Package ‘SpatialPack’

June 21, 2020

Type Package

Title Tools for Assessment the Association Between Two Spatial Processes

Version 0.3-819

Description Tools to assess the association between two spatial processes. Currently, several methodologies are implemented: A modified t-test to perform hypothesis testing about the independence between the processes, a suitable nonparametric correlation coefficient, the codispersion coefficient, and an F test for assessing the multiple correlation between one spatial process and several others. Functions for image processing and computing the spatial association between images are also provided.

Author Felipe Osorio [aut, cre] (<https://orcid.org/0000-0002-4675-5201>),
Ronny Vallejos [aut] (<https://orcid.org/0000-0001-5519-0946>),
Francisco Cuevas [ctb],
Diego Mancilla [ctb]

Maintainer Felipe Osorio <felipe.osorios@usm.cl>

Depends R (>= 2.10)

Imports stats, graphics

License GPL-3

URL http://spatialpack.mat.utfsm.cl

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-06-21 13:40:09 UTC

R topics documented:

`clipping` *Clipping image*

Description

This function returns the image which restricts pixel value from the specified range.

Usage

```
clipping(img, low = 0, high = 1)
```

Arguments

`img`	input grayscale image matrix.
`low`	lowest value.
`high`	highest value.

Value

grayscale image matrix with the same size as 'img'.

Examples

```
data(texmos2)
plot(as.raster(texmos2))

# the appearance of next one doesn't change because of normalization
x <- normalize(2 * texmos2)
plot(as.raster(x))
title(main = "Doubled pixel value with normalization", font.main = 1)

# the next one is saturated as expected
x <- clipping(2 * texmos2)
plot(as.raster(x))
title(main = "Doubled pixel value with clipping", font.main = 1)
```

codisp — *Codispersion Coefficient*

Description

Computes the codispersion coefficient between two spatial variables for a given number of classes for the lag distance.

Usage

```
codisp(x, y, coords, nclass = 13)
```

Arguments

`x`	an n-dimensional vector of data values.
`y`	an n-dimensional vector of data values.
`coords`	an `n-by-2` matrix containing coordinates of the `n` data locations in each row.
`nclass`	a single number giving the number of cells for the codispersion coefficient. The default is 13. If this argument is `NULL` Sturges' formula is used.

Details

The procedure computes the codispersion coefficient for two spatial sequences defined on general (non-rectangular) grids. First, a given number of bins are constructed for the lag distance. Then the codispersion is computed for each bin.

Value

A list with class `"codisp"` containing the following components:

`coef`	a vector of size `nclass` containing the values of the codispersion coefficient.
`upper.bounds`	upper bounds of the intervals constructed to compute the codispersion coefficient.
`card`	number of elements in each interval generated to compute the codispersion coefficient.

The function `plot` can be used to obtain a graph of the codispersion coefficient versus the lag distance.

References

Matheron, G. (1965), *Les Variables Regionalisees et leur Estimation*. Masson, Paris.

Rukhin, A., Vallejos, R. (2008), Codispersion coefficient for spatial and temporal series. *Statistics and Probability Letters* **78**, 1290–1300.

Vallejos, R. (2008). Assessing the association between two spatial or temporal sequences. *Journal of Applied Statistics* **35**, 1323–1343.

Examples

```
# Murray Smelter site dataset
data(murray)

# defining the arsenic (As) and lead (Pb) variables from the murray dataset
x <- murray$As
y <- murray$Pb

# extracting the coordinates from Murray dataset
coords <- murray[c("xpos","ypos")]

# computing the codispersion coefficient
z <- codisp(x, y, coords)
z

## plotting the codispersion coefficient vs. the lag distance
plot(z)

# Comovement between two time series representing the monthly deaths
# from bronchitis, emphysema and asthma in the UK for 1974-1979
x <- mdeaths
y <- fdeaths
coords <- cbind(1:72, rep(1,72))
z <- codisp(x, y, coords)

# plotting codispersion and cross-correlation functions
par(mfrow = c(1,2))
ccf(x, y, ylab = "cross-correlation", max.lag = 20)
plot(z)
```

`cor.spatial` *Tjostheim's Coefficient*

Description

Computes Tjostheim's coefficient for two spatial sequences observed over the same locations on the plane.

Usage

```
cor.spatial(x, y, coords)
```

Arguments

`x`	an n-dimensional vector of data values.
`y`	an n-dimensional vector of data values.
`coords`	an `n-by-2` matrix containing coordinates of the `n` data locations in each row.

Details

The implemented technique is a nonparametric coefficient that summarizes the association between two spatial variables. This coefficient was first introduced by Tjostheim (1978) and later generalized by Hubert and Golledge (1992). The computation of the coefficient is based on the construction of ranks associated to suitable modifications of the coordinates. Tjostheim's coefficient is a variant of the correlation coefficient (`cor`) to be used in a spatial statistics context.

Value

Tjostheim's coefficient. The variance is returned as the attribute `"variance"`.

References

Tjostheim, D., (1978), A measure of association for spatial variables. *Biometrika* **65**, 109–114.

Hubert, L., Golledge, R.G., (1982), Measuring association between spatially defined variables: Tjostheim's coefficient index and some extensions. *Geographical Analysis* **14**, 273–278.

Examples

```
# Murray Smelter site dataset
data(murray)

# defining the arsenic (As) and lead (Pb) variables from the murray dataset
x <- murray$As
y <- murray$Pb

# extracting the coordinates from Murray dataset
coords <- murray[c("xpos","ypos")]

# computing Tjostheim's coefficient
z <- cor.spatial(x, y, coords)
z
```

CQ *Codispersion based similarity index*

Description

This function computes a similarity index (CQ) based on the codispersion coefficient.

Usage

```
CQ(x, y, h = c(0,1), eps = c(0.01, 0.03), L = 255)
```

Arguments

`x`	reference image matrix (grayscale)
`y`	distorted image matrix (grayscale)
`h`	2-dimensional vector of the spatial lag. Default value is `h = c(0,1)`.
`eps`	rescaling constants, by default `eps = c(0.01,0.03)`
`L`	dynamic range of the images, by default `L = 255`.

Value

A list containing the following components:

`CQ`	codispersion based similarity index between images `x` and `y`.
`direction`	vector of spatial lag.
`comps`	components of `CQ`, that is `luminance`, `contrast` and `codispersion`
`stats`	sample statistics (means, variances and covariance) for each image.
`speed`	Running time taken by the procedure.

References

Ojeda, S.M., Lamberti, P.W., Vallejos, R. (2012). Measure of similarity between images based on the codispersion coefficient. *Journal of Electronic Imaging* **21**, 023019.

Vallejos, R., Mancilla, D., Acosta, J. (2016). Image similarity assessment based on coefficients of spatial association. *Journal of Mathematical Imaging and Vision* **56**, 77-98.

Examples

```
data(texmos2)

y <- imnoise(texmos2, type = "gaussian")
plot(as.raster(y))
o <- CQ(texmos2, y, h = c(0,1))
o

y <- imnoise(texmos2, type = "speckle")
plot(as.raster(y))
o <- CQ(texmos2, y, h = c(0,1))
o
```

denoise *Remove noise from an image*

Description

This function removes noise from an input image.

Usage

```
denoise(img, type = "Lee", looks = 1, damping = 1)
```

Arguments

`img`	input grayscale image matrix.
`type`	character string, specifying the type of filter: `"median"`, `"Lee"`, `"enhanced"` (enhanced Lee filter), `"Kuan"` and `"Nathan"`.
`looks`	specifies the equivalent (or effective) number of looks used to estimate noise variance, and it effectively controls the amount of smoothing applied to the image by the filter. A smaller value leads to more smoothing; a larger value preserves more distinct image features. The default value is `looks = 1`.
`damping`	specifies the extent of exponential damping effect on filtering, by default `damping = 1`.

Details

The median filter, in which each pixel is replaced by the median of nearby values is suitable to remove additive noise from an image.

The Lee filter reduces the speckle noise by applying a spatial filter to each pixel in an image, which filters the data based on local statistics calculated within a square window. The value of the center pixel is replaced by a value calculated using the neighboring pixels. Use the Lee filter to smooth speckled data that has a multiplicative component.

The Enhanced Lee filter is a refined version of the Lee filter, reducing the speckle noise effectively by preserving image sharpness and detail. Use the Enhanced Lee filter to reduce speckle while preserving texture information.

The Kuan filter follows a similar filtering process to the Lee filter in reducing speckle noise. This filter also applies a spatial filter to each pixel in an image, filtering the data based on local statistics of the centered pixel value that is calculated using the neighboring pixels.

Nathan filter is a particular case of the Kuan filter, obtained by puting `looks = 1`, and is thus applicable to 1-look SAR images only.

The size of the pixel window used to each filter is 3-by-3.

Value

Filtered image, returned as a numeric matrix. It allows a better image interpretation. The `denoise` function clips output pixel values to the range $[0, 1]$ after removing the noise.

References

Lee, J.S. (1980). Digital image enhancement and noise filtering by use of local statistics. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-2, 165-168.

Examples

```
data(texmos2)
x <- imnoise(texmos2, type = "saltnpepper", epsilon = 0.10)
plot(as.raster(x))

y <- denoise(x, type = "median")
plot(as.raster(y))

x <- imnoise(texmos2, type = "speckle")
plot(as.raster(x))

y <- denoise(x, type = "Lee")
plot(as.raster(y))
```

---

imnoise *Add noise to image*

Description

This function adds noise to an input image.

Usage

```
imnoise(img, type = "gaussian", mean = 0, sd = 0.01, epsilon = 0.05, var = 0.04,
       looks = 1)
```

Arguments

`img`	input grayscale image matrix.
`type`	character string, specifying the type of contamination: `"gaussian"` (Gaussian white/additive noise), `"saltnpepper"` (salt and pepper noise or on-off noise), `"speckle"` (uniform multiplicative noise) and `"gamma"` (gamma multiplicative noise).
`mean`	mean for the Gaussian noise, default value is `mean = 0`.
`sd`	standard deviation for the Gaussian noise, default value is `sd = 0.01`.
`epsilon`	contamination percentage for the salt and pepper noise with default noise density `0.05`. This affects approximately `epsilon`% of pixels.
`var`	variance of uniform multiplicative noise using the equation $noise = img + unif * img$, with $unif$ is uniformly distributed with mean 0 and variance `var`. Default value is `var = 0.04`.
`looks`	parameter of gamma multiplicative noise. The default value is `looks = 1`.

Value

Noisy image, returned as a numeric matrix. The `imnoise` function clips output pixel values to the range $[0, 1]$ after adding noise.

Examples

```
data(texmos2)
x <- imnoise(texmos2, type = "saltnpepper", epsilon = 0.10)
plot(as.raster(x))

y <- imnoise(texmos2, type = "speckle")
plot(as.raster(y))

z <- imnoise(texmos2, type = "gamma", looks = 4)
plot(as.raster(z))
```

`modified.Ftest` *Modified F test*

Description

Performs a modified version of the F test to assess the multiple correlation between one spatial processes and several others.

Usage

```
modified.Ftest(x, y, coords, nclass = 13)
```

Arguments

`x`	an `n-by-q` matrix of data values.
`y`	an `n`-dimensional vector of data values.
`coords`	an `n-by-2` matrix containing coordinates of the `n` data locations in each row.
`nclass`	a single number giving the number of cells for Moran's index. The default is 13. If this argument is `NULL` Sturges' formula us used.

Details

The methodology implemented is a modified F test for assessing the multiple correlation between one spatial process and several others. The test is based on corrections of the multiple correlation coefficient between the two spatially correlated sequences and required the estimation of an effective sample size. This factor takes into account the spatial association of both processes.

Value

A list with class "mod.Ftest" containing the following components:

- `corr` the sample correlation coefficient.
- `ESS` the estimated effective sample size.
- `Fstat` the value of the (unscaled) F-statistic.
- `df1,df2` degrees of freedom for the F-statistic.
- `p.value` the p-value for the test.
- `upper.bounds` upper bounds of the intervals constructed to compute Moran's I.
- `card` number of elements in each interval generated to compute Moran's I.
- `imoran` a matrix containing Moran's index for each interval associated with the response and predicted variables.

The generic functions `print` and `summary` are used to obtain and print additional details about the modified F test.

References

Dutilleul, P., Pelletier, B., Alpargu, G. (2008). Modified F tests for assessing the multiple correlation between one spatial process and several others. *Journal of Statistical Planning and Inference* **138**, 1402–1415.

Examples

```
# The Pinus Radiata data set
data(radiata)

# defining the response and predictor variables from the radiata data set
y <- radiata$height
x <- radiata[c("basal","altitude","slope")]

# extracting the coordinates from the radiata data set
coords <- radiata[c("xpos","ypos")]

# computing the modified F-test of spatial association
z <- modified.Ftest(x, y, coords)
z

# display the upper bounds, cardinality and the computed Moran's index
summary(z)
```

`modified.ttest` *Modified t test*

Description

Performs a modified version of the *t* test to assess the correlation between two spatial processes.

Usage

```
modified.ttest(x, y, coords, nclass = 13)
```

Arguments

`x`	an `n`-dimensional vector of data values.
`y`	an `n`-dimensional vector of data values.
`coords`	an `n-by-2` matrix containing coordinates of the `n` data locations in each row.
`nclass`	a single number giving the number of cells for Moran's index. The default is 13. If this argument is `NULL` Sturges' formula us used.

Details

The methodology implemented is a modified *t* test of spatial association based on the work of Clifford and Richardson (1989). The test is based on corrections of the sample correlation coefficient between the two spatially correlated sequences and required the estimation of an effective sample size. This factor takes into account the spatial association of both processes.

Value

A list with class `"mod.ttest"` containing the following components:

`corr`	the sample correlation coefficient.
`ESS`	the estimated effective sample size.
`Fstat`	the value of the (unscaled) *F*-statistic.
`dof`	the estimated degrees of freedom for the *F*-statistic.
`p.value`	the *p*-value for the test.
`upper.bounds`	upper bounds of the intervals constructed to compute Moran's *I*.
`card`	number of elements in each interval generated to compute Moran's *I*.
`imoran`	a matrix containing Moran's index for each interval associated with both variables.

The generic functions `print` and `summary` are used to obtain and print additional details about the modified *t* test.

References

Clifford, P., Richardson, S., Hemon, D. (1989). Assessing the significance of the correlation between two spatial processes. *Biometrics* **45**, 123–134.

Dutilleul, P. (1993). Modifying the *t* test for assessing the correlation between two spatial processes. *Biometrics* **49**, 305–314.

Examples

```
# Murray Smelter site dataset
data(murray)

# defining the arsenic (As) and lead (Pb) variables from the murray dataset
x <- murray$As
y <- murray$Pb

# extracting the coordinates from Murray dataset
coords <- murray[c("xpos","ypos")]

# computing the modified t-test of spatial association
z <- modified.ttest(x, y, coords)
z

# display the upper bounds, cardinality and the computed Moran's index
summary(z)
```

---

murray — *The Murray smelter site dataset*

Description

The dataset consists of soil samples collected in and around the vacant, industrially contaminated, Murray smelter site (Utah, USA). This area was polluted by airborne emissions and the disposal of waste slag from the smelting process. A total of 253 locations were included in the study, and soil samples were taken from each location. Each georeferenced sample point is a pool composite of four closely adjacent soil samples in which the concentration of the heavy metals arsenic (As) and lead (Pb) was determined.

Usage

```
data(murray)
```

Format

A data frame with 253 observations on the following 5 variables.

As arsenic concentrations measurements.

Pb lead concentrations measurements.

xpos x-coordinates.

ypos y-coordinates.

quad a factor where numbers indicate different sub-regions within the area.

Source

Griffith, D., Paelinck, J.H.P. (2011). *Non-Standard Spatial Statistics*. Springer, New York.

`normalize` *Normalization for a matrix*

Description

This function normalizes an image matrix so that the minimum value is 0 and the maximum value is 1.

Usage

```
normalize(img)
```

Arguments

`img`	target image

Value

Image matrix in which minimum value is 0 and maximum value is 1.

Examples

```
data(twelve)
x <- RGB2gray(twelve, method = "RMY")
x <- normalize(x)
plot(as.raster(x))
```

`radiata` *The Pinus Radiata dataset*

Description

Pinus radiata is one of the mostly widely planted species in Chile and is planted in a wide array of soil types and regional climates. The plots were located in the Escuadron sector, south of Concepcion, in the southern portion of Chile and has an area of 1244.43 hectares.

Usage

```
data(radiata)
```

Format

A data frame with 468 observations on the following 6 variables.

xpos x-coordinates.

ypos y-coordinates.

basal basal area measurements.

height dominant tree height.

altitude altitude in meters.

slope slope of the terrain plot.

Source

Cuevas, F., Porcu, E., Vallejos, R. (2013). Study of spatial relationships between two sets of variables: A nonparametric approach. *Journal of Nonparametric Statistics* **25**, 695–714.

Vallejos, R., Osorio, F., Bevilacqua, M. (2020+). *Spatial Relationships Between Two Georeferenced Variables: with Applications in R*. Springer, New York.

`RGB2gray` *Convert RGB image or colormap to grayscale image*

Description

This function converts color image to gray image.

Usage

```
RGB2gray(img, method = "average", weights = NULL)
```

Arguments

`img`	target image, specified as an `nrow`-by-`ncol`-by-3 numeric array.
`method`	character, procedure for converting color to grayscale. Available methods are `"average"`, `"BT240"`, `"brighter"` (or maximum decomposition), `"darker"` (or minimu decomposition), `"ITU"` (or BT.709), `"lightness"` (or desaturation), `"LUMA"` (or BT.601), `"RMY"` and `"weighted"` (user provided).
`weights`	weights for red (R), green (G), and blue (B) channels. Required if `method = "weighted"`.

Value

Grayscale image, returned as an `nrow`-by-`ncol` numeric matrix with values in the range $[0, 1]$.

`RGB2gray` converts RGB values to grayscale values by forming a weighted sum of the R, G, and B channels.

Examples

```
data(twelve)
par(pty = "s", mfrow = c(1,3))
plot(as.raster(twelve)) # in RGB
title(main = "original", font.main = 1)

x <- RGB2gray(twelve, method = "RMY")
plot(as.raster(x)) # in grayscale
title(main = "RMY", font.main = 1)

x <- RGB2gray(twelve, method = "ITU")
plot(as.raster(x)) # OMG! 12 is gone...
title(main = "ITU", font.main = 1)
```

`SSIM` *Structural similarity index*

Description

This function computes the structural similarity index (SSIM) proposed by Wang et al. (2004).

Usage

```
SSIM(x, y, alpha = 1, beta = 1, gamma = 1, eps = c(0.01, 0.03), L = 255)
```

Arguments

`x`	reference image matrix (grayscale)
`y`	distorted image matrix (grayscale)
`alpha`	weight associated with luminance, default value is `alpha = 1`.
`beta`	weight associated with contrast, default value is `beta = 1`.
`gamma`	weight associated with structure, default value is `gamma = 1`.
`eps`	rescaling constants, by default `eps = c(0.01,0.03)`
`L`	dynamic range of the images, by default `L = 255`.

Value

A list containing the following components:

`SSIM`	structural similarity index between images `x` and `y`.
`coefficients`	weights (`alpha`, `beta`, `gamma`) associated with each component of `SSIM`
`comps`	components of `SSIM`, that is `luminance`, `contrast` and `structure`
`stats`	sample statistics (means, variances and covariance) for each image.
`speed`	Running time taken by the procedure.

References

Wang, Z., Bovik, A.C. (2002). A universal image quality index. *IEEE Signal Processing Letters* **9**, 81-84.

Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing* **13**, 600-612.

Examples

```
data(texmos2)

y <- imnoise(texmos2, type = "gaussian")
plot(as.raster(y))
o <- SSIM(texmos2, y)
o

y <- imnoise(texmos2, type = "speckle")
plot(as.raster(y))
o <- SSIM(texmos2, y)
o
```

---

texmos2 — *USC texture mosaic number 2*

Description

Gray-level texture map with information about mosaic composed of eight different texture samples taken from the Brodatz texture book, available from USC-SIPI image database.

Usage

```
data(texmos2)
```

Format

A grayscale matrix of size 512-by-512.

Source

Brodatz, P. (1966). *Textures: A Photographic Album for Artist and Designers*. Dover Publications, New York.

USC-SIPI image database, URL: http://sipi.usc.edu/database/

`twelve` *Ishihara plate number 1*

Description

Ishihara plate number 1, with the numeral `'12'` designed to be visible by all persons.

Usage

```
data(twelve)
```

Format

An array of 380-by-380-by-3 representing a RGB image.

Index