Package 'spTest'

November 10, 2015

Title Nonparametric Hypothesis Tests of Isotropy and Symmetry

Date 2015-11-10

Version 0.2.3

Description Implements nonparametric hypothesis tests to check isotropy and symmetry properties for two-dimensional spatial data.

Imports stats,fields,methods

Suggests mvtnorm,geoR

Depends R (>= 3.0.0)

License CC0

LazyData TRUE

RoxygenNote 5.0.0

NeedsCompilation no

Author Zachary Weller [aut, cre]

Maintainer Zachary Weller <wellerz@stat.colostate.edu>

Repository CRAN

Date/Publication 2015-11-10 19:04:31

R topics documented:

coords.aniso 2
GuanTestGrid 3
GuanTestUnif 6
htestIso-class 9
LuTest 9
MaityTest 11
print.htestIso 14
spTest 15
WRFG 16

Index **18**

`coords.aniso` *Geometric Anisotropy Correction from geoR*

Description

This is the function `coords.aniso` from the R package **geoR**. It performs the linear transformation or reverse transformation of a set of spatial coordinates in R^2 according to geometric anisotropy parameters. See geoR documentation for more details.

Usage

```
coords.aniso(coords, aniso.pars, reverse = FALSE)
```

Arguments

`coords`	An $n \times 2$ matrix of spatial coordinates.
`aniso.pars`	A vector of length two containing the anisotropy angle and anisotropy ratio, respectively.
`reverse`	Logical. If `reverse = TRUE` the reverse transformation is performed.

Details

Details on the function can be found in the geoR documentation. The function is included in this package to avoid the necesity of loading the **geoR** package when simulating anisotropic spatial processes.

Value

An $n \times 2$ matrix of transformed coordinates.

Author(s)

Paulo Justiniano Ribeiro Jr.

Peter J. Diggle

References

Paulo J. Ribeiro Jr & Peter J. Diggle geoR: a package for geostatistical analysis R-NEWS, 1(2):15-18. June, 2001.

See Also

`coords.aniso`

Examples

```
#Set parameter values for exponential covariance function
sigma.sq <- 1
tau.sq <- 0.0
phi <- 1/4
#Set anisotropy parameters
aniso.angle <- pi/4
aniso.ratio <- 2
#function to compute distance corresponding to a correlation contour
#for a spatial process in R^2 with an exponential covariance function
cor.dist = function(cv, ss, ts, phi)
{
h <- -1*( log(cv*(ss+ts)/ss) ) /(phi)
return(h)
}
#compute the distance of 0.5 correlation for isotropic process
r <-  cor.dist(0.5, sigma.sq, tau.sq, phi)
#create a sequence of points corresponding to equicorrelation
#with a data point observed at the location (0,0)
angle <- seq(0, 2*pi, by = 0.01)
x <- r*sin(angle)
y <- r*cos(angle)
cor.coords <- cbind(x,y)
#plot the contour of equicorrelation for isotropic process
plot(cor.coords, type = "l", xlim = c(-5,5), ylim = c(-5,5), pty = "s", lwd = 2)
points(0,0, pch = 20, cex = 1.3)
#transform and plot the coordinates according to the anisotropy parameters
aniso.cor.coords <- coords.aniso(cor.coords, c(aniso.angle, aniso.ratio), rev = TRUE)
lines(aniso.cor.coords, lwd = 2, lty = 2)
```

`GuanTestGrid` *Nonparametric Test of Isotropy Using the Sample Semivariogram*

Description

This function performs the nonparametric test of isotropy using the sample semivariogram from Guan et. al. (2004) for spatial data with sampling locations on a grid. See Guan et. al. (2004) for more details.

Usage

```
GuanTestGrid(spdata, delta = 1, lagmat = rbind(c(1, 0), c(0, 1), c(1, 1),
  c(-1, 1)), A = rbind(c(1, -1, 0, 0), c(0, 0, 1, -1)), df = 2,
  window.dims = c(2, 2), pt.est.edge = TRUE, sig.est.edge = TRUE,
  sig.est.finite = TRUE)
```

Arguments

spdata	An $n \times 3$ matrix. The first two columns provide (x, y) spatial coordinates. The third column provides data values at the coordinates.
delta	A scalar indicating the distance between grid locations. Defaults to 1 (integer grid) and assumes equal spacing between locations in the x and y directions.
lagmat	A $k \times 2$ matrix of spatial lags. Each row corresponds to a lag of the form $(x.lag, y.lag)$ for which the semivariogram value will be estimated. The scale of the lags provided in 'lagmat' are in units of delta. See details for more information.
A	A $d \times k$ contrast matrix. The contrasts correspond to contrasts of the estimated semivariogram at the lags given in lagmat.
df	A scalar indicating the row rank of the matrix A. This value gives the degrees of freedom for the asymptotic χ^2 distribution used to compute the p-value.
window.dims	A vector of length two corresponding to the width and height (in number of columns and rows, respectively) of the moving windows used to estimate the asymptotic variance-covariance matrix. If window width does not evenly divide the number of columns of spatial data, some data will be ommited during subsampling, i.e., function does not handle partial windows. Same applies to window height and number of rows of spatial data.
pt.est.edge	Logical. TRUE corrects for edge effects in the point estimate (see Guan et. al. (2004), Section 4.2.1 for details).
sig.est.edge	Logical. Defaults to TRUE, which corrects for edge effects when estimating the semivariogram in the moving windows (see Guan et. al. (2004), Section 4.2.1 for details).
sig.est.finite	Logical. Defaults to TRUE, which provides a finite sample correction in estimating Σ, the asymptotic variance-covariance matrix. (see Guan et. al. (2004) Section 3.2.1, Equation 5). False provides the empirical variance-covariance matrix of sample semivariogram values computed via the moving windows.

Details

This function currently only supports square and rectangular sampling regions and does not currently support partial blocks. For example, suppose the sampling grid contains 20 columns and 30 rows of data. Then an ideal value of window.dims would be (2,3) since its entries evenly divide the number of columns (20) and rows (30), respectively, of data. To preserve the spatial dependence structure, the moving window should have the same shape (i.e. square or rectangle) and orientation as the entire sampling domain.

The parameter delta serves to scale the samplng locations to the integer grid. Thus the lags provided in lagmat are automatically scaled by delta by the function. For example, suppose spatial locations are observed on grid boxes of 0.5 degrees by 0.5 degrees and referenced by longitude and latitude coordinates in degrees. Then, delta should be 0.5 and a spatial lag of (0,1) corresponds to a change in coordinates of (0, 0.5), i.e., moving one sampling location north in the y-direction.

Value

gamma.hat	A matrix of the spatial lags provided and the semivariogram point estimates,$\hat{\gamma}()$, at the lags used to construct the test statistic.

`sigma.hat`	The estimate of asymptotic variance-covariance matrix, $\widehat{\Sigma}$, used to construct the test statistic.
`n.subblocks`	The number of subblocks created by the moving window used to estimate Σ.
`test.stat`	The calculated test statistic.
`pvalue.finite`	The approximate, finite-sample adjusted p-value computed by using the subblocks created by the moving windows (see Guan et. al. (2004), Section 3.3 for details).
`pvalue.chisq`	The p-value computed using the asymptotic χ^2 distribution.

References

Guan, Y., Sherman, M., & Calvin, J. A. (2004). A nonparametric test for spatial isotropy using subsampling. *Journal of the American Statistical Association*, 99(467), 810-821.

See Also

`GuanTestUnif` `MaityTest`

Examples

```
library(mvtnorm)
set.seed(1)
#number of rows and columns
nr <- 12
nc <- 18
n <- nr*nc
#Set up the coordinates
coords <- expand.grid(0:(nr-1), 0:(nc-1))
coords <- cbind(coords[,2], coords[,1])
#compute the distance between sampling locations
D <- as.matrix(dist(coords))
#Set parameter values for exponential covariance function
sigma.sq <- 1
tau.sq <- 0.0
phi <- 1/4
R <- sigma.sq * exp(-phi*D)
R <- R + diag(tau.sq, nrow = n, ncol = n)
#Simulate Gaussian spatial data
z <- rmvnorm(1,rep(0,n), R, method = c("chol"))
z <-  z-mean(z)
z <- t(z)
mydata <- cbind(coords, z)
mylags <-  rbind(c(1,0), c(0, 1), c(1, 1), c(-1,1))
myA <-  rbind(c(1, -1, 0 , 0), c(0, 0, 1, -1))
tr <- GuanTestGrid(mydata, delta = 1, mylags, myA, df = 2, window.dims = c(3,2),
pt.est.edge = TRUE, sig.est.edge = TRUE, sig.est.finite = TRUE )
tr

#Simulate data from anisotropic covariance function
aniso.angle <- pi/4
aniso.ratio <- 2
```

```
coordsA <- coords.aniso(coords, c(aniso.angle, aniso.ratio))
Da <- as.matrix(dist(coordsA))
R <- sigma.sq * exp(-phi*Da)
R <- R + diag(tau.sq, nrow = n, ncol = n)
z <- rmvnorm(1,rep(0,n), R, method = c("chol"))
z <-  z-mean(z)
z <- t(z)
mydata <- cbind(coords, z)
#Run the test on the data generated from an anisotropic covariance function
tr <- GuanTestGrid(mydata, delta = 1, mylags, myA, df = 2, window.dims = c(3,2),
pt.est.edge = TRUE,sig.est.edge = TRUE, sig.est.finite = TRUE)
tr
```

`GuanTestUnif` *Nonparametric Test of Isotropy Using the Sample Semivariogram*

Description

This function performs the nonparametric test of isotropy using the sample semivariogram from Guan et. al. (2004) for spatial data with uniformly distributed sampling locations. See Guan et. al. (2004) for more details.

Usage

```
GuanTestUnif(spdata, lagmat, A, df, h = 0.7, kernel = "norm",
  truncation = 1.5, xlims, ylims, grid.spacing = c(1, 1),
  window.dims = c(2, 2), subblock.h = h, sig.est.finite = TRUE)
```

Arguments

`spdata`	An $n \times 3$ matrix. The first two columns provide (x, y) spatial coordinates. The third column provides data values at the coordinates.
`lagmat`	A $k \times 2$ matrix of spatial lags. Each row corresponds to a lag of the form $(x.lag, y.lag)$ for which the semivariogram value will be estimated.
`A`	A $d \times k$ contrast matrix. The contrasts correspond to contrasts of the estimated semivariogram at the lags given in `lagmat`.
`df`	A scalar indicating the row rank of the matrix `A`. This value gives the degrees of freedom for the asymptotic χ^2 distribution used to compute the p-value.
`h`	A scalar giving the bandwidth for the kernel smoother. The same bandwidth is used for lags in both the x and y directions.
`kernel`	A string taking one of the following values: `norm`, `ep`, `cos`, or `unif`, for the normal, Epanechnikov, cosine, or uniform kernel functions. Defaults to `norm`.
`truncation`	A scalar providing the truncation value for the normal density if `kernel = "norm"`.
`xlims`	A vector of length two providing the lower and upper x-limits of the sampling region. To ensure all sampling locations are included in the subsampling procedure, the x-limits should be **slightly** wider than than the minimum and maximum observed x-coordinates of sampling locations.

`ylims` A vector of length two providing the lower and upper y-limits of the sampling region. To ensure all sampling locations are included in the subsampling procedure, the y-limits should be **slightly** wider than than the minimum and maximum observed y-coordinates of sampling locations.

`grid.spacing` A vector of length two providing the x (width) and y (height) spacing, respectively, of the underlying grid laid on the sampling region to create moving windows. If the grid spacing width does not evenly divide the width of the sampling region, some data will be ommited during subsampling, i.e., the function does not handle partial windows. Same applies to grid spacing height and height of sampling region. See details for an example.

`window.dims` A vector of length two corresponding to the width and height of the moving windows used to estimate the asymptotic variance-covariance matrix. The width and height are given in terms of the spacing of the grid laid on the sampling region. See details for an example.

`subblock.h` A scalar giving the bandwidth used for the kernel smoother when estimating the semivariogram on the moving windows (sub-blocks of data). Defaults to the same bandwidth used for the entire domain.

`sig.est.finite` Logical. Defaults to `TRUE`, which provides a finite sample correction in estimating Σ (see Guan et. al. (2004) Section 4.2.2). False provides the empirical variance-covariance matrix of sample semivariogram values computed via the moving windows.

Details

This function currently only supports square and rectangular sampling regions and does not support partial blocks. For example, suppose the sampling region runs from 0 to 20 in the x-direction and from 0 to 30 in the y-direction and an underlying grid of 1 by 1 is laid over the sampling region. Then an ideal value of `window.dims` would be (2,3) since its entries evenly divide the width (20) and height (30), respectively, of the sampling region. Using `window.dims` (3, 4.5) would imply that some data will not be used in the moving windows since these values would create partial blocks in the sampling region.

The value `window.dims` provides the width and height of the moving window in terms of the underlying grid laid on the sampling region. For example, if a grid with dimensions of grid.spacing = c(0.1, 0.2) is laid on the sampling region and window.dims = c(2,3) then the dimensions of the subblocks created by the moving windows are (0.2, 0.6). Thus, the user must take care to ensure that the values of `grid.spacing` and `window.dims` are compatible with the dimensions of the sampling region. The easiest way to meet this constrain is to make the `grid.spacing` values a function of the `xlims` and `ylims` values. For example, to put down a 10×10 grid on the domain, use `grid.spacing = (xlims[2]-xlims[1], ylim[2]-ylims[1])/10`. Then, setting `window.dims = c(2,2)` ensures that no data will be omitted during the subsampling.

To preserve the spatial dependence structure, the moving window should have the same shape (i.e. square or rectangle) and orientation as the entire sampling domain.

Value

`gamma.hat` A matrix of the spatial lags provided and the semivariogram point estimates, $\hat{\gamma}()$, at those lags used to construct the test statistic.

`sigma.hat`	The estimate of asymptotic variance-covariance matrix, $\widehat{\Sigma}$, used to construct the test statistic.
`n.subblocks`	The number of subblocks created by the moving window used to estimate Σ.
`test.stat`	The calculated test statistic.
`pvalue.finite`	The approximate, finite-sample adjusted p-value computed by using the subblocks created by the moving windows (see Guan et. al. (2004), Section 3.3 for details).
`pvalue.chisq`	The p-value computed using the asymptotic χ^2 distribution.

References

Guan, Y., Sherman, M., & Calvin, J. A. (2004). A nonparametric test for spatial isotropy using subsampling. *Journal of the American Statistical Association*, 99(467), 810-821.

See Also

MaityTest GuanTestGrid

Examples

```
library(mvtnorm)
set.seed(1)
#Sample Size
N <- 300
#Set parameter values for exponential covariance function
sigma.sq <- 1
tau.sq <- 0.0
phi <- 1/4
#Generate sampling locations
coords <-  cbind(runif(N,0,16), runif(N,0,16))
D <-  as.matrix(dist(coords))
R <- sigma.sq * exp(-phi*D)
R <- R + diag(tau.sq, nrow = N, ncol = N)
#Simulate Gaussian spatial data
z <- rmvnorm(1,rep(0,N), R, method = "chol")
z <- z - mean(z)
z <- t(z)
mydata <- cbind(coords, z)
mylags = rbind(c(1,0), c(0, 1), c(1, 1), c(-1,1))
myA = rbind(c(1, -1, 0 , 0), c(0, 0, 1, -1))
my.grid = c(1,1)
my.windims = c(4,4)
myh = 0.7
myh.sb = 0.8
my.xlims = c(0, 16)
my.ylims = c(0, 16)
tr <- GuanTestUnif(mydata, mylags, myA, df = 2, myh, "norm", 1.5,
 my.xlims, my.ylims, my.grid,my.windims, myh.sb)
tr
```

```
##Not run ##
#Simulate data from anisotropic covariance function
#aniso.angle <- pi/4
#aniso.ratio <- 2
#coordsA <- coords.aniso(coords, c(aniso.angle, aniso.ratio))
#Da <- as.matrix(dist(coordsA))
#R <- sigma.sq * exp(-phi*Da)
#R <- R + diag(tau.sq, nrow = N, ncol = N)
#z <- rmvnorm(1,rep(0,N), R, method = c("chol"))
#z <-  z-mean(z)
#z <- t(z)
#mydata <- cbind(coords, z)
#Run the test on the data generated from an anisotropic covariance function
#tr <- GuanTestUnif(mydata, mylags, myA, df = 2, myh, "norm", 1.5,
# my.xlims, my.ylims, my.grid,my.windims, myh.sb)
#tr
```

`htestIso-class` *An S4 class to hold output from a hypothesis test of isotropy. Extends the S3 class 'htest'.*

Description

An S4 class to hold output from a hypothesis test of isotropy. Extends the S3 class 'htest'.

Slots

`p.value.finite` A length-one numeric vector containing the p-value approximated by using a finite sample adjustment.

`sigma.hat` A matrix containing the estimated asymptotic variance-covariance matrix.

`n.subblocks` A length-one numeric vector containing the number of subblocks used in estimating the asymptotic variance-covariance.

`n.boot` A length-one numeric vector containing the number of bootstrap samples used in estimating the asymptotic variance-covariance.

`LuTest` *Nonparametric Test of Symmetry Using the Periodogram*

Description

This function performs the nonparametric tests of reflection and complete symmetry using the periodogram from Lu and Zimmerman (2005) for spatial data with sampling locations on the integer grid. See Lu and Zimmerman (2005) for more details.

Usage

```
LuTest(spdata, nrows, ncols, test = "complete", nsim = 5000)
```

Arguments

spdata	An $n \times 3$ matrix. The first two columns provide (x, y) spatial coordinates. The third column provides data values at the coordinates.
nrows	The number of rows of observed data.
ncols	The number of columns of observed data.
test	A string taking the value `reflection` or `complete` for a test of reflection or complete symmetry. If `test = "complete"`, a test for complete symmetry is performed after a test of reflection symmetry is performed and both p-values are returned.
nsim	The number simulations used to approximate the sampling distribution of CvM and CvM* from Lu ad Zimmerman (2005).

Details

The function assumes data are on the integer grid, Z^2. It uses the (unsmoothed) periodogram, the Fourier transform of the sample covariance function, to test symmetry properties.

Value

pvalue.refl	The p-value for the test of reflection symmetry computed by the CvM GoF test.
pvalue.comp	If `test = "complete"`, the p-value for the test of complete symmetry computed by using the CvM* GoF test.

References

Lu, N., & Zimmerman, D. L. (2005). Testing for directional symmetry in spatial dependence using the periodogram. *Journal of Statistical Planning and Inference*, 129(1), 369-385.

See Also

GuanTestGrid

Examples

```
library(mvtnorm)
set.seed(1)
#Number of rows and columns
nr <- 15
nc <- 15
n <- nr*nc
#Set up the coordinates
coords <- expand.grid(0:(nr-1), 0:(nc-1))
coords <- cbind(coords[,2], coords[,1])
#Compute the distance between sampling locations
D <- as.matrix(dist(coords))
#Set parameter values for exponential covariance function
sigma.sq <- 1
tau.sq <- 0.0
phi <- 1/4
```

```
#Simulate data using isotropic covariance function
D <- as.matrix(dist(coords))
R <- sigma.sq * exp(-phi*D)
R <- R + diag(tau.sq, nrow = n, ncol = n)
z <- rmvnorm(1,rep(0,n), R, method = c("chol"))
z <-  z-mean(z)
z <- t(z)
mydata <- cbind(coords, z)
#Run the test on the data from an isotropic (symmetric) covariance function
tr <- LuTest(mydata, nr, nc, test = "complete", nsim = 1000)
tr

#Simulate data from anisotropic (non-symmetric) covariance function
aniso.angle <- pi/4
aniso.ratio <- 2
coordsA <- coords.aniso(coords, c(aniso.angle, aniso.ratio))
Da <- as.matrix(dist(coordsA))
R <- sigma.sq * exp(-phi*Da)
R <- R + diag(tau.sq, nrow = n, ncol = n)
z <- rmvnorm(1,rep(0,n), R, method = c("chol"))
z <-  z-mean(z)
z <- t(z)
mydata <- cbind(coords, z)
#Run the test on the data generated from an anisotropic
#(and non reflection and non completely symmetric) covariance function
tr <- LuTest(mydata, nr, nc, test = "complete", nsim = 1000)
tr
```

`MaityTest` *Nonparametric Test of Isotropy Using the Sample Covariogram*

Description

This function performs the nonparametric test of isotropy using the sample covariogram from Maity and Sherman (2012) for spatial data with sampling locations following any general spatial sampling design. It uses the Epanechnikov kernel function with an empirical bandwidth parameter to smooth over spatial lags. The asymptotic variance-covariance matrix is estimated using the grid based block bootstrap (GBBB) from Lahiri and Zhu (2006). See Maity and Sherman (2012) for more details.

Usage

```
MaityTest(spdata, lagmat, A, df, xlims, ylims, grid.spacing = c(1, 1),
  block.dims, nBoot = 100, kappa = 1, user.bandwidth = F,
  bandwidth = c(1, 1))
```

Arguments

`spdata`	An $n \times 3$ matrix. The first two columns provide (x, y) spatial coordinates. The third column provides data values at the coordinates.

lagmat	A $k \times 2$ matrix of spatial lags. Each row corresponds to a lag of the form $(x.lag, y.lag)$ for which the covariogram value will be estimated.
A	A $d \times k$ contrast matrix. The contrasts correspond to contrasts of the estimated covariogram at the lags given in 'lagmat'.
df	A scalar indicating the row rank of the matrix A. This value gives the degrees of freedom for the asymptotic χ^2 distribution used to compute the p-value.
xlims	A vector of length two providing the lower and upper x-limits of the sampling region. To ensure all sampling locations are included in the subsampling procedure, the x-limits should be **slightly** wider than than the minimum and maximum observed x-coordinates of sampling locations.
ylims	A vector of length two providing the lower and upper y-limits of the sampling region. To ensure all sampling locations are included in the subsampling procedure, the y-limits should be **slightly** wider than than the minimum and maximum observed y-coordinates of sampling locations.
grid.spacing	A vector of length two providing the x (width) and y (height) spacing, respectively, of the underlying grid laid on the sampling region to create spatial blocks. If the grid spacing width does not evenly divide the width of the sampling region, some data will be ommited during subsampling, i.e., the function does not handle partial windows. Same applies to grid spacing height and height of sampling region. See details for an example.
block.dims	A vector of length two corresponding to the width and height of the blocks used in the GBBB. The width and height are given in terms of the spacing of the grid laid on the sampling region. See details for an example.
nBoot	A scalar indicating the number of grid based block bootstrap (GBBB) samples to compute (see Lahiri and Zhu (2006) for details on GBBB). Recommended to be at least 50.
kappa	A scalar corresponding to the tuning parameter to adjust empirical bandwidth.
user.bandwidth	Logical. Defaults to FALSE. Set to TRUE to manually override the empirical bandwidth.
bandwidth	When user.bandwidth = TRUE, a vector of length two providing the user-definted bandwidths for smoothing over spatial lags in the x and y directions.

Details

This function currently only supports square and rectangular sampling regions and does not support partial blocks. For example, suppose the sampling region runs from 0 to 20 in the x-direction (xlims = c(0, 20)) and from 0 to 30 in the y-direction (ylims = c(0, 30)) and an underlying grid of 1 by 1 is laid over the sampling region. Then an ideal value of block.dims would be (2,3) since this would imply the blocks created have dimension 2 by 3 and these values evenly divide the width (20) and height (30), respectively, of the sampling region. Using the vector (3, 4.5) would imply that some data will not be used in the GBBB since these values would create partial blocks in the sampling region.

The value block.dims provides the width and height of the blocks in terms of the underlying grid laid on the sampling region. For example, if a grid with dimensions of grid.spacing = c(0.1, 0.2) is laid on the sampling region and block.dims = c(2,3) then the dimensions of the blocks created

by the moving windows are (0.2, 0.6). The block dimensions of 0.2 and 0.6 should evenly divide the width and height, respectively, of the entire sampling region. Thus, the user must take care to ensure that the values of `grid.spacing` and `block.dims` are compatible with the dimensions of the sampling region. The easiest way to meet this constrain is to make the `grid.spacing` values a function of the `xlims` and `ylims` values. For example, to put down a 10×10 grid on the domain, use `grid.spacing = (xlims[2]-xlims[1], ylim[2]-ylims[1])/10`. Then, setting `block.dims = c(2,2)` ensures that no data will be omitted during thes subsampling.

To preserve the spatial dependence structure, the spatial blocks should have the same shape (i.e. square or rectangle) and orientation as the entire sampling domain.

Value

`C.hat`	A matrix of the spatial lags provided and the covariogram, $\hat{C}()$, point estimates at those lags used to construct the test statistic.
`V.hat`	The estimate of the asymptotic variance-covariance matrix, $\hat{V}$, used to construct the test statistic.
`n.boot`	The number of bootstrap samples used to estimate V.
`test.stat`	The calculated test statistic.
`pvalue.chisq`	The p-value computed using the asymptotic χ^2 distribution.

References

Maity, A., & Sherman, M. (2012). Testing for spatial isotropy under general designs. *Journal of Statistical Planning and Inference*, 142(5), 1081-1091.

Lahiri, S. N., & Zhu, J. (2006). Resampling methods for spatial regression models under a class of stochastic designs. *The Annals of Statistics*, 34(4), 1774-1813.

See Also

`GuanTestUnif`

Examples

```
library(mvtnorm)
set.seed(1)
#Sample Size
N <- 300
#Set parameter values for exponential covariance function
sigma.sq <- 1
tau.sq <- 0.0
phi <- 1/4
#Generate sampling locations
coords <-  cbind(runif(N,0,16), runif(N,0,16))
D <-  as.matrix(dist(coords))
R <- sigma.sq * exp(-phi*D)
R <- R + diag(tau.sq, nrow = N, ncol = N)
#Simulate Gaussian spatial data
z <- rmvnorm(1,rep(0,N), R, method = "chol")
z <- z - mean(z)
```

```
z <- t(z)
mydata <- cbind(coords, z)
mylags <- rbind(c(1,0), c(0, 1), c(1, 1), c(-1,1))
myA <- rbind(c(1, -1, 0 , 0), c(0, 0, 1, -1))
my.xlims <- c(0, 16)
my.ylims <- c(0, 16)
my.grid <- c(1,1)
my.blockdims <- c(4,4)
#Number of bootstraps for demonstration only.
#In practice, use nBoot > 50
my.nBoot <- 3
tr <- MaityTest(mydata, mylags, myA, df = 2, my.xlims, my.ylims, my.grid,
my.blockdims, nBoot = my.nBoot)
tr

####NOT RUN####
# #Simulate data from anisotropic covariance function
# aniso.angle <- pi/4
# aniso.ratio <- 2
# coordsA <- coords.aniso(coords, c(aniso.angle, aniso.ratio))
# Da <- as.matrix(dist(coordsA))
# R <- sigma.sq * exp(-phi*Da)
# R <- R + diag(tau.sq, nrow = N, ncol = N)
# z <- rmvnorm(1,rep(0,N), R, method = c("chol"))
# z <-  z-mean(z)
# z <- t(z)
# mydata <- cbind(coords, z)
# #Run the test on the data generated from an anisotropic covariance function
# #Increase the number of bootstraps
# my.nBoot = 100
# tr <- MaityTest(mydata, mylags, myA, df = 2, my.xlims, my.ylims,
# my.grid, my.blockdims, nBoot = my.nBoot)
# tr
```

`print.htestIso` *Print hypothesis test results.*

Description

Print the results from a nonparametric hypothesis test of isotropy/symmetry.

Usage

```
## S3 method for class 'htestIso'
print(x, digits = getOption("digits"), prefix = "\t",
  ...)
```

Arguments

x	An object of class 'htestIso' from the LuTest, GuanTestGrid, GuanTestUnif, or MaityTest functions
digits	Number of significant digits in printed output.
prefix	A character defining the prefix used for lines of output. Default is a tab.
...	Other arguments to print.

Value

Summary results of the hypothesis test.

Examples

```
library(mvtnorm)
set.seed(1)
#number of rows and columns
nr <- 12
nc <- 18
n <- nr*nc
#Set up the coordinates
coords <- expand.grid(0:(nr-1), 0:(nc-1))
coords <- cbind(coords[,2], coords[,1])
#compute the distance between sampling locations
D <- as.matrix(dist(coords))
#Set parameter values for exponential covariance function
sigma.sq <- 1
tau.sq <- 0.0
phi <- 1/4
R <- sigma.sq * exp(-phi*D)
R <- R + diag(tau.sq, nrow = n, ncol = n)
#Simulate Gaussian spatial data
z <- rmvnorm(1,rep(0,n), R, method = c("chol"))
z <-  z-mean(z)
z <- t(z)
mydata <- cbind(coords, z)
mylags <-  rbind(c(1,0), c(0, 1), c(1, 1), c(-1,1))
myA <-  rbind(c(1, -1, 0 , 0), c(0, 0, 1, -1))
tr <- GuanTestGrid(mydata, delta = 1, mylags, myA, df = 2, window.dims = c(3,2),
pt.est.edge = TRUE, sig.est.edge = TRUE, sig.est.finite = TRUE )
print.htestIso(tr) #print the summary
tr #can also print it using this command
```

---

spTest *Nonparametric Hypothesis Tests of Isotropy and Symmetry*

Description

The package **spTest** implements nonparametric hypothesis tests of isotropy and symmetry in spatial data. The available functions are LuTest,GuanTestGrid, GuanTestUnif, and MaityTest.

References

- Lu, N., & Zimmerman, D. L. (2005). Testing for directional symmetry in spatial dependence using the periodogram. *Journal of Statistical Planning and Inference*, 129(1), 369-385.
- Guan, Y., Sherman, M., & Calvin, J. A. (2004). A nonparametric test for spatial isotropy using subsampling. *Journal of the American Statistical Association*, 99(467), 810-821.
- Maity, A., & Sherman, M. (2012). Testing for spatial isotropy under general designs. *Journal of Statistical Planning and Inference*, 142(5), 1081-1091.

`WRFG` *Average temperature from WRFG RCM output from NARCCAP*

Description

A list containing coordinates and averages of yearly average temperatures from a 24-year period of WRFG-NCEP RCM output on 14606 grid boxes over North America (Weather Research and Forecasting - Grell scheme [WRFG] with boundary conditions provided by the National Center for Environmental Prediciton [NCEP] Regional Climate Model [RCM]).

Usage

```
WRFG
```

Format

A list containing five items:

lon longitude of the grid boxes

lat latitude of the grid boxes

xc x coordinates of the grid boxes (Lambert Conformal projection)

yc y coordinates of the grid boxes (Lambert Conformal projection)

WRFG.NCEP.tas matrix of average temperatures

Details

RCM output is available through the North American Regional Climate Change Assessment Program (NARCCAP). Data used for this example are from the WRFG model with NCEP boundary conditions. For the years 1981-2004, grid box temperatures (given by the variable 'tas') are averaged over the entire year. Yearly average temperatures for each grid box were then averaged over the 24 year period to create the matrix WRFG.NCEP.tas.

Source

https://www.earthsystemgrid.org/project/narccap.html

Examples

```
data(WRFG)
library(fields)
image.plot(WRFG$lon-360, WRFG$lat, WRFG$WRFG.NCEP.tas)
world(add = TRUE)
```

Index

∗Topic **datasets**
WRFG, 16
∗Topic **external**
coords.aniso, 2
GuanTestGrid, 3
GuanTestUnif, 6
LuTest, 9
MaityTest, 11
print.htestIso, 14

coords.aniso, *2*, 2

GuanTestGrid, 3, *8*, *10*, *15*
GuanTestUnif, *5*, 6, *13*, *15*

htestIso *(*htestIso-class*)*, 9
htestIso-class, 9

LuTest, 9, *15*

MaityTest, *5*, *8*, 11, *15*

print.htestIso, 14

spTest, 15
spTest-package *(*spTest*)*, 15

WRFG, 16