Package 'ExceedanceTools'

February 19, 2015

Type Package

Title Confidence regions for exceedance sets and contour lines

Version 1.2.2

Date 2014-07-30

Author Joshua French

Maintainer Joshua French <joshua.french@ucdenver.edu>

Description Tools for constructing confidence regions for exceedance regions and contour lines.

License GPL (>= 2)

LazyLoad yes

Depends R (>= 2.12.0)

Imports splancs, SpatialTools

Suggests spBayes

NeedsCompilation no

Repository CRAN

Date/Publication 2014-07-30 22:02:11

R topics documented:

colorado 2
create.pgrid 2
create.pgrid2 4
exceedance.ci 5
ExceedanceTools 6
plot.pgrid 7
sdata 8
statistic.cv 9
statistic.sim 10

Index **14**

colorado *Colorado precipitation data*

Description

Data related to Colorado precipitation in May 1997. Taken from http://www.image.ucar.edu/Data/US.monthly.met/. Data is contained in a list with components odata (containing a transformed precipitation variable) and ocoords containing the longitude and latitude of the associated sites.

Usage

```
data(colorado)
```

Format

A list.

Author(s)

Joshua French

Source

National Center for Atmospheric Research

create.pgrid *Create grid of locations.*

Description

create.pgrid creates a grid of locations from the boundaries of domain and other information.

Usage

```
create.pgrid(xmin, xmax, ymin, ymax, nx, ny, midpoints = FALSE,
  poly.coords = NULL)
```

Arguments

xmin	The minimum value of the boundary of the x coordinates of the spatial domain.
xmax	The maximum value of the boundary of the x coordinates of the spatial domain.
ymin	The minimum value of the boundary of the y coordinates of the spatial domain.
ymax	The maximum value of the boundary of the y coordinates of the spatial domain.
nx	The number of gridpoints/cells/pixels in the x direction.
ny	The number of gridpoints/cells/pixels in the y direction.
midpoints	A logical value (TRUE or FALSE) indicating whether the boundary values are for the midpoint of a pixel (midpoints = TRUE) or for the boundary of the spatial domain in general (midpoints = FALSE), in which case the midpoints are calculated internally). Default is FALSE.
poly.coords	An $n \times 2$ matrix with the coordinates specifying the polygon vertices of the true spatial domain of interest within the rectangular boundaries provided by xmin, xmax, ymin, and ymax. If this is provided, the pgrid returned will be within the convex hull of poly.coords.

Details

The key argument in the function midpoints. If this is TRUE, it is assumed that the boundaries of the spatial domain correspond to the midpoints of the cell/pixel in the grid. Otherwise, it is assumed that the boundaries correspond to the actual borders of the region of interest. If poly.coords is supplied, the grid returned is the grid of midpoints contained in the convex hull of poly.coords.

Value

Returns an object of class pgrid with the following components:

pgrid	An $n \times 2$ matrix of locations (the midpoints of the pixelized grid).
m	The number of rows in pgrid.
p.in.grid	A vector of 0s and 1s indicating whether the midpoint of each pixel is in the convex hull of poly.coords. If poly.coords is not provided, this is a vector of 1s.
ubx	The pixel boundaries in the x direction.
uby	The pixel boundaries in the y direction.
upx	The pixel midpoints in the x direction.
upy	The pixel midpoints in the y direction.

Author(s)

Joshua French

Examples

```
pgrida <- create.pgrid(0, 1, 0, 1, nx = 50, ny = 50, midpoints = FALSE)
pgridb <- create.pgrid(.01, .99, .01, .99, nx = 50, ny = 50, midpoints = TRUE)
```

`create.pgrid2` *Create grid of locations.*

Description

`create.pgrid2` creates a grid of locations fusing vectors of x and y coordinates.

Usage

```
create.pgrid2(xgrid, ygrid, midpoints = FALSE, poly.coords = NULL)
```

Arguments

`xgrid`	A vector of locations in the x direction.
`ygrid`	A vector of location in the y direction.
`midpoints`	A logical value (`TRUE` or `FALSE`) indicating whether the boundary values are for the midpoint of a pixel (`midpoints = TRUE`) or for the boundary of the spatial domain in general (`midpoints = FALSE`, in which case the midpoints are calculated internally). Default is `FALSE`.
`poly.coords`	An $n \times 2$ matrix with the coordinates specifying the polygon vertices of the true spatial domain of interest within the rectangular boundaries provided by `xmin`, `xmax`, `ymin`, and `ymax`. If this is provided, the `pgrid` returned will be within the convex hull of `poly.coords`.

Details

The key argument in the function midpoints. If this is `TRUE`, it is assumed that the boundaries of the spatial domain correspond to the midpoints of the cell/pixel in the grid. Otherwise, it is assumed that the boundaries correspond to the actual borders of the region of interest. If `poly.coords` is supplied, the grid returned is the grid of midpoints contained in the convex hull of `poly.coords`.

Value

Returns an object of class pgrid with the following components:

`pgrid`	An $n \times 2$ matrix of locations (the midpoints of the pixelized grid).
`m`	The number of rows in pgrid.
`p.in.grid`	A vector of 0s and 1s indicating whether the midpoint of each pixel is in the convex hull of `poly.coords`. If `poly.coords` is not provided, this is a vector of 1s.
`ubx`	The pixel boundaries in the x-direction.
`uby`	The pixel boundaries in the y-direction.
`upx`	The pixel midpoints in the x-direction.
`upy`	The pixel midpoints in the y-direction.

Author(s)

Joshua French

Examples

```
seq1 = seq(0, 1, len = 101)
pgrida <- create.pgrid2(seq1, seq1, midpoint = FALSE)
seq2 = seq(.005, .995, len = 100)
pgridb <- create.pgrid2(seq2, seq2, midpoint = TRUE)
# pgrids produced match
range(pgrida$pgrid - pgridb$pgrid)
```

`exceedance.ci` *Return confidence region*

Description

`exceedance.ci` returns a confidence set for an exceedance region or contour line.

Usage

```
exceedance.ci(statistic.sim.obj, conf.level = 0.95, type = "null")
```

Arguments

`statistic.sim.obj`	An object returned from the `statistic.sim` function.
`conf.level`	The desired confidence level of the confidence region.
`type`	Whether the function should return the null region or rejection region of exceedance confidence region Options are `"null"` or `"rejection"`. Default is `"null"`.

Value

Returns a numeric vector with the set of pixels comprising the null or rejection region related to `statistic.sim.obj`.

Author(s)

Joshua French

Examples

```
library(SpatialTools)

# Example for exceedance regions

set.seed(10)
# Load data
data(sdata)
# Create prediction grid
pgrid <- create.pgrid(0, 1, 0, 1, nx = 26, ny = 26)
pcoords <- pgrid$pgrid
# Create design matrices
coords = cbind(sdata$x1, sdata$x2)
X <- cbind(1, coords)
Xp <- cbind(1, pcoords)

# Generate covariance matrices V, Vp, Vop using appropriate parameters for
# observed data and responses to be predicted
spcov <- cov.sp(coords = coords, sp.type = "exponential",
 sp.par = c(1, 1.5), error.var = 1/3, finescale.var = 0, pcoords = pcoords)

# Predict responses at pgrid locations
krige.obj <- krige.uk(y = as.vector(sdata$y), V = spcov$V, Vp = spcov$Vp,
 Vop = spcov$Vop, X = X, Xp = Xp, nsim = 100,
 Ve.diag = rep(1/3, length(sdata$y)) , method = "chol")

# Simulate distribution of test statistic for different alternatives
statistic.sim.obj.less <- statistic.sim(krige.obj = krige.obj, level = 5,
 alternative = "less")
statistic.sim.obj.greater <- statistic.sim(krige.obj = krige.obj, level = 5,
 alternative = "greater")
# Construct null and rejection sets for two scenarios
n90 <- exceedance.ci(statistic.sim.obj.less, conf.level = .90, type = "null")
r90 <- exceedance.ci(statistic.sim.obj.greater,conf.level = .90, type = "rejection")
# Plot results
plot(pgrid, n90, col="blue", add = FALSE, xlab = "x", ylab = "y")
plot(pgrid, r90, col="orange", add = TRUE)
legend("bottomleft",
 legend = c("contains true exceedance region with 90 percent confidence",
   "is contained in true exceedance region with 90 percent confidence"),
   col = c("blue", "orange"), lwd = 10)
```

ExceedanceTools *ExceedanceTools.*

Description

ExceedanceTools.

plot.pgrid *Plot set of pixels on grid*

Description

plot plots a grid of pixels based on an object from pgrid or confreg.

Usage

```
## S3 method for class 'pgrid'
plot(x, set, col = "gray", add = FALSE,
  type = "confidence", ...)
```

Arguments

x	An object returned from the pgrid function.
set	A vector which contains the indices of the pixels/cells that should be plotted. OR a confreg object from the confreg function. See Details.
col	The color of the plotted pixels.
add	A logical value indicating whether the pixels should be added to an existing plot (add = TRUE) or should the pixels be plotted on a new plot (add = FALSE).
type	The type of set of plot if set of of class confreg. Th default is "confidence", while the other option is complement, based on the components of the confreg object.
...	Additional arguments that will be passed to the plot function (assuming add = FALSE.)

Details

If a vector of pixel indices is supplied to set, then those pixels will be colored col by this function and the type argument has no effect. On the other hand, if the set argument is of class confreg, then the function digs in to display either the confidence or complement set in the confreg object. In that case, type is used to decide which set to display.

Value

This function does not return anything; it only creates a new plot or modifies an existing plot.

Author(s)

Joshua French

Examples

```
library(SpatialTools)

# Example for exceedance regions

set.seed(10)
# Load data
data(sdata)
# Create prediction grid
pgrid <- create.pgrid(0, 1, 0, 1, nx = 26, ny = 26)
pcoords <- pgrid$pgrid
# Create design matrices
coords = cbind(sdata$x1, sdata$x2)
X <- cbind(1, coords)
Xp <- cbind(1, pcoords)

# Generate covariance matrices V, Vp, Vop using appropriate parameters for
# observed data and responses to be predicted
spcov <- cov.sp(coords = coords, sp.type = "exponential",
 sp.par = c(1, 1.5), error.var = 1/3, finescale.var = 0, pcoords = pcoords)

# Predict responses at pgrid locations
krige.obj <- krige.uk(y = as.vector(sdata$y), V = spcov$V, Vp = spcov$Vp,
 Vop = spcov$Vop, X = X, Xp = Xp, nsim = 100,
 Ve.diag = rep(1/3, length(sdata$y)) , method = "chol")

# Simulate distribution of test statistic for different alternatives
statistic.sim.obj.less <- statistic.sim(krige.obj = krige.obj, level = 5,
 alternative = "less")
statistic.sim.obj.greater <- statistic.sim(krige.obj = krige.obj,
 level = 5, alternative = "greater")
# Construct null and rejection sets for two scenarios
n90 <- exceedance.ci(statistic.sim.obj.less, conf.level = .90,
 type = "null")
r90 <- exceedance.ci(statistic.sim.obj.greater,conf.level = .90,
 type = "rejection")
# Plot results
plot(pgrid, n90, col="blue", add = FALSE, xlab = "x", ylab = "y")
plot(pgrid, r90, col="orange", add = TRUE)
legend("bottomleft",
 legend = c("contains true exceedance region with 90 percent confidence",
 "is contained in true exceedance region with 90 percent confidence"),
 col = c("blue", "orange"), lwd = 10)
```

---

sdata *Synthetic data*

Description

A synthetic data set for use in examples. A 100x3 data frame with vectors x1 and x2 (specifying spatial location) and y, the response.

Usage

```
data(sdata)
```

Format

A data frame.

Author(s)

Joshua French

---

statistic.cv *Return critical value of distribution.*

Description

`statistic.cv` returns the critical value of the distribution of the test statistics from `statistic.sim` based on the specified confidence level. However, it is not recommended for general usage. It is recommedned that the `exceedance.ci` function be used to automatically create confidence regions.

Usage

```
statistic.cv(statistic.sim.obj, conf.level = 0.95)
```

Arguments

`statistic.sim.obj`	An object returned from the `statistic.sim` function.
`conf.level`	The desired confidence level of the confidence interval we want to construct.

Value

Returns the desired critical value.

Author(s)

Joshua French

Examples

```
library(SpatialTools)

# Example for exceedance regions

set.seed(10)
# Load data
data(sdata)
# Create prediction grid
```

```
pgrid <- create.pgrid(0, 1, 0, 1, nx = 26, ny = 26)
pcoords <- pgrid$pgrid
# Create design matrices
coords = cbind(sdata$x1, sdata$x2)
X <- cbind(1, coords)
Xp <- cbind(1, pcoords)

# Generate covariance matrices V, Vp, Vop using appropriate parameters for
# observed data and responses to be predicted
spcov <- cov.sp(coords = coords, sp.type = "exponential", sp.par = c(1, 1.5),
 error.var = 1/3, finescale.var = 0, pcoords = pcoords)

# Predict responses at pgrid locations
krige.obj <- krige.uk(y = as.vector(sdata$y), V = spcov$V, Vp = spcov$Vp,
 Vop = spcov$Vop, X = X, Xp = Xp, nsim = 100,
 Ve.diag = rep(1/3, length(sdata$y)) , method = "chol")

# Simulate distribution of test statistic for different alternatives
statistic.sim.obj.less <- statistic.sim(krige.obj = krige.obj, level = 5,
 alternative = "less")
statistic.sim.obj.greater <- statistic.sim(krige.obj = krige.obj, level = 5,
 alternative = "greater")
# Calculate quantiles of distribution of statistic
q90.less <- statistic.cv(statistic.sim.obj.less, conf.level = .90)
q90.greater <- statistic.cv(statistic.sim.obj.greater, conf.level = .90)
```

`statistic.sim` *Simulates statistics related to exceedance region.*

Description

`statistic.sim` simulates statistics related to the construction of confidence regions for exceedance sets and contour lines.

Usage

```
statistic.sim(krige.obj, level, alternative = "less", ...)
```

Arguments

`krige.obj`	An object from the function `krige.uk` in the `SpatialTools` package.
`level`	The threshold/exceedance level under consideration.
`alternative`	Indicates the type of exceedance region or level curve under consideration. For exceedances above a threshold, use (`alternative = "less"`). For exceedances below a threshold, use (`alternative = "greater"`). For contour lines, use (`alternative = "two.sided"`). Defaults to "less".
`...`	Additional arguments when `alternative = "two.sided"`. See Details.

Details

When `alternative = "two.sided"`, the `...` argument must include `user.cov` (a user-specified covariance function), `pgrid` (the grid of locations to be predicted, produced by `create.pgrid` or `create.pgrid2`), `X` (the matrix of covariates for the observed data), and any other arguments needed by `user.cov`. Note that `user.cov` should take `cLcoords` as its first argument (a matrix containing the coordinates of contour lines under consideration). Additional arguments to `user.cov` are passed internally using the `...` argument. The `user.cov` function should return a list with values `V` (the covariance matrix of the observed data), `Vop` (the cross-covariance matrix between the observed data and the responses with coordinates in cL), `Vp` (the covariance matrix of the responses with coordinates in `cL`), and `Xp` (the matrix of covariates for the coordinates contained in `cL`). See the Examples section.

Value

Returns a list with components:

`statistic`	A vector with the observed values of the test statistic.
`statistic.sim`	A vector with the observed values of the test statistic.
`alternative`	The alternative hypothesis provided to `statistic.sim`.
`level`	The threshold level under consideration.

Author(s)

Joshua French

Examples

```
library(SpatialTools)

# Example for exceedance regions

set.seed(10)
# Load data
data(sdata)
# Create prediction grid
pgrid <- create.pgrid(0, 1, 0, 1, nx = 26, ny = 26)
pcoords <- pgrid$pgrid
# Create design matrices
coords = cbind(sdata$x1, sdata$x2)
X <- cbind(1, coords)
Xp <- cbind(1, pcoords)

# Generate covariance matrices V, Vp, Vop using appropriate parameters for
# observed data and responses to be predicted
spcov <- cov.sp(coords = coords, sp.type = "exponential", sp.par = c(1, 1.5),
 error.var = 1/3, finescale.var = 0, pcoords = pcoords)

# Predict responses at pgrid locations
krige.obj <- krige.uk(y = as.vector(sdata$y), V = spcov$V, Vp = spcov$Vp,
 Vop = spcov$Vop, X = X, Xp = Xp, nsim = 50,
```

```
 Ve.diag = rep(1/3, length(sdata$y)) , method = "chol")

# Simulate distribution of test statistic for different alternatives
statistic.sim.obj.less <- statistic.sim(krige.obj = krige.obj, level = 5,
 alternative = "less")
statistic.sim.obj.greater <- statistic.sim(krige.obj = krige.obj, level = 5,
 alternative = "greater")
# Construct null and rejection sets for two scenarios
n90 <- exceedance.ci(statistic.sim.obj.less, conf.level = .90, type = "null")
r90 <- exceedance.ci(statistic.sim.obj.greater,conf.level = .90,
 type = "rejection")
# Plot results
plot(pgrid, n90, col="blue", add = FALSE, xlab = "x", ylab = "y")
plot(pgrid, r90, col="orange", add = TRUE)
legend("bottomleft",
 legend = c("contains true exceedance region with 90 percent confidence",
 "is contained in true exceedance region with 90 percent confidence"),
 col = c("blue", "orange"), lwd = 10)

# Example for level curves
data(colorado)
ocoords <- colorado$ocoords
odata <- colorado$odata

# Set up example
nsim <- 50
u <- log(16)
np <- 26
conf.level <- 0.90
x.min <- min(ocoords[,1])
x.max <- max(ocoords[,1])
y.min <- min(ocoords[,2])
y.max <- max(ocoords[,2])

#pixelize the domain
pgrid <- create.pgrid(x.min, x.max, y.min, y.max, nx = np, ny = np)
pcoords <- pgrid$pgrid; upx <- pgrid$upx; upy <- pgrid$upy
names(pcoords) <- c("lon", "lat")

# Set up covariates matrices
X <- cbind(1, ocoords)
Xp <- cbind(1, pcoords)

# Estimate covariance parameters
cov.est <- maxlik.cov.sp(X, odata, sp.type = "exponential", range.par = 1.12,
 error.ratio = 0.01, reml = TRUE, coords = ocoords)

# Create covariance matrices
myCov <- cov.sp(coords = ocoords, sp.type = "exponential",
 sp.par = cov.est$sp.par, error.var = cov.est$error.var, pcoords = pcoords)

# Krige and do conditional simulation
krige.obj <- krige.uk(y = odata, V = myCov$V, Vp = myCov$Vp, Vop = myCov$Vop,
```

```
 X = X, Xp = Xp, nsim = nsim, Ve.diag = rep(cov.est$error.var,
 length(odata)))

# Create user covariance function for simulating statistic for confidence
# regions
user.cov <- function(cLcoords,...)
{
   arglist <- list(...)
   coords <- arglist$coords
   sp.type <- arglist$sp.type
   sp.par <- arglist$sp.par
   V <- arglist$V
   out <- list(V = arglist$V,
               Vp = sp.par[1] * exp(-dist1(cLcoords)/sp.par[2]),
               Vop = sp.par[1] * exp(-dist2(coords, cLcoords)/sp.par[2]))
   out$Xp <- cbind(1, cLcoords)
   return(out)
}

# Simulation statistic for confidence regions
statistic.sim.obj <- statistic.sim(krige.obj = krige.obj, level = u,
 alternative = "two.sided", user.cov = user.cov, y = odata, pgrid = pgrid,
 X = X, coords = ocoords, pcoords = pcoords, V = myCov$V,
 sp.type = "exponential", sp.par = cov.est$sp.par)

# Create 90% confidence region
n90 <- exceedance.ci(statistic.sim.obj, conf.level = conf.level,
 type = "null")
# Get estimated contour lines
cL <- contourLines(pgrid$upx, pgrid$upy, matrix(krige.obj$pred, nrow = np),
 level = u)

# Plot results
plot(ocoords, xlab = "longitude", ylab = "latitude", type = "n",
 cex.lab = 1.5, cex.axis = 1.5)
plot(pgrid, n90, col = "grey", add = TRUE)
plot.contourLines(cL, col="black", lwd=2, lty = 2, add = TRUE)
```

Index

colorado, 2
create.pgrid, 2
create.pgrid2, 4

exceedance.ci, 5
ExceedanceTools, 6
ExceedanceTools-package
        (ExceedanceTools), 6

plot.pgrid, 7

sdata, 8
statistic.cv, 9
statistic.sim, 10