential squared kernel.

mcmc_sample A matrix of real numbers for saved MCMC samples.

elapsed A real number indicating the computing time in second.

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
library(lattice)
set.seed(1227)
dat = list()
dat$x = GP.generate.grids(d=2,num_grids = 100)
curve = GP.simulate.curve.fast(dat$x,a=0.01,b=0.5,poly_degree=20L)
dat$f = curve$f + rnorm(length(curve$f),sd=1)
fast_fit = GP.fast.Bayes.fit(dat$f,dat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
reg_fit = GP.Bayes.fit(dat$f,dat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
mse = c(reg = mean((reg_fit$f - curve$f)^2),
       fast = mean((fast_fit$f - curve$f)^2))
print(mse)
plot(GP.plot.curve(curve,main="True curve"),split=c(1,2,2,2),more=TRUE)
plot(GP.plot.curve(fast_fit,main="Posterior mean estimates (fast)"),split=c(2,2,2,2),more=TRUE)
plot(GP.plot.curve(reg_fit,main="Posterior mean estimates (Regular)"),split=c(2,1,2,2))
```

GP.create.cols *Create 256 colors gradually transitioning from Blue to Yellow to Red.*

Description

Create 256 colors gradually transitioning from Blue to Yellow to Red.

Usage

```
GP.create.cols(num = 256L)
```

Arguments

num	A integer number to specify the number of colors to generate. The default value is 256.

Value

A vector of RGB colors

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
colors = GP.create.cols(101L)
require(graphics)
filled.contour(volcano,col=colors,nlevels=length(colors)-1,asp=1)
filled.contour(volcano,color.palette = GP.create.cols, nlevels=256, asp = 1)
```

GP.eigen.funcs.fast *Compute eigen functions*

Description

Compute eigen functions for the standard modified exponential squared correlation kernel.

Usage

```
GP.eigen.funcs.fast(grids, poly_degree = 10L, a = 0.01, b = 1)
```

Arguments

grids	A matrix where rows represent points and columns are coordinates.
poly_degree	A integer number specifies the highest degree of Hermite polynomials. The default value is 10L.
a	A positive real number specifies the concentration parameter in the modified exponetial squared kernel. The larger value the more the GP concentrates around the center. The default value is 0.01.
b	A positive real number specifies the smoothness parameter in the modeified exponetial squared kernel. The smaller value the smoother the GP is. The default value is 1.0.

Details

Compute eigen values of the standard modified exponential squared kernel on d-dimensional grids

$cor(X(s_1), X(s_2)) = \exp -a * (s_1^2 + *s_2^2) - b * (s_1 - s_2)^2$

where a is the concentration parameter and b is the smoothness parameter. The expected ranges of each coordinate is from -6 to 6.

Value

A matrix represents a set of eigen functions evaluated at grid points. The number of rows is equal to the number of grid points. The number of columns is choose(poly_degree+d,d), where d is the dimnension of the grid points.

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(lattice)
grids = GP.generate.grids(d=2L)
Psi_mat = GP.eigen.funcs.fast(grids)
fig = list()
for(i in 1:4){
   fig[[i]] = levelplot(Psi_mat[,i]~grids[,1]+grids[,2])
}
plot(fig[[1]],split=c(1,1,2,2),more=TRUE)
plot(fig[[2]],split=c(1,2,2,2),more=TRUE)
plot(fig[[3]],split=c(2,1,2,2),more=TRUE)
plot(fig[[4]],split=c(2,2,2,2))
```

`GP.eigen.funcs.fast.orth`

Create orthogonal eigen functions

Description

Create orthogonal eigen functions based on the standard modified exponential squared correlation kernel and Gram-Schimit Process

Usage

```
GP.eigen.funcs.fast.orth(grids, poly_degree = 10L, a = 0.01, b = 1)
```

Arguments

`grids`	A matrix where rows represent points and columns are coordinates.
`poly_degree`	A integer number specifies the highest degree of Hermite polynomials. The default value is 10L.
`a`	A positive real number specifies the concentration parameter in the modified exponetial squared kernel. The larger value the more the GP concentrates around the center. The default value is 0.01.
`b`	A positive real number specifies the smoothness parameter in the modeified exponetial squared kernel. The smaller value the smoother the GP is. The default value is 1.0.

Details

Compute eigen values of the standard modified exponential squared kernel on d-dimensional grids

$cor(X(s_1), X(s_2)) = \exp -a * (s_1^2 + *s_2^2) - b * (s_1 - s_2)^2$

where a is the concentration parameter and b is the smoothness parameter. The expected ranges of each coordinate is from -6 to 6.

Value

A matrix represents a set of eigen functions evaluated at grid points. The number of rows is equal to the number of grid points. The number of columns is choose(poly_degree+d,d), where d is the dimnension of the grid points.

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(lattice)
grids = GP.generate.grids(d=2L)
Psi_mat = GP.eigen.funcs.fast.orth(grids)
fig = list()
for(i in 1:4){
   fig[[i]] = levelplot(Psi_mat[,i]~grids[,1]+grids[,2])
}
plot(fig[[1]],split=c(1,1,2,2),more=TRUE)
plot(fig[[2]],split=c(1,2,2,2),more=TRUE)
plot(fig[[3]],split=c(2,1,2,2),more=TRUE)
plot(fig[[4]],split=c(2,2,2,2))
```

GP.eigen.value *Compute eigen values for the standard modified exponential squared correlation kernel.*

Description

Compute eigen values for the standard modified exponential squared correlation kernel.

Usage

```
GP.eigen.value(poly_degree = 10, a = 1, b = 1, d = 2)
```

Arguments

poly_degree	A positive integer number specifies the highest degree of Hermite polynomials. The default value is 10L.
a	A positive real number specifying the concentration parameter in the modified exponetial squared kernel. The larger value the more the GP concentrates around the center. The default value is 0.01.
b	A positive real number specifying the smoothness parameter in the modeified exponetial squared kernel. The smaller value the smoother the GP is. The default value is 1.0.
d	A positive integer number specifying the dimension of grid points.

Details

Compute eigen values of the standard modified exponential squared kernel on d-dimensional grids

$cor(X(s_1), X(s_2)) = \exp -a * (s_1^2 + *s_2^2) - b * (s_1 - s_2)^2$

where a is the concentration parameter and b is the smoothness parameter. The expected ranges of each coordinate is from -6 to 6.

Value

A matrix represents a set of eigen functions evaluated at grid points. The number of rows is equal to the number of grid points. The number of columns is choose(poly_degree+d,d), where d is the dimnension of the grid points.

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
Lambda = GP.eigen.value(poly_degree=10L,a=0.01,b=0.5,d=2)
plot(Lambda)
```

`GP.fast.Bayes.fit` *Fast Bayesian fitting of Gaussian process*

Description

Fast Bayesian fitting of Gaussian process regression on regular grid points with the modified exponential sqaured kernel.

Usage

```
GP.fast.Bayes.fit(
  y,
  x,
  poly_degree = 10L,
  a = 0.01,
  b = 1,
  center = NULL,
  scale = NULL,
  max_range = NULL,
  num_results = 500L,
  iters_between_results = 2L,
  burn_in = 500L,
  a_sigma = 0.01,
  b_sigma = 0.01,
```

```
  a_tau = 0.01,
  b_tau = 0.01,
  progress_bar = FALSE
)
```

Arguments

y	A vector of real numbers as the observations for the reponse variable.
x	A matrix of real numbers as grid points where rows are observations and columns are coordinates.
poly_degree	A integer number to specify the highest degree of Hermite polynomials. The default value is 10L.
a	A positive real number to specify the concentration parameter in the standard modified exponential squared kernel. The larger value the more the GP concentrates around the center. The default value is 0.01.
b	A positive real number to specify the smoothness parameter in the standard modified exponential squared kernel. The smaller value the smoother the GP is. The default value is 1.0.
center	A vector of real numbers specifying the centroid parameters in the modified exponential squared kernel. The default value is NULL and set to the center of the grid points: apply(x,2,mean).
scale	A vector of positive numbers specifying the scale parameters in the modified exponential squared kernel. The default value is NULL and set to values such that grid points in a range of (-max_range,max_range) in each dimension.
max_range	A positive real number indicating the maximum range of the grid points to specify the scale parameter. The default value is NULL and set to 6.
num_results	An integer number to specify the number of posterior samples to save over MCMC iterations.
iters_between_results	An integer number to specify the number of iterations to skip between two saved iterations.
burn_in	An integer number to specify the burn-in number. The default value is 500L.
a_sigma	A real number for the shape parameter in the Gamma prior of sigma2. The default value is 0.01.
b_sigma	A real number for the rate parameter in the Gamma prior of sigma2. The default value is 0.01.
a_tau	A real number for the shape parameter in the Gamma prior of tau2. The default value is 0.01.
b_tau	A real number for the rate parameter in the Gamma prior of tau2. The default value is 0.01.
progress_bar	A logical value to indicate whether a progress bar will be shown.

Value

A list of variables including the model fitting results

f A vector of real numbers for the posterior mean of the fitted curve.

x A matrix of real numbers for the grid points where rows are observations and columns are coordinates.

work_x A matrix of real numbers for the standardized grid points for the model fitting. It has the same dimension as "x".

sigma2 A real number for the posterior mean of the variance parameter of random errors.

tau2 A real number for the posterior mean of the variance parameter for the Gaussian process prior.

theta A vector of real numbers for the posterior mean of the basis coefficients for the Gaussian process.

Xmat A matrix real numbers for the basis functions evaluated at the standardized grid points (work_x), where rows are observations and columns are the basis functions

grid_size A real scalar for the grid size

center A vector of real numbers for the centroid parameters in the modified exponential squared kernel.

scale A vector of positive numbers for the scale parameters in the modified exponential squared kernel.

max_range A positive real number indicating the maximum range of the grid points to specify the scale parameter.

poly_degree An integer number to specify the highest degree of Hermite polynomials.

a A positive real number to specify the concentration parameter in the standard modified exponential squared kernel.

b A positive real number to specify the smoothness parameter in the standard modified exponential squared kernel.

mcmc_sample A matrix of real numbers for saved MCMC samples.

elapsed A real number indicating the computing time in second

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
library(lattice)
set.seed(1224)
dat = list()
dat$x = GP.generate.grids(d=2,num_grids = 100)
curve = GP.simulate.curve.fast(dat$x,a=0.01,b=0.5,poly_degree=20L)
dat$f = curve$f + rnorm(length(curve$f),sd=1)
fit = GP.fast.Bayes.fit(dat$f,dat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
plot(GP.plot.curve(dat,main="Data"),split=c(1,1,2,2),more=TRUE)
plot(GP.plot.curve(curve,main="True curve"),split=c(1,2,2,2),more=TRUE)
```

```
plot(GP.plot.curve(fit,main="Posterior mean estimates"),split=c(2,2,2,2),more=TRUE)
```

`GP.generate.grids` *Create spatial grids.*

Description

Create spatial grids.

Usage

```
GP.generate.grids(
  d = 1L,
  num_grids = 50L,
  grids_lim = c(-1, 1),
  random = FALSE
)
```

Arguments

`d`	An integer number for the dimension of the space. The default value is 1.
`num_grids`	An integer number for the number of grids in each dimension. The default value is 50.
`grids_lim`	A vector of two real numbers for the range of the grids in each dimension. The default value is c(-1,1).
`random`	A logical value indicating whether each dimension of the grids is generated from a uniform distribution or fixed as equally-spaced.

Value

A matrix with d columns and num_grids^d rows.

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
x = GP.generate.grids(d=2L)
require(lattice)
y = sin(abs(x[,1]+x[,2]))
levelplot(y~x[,1]+x[,2])
```

GP.plot.curve *Graphical representation of one, two, three-dimensional curves*

Description

Graphical representation of one, two, three-dimensional curves

Usage

```
GP.plot.curve(
  curve,
  xlab = NULL,
  ylab = NULL,
  zlab = NULL,
  xlim = NULL,
  ylim = NULL,
  zlim = NULL,
  col.regions = NULL,
  cut = NULL,
  num_slices = NULL,
  ...
)
```

Arguments

curve	A list object with two elements: **f** A vector of real numbers for the curve. **x** A matrix of real numbers for the grid points where rows are observations and columns are coordinates.
xlab	A character specifying the label of x-axis for 1D, 2D and 3D case. The default value is NULL and set to "x" for 1D case and "x1" for 2D and 3D cases.
ylab	A character specifying the label of y-axis for 1D curve or coords for 2D and 3D case. The default value is NULL and set to "x2" for 2D and 3D cases.
zlab	A character specifying the label of z-axis only for 3D case. The default value is NULL and set to "x3".
xlim	A vector of two real numbers specifying the range of x-axis for 1D, 2D and 3D case. The default value is NULL and set to range(curve$x[,1]).
ylim	A vector of two real numbers specifying the range of y-axis only for 2D, 3D case. The default value is NULL and set to range(curve$x[,2]).
zlim	A vector of two real numbers specifying the range of z-axis only for 3D case. The default value is NULL and set to range(curve$x[,3]).
col.regions	A vector of RGB colors for 2D and 3D plots. See GP.create.cols. The default value is NULL and set to GP.create.cols().

cut	An integer specifying the number of colors in 2D and 3D plots. The default value is NULL and set to length(col.regions)-1.
num_slices	An integer specifying the number of slices cutting through the 3rd dimension to show.
...	All other parameters for plot (1D case) and levelplot (2D and 3D cases).

Value

NULL for 1D case. An object of class "trellis" for 2D and 3D cases.

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
library(lattice)
set.seed(1224)
##plot 1D curve
x1d = GP.generate.grids(d=1,num_grids = 1000)
curve1d = GP.simulate.curve.fast(x1d,a=0.01,b=0.5,
                                 poly_degree=10L)
GP.plot.curve(curve1d,main="Simulated 1D Curve")

##plot 2D curve
x2d = GP.generate.grids(d=2L,num_grids = 100)
curve2d = GP.simulate.curve.fast(x2d,a=0.01,b=0.5,
                                 poly_degree=10L)
GP.plot.curve(curve2d,main="Simulated 2D Curve")

##plot 3D curve
x3d = GP.generate.grids(d=3,num_grids = 50)
curve3d = GP.simulate.curve.fast(x3d,a=0.01,b=0.5,
                                 poly_degree=10L)
GP.plot.curve(curve3d,main="Simulated 3D Curve",num_slices=10,zlim=c(-0.5,0.5))
```

GP.plot.curves *Graphical representation of multiple curves in one and two-dimensional curves*

Description

Graphical representation of multiple curves in one and two-dimensional curves

Usage

```
GP.plot.curves(
  curves,
  xlab = NULL,
  ylab = NULL,
  cols = NULL,
  lwd = NULL,
  type = NULL,
  leg_pos = NULL,
  xlim = NULL,
  ylim = NULL,
  col.regions = NULL,
  cut = NULL,
  nms = NULL,
  ...
)
```

Arguments

curves	A list object of multiple curves and each curve is a list with two elements: **f** A vector of real numbers for the curve. **x** A matrix of real numbers for the grid points where rows are observations and columns are coordinates. or a list object of two components **f** A matrix of real numbers for the multiple curves. **x** A matrix of real numbers for the grid points where rows are observations and columns are coordinates.
xlab	A character specifying the label of x-axis for 1D, 2D and 3D case. The default value is NULL and set to "x" for 1D case and "x1" for 2D case.
ylab	A character specifying the label of y-axis for 1D curve or coords for 2D and 3D case. The default value is NULL and set to "x2" for 2D case.
cols	A vector of integer numbers or characters to specify the plot colors for 1D curve. The default value is NULL and set to 1:length(curves).
lwd	A positive number to specify the width of lines for 1D curve.
type	A character specifying what type of plot should be drawn for 1D curve. Possible types are the same as plot.
leg_pos	A character specifying the position of legend for multiple 1D curves. Possible valeus are "topleft", "topright","bottemright","bottemleft".
xlim	A vector of two real numbers specifying the range of x-axis for 1D, 2D and 3D case. The default value is NULL and set to range(curve$x[,1]).
ylim	A vector of two real numbers specifying the range of y-axis only for 2D, 3D case. The default value is NULL and set to range(curve$x[,2]).
col.regions	A vector of RGB colors for 2D and 3D plots. See GP.create.cols. The default value is NULL and set to GP.create.cols().

cut	An integer specifying the number of colors in 2D plots. The default value is NULL and set to length(col.regions)-1.
nms	A vector of charterers for figure titles. Default is Null in which case it is set to the names of the list object curves
...	All other parameters for plot (1D case) and levelplot (2D case).

Value

NULL for 1D case. An object of class "trellis" for 2D and 3D cases.

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
library(lattice)
set.seed(1227)
dat = list()
dat$x = GP.generate.grids(d=2L,num_grids = 100)
curve = GP.simulate.curve.fast(dat$x,a=0.01,b=0.5,poly_degree=20L)
dat$f = curve$f + rnorm(length(curve$f),sd=1)
fast_fit = GP.fast.Bayes.fit(dat$f,dat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
reg_fit = GP.Bayes.fit(dat$f,dat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
curves = list(True = curve,
Bayes_fast = fast_fit,
Bayes = reg_fit)
GP.plot.curves(curves,
              main="Comparisons of Bayesian model fitting")
```

GP.predict *Gaussian process predictions*

Description

Gaussian process predictions

Usage

```
GP.predict(GP_fit, newx, CI = TRUE)
```

Arguments

GP_fit	An output object of function GP.Bayes.fit or GP.fast.Bayes.fit. Please refer to them for details.
newx	A matrix of real numbers as new grid points for preditions.
CI	A logical value indicating prediction.

Value

A list object When CI is FALSE, the object consists of three elements:

f Posterior predictive mean values of the curves.

x The grid points for prediction, i.e. "newx".

work_x The standardized grid points for prediction.

When CI is FALSE, the object consists of four elements:

mean A list object for posterior predictive mean of the curve,consisting of two elements (f is a vector for the curve values; x is a vector or matrix for points evaluated).

work_x A matrix of real numbers for the standardized grid points for the model fitting. It has the same dimension as "newx".

uci A list object for 95% upper bound of the predictive creditible interval (uci) of the curve, consisting of two elements (f is a vector for curve values; x is a vector or matrix for points evaluated).

lci A list object for 95% lower bound of the predictive creditibel interval (lci) of the curve, consisting of two elements (f is a vector for curve value; x is a vector or matrix for points evaluated).

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
set.seed(1224)
traindat = list()
traindat$x = GP.generate.grids(d=2,num_grids=30,random=TRUE)
testdat = list()
testdat$x = GP.generate.grids(d=2,num_grids=30,random=FALSE)
curve = GP.simulate.curve.fast(rbind(traindat$x,testdat$x),a=0.01,b=0.5,poly_degree=20L)
train_curve = list(f=curve$f[1:nrow(traindat$x)],x=traindat$x)
test_curve = list(f=curve$f[nrow(traindat$x)+1:nrow(testdat$x)],x=testdat$x)
traindat$f = train_curve$f + rnorm(length(train_curve$f),sd=1)
testdat$f = test_curve$f + rnorm(length(test_curve$f),sd=1)
fast_fit = GP.fast.Bayes.fit(traindat$f,traindat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
reg_fit = GP.Bayes.fit(traindat$f,traindat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
fast_pred = GP.predict(fast_fit,testdat$x,CI=TRUE)
reg_pred = GP.predict(reg_fit,testdat$x,CI=TRUE)
pmse = c(fast = mean((fast_pred$mean$f-test_curve$f)^2),
        reg = mean((reg_pred$mean$f-test_curve$f)^2))
print(pmse)
curves = list(true = test_curve,
             Bayes = reg_pred$mean,
             fast = fast_pred$mean)
GP.plot.curves(curves,main="Posterior predictive mean")
```

`GP.simulate.curve.fast` *Simulate curve on d-dimensional Euclidean space based on Gaussian processes via modified exponential squared kernel.*

Description

Simulate curve on d-dimensional Euclidean space based on Gaussian processes via modified exponential squared kernel.

Usage

```
GP.simulate.curve.fast(
  x,
  poly_degree,
  a,
  b,
  center = NULL,
  scale = NULL,
  max_range = 6
)
```

Arguments

`x`	A matrix of real numbers as grid points where rows are observations and columns are coordinates.
`poly_degree`	A integer number to specify the highest degree of Hermite polynomials. The default value is 10L.
`a`	A positive real number to specify the concentration parameter in the standard modified exponential squared kernel. The larger value the more the GP concentrates around the center. The default value is 0.01.
`b`	A positive real number to specify the smoothness parameter in the standard modified exponential squared kernel. The smaller value the smoother the GP is. The default value is 1.0.
`center`	A vector of real numbers specifying the centroid parameters in the modified exponential squared kernel. The default value is NULL and set to the center of the grid points: apply(x,2,mean).
`scale`	A vector of positive numbers specifying the scale parameters in the modified exponential squared kernel. The default value is NULL and set to values such that grid points in a range of (-max_range,max_range) in each dimension.
`max_range`	A positive real number indicating the maximum range of the grid points to specify the scale parameter. The default value is NULL and set to 6.

Value

A list of variables representing the simulated curve:

f A vector of real numbers for the simulated curve.

x A matrix of real numbers for the grid points where rows are observations and columns are coordinates.

work_x A matrix of real numbers for the standardized grid points for the simulated curve. It has the same dimension as "x".

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
library(lattice)
set.seed(1224)
dat = list()
dat$x = GP.generate.grids(d=2,num_grids = 100)
curve = GP.simulate.curve.fast(dat$x,a=0.01,b=0.5,poly_degree=20L)
GP.plot.curve(curve,main="Simulated Curve")
```

`GP.simulate.curves.fast`

Simulate multiple curves on d-dimensional Euclidean space based on Gaussian processes via modified exponential squared kernel.

Description

Simulate multiple curves on d-dimensional Euclidean space based on Gaussian processes via modified exponential squared kernel.

Usage

```
GP.simulate.curves.fast(
  n,
  x,
  poly_degree,
  a,
  b,
  center = NULL,
  scale = NULL,
  max_range = 6
)
```

Arguments

n	An integer number to specify the number of curves to simulate
x	A matrix of real numbers as grid points where rows are observations and columns are coordinates.
poly_degree	An integer number to specify the highest degree of Hermite polynomials. The default value is 10L.
a	A positive real number to specify the concentration parameter in the standard modified exponential squared kernel. The larger value the more the GP concentrates around the center. The default value is 0.01.
b	A positive real number to specify the smoothness parameter in the standard modified exponential squared kernel. The smaller value the smoother the GP is. The default value is 1.0.
center	A vector of real numbers specifying the centroid parameters in the modified exponential squared kernel. The default value is NULL and set to the center of the grid points: apply(x,2,mean).
scale	A vector of positive numbers specifying the scale parameters in the modified exponential squared kernel. The default value is NULL and set to values such that grid points in a range of (-max_range,max_range) in each dimension.
max_range	A positive real number indicating the maximum range of the grid points to specify the scale parameter. The default value is NULL and set to 6.

Value

A list of variables representing the simulated curve:

f A matrix of real numbers for the multiple simulated curves.

x A matrix of real numbers for the grid points where rows are observations and columns are coordinates.

work_x A matrix of real numbers for the standardized grid points for the simulated curve. It has the same dimension as "x".

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
library(lattice)
set.seed(1224)
dat = list()
dat$x = GP.generate.grids(d=2,num_grids = 100)
curves = GP.simulate.curves.fast(n = 10, dat$x,a=0.01,b=0.5,poly_degree=20L)
GP.plot.curves(curves,main="Simulated Curves")
```

GP.std.grids *Compute the standardized grids*

Description

Compute the standardized grids

Usage

```
GP.std.grids(
  grids,
  center = apply(grids, 2, mean),
  scale = NULL,
  max_range = 6
)
```

Arguments

grids	A matrix where rows represent grid points and columns are coordinates.
center	A vector of real numbers specifying the centroid parameters in the modified exponential squared kernel. The default value is NULL and set to the center of the grid points: apply(x,2,mean).
scale	A vector of positive numbers specifying the scale parameters in the modified exponential squared kernel. The default value is NULL and set to values such that grid points in a range of (-max_range,max_range) in each dimension.
max_range	A positive real number indicating the maximum range of the grid points to specify the scale parameter. The default value is NULL and set to 6.

Value

A matrix where rows represent the standardized grids.

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
grids = GP.generate.grids(d=2L)
std_grids = GP.std.grids(grids)
plot(grids[,1],std_grids[,1],asp=1,type="l")
abline(a=0,b=1,lty=2)
```

GP.summary *Summary of posterior inference on the Bayesian Gaussian process regression model*

Description

Summary of posterior inference on the Bayesian Gaussian process regression model

Usage

```
GP.summary(GP_fit)
```

Arguments

GP_fit	An output object of function GP.Bayes.fit or GP.fast.Bayes.fit. Please refer to them for details.

Value

A list object consisting of the following elements:

mean A list object for posterior mean of the target function,consisting of two elements (f is a vector for function values; x is a vector or matrix for points evaluated).

work_x A matrix of real numbers for the standardized grid points for the model fitting. It has the same dimension as "x".

uci A list object for 95% upper bound of the creditible interval (uci) of the taget function, consisting of two elements (f is a vector for function values; x is a vector or matrix for points evaluated).

lci A list object for 95% lower bound of the creditibel interval (lci) of the taget function, consisting of two elements (f is a vector for function values; x is a vector or matrix for points evaluated).

sigma2 A vector of posteror mean, the 95% lcl and ucl for variance of the random error.

tau2 A vector of posterior mean, the 95% lcl and ucl for variance of the target function (hyperparameters).

Author(s)

Jian Kang <jiankang@umich.edu>

Examples

```
library(BayesGPfit)
library(lattice)
set.seed(1224)
dat = list()
dat$x = GP.generate.grids(d=2,num_grids = 30)
curve = GP.simulate.curve.fast(dat$x,a=0.01,b=0.5,poly_degree=20L)
dat$f = curve$f + rnorm(length(curve$f),sd=1)
fast_fit = GP.fast.Bayes.fit(dat$f,dat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
```

```
reg_fit = GP.Bayes.fit(dat$f,dat$x,a=0.01,b=0.5,poly_degree=20L,progress_bar = TRUE)
sum_fast_fit = GP.summary(fast_fit)
sum_reg_fit = GP.summary(reg_fit)
curves = list(mean_fast = sum_fast_fit$mean,
              mean = sum_reg_fit$mean,
              lci_fast = sum_fast_fit$lci,
              lci = sum_reg_fit$lci,
              uci_fast = sum_fast_fit$uci,
              uci = sum_reg_fit$uci)
GP.plot.curves(curves,layout=c(2,3))
```

Index