Package: spectralGP (via r-universe)

June 6, 2026

Version 1.3.4

Date 2026-02-05

Maintainer Christopher Paciorek <paciorek@stat.berkeley.edu>

Title Approximate Gaussian Processes Using the Fourier Basis

Description Routines for creating, manipulating, and performing Bayesian inference about Gaussian processes in one and two dimensions using the Fourier basis approximation: simulation and plotting of processes, calculation of coefficient variances, calculation of process density, coefficient proposals (for use in MCMC). It uses R environments to store GP objects as references/pointers.

Depends R (>= 1.9.0)

Imports graphics, stats, grDevices

LazyLoad yes

License GPL (>= 2)

URL https://doi.org/10.18637/jss.v019.i02

NeedsCompilation no

Author Christopher Paciorek [aut, cre]

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

Date/Publication 2026-02-05 22:51:31 UTC

RemoteUrl https://github.com/cran/spectralGP

RemoteRef HEAD

RemoteSha 05371b51b27ff4e66a87dbe6fa1a28f98d87a832

Contents

`add.blocks.gp` *Adds coefficient block structure to a spectral GP object*

Description

Adds block structure to a GP object, allowing simulating and sampling (in an MCMC setup) from blocks of coefficients, as opposed to all the coefficients at once. The size of the blocks will usually increase with increasing frequency as the most important coefficients are those for the low frequency basis functions.

Usage

```
## S3 method for class 'gp'
add.blocks(object, breaks = NULL,...)
```

Arguments

object	A GP object, created by gp.
breaks	An optional vector of increasing frequency values that allow for binning the coefficients based on the frequencies of the basis functions with which they are associated. The maximum value of the vector should be one half the number of gridpoints in the dimension with the largest number of gridpoints (e.g., 64, if gridsize=c(32,128)).
...	Other arguments.

Details

The function sets up a block structure with blocks with increasing numbers of coefficients as the frequencies increase. The frequency in question is the highest frequency of the $(omega_1, omega_2)$ pair in the two-dimensional situation. E.g., the default block structure is c(1,2,4,8,16,...), which means that the first block is the coefficients whose maximum frequency is 1, the second with maximum frequency is 2, the third with maximum frequency of 3 and 4, the fourth with maximum frequency between 5 and 8, etc.

Value

The function modifies the GP object, which is essentially a pointer (an R environment in this case), so NULL is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, propose.coeff.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
add.blocks(gp1,c(1,3,7,15,31,64))
add.blocks(gp2)
propose.coeff(gp1,block=5)
plot(gp1)
propose.coeff(gp2,block=2,proposal.sd=0.1)
plot(gp2)
```

`calc.variances.gp` *Calculates prior variances of coefficients in a spectral GP object*

Description

Calculates the prior variances of the spectral coefficients in a GP object. The variances are based on the spectral density function chosen in gp and the correlation function parameters supplied.

Usage

```
## S3 method for class 'gp'
calc.variances(object,...)
```

Arguments

`object`	A GP object, created by gp.
`...`	Other arguments.

Details

This function is an internal function not meant to be called by the user. The prior variances for each coefficient are calculated based on the frequency of the corresponding basis function, the spectral density function, the parameters of the spectral density/correlation function, and the (optional) coefficient variance parameter. The function creates `variances`, a matrix of variances corresponding to `coeff`, the matrix of coefficients.

Value

The function modifies the GP object, which is essentially a pointer (an R environment in this case), so NULL is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

`gp`, `propose.coeff.gp`, `simulate.gp`, `logdensity.gp`, `change.param.gp`

change.param.gp *Changes correlation function parameter values of a spectral GP object*

Description

Changes the correlation parameter values or the (optional) variance parameter and recalculates the prior variances of the coefficients using `calc.variances.gp`.

Usage

```
## S3 method for class 'gp'
change.param(object,new.specdens.param=NULL,
  new.variance.param=NULL,...)
```

Arguments

`object`	A GP object, created by gp.
`new.specdens.param`	A vector of new parameter values, matching the length of the original parameter vector.
`new.variance.param`	The new variance parameter value.
`...`	Other arguments.

Details

This function allows the user to change the parameter values of the spectral GP object and recalculate the prior variances for the coefficients. This is particularly useful for implementing MCMC with the spectral GP.

Value

The function modifies the GP object, which is essentially a pointer (an R environment in this case), so NULL is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, calc.variances.gp, logdensity.gp

Examples

```
library(spectralGP)
rho=1
gp1=gp(128,matern.specdens,c(rho,4))
gp2=gp(c(64,64),matern.specdens,c(rho,4),0.5)
propose.coeff(gp1)
propose.coeff(gp2)
print(logdensity(gp1))
print(logdensity(gp2))
rho=2
sigma=2.5
change.param(gp1,c(rho,4)) # change parameter values of correlation function
change.param(gp2,c(rho,4),sigma)
print(logdensity(gp1))
print(logdensity(gp2))
```

`copy.gp` *Copy a spectral GP object.*

Description

Creates a new copy of a spectral GP object, with new memory allocated for the object, or copies the elements of one spectral GP object to another one that is already in existence.

Usage

```
## S3 method for class 'gp'
copy(object, object2 = NULL,...)
```

Arguments

`object`	Spectral GP object to be copied.
`object2`	Already existing spectral GP object to which the elements of `object` should be copied. If NULL, the function returns a newly-created copy of `object`.
`...`	Other arguments.

Details

This function copies an object of class gp. More details on the spectral representation of GPs can be found in Paciorek (2006).

Value

An object of class gp. If `object2` is specified, returns NULL.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, is.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(128,matern.specdens,c(0.5,4))
copy(gp1,gp2)  # gp2 is now a copy of gp1, with first parameter equal to 1
gp3=copy(gp1)
```

---

expand.gpgrid.gp *Calculate grid locations for a spectral GP object.*

Description

This is a version of expand.grid that calculates the grid locations for a spectral GP object. Grid-points representing the part of the domain in which the periodicity of the GP emerges are omitted.

Usage

```
## S3 method for class 'gp'
expand.gpgrid(object,...)
```

Arguments

object	A GP object, created by gp.
...	Other arguments.

Details

Note that this function is not named expand.grid.gp because expand.grid is a function, and not an S3 method.

Value

A matrix of grid locations with the first column the x-dimension and the second the y-dimension, or for one dimensional processes, a vector of grid locations.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, getgrid.gp, predict.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
grid1=expand.gpgrid(gp1)
grid2=expand.gpgrid(gp2)
plot(grid2)
```

---

getgrid.gp *Calculates the gridpoints in a spectral GP object*

Description

Calculates the sequence of gridpoints in each dimension for a spectral GP object. Gridpoints representing the part of the domain in which the periodicity of the GP emerges are omitted.

Usage

```
## S3 method for class 'gp'
getgrid(object,...)
```

Arguments

object	A GP object, created by gp.
...	Other arguments.

Details

Not meant to be used directly by the user, unless the user needs the unique gridpoints in each dimension. For the expanded grid that corresponds to the process values, with each row containing the two coordinates of a grid location, use expand.gpgrid.

Value

For two dimensions, a list containing the gridpoints in each dimension, with the first element containing the unique gridpoints in the first dimension and the second element the unique gridpoints in the second dimension, or for one-dimensional processes, a vector of gridpoints.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, getgrid.gp, predict.gp, expand.gpgrid.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
grid1=getgrid(gp1)
grid2=getgrid(gp2)
```

Gibbs.sample.coeff.gp *Samples new coeffients via Gibbs sampling in a spectral GP object.*

Description

Samples new coefficients via Gibbs sampling in a spectral GP object following the Gibbs sampling scheme of Wikle (2002), which involves an extra variance component (sig2e and a noisy version of the process (z).

Usage

```
## S3 method for class 'gp'
Gibbs.sample.coeff(object, z, sig2e, meanVal=0,
sdVal=1,returnHastings=FALSE, ...)
```

Arguments

object	A GP object, created by gp.
z	Vector of values for z, the noisy version of the process.
sig2e	Noise variance component that distorts z as a version of the process.
meanVal	Optional mean value for z.
sdVal	Optional standard deviation value for z.
returnHastings	Optional argument telling whether to return the logdensity of the proposal for use in a Metropolis-Hastings correction calculation.
...	Other arguments.

Details

This function can be used in an MCMC context to take Gibbs samples of the process coefficients, as part of the algorithm of Wikle (2002). The function modifies the GP object, updating the `coeff` and `process` components.

Value

The function modifies the GP object, which is essentially a pointer (an R environment in this case), so NULL is returned, unless `returnHastings=TRUE`.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, propose.coeff.gp, updateprocess.gp

---

gp *Create a new GP object*

Description

Creates a Gaussian process (GP) object based on the spectral basis approximation of a GP on a grid. The advantage of this approach is that GPs can be simulated and sampled much more efficiently than standard GP representations. E.g., GPs can be simulated on fine grids of 256X256 locations, many more locations than can usually be predicted with standard computational approaches. Currently one and two dimensional GPs are supported.

Usage

```
gp(gridsize = c(64, 64), specdens = matern.specdens,
  specdens.param = c(1, 4),variance.param=1,const.fixed=FALSE)
```

Arguments

`gridsize`	Vector (or scalar for one dimension) of number of gridpoints in each direction. Number of gridpoints should be a power of two, and it is recommended that the number be the same for each dimension.
`specdens`	Function (as a function or text string of the function name) that calculates spectral density of correlation function desired; function should take a vector (scalar) of parameter values. See matern.specdens() for an example.
`specdens.param`	Vector of parameters to be supplied to the specdens.function function.

`variance.param` Variance parameter used to scale the variances of all the coefficients. Note that this can also be done outside of the GP framework by scaling the predictions as in Wikle (2002).

`const.fixed` Logical indicating whether the coefficient of the constant basis function is fixed at zero. Since this coefficient does not have sufficient flexibility under the prior in most situations, it is advisable to fix this coefficient and have a separate mean value/parameter outside of the gp object. However, in simulating realizations, one should not fix this parameter, so as to ensure the correct approximate covariance structure induced by the spectral density and parameter values chosen.

Details

This function produces an object of class gp. More details on the spectral representation of GPs can be found in Paciorek (2006); see below.

Value

An object of class gp. This includes the dimension of the space, the spectral density information, a matrix of coefficients, the Fourier frequencies, and prior variances.

`gridsize` Vector (or scalar for one dimension) of number of gridpoints in each direction.

`d` Dimension of the space (1 or 2).

`specdens` Spectral density function of the correlation function of the GP.

`coeff` Matrix of coefficient values (a one-column matrix for one-dimensional processes).

`omega` A matrix of Fourier frequency values corresponding the basis functions in expand.grid() format.

`variances` A matrix of coefficient variances.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

`simulate.gp`, `plot.gp`, `propose.coeff.gp`, `calc.variances.gp`, `new.mapping`, `logdensity.gp`, `predict.gp`, `add.blocks.gp`

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
simulate(gp1)
simulate(gp2)
```

```
plot(gp1)
plot(gp2)

n=100
locs=cbind(runif(n,0.2,1.2),runif(n,-0.2,1.4))
locs.predict=cbind(runif(n,-0.4,0.8),runif(n,-0.1,1.7))
scaled.locs=xy2unit(locs,rbind(locs,locs.predict))
scaled.locs.predict=xy2unit(locs.predict,rbind(locs,locs.predict))
train.map=new.mapping(gp2,scaled.locs)
predict.map=new.mapping(gp2,scaled.locs.predict)
vals.train=predict(gp2,mapping=train.map)
vals.predict=predict(gp2,mapping=predict.map)
```

`Hastings.coeff.gp` *Calculates Hastings value of coefficients*

Description

Calculates Hastings value of coefficients, the logdensity of the current coefficients given proposal mean and variance based on a Gibbs sample of the form in `Gibbs.sample.coeff.gp`.

Usage

```
## S3 method for class 'gp'
Hastings.coeff(object, z, sig2e, meanVal=0, sdVal=1, ...)
```

Arguments

`object`	A GP object, created by gp.
`z`	Vector of values for z, the noisy version of the process.
`sig2e`	Noise variance component that distorts z as a version of the process.
`meanVal`	Optional mean value for z.
`sdVal`	Optional standard deviation value for z.
`...`	Other arguments.

Details

This function can be used in an MCMC context to calculate the Hastings correction that may be necessary in taking a quasi-Gibbs sample of the process coefficients, as part of one of the algorithms of Paciorek (2006). The function calculates and returns the logdensity.

Value

The function returns the logdensity.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, propose.coeff.gp, updateprocess.gp

---

image_plot *Draws image plot with a legend strip for the color scale.*

Description

This function combines the R image function with some automatic placement of a legend. This is done by splitting the plotting region into two parts. Putting the image in one and the legend in the other.

This function and help file are copied from the fields package.

Usage

```
image_plot(..., add = FALSE, nlevel = 64, legend.shrink = 0.9,
    legend.width = 0.05, graphics.reset = FALSE, horizontal = FALSE,
    offset = 2 * legend.width, bigplot = NULL, smallplot = NULL,
    legend.only = FALSE, col = topo.colors(nlevel))
```

Arguments

...	The usual arguments to the image function. because this function may change the size of the plotting region. (See details below)
add	If true add image and a legend strip to the existing plot.
nlevel	Number of color levels used in legend strip
legend.shrink	Amount to shrink the size of legend relative to the full height or width of the plot.
legend.width	Width in plotting coordinates (the full size plot is [0,1]X[0,1]) of the legend strip.
offset	Amount that the legend strip is set in from the left edge or the bottom of the plotting region. Units are with respect to the plotting coordinates.
graphics.reset	If false (default) the plotting region (plt in par) will not be reset and one can add more information onto the image plot. (e.g. using functions such as points or lines.) If true will reset plot parameters to the values before entering the function.

horizontal	If false (default) legend will be a vertical strip on the right side. If true the legend strip will be along the bottom.
bigplot	Plot coordinates for image plot. If not passed these will be determined within the function.
smallplot	Plot coordinates for legend. If not passed these will be determined within the function.
legend.only	If true just add the legend to a the plot in the plot region defined by the coordinates in smallplot.
col	Color table to use for image (see help file on image for details). Default is a pleasing range of 64 divisions on a topgraphic scale.

Details

It is surprising how hard it is just to automatically add the legend! All "plotting coordinates" mentioned here are in device coordinates. The plot region is assumed to be [0,1]X[0,1] and plotting regions are defined as rectangles within this square. We found these easier to work with than user coordinates. There are always problems with default solutions to placing information on graphs but the choices made here may be useful for most cases. The most annoying thing is that after using plot.image and adding information the next plot that is made may have the slightly smaller plotting region set by the image plotting.

The strategy is simple, divide the plotting region into two smaller regions. The image goes in one and the legend in the other. This way there is always room for the legend. Some adjustments are made to this rule by not shrinking the image plot if there is already room for the legend strip and also sticking the legend strip close to the image plot. Also, one can specify the plot regions explicitly by bigplot and small plot if the default choices do not work. There may be problems with small plotting regions in fitting both of these plot and one may have to change the default character sizes or margins to make things fit.

By keeping the zlim argument the same across images one can generate the same color scale. (See `image` help file) One useful technique for a panel of images is to just draw the first with image.plot to get a legend and just use image for subsequent plots. Also keep in mind one can just add a legend to an existing plot without changing plotting parameters. Usually a square plot (pty="s") done in a rectangular plot region will have room for the legend with any adjustments stuck to the right side.

Side Effects

After exiting, the plotting region may be changed to make it possible to add more features to the plot. To be explicit, `par()$plt` may be changed to reflect a smaller plotting region that includes a legend subplot.

See Also

`image`

Examples

```
x<- 1:10
y<- 1:15
z<- outer( x,y,"+")
```

```
image_plot(x,y,z)
# now add some points on diagonal
points( 5:10, 5:10)
#
#fat (5% of figure) and short (50% of figure)  legend strip on the bottom
image_plot( x,y,z,legend.width=.05, legend.shrink=.5, horizontal=TRUE)

# add a legend on the bottom but first change margin for some room
par( mar=c(10,5,5,5))

image( x,y,z)
image_plot( zlim=c(0,25), legend.only=TRUE, horizontal=TRUE)
```

`is.gp` *Test if object is a spectral GP*

Description

Tests if the argument is a spectral GP object.

Usage

```
is.gp(object)
```

Arguments

`object`	A GP object, created by gp.

Value

Returns 'TRUE' if the argument is a gp, and 'FALSE' otherwise.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
is.gp(gp1)
```

`lines.gp` *Add a line plot for a one-dimensional process based on a spectral GP object*

Description

Adds a line plot to an existing plot.

Usage

```
## S3 method for class 'gp'
lines(x, ...)
```

Arguments

`x`	A GP object, created by gp.
`...`	Extra arguments to plotting functions.

Value

No value is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, simulate.gp, getgrid.gp, predict.gp

Examples

```
library(spectralGP)
gp1=gp(c(128),matern.specdens,c(1,4))
simulate(gp1)
plot(gp1)
simulate(gp1)
lines(gp1,col=2)
```

`logdensity.gp` *Calculates log prior density of a spectral GP object*

Description

Calculates the log prior density of a spectral GP object as the log prior density of the basis coefficients, based on the prior variances and a prior of independent Gaussians.

Usage

```
## S3 method for class 'gp'
logdensity(object,...)
```

Arguments

`object`	A GP object, created by gp.
`...`	Other arguments.

Details

The log density is calculated based on the real and imaginary components of the basis function coefficients, but only those coefficients that are not determined as the complex conjugates of other coefficients. The density function is that the coefficients are IID normal with mean zero and prior variance based on the spectral density and correlation parameters.

Value

The logarithm of the prior density.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, propose.coeff.gp, calc.variances.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
propose.coeff(gp1)
propose.coeff(gp2)
print(logdensity(gp1))
print(logdensity(gp2))
```

`lonlat2xy` *Projects lon/lat coordinates to x/y Euclidean coordinate system*

Description

The projection calculates, for all points, the great circle distance in the x direction to the mean longitude and in the y direction to the mean latitude, and uses these distances as the x-y coordinates of the location. This function is copied from the fields library.

Usage

```
lonlat2xy(lnlt,miles=FALSE)
```

Arguments

`lnlt`	A two-column matrix-like object of lon/lat coordinates to be projected, with longitude in the first column.
`miles`	Indicator of whether distances should be calculated in miles or kilometers (FALSE, the default).

Details

Note that this is an ad hoc projection best used only for small portions of the globe.

Value

A two-column matrix of projected x/y coordinates, with the x-coordinate in the first column.

Author(s)

copied from the fields library by Christopher Paciorek <paciorek@alumni.cmu.edu>

See Also

xy2unit, new.mapping

Examples

```
library(spectralGP)
gp1=gp(c(128,128),matern.specdens,c(1,4))
n=100
locs=cbind(runif(n,20,80),runif(n,40,50))
locs.predict=cbind(runif(n,30,90),runif(n,38,48))
locs=lonlat2xy(locs)
locs.predict=lonlat2xy(locs.predict)
scaled.locs=xy2unit(locs,rbind(locs,locs.predict))
scaled.locs.predict=xy2unit(locs.predict,rbind(locs,locs.predict))
train.map=new.mapping(gp1,scaled.locs)
predict.map=new.mapping(gp1,scaled.locs.predict)
plot(locs,xlim=c(min(locs[,1],locs.predict[,1]),max(locs[,1],
  locs.predict[,1])),ylim=c(min(locs[,2],locs.predict[,2]),
  max(locs[,2],locs.predict[,2])))
points(locs.predict,col=2)
plot(scaled.locs,xlim=c(0,1),ylim=c(0,1))
points(scaled.locs.predict,col=2)
```

---

matern.specdens *Matern correlation spectral density function*

Description

Calculates the Matern spectral density for supplied frequencies and Matern correlation parameters. Spectral density is evaluated for each supplied frequency or pair of frequencies. The output is generally used as the prior variances for spectral GP basis coefficients.

Usage

```
matern.specdens(omega, param, d = 2)
```

Arguments

omega	Vector or two-column matrix-like object of frequencies, with the first column the frequencies in the first dimension and the second column in the second dimension.
param	Vector of two Matern parameter values, first the spatial range and second the differentiability parameter.
d	Dimension of the domain.

Details

The spectral density,

$$\frac{\Gamma(\nu+d/2)(4\nu)^{\nu}}{\pi^(d/2)\Gamma(\nu)(\pi\rho)^{2\nu}}\left(\frac{4\nu}{(\pi\rho)^2}+\omega^T\omega\right)^{-(\nu+d/2)},$$

corresponds to the following functional form of the Matern correlation function,

$$\frac{1}{\Gamma(\nu)2^{\nu-1}}\left(\frac{2\sqrt{\nu}\tau}{\rho}\right)^{\nu}\mathcal{K}_{\nu}\left(\frac{2\sqrt{\nu}\tau}{\rho}\right),$$

where rho is the range and nu the differentiability. Rho is interpreted on the scale $(0,1)^d$. Nu of 0.5 is the exponential correlation, and as nu goes to infinity the correlation approaches the squared exponential (Gaussian). Nu of 0.5 gives Gaussian processes with continuous but not differentiable sample paths, while nu of infinity gives infinitely-differentiable (and analytic) sample paths. In the spectral GP approximation, the frequencies are a sequence of integers from 0 to half the gridsize in each dimension.

Value

A vector of spectral density values corresponding to the supplied frequencies.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, calc.variances.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
dens1=matern.specdens(gp1$omega,c(1,4),d=1)
dens2=matern.specdens(gp2$omega,c(1,4),d=2)
```

---

names.gp *The names of the elements of a GP object*

Description

Gives the names of the elements of the GP object.

Usage

```
## S3 method for class 'gp'
names(x,...)
```

Arguments

x	Spectral GP object.
...	Other arguments.

Value

A vector of strings of the names of the elements of the GP object.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
print(names(gp1))
add.blocks(gp1)
print(names(gp1))
```

---

new.mapping *Map arbitrary locations to gridpoints of spectral GP object*

Description

Finds the nearest gridpoint in a spectral GP representation for each supplied location based on Euclidean distance.

Usage

```
new.mapping(object, locations)
```

Arguments

object	A GP object, created by gp.
locations	A two-column matrix-like object (vector for one-dimensional data) of locations of interest, for which the first column is the first coordinate and the second column the second coordinate. Locations should lie in $(0,1)^d$, as the process representation is on a grid on $(0,1)^d$.

Value

A vector for which each element is the index of the gridpoint nearest the location. The indices run from 1 to $(k/2)^d$ where k the number of gridpoints in each direction (assuming there are an equal number in each direction). The indices run along the first dimension from the lower right corner of the space, e.g., 13 14 15 16 9 10 11 12 5 6 7 8 1 2 3 4

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, predict.gp

Examples

```
library(spectralGP)
loc1=runif(100)
loc2=cbind(runif(100),runif(100,0,1))
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
map1=new.mapping(gp1,loc1)
map2=new.mapping(gp2,loc2)
simulate(gp1)
simulate(gp2)
vals1=predict(gp1,mapping=map1)
vals2=predict(gp2,mapping=map2)
plot(gp1)
points(loc1,vals1)
```

---

plot.gp *Plot a process based on a spectral GP object*

Description

Makes a line plot (for one-dimensional processes) or image plot (two-dimensional processes) of a process represented in a spectral GP object.

Usage

```
## S3 method for class 'gp'
plot(x, type = "l", col = terrain.colors(32), ...)
```

Arguments

x	A GP object, created by gp.
type	Type of plot if process is one-dimensional, "l" for line, "p" for points, etc.
col	Color scheme for image plot if process is two-dimensional. E.g., topo.colors(64) is the default for image_plot; I prefer terrain.colors(64) as topo.colors has sharp color changes between adjacent bins.
...	Extra arguments to plotting functions.

Value

No value is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, simulate.gp, getgrid.gp, predict.gp

Examples

```
library(spectralGP)
gp1=gp(c(128),matern.specdens,c(1,4))
simulate(gp1)
plot(gp1)
gp2=gp(c(256,256),matern.specdens,c(1,0.5))
simulate(gp2)
plot(gp2)
```

---

points.gp *Add points for a one-dimensional process based on a spectral GP object*

Description

Adds points to an existing plot.

Usage

```
## S3 method for class 'gp'
points(x, ...)
```

Arguments

x	A GP object, created by gp.
...	Extra arguments to plotting functions.

Value

No value is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, simulate.gp, getgrid.gp, predict.gp

Examples

```
library(spectralGP)
gp1=gp(c(128),matern.specdens,c(1,4))
simulate(gp1)
plot(gp1)
simulate(gp1)
points(gp1,col=2)
```

---

predict.gp *Prediction from a spectral GP object*

Description

Produces the process values of a spectral GP object on the defined grid or predicts process values for a new set of inputs (domain points).

Usage

```
## S3 method for class 'gp'
predict(object,newdata=NULL,mapping=NULL,...)
```

Arguments

object	A GP object, created by gp.
newdata	An optional two-column matrix-like object (vector for one-dimensional data) of locations of interest, for which the first column is the first coordinate and the second column the second coordinate. Locations should lie in $(0,1)^d$, as the process representation is on a grid on $(0,1)^d$.
mapping	Optional output of new.mapping, which creates a vector of indices mapping the prediction locations to their nearest gridpoints.
...	Other arguments.

Details

Does prediction for a spectral GP, either at the gridpoints or for locations by associating locations with the nearest gridpoint, depending on the arguments supplied. If newdata and mapping are both NULL, then prediction is done on the grid. If only newdata is supplied, the mapping is done using new.mapping and then the prediction is done. If mapping is supplied (this should be done for computational efficiency if prediction at the same locations will be done repeatedly) then the mapping is used directly to calculate the predictions.

Value

A vector of process values (matrix for two-dimensional processes in which prediction on the grid is requested).

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, new.mapping, plot.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
simulate(gp1)
simulate(gp2)
gridvals=predict(gp1)
gridvals2=predict(gp2)

loc1=runif(100)
loc2=cbind(runif(100),runif(100,0,1))
map1=new.mapping(gp1,loc1)
```

```
map2=new.mapping(gp2,loc2)
vals1=predict(gp1,mapping=map1)
vals2=predict(gp2,mapping=map2)
#equivalently:
vals1=predict(gp1,loc1)
vals2=predict(gp2,loc2)
plot(gp1)
points(loc1,vals1)
```

`print.gp` *Spectral GP default print statement*

Description

This is the default print statement for a spectral GP object. If you need a list of everything that is part of a spectral GP object, use 'names()'.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, names.gp

`propose.coeff.gp` *Proposes new coeffients in a spectral GP object.*

Description

Proposes new coefficients in a spectral GP object as normal deviates centered around the current values, with the proposal standard deviation the product of the supplied standard deviation(s) and the square root of the prior variances. The proposal can be done for all coefficients at once (`block=0`) or for individual blocks.

Usage

```
## S3 method for class 'gp'
propose.coeff(object, block = 0, proposal.sd = 1,...)
```

Arguments

object	A GP object, created by gp.
block	The block of coefficients to be proposed, or 0 if all coefficients are to be proposed.
proposal.sd	Proposal standard deviation. This is multiplied by the square root of the prior variance for each coefficient to produce the final proposal standard deviation.
...	Other arguments.

Details

This function can be used to simulate a GP by using proposal.sd=1 to sample coefficients, coeff, from the prior or in a MCMC context to propose new coefficient values via the Metropolis algorithm. The function automatically updates the process values in the process component of the gp list based on the new coefficient values.

Value

The function modifies the GP object, which is essentially a pointer (an R environment in this case), so NULL is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, calc.variances.gp, simulate.gp, updateprocess.gp, zero.coeff.gp

Examples

```
library(spectralGP)
rho=1
gp1=gp(128,matern.specdens,c(rho,4))
gp2=gp(c(64,64),matern.specdens,c(rho,4))
propose.coeff(gp1)
propose.coeff(gp2)
plot(gp1)
plot(gp2)
prior1=logdensity(gp1)
prior2=logdensity(gp2)
add.blocks(gp1)
add.blocks(gp2)
propose.coeff(gp1,block=2,proposal.sd=0.1)
propose.coeff(gp2,block=3,proposal.sd=0.1)
priorstar1=logdensity(gp1)
priorstar2=logdensity(gp2)
```

```
plot(gp1)
plot(gp2)
```

`rdist.earth` *Great circle distance matrix*

Description

Given two sets of longitude/latitude locations computes the Great circle (geogrpahic) distance matrix among all pairings. This function and help file are copied from the fields library.

Usage

```
rdist.earth(loc1, loc2, miles = TRUE, R = NULL)
```

Arguments

`loc1`	Matrix of first set of lon/lat coordinates first column is the longitudes and second is the latitudes.
`loc2`	Matrix of second set of lon/lat coordinates first column is the longitudes and second is the latitudes. If missing x1 is used.
`miles`	If true distances are in statute miles if false distances in kilometers.
`R`	Radius to use for sphere to find spherical distances. If NULL the radius is either in miles or kilometers depending on the values of the miles argument. If R=1 then distances are of course in radians.

Details

Surprisingly this all done efficiently in S.

Value

The great circle distance matrix if nrow(x1)=m and nrow(x2)=n then the returned matrix will be mXn.

See Also

rdist, exp.earth.cov

Examples

```
 lon.lat=cbind(runif(20,0,360),runif(20,-90,90))
 out<- rdist.earth (lon.lat)
#out is a 20X20 distance matrix
```

`simulate.gp` *Simulates a process realization from a spectral GP object*

Description

Simulates a process realization by drawing a random draw of coefficients from their prior distribution and updating the process values.

Usage

```
## S3 method for class 'gp'
simulate(object,...)
```

Arguments

`object`	A GP object, created by gp.
`...`	Other arguments.

Details

Modifies the `coeff` and `process` elements of the object.

Value

The function modifies the GP object, which is essentially a pointer (an R environment in this case), so NULL is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, zero.coeff.gp, propose.coeff.gp, updateprocess.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
simulate(gp1)
simulate(gp2)
plot(gp1)
plot(gp2)
```

spectralGP *spectralGP - tools for specifying Gaussian processes using the computationally efficient Fourier basis*

Description

SpectralGP is a collection of functions for creating Gaussian processes in one and two dimensions using the Fourier basis approximation. It provides fast simulation and plotting of process realizations by use of the FFT, allowing simulation and plotting on very dense grids. For inference, it provides tools for use in setting up an MCMC: calculation of coefficient variances, calculation of process density, and coefficient proposals. It uses R environments to store GP objects as references/pointers.

Some major methods include:

- `gp` Create a Gaussian process object
- `simulate.gp` Simulate a Gaussian process realization
- `plot.gp` Plot a Gaussian process
- `predict.gp` Extract process values at specified domain points

DISCLAIMER:

This is software for statistical research and not for commercial uses. The author does not guarantee the correctness of any function or program in this package. Any changes to the software should not be made without the author's permission.

ACKNOWLEDGEMENT:

Many thanks to Chris Wikle who first suggested I use the Fourier basis approximation for Gaussian processes.

REFERENCES:

For more details, type 'citation("spectralGP")' for references.

See also:

Royle, J.A., and C.K. Wikle, (2005). Efficient Statistical Mapping of Avian Count Data. Ecological and Environmental Statistics 12:225-243.

Wikle, C.K., (2002). Spatial modeling of count data: A case study in modelling breeding bird survey data on large spatial domains. In Spatial Cluster Modelling, A. Lawson and D. Denison, eds. Chapman and Hall, 199-209.

Examples

```
gp1=gp(128,matern.specdens,c(1,4))
gp2=gp(c(64,64),matern.specdens,c(1,4))
simulate(gp1)
simulate(gp2)
plot(gp1)
plot(gp2)
gridvals=predict(gp1)
newlocs=runif(100)
offgridvals=predict(gp1,newlocs)
```

spectralGP-generic *spectralGP generic functions*

Description

These functions are generics; see the help files associated with the spectralGP methods.

updateprocess.gp *Recalculate process values in a spectral GP object*

Description

Calculates the process values in a spectral GP object based on the current coefficient values. The process values are calculated by multiplying the coefficient values by the basis matrix, which is done by the inverse FFT.

Usage

```
## S3 method for class 'gp'
updateprocess(object,...)
```

Arguments

object	A GP object, created by gp.
...	Other arguments.

Details

Modifies the process values of the object.

Value

The function modifies the GP object, which is essentially a pointer (an R environment in this case), so NULL is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, propose.coeff.gp, simulate.gp, zero.coeff.gp

Examples

```
library(spectralGP)
gp1=gp(128,matern.specdens,c(1,4))
propose.coeff(gp1)
gp1$coeff[1,1]=0
updateprocess(gp1)
```

`xy2unit` *Scales locations to the unit hypercube for use in spectral GP*

Description

Scales locations to $(0,1)^d$ so that they can be related to the gridpoints in a spectral GP representation. The `locations.scale` argument allows one to scale the `locations` to a separate set of locations. E.g., if one wants to predict over a certain set of locations, but has a separate training set of locations that lie within the prediction set, one would use the prediction locations as the `locations.scale` argument.

Usage

```
xy2unit(locations, locations.scale = NULL)
```

Arguments

`locations`	A two-column matrix-like object (vector for one-dimensional data) of locations to be scaled.
`locations.scale`	A two-column matrix-like object (vector for one-dimensional data) of locations that provides the function with the min and max coordinates in each direction.

Details

One may want to use both training and prediction locations as the `locations.scale` argument to ensure that all locations of interest will lie in $(0,1)^d$ and be able to be related to the gridpoints.

Value

A matrix (vector for one-dimensional data) of scaled locations lying in $(0,1)^d$.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, new.mapping

Examples

```
library(spectralGP)
gp1=gp(c(128,128),matern.specdens,c(1,4))
n=100
locs=cbind(runif(n,0.2,1.2),runif(n,-0.2,1.4))
locs.predict=cbind(runif(n,-0.4,0.8),runif(n,-0.1,1.7))
scaled.locs=xy2unit(locs,rbind(locs,locs.predict))
scaled.locs.predict=xy2unit(locs.predict,rbind(locs,locs.predict))
train.map=new.mapping(gp1,scaled.locs)
predict.map=new.mapping(gp1,scaled.locs.predict)
plot(locs,xlim=c(min(locs[,1],locs.predict[,1]),max(locs[,1],
  locs.predict[,1])),ylim=c(min(locs[,2],locs.predict[,2]),
  max(locs[,2],locs.predict[,2])))
points(locs.predict,col=2)
plot(scaled.locs,xlim=c(0,1),ylim=c(0,1))
points(scaled.locs.predict,col=2)
```

---

zero.coeff.gp *Sets coefficients to zero in a spectral GP object*

Description

Sets coefficients to zero in a spectral GP object. Used to zero out the coefficients before simulating a new GP realization from the prior distribution.

Usage

```
## S3 method for class 'gp'
zero.coeff(object,...)
```

Arguments

object	A GP object, created by gp.
...	Other arguments.

Details

Modifies the coeff and process components of the object.

Value

The function modifies the GP object, which is essentially a pointer (an R environment in this case), so NULL is returned.

Author(s)

Christopher Paciorek <paciorek@alumni.cmu.edu>

References

Type 'citation("spectralGP")' for references.

See Also

gp, simulate.gp, updateprocess.gp

Index