Package 'spatialCovariance'

July 8, 2015

Version 0.6-9

Date 2015-7-5

Title Computation of Spatial Covariance Matrices for Data on Rectangles

Author David Clifford <david.clifford+CRAN@gmail.com>

Maintainer David Clifford <david.clifford+CRAN@gmail.com>

Description Functions that compute the spatial covariance matrix for the matern and power classes of spatial models, for data that arise on rectangular units. This code can also be used for the change of support problem and for spatial data that arise on irregularly shaped regions like counties or zipcodes by laying a fine grid of rectangles and aggregating the integrals in a form of Riemann integration.

License GPL

SystemRequirements

NeedsCompilation no

Repository CRAN

Date/Publication 2015-07-08 10:57:37

R topics documented:

computeV *Compute Covariance Matrix*

Description

Observations are averages over congruent rectangular plots that like in a lattice. For extensive observations one needs to multiply the matrix by the $area^2$ where $area$ is the common area of each plot.

Various different classes of covariance functions, generalised covariance functions and their derivatives wrt parameters are built into this library. These include the Cauchy and Mat\'ern covariance functions as well as specific sub models such as the Bessel$_0$, Exponential, Bessel$_1$, spline and logarithmic covariance functions.

Usage

```
computeV(info, class = "ldt", params, rel.tol = .Machine$double.eps^0.25,
         abs.tol = rel.tol, cat.level = 0, K = NULL)
```

Arguments

info	Result of the precompute stage
class	The class of covariance functions,"ldt", "bess0", "exp", "bess1", "power", "powerNI", "matern", "spline", "cauchy". Can also be used to compute the derivatives of the covariance matrices for specific models, for example "dbess0", "dexp", "dexp2", "dbess1", "dpowerNI". Can also be used for any isotropic function K, simply define a function K in the workspace that has two arguments, distance and a vector of parameters. Then call computeV with class="special".
params	Parameters that go with a specific class of models, for the "matern" class it requires an inverse range parameter and a smoothness parameter, for example params=c(1,0.5), this corresponds to the case when class="exp", params=c(1).
rel.tol	Relative Tolerance for one dimensional numerical integration
abs.tol	Absolute Tolerance for one dimensional numerical integration
cat.level	Controls level of time output, takes values 0, 0.5, 1
K	If class="misc" pass your own covariance function K here, see example below

Author(s)

David Clifford

Examples

```
## Example for extensive variables - variances of combined plots
library(spatialCovariance)
nrows <- 1
```

```
ncols <- 2
rowwidth <- 1.1
colwidth <- 1.2
rowsep <- 0
colsep <- 0

info <- precompute(nrows,ncols,rowwidth,colwidth,rowsep,colsep)
V <- computeV(info,class="matern",params=c(1,1))

info2 <- precompute(nrows=1,ncols=1,rowwidth=rowwidth,colwidth=colwidth*2,0,0)
V2 <- computeV(info2,class="matern",params=c(1,1))

c(1,1)

(rowwidth * (2*colwidth))^2 * V2

## Bring in anisotropy
V
info <- precompute.update(info,aniso=2)  ## geometric anisotropy update
V <- computeV(info,class="matern",params=c(1,1))
V
info <- precompute.update(info,aniso=5)  ## geometric anisotropy update
V

## Second Example - define your own covariance function, here we use a
## spherical one

library(spatialCovariance)

K <- function(d,params) {
  frac <- d/params
  ret <- rep(0,length(d))
  ind <- which(frac<1)
  if(length(ind)) ret[ind] <- (1 - 2/pi*(frac[ind]*sqrt(1 - frac[ind]^2) + asin(frac[ind])))
  return(ret)
}

dVals <- seq(0,10,l=1001)
plot(dVals,K(dVals,8),type="l")
lines(dVals,K(dVals,7),col=2)

nrows <- 1
ncols <- 3
rowwidth <- 2
colwidth <- 2
rowsep <- 0
colsep <- 0

info <- precompute(nrows,ncols,rowwidth,colwidth,rowsep,colsep)
V <- computeV(info,class="misc",params=c(8),K=K)
V

## Now if we have a low value of theta_2 we should see that the first
```

```
## and third plot are independent as there is a 2 unit gap between
## them, so that term in V will be zero
V <- computeV(info,class="misc",params=c(1),K=K)
V

## If theta_2 gets a little bigger than 2 then we should see no
## non-zero entries in V
V <- computeV(info,class="misc",params=c(2.005),K=K)
V

## Check V is positive definite
eigen(V)$values ## should all be positive
```

---

f *Density For Distance Between Two Points In Rectangles*

Description

This evaluates the density for the distance between two points, each distribution uniformly and independently in rectangles. The rectangles are congruent and lie on a lattice. Three special cases exist, when the two rectangles coincide, when the two rectangles lie on the same row (or column) of the lattice and when the two rectangles lie on different rows and columns.

Usage

```
f(d, rowwidth, colwidth, ax, bx, i, j)
```

Arguments

d	distance
rowwidth, colwidth	Dimensions of the rectangle
ax,bx	Coordinate of the lower left corner of the second rectangle. Lower left corner of the first is at the origin
i,j	Second rectangle lies in i-th row and j-th column of lattice.

Author(s)

David Clifford

References

B. Ghosh "Random distances within a rectangle and between two rectangles", Bulletin of the Calcutta Mathematical Society, 1951, 43, 17-24

D. Clifford, "Computation of Spatial Covariance Matrices", JCGS, March 2005, vol. 14, no. 1, pp. 155-167(13)

Examples

```
d <- 0.75
f(d,rowwidth=1,colwidth=1,ax=0,bx=0,i=1,j=1)  ## two points in a unit square
f(d,rowwidth=1,colwidth=1,ax=1,bx=0,i=2,j=1)  ## two points in squares, squares are side by side
```

`precompute` *Precompute Step for Computing Covariance Matrix*

Description

For a lattice with `nr` rows and `nc` columns on only needs to compute $n=nr X nc$ entries to fill the whole covariance matrix (of size $n X n$). For example, the diagonal entries will all be the same so one only needs to compute it once and know that the value needs to be placed along the diagonal. This algorithm figures out which entries need to be computed, and how to insert them into the covariance matrix.

When an anisotropy term `aniso` is included in the direction of rows and columns it changes how distance is measure from $sqrt (x^2+y^2)$ to $sqrt (x^2+ alpha^2 y^2)$. This amounts to stretching the lattice in the appropriate direction by a factor of $alpha$. We can update the results of the `precompute` stage very easily.

Usage

```
precompute(nrows,ncols,rowwidth,colwidth,rowsep,colsep,cat.level)
precompute.update(info,cat.level=0,aniso=1)
```

Arguments

`nrows,ncols`	Number of rows and columns in the lattice
`rowwidth, colwidth`	Dimensions of the rectangle
`rowsep,colsep`	Vectors of separations between rows and columns. Pass scalars if the separations are constant in each direction.
`cat.level`	0,0.5,1, changes the amount of output. Output is limited to times for various stages of the computation
`aniso`	Value of anisotropy parameter in the direction of rows and columns. Should be a positive number.
`info`	Result of the precompute stage

Author(s)

David Clifford

Examples

```
## See computeV help page for more details and examples
```

Index