Package 'ShapePattern'

May 7, 2026

Type Package

Title Tools for Analyzing Shapes and Patterns

Version 3.1.0

Maintainer Tarmo K. Remmel <remmelt@yorku.ca>

Description

This is an evolving and growing collection of tools for the quantification, assessment, and comparison of shape and pattern. This collection provides tools for: (1) the spatial decomposition of planar shapes using 'ShrinkShape' to incrementally shrink shapes to extinction while computing area, perimeter, and number of parts at each iteration of shrinking; the spectra of results are returned in graphic and tabular formats (Remmel 2015) <doi:10.1111/cag.12222>, (2) simulating landscape patterns, (3) provision of tools for estimating composition and configuration parameters from a categorical (binary) landscape map (grid) and then simulates a selected number of statistically similar landscapes. Class-focused pattern metrics are computed for each simulated map to produce empirical distributions against which statistical comparisons can be made. The code permits the analysis of single maps or pairs of maps (Remmel and Fortin 2013) <doi:10.1007/s10980-013-9905-x>, (4) counting the number of each first-order pattern element and converting that information into both frequency and empirical probability vectors (Remmel 2020) <doi:10.3390/e22040420>, and (5) computing the porosity of raster patches <doi:10.3390/su10103413>. NOTE: This is a consolidation of existing packages ('PatternClass', 'ShapePattern') to begin warehousing all shape and pattern code in a common package. Additional utility tools for handling data are provided and this package will be added to as more tools are created, cleaned-up, and documented. Note that all future developments will appear in this package and that 'PatternClass' will eventually be archived.

License GPL-3

LazyData true

LazyDataCompression xz

Depends R (>= 4.3.0), sp, igraph, terra

Imports landscapemetrics, raster

NeedsCompilation no

Author Tarmo K. Remmel [aut, cre] (ORCID:
<https://orcid.org/0000-0001-6251-876X>),
Marie-Josee Fortin [ctb],
Ferenc Csillag [ctb],
Sandor Kabos [ctb]

Repository CRAN
Date/Publication 2024-12-10 04:40:01 UTC

Contents

`ShapePattern-package` *Tools for Analyzing Shapes and Patterns*

Description

This is an evolving and growing collection of tools for the quantification, assessment, and comparison of shape and pattern. This collection provides tools for: (1) the spatial decomposition of planar shapes using 'ShrinkShape' to incrementally shrink shapes to extinction while computing area, perimeter, and number of parts at each iteration of shrinking; the spectra of results are returned in graphic and tabular formats (Remmel 2015) <doi:10.1111/cag.12222>, (2) simulating landscape patterns, (3) provision of tools for estimating composition and configuration parameters from a categorical (binary) landscape map (grid) and then simulates a selected number of statistically similar landscapes. Class-focused pattern metrics are computed for each simulated map to produce empirical distributions against which statistical comparisons can be made. The code permits the analysis of single maps or pairs of maps (Remmel and Fortin 2013) <doi:10.1007/s10980-013-9905-x>, (4) counting the number of each first-order pattern element and converting that information into both frequency and empirical probability vectors (Remmel 2020) <doi:10.3390/e22040420>, and (5) computing the porosity of raster patches <doi:10.3390/su10103413>. NOTE: This is a consolidation of existing packages ('PatternClass', 'ShapePattern') to begin warehousing all shape and pattern code in a common package. Additional utility tools for handling data are provided and this package will be added to as more tools are created, cleaned-up, and documented. Note that all future developments will appear in this package and that 'PatternClass' will eventually be archived.

Details

The DESCRIPTION file:

Type:	Package
Package:	ShapePattern
Title:	Tools for Analyzing Shapes and Patterns
Version:	3.1.0
Authors@R:	c(person("Tarmo K.", "Remmel", email = "remmelt@yorku.ca", role = c("aut", "cre"), comment =
Maintainer:	Tarmo K. Remmel <remmelt@yorku.ca>
Description:	This is an evolving and growing collection of tools for the quantification, assessment, and comparis
License:	GPL-3
LazyData:	true
LazyDataCompression:	xz
Depends:	R (>= 4.3.0), sp, igraph, terra
Imports:	landscapemetrics, raster
NeedsCompilation:	no
Packaged:	2024-12-09 02:34:55 UTC; tarmo
Author:	Tarmo K. Remmel [aut, cre] (<https://orcid.org/0000-0001-6251-876X>), Marie-Josee Fortin [ctb],
Repository:	CRAN
Date/Publication:	2024-12-09 07:20:09 UTC

Index of help topics:

	it were a raster landscape.
patternbits	Decompose a binary landscape (grid) to count its pattern elements
porosity	Compute the porosity of raster zones (patches)
singlemap	Process a single binary map
singleplotter	Depict the empirical (null) distribution for class-focused pattern metrics.
ssr	Decompose a planar shape (polygon) to produce area, perimeter, and number of parts spectra
surfplot	Produce graphic plots providing class-focused pattern metric context for a landcape map
wi	An internal utility that helps with the Whittle estimation.
wtest.run	Whittle estimation for binary map

This package combines the legacy functionality of package PatternClass and package ShapePattern and sets the environment for continuing to contribute to shape and pattern analyses. Eventually, package PatternClass will be archived (please point to this package to maintain functionality). Functions are provided to perform (1) ShrinkShape, a spatial decomposition by iterative shrinking of planar shapes (ploygons); (2) PatternClass, a tool for computing and visualizing statistical differences between class-level landscape maps; (3) Patternbits, a means for producing probability distributions for the most elemental hyper-local pattern elements in a binary landscape map and using Kullback-Leibler divergence to assess similarity and differnece between two such distributions; (4) CAR 2D simulation of stationary and isotropic landscape patterns, and (5) a tool for computing raster zone (patch) porosity.

Author(s)

Tarmo K. Remmel [aut, cre] (<https://orcid.org/0000-0001-6251-876X>), Marie-Josee Fortin [ctb], Ferenc Csillag [ctb], Sandor Kabos [ctb]

Maintainer: Tarmo K. Remmel <remmelt@yorku.ca>

References

1. Polygon shape analysis:

Remmel, T.K. 2015. ShrinkShape2: a FOSS toolbox for computing rotation-invariant shape spectra for characterizing and comparing polygons. The Canadian Geographer 59(4):532-547.

2. Comparing binary landscapes:

Remmel, T.K. and M.-J. Fortin. 2016. What constitutes a significant difference in landscape pattern? (using R). In Gergel, S.E. and M.G. Turner. Learning landscape ecology: concepts and techniques for a sustainable world (2nd ed.). New York: Springer. http://sarahgergel.net/lel/learning-landscape-ecology/

Remmel, T.K. and M.-J. Fortin. 2013. Categorical class map patterns: characterization and comparison. Landscape Ecology. DOI: 10.1007/s/10980-013-9905-x.

Remmel, T.K. and F. Csillag. 2003. When are two landscape pattern indices significantly different? Journal of Geographical Systems 5(4):331-351.

3. For computing pattern elements:

Remmel, T.K. 2020. Distributions of hyper-local configuration elements to characterize, compare, and assess landscape-level spatial patterns. Entropy 22(4):420.

4. Simulating Landscapes:

Remmel, T.K. and F. Csillag. When are two landscape pattern indices significantly different? Journal of Geographical Systems 5(4):331-351.

5. For computing raster patch porosity:

Remmel, T.K. 2018. An incremental and philosophically different approach to measuring raster patch porosity. Sustainability 10:3413.

See Also

It would be prudent to also observe the `landscapemetrics` package.

Examples

```
# Specific examples are provided on the individual function manual pages.
# This package provides functions for different areas of analysis:
#
# 1. ShrinkShape
# ssr()
#
# 2. PatternClass (legacy package moved here)
# The order of function calls should be: singlemap() > singleplotter()
# or
# singlemap(), singlemap() > doubleplotter()
#
# 3. Patternbits
# patternbits() and KLPQ()
#
# 4. Conditional AutoRegressive 2D pattern simulator
# CARsimu()
```

`build.lut` *Bias correction lookup table (LUT) builder based on simulations.*

Description

This function needs to be run only once, at the beginning of analysis, to produce a lookup table of spatial autocorrelation (rho) bias correction factors. Since this process takes a long time, due to the need for simulating numerous replicates (generally 50 or more) landscapes at each pair-wise combination of composition and configuration parameterization, we provide a lookup table with the package that is used by the `singlemap` function internally. Currently, the implementation is limited to 64x64 pixel maps; however, future developments will expand on this limitation and thus the build.lut function is provided for those that may wish to fast-track the construction of scaled lookup tables for analyses of landscapes of different size. Bias corrections are achieved by comparing Whittle estimates of spatial autocorrelation on binary versus continuous surfaces with identical parameterizations.

Usage

```
build.lut(LEVEL = 6, REPSIM = 5, RAJZ = FALSE, CIM = "", ENV="data")
```

Arguments

LEVEL	A power (n) of base 2 that controls the size of the simulated maps used to process the bias correction surface: 2^n x 2^n cells.
REPSIM	The number of simulation replicates at each combination of composition and configuration.
RAJZ	A Boolean flag that when TRUE will draw the result.
CIM	An option that permits the naming of image objects. This is provided for potential future expansion.
ENV	The name of the environment where temporary objects are stored. The default is data.

Details

This does not need to be run if you have downloaded and installed the `PatternClass` package and are using it to analyze 64x64 pixel image subsets as proposed. The `singlemap` function will call on the provided internal lookup table and thus save on substantial processing time. The current implementation provides a lookup table with bias correction values based on the mean of 50 replicate simulations for each pair-wise combination of composition and configuration parameterization. hus, the 9 levels of composition, 11 levels of spatial autocorrelation, and 50 replicates, means that 9*11*50 = 4950 landscapes are simulated and used to produce the bias correction lookup table that is provided with the package.

Value

The result is a lookuptable (LUT) that is a 9x11 matrix of values used by singlemap().

Note

The current implementation is for 64x64 image subsets. Future developments will expand this extent limitation. However, our work has shown that this extent provides a reasonable compromise between statistical power and computing efficiency.

Author(s)

Tarmo K. Remmel

References

Remmel, T.K. and M.-J. Fortin. 2013. Categorical class map patterns: characterization and comparison. Landscape Ecology. DOI: 10.1007/s/10980-013-9905-x.

Remmel, T.K. and M.-J. Fortin. What constitutes a significant difference in landscape pattern? (using R). 2016. In Gergel, S.E. and M.G. Turner. Learning landscape ecology: concepts and techniques for a sustainable world (2nd ed.). New York: Springer.

See Also

See Also singlemap

Examples

```
build.lut(LEVEL = 6, REPSIM = 5, RAJZ = FALSE, CIM = "", ENV="data")
```

---

buildsurfs *Build an array of surfaces across compositional and configuration analysis space.*

Description

This function iteratively cycles through combinations of 9 binary proportion intervals and 11 spatial autocorrelation intervals and at each simulates a specified number of statistically similar binary landscape patterns. Each landscape is processed by landscapemetrics to produce empirical (null) distributions for class-focused pattern metrics. These surfaces can then be plotted or assessed for their mean values at any point (along with their variability).

Usage

```
buildsurfs(reps = 1000, verbose = TRUE)
```

Arguments

reps	An argument that defines the number of simulated maps that are produced for each combination of composition and configuration across the parameter space. The default value, 1000, is appropriate, but will take a long time to compute. The example below uses 2 replicates to demonstrate the process; however, the result should be used only for demonstrating that the code is working. Using 1000 simulations ensures that a valid empirical (null) distribution is constructed at each point on this surface.
verbose	When this argument is set to TRUE, additional information is provided during processing on screen. When FALSE, the code runs in relatively quiet mode, with little information provided on screen. When set to TRUE, it is easier to assess the time required to complete the processing.

Details

This is a simulation-intensive procedure and takes a long time. It is only necessary for advanced users who wish to assess the infulence of composition and configuration (jointly) on specific pattern metrics. Once this function is run and the surfaces object is created, all further analysis can be performed quickly. This lookup reference set requires a lot of up-front processing, but simplifies all future analyses. The produced reference set is also necessary for asking specific questions regarding the expectation and variability of pattern metrics within the context of composition and configuration values (i.e., distributions at all pair-wise combinations of compostion and configuration parameterizations).

Value

The returned value is an array called surfaces having dimensions [metrics, proportions, configurations, reps].

Note

The enclosed code is currently limited to 64x64 pixel subsets; however, future developments will expand the applicability to incorporate other dimensions. Since this code requires the simulation and storage of numerous landscape (raster) maps, the processing time (up-front) can be quite time consuming; however, once complete, plotting and analysis of mean and variance for any combination of composition and configuration and class-focused pattern metric are nearly instantaneous.

Author(s)

Tarmo K. Remmel

References

Remmel, T.K. and M.-J. Fortin. What constitutes a significant difference in landscape pattern? (using R). 2016. In Gergel, S.E. and M.G. Turner. Learning landscape ecology: concepts and techniques for a sustainable world (2nd ed.). New York: Springer.

See Also

See Also singlemap, singleplotter, and doubleplotter

Examples

```
surfaces <- data$surfaces
# COMPUTE SURFACES
surfaceexample <- buildsurfs(reps = 2, verbose = TRUE)
# COMPUTE MEDIAN SURFACE FOR A SINGLE METRIC
tempmed <- apply(surfaces[9,,,], MARGIN=c(1,2), median)
# PRODUCE A PERSPECTIVE PLOT OF MEDIAN VALUES RESULTS FOR THE SELECTED METRIC
persp(tempmed, ticktype="detailed", cex.axis=0.7, zlab="Metric",
ylab="Proportion", xlab="Rho", theta=-45)
# COMPUTE VARIANCE SURFACE FOR A SINGLE METRIC
tempvar <- apply(surfaces[9,,,], MARGIN=c(1,2), var)
# PRODUCE A PERSPECTIVE PLOT OF VARIANCE VALUES RESULTS FOR THE SELECTED METRIC
persp(tempvar, ticktype="detailed", cex.axis=0.7, zlab="Metric",
ylab="Proportion", xlab="Rho", theta=-45)
```

CARsimu *This function simulates an image using a FFT implementation of a Conditional Autoregressive (CAR) model.*

Description

A CAR landscape simulator for isotropic conditions.

Usage

```
CARsimu(LEVEL = 6, rho = 0.2499, row2 = 0, col2 = 0, rc1 = 0, cr1 = 0,
maindi = 1, rajz = TRUE)
```

Arguments

LEVEL	This is the power (n) with base 2 that defines the image size: 2^n x 2^n.
rho	This is the spatial autocorrelation parameter.
row2	For later implementation - 2nd order neighbour spatial autocorrelation parameter (rows).
col2	For later implementarion - 2nd order neighbour spatial autocorrelation parameter (columns).
rc1	For later implementation - diagonal spatial autocorreltion parameter.
cr1	For later implementation - diagonal spatial autocorrelation parameter.
maindi	For later implementation.
rajz	When rajz = TRUE, the simulated map will be drawn.

Details

This function can actually be parameterized in many ways, however, for isotropic landscapes, only the rho parameter is to be used. Additional parameterizations, although possible, are not permitted with the use of the other functions provided with the PatternClass package

Value

The output is a simulated map on a grid (provided as a matrix).

Note

All simulated landscapes are stationary and isotropic, thus extent cannot be larger than what would be expected to generally adhere to these constraints. The parameterization requires that the sum of all spatial autocorrelation parameters sum to < 0.5. This is because the simulator is isotropic. A perfect 1.0 spatial autocorrelation parameter can be approximated by 0.499999 (infinitely repeating), due to computational limits. Outputs of this function can be saved to objects and used as continuous simulated landscape maps. To obtain categorical maps, these continuous maps can be sliced based on their distributions, as necessary.

Author(s)

Ferenc (Ferko) Csillag and Sandor Kabos; modified by Tarmo K. Remmel

References

Remmel, T.K. and F. Csillag. 2003. When are two landscape pattern indices significantly different? Journal of Geographical Systems 5(4):331-351.

Remmel, T.K. and M.-J. Fortin. 2013. Categorical class map patterns: characterization and comparison. Landscape Ecology. DOI: 10.1007/s/10980-013-9905-x.

Remmel, T.K. and M.-J. Fortin. What constitutes a significant difference in landscape pattern? (using R) (Pending 2014). In Gergel, S.E. and M.G. Turner. Learning landscape ecology: concepts and techniques for a sustainable world (2nd ed.). New York: Springer.

See Also

See Also as `singlemap`

Examples

```
CARsimu(LEVEL = 6, rho = 0.2499, row2 = 0, col2 = 0, rc1 = 0, cr1 = 0,
maindi = 1, rajz = TRUE)
```

---

data *This is a generic data environment that provides some demo data and control functionality.*

Description

This is a generic data environment that provides some demo data and control functionality. for the operation of functions within this package. Two imported shapefiles are provided. One is the perimeter of a pond, the second is the same pond but with the internal islands (holes) included. There are demo images for comparing class patterns, a matrix of Whittle correction factors, and some additional interal objects that hold settings for the functions to operate properly.

Usage

```
data("data")
```

Format

The lake outlines (`p4is` and `p4no`) are in imported shapefile format; they were imported using `rgdal:readOGR`. The demo images for class-level pattern comparison (`demoimage1` and `demoimage2`) are matrix objects. DIFF50 is a matrix of Whittle correction factors (which should not be modified unless the user is certain that this needs to be done). The current factors are designed for 64x64 cell landscape subsets; corrections for other sizes can be constructed with `build.lut`; this can take a very long time! The raster 'rst' can be used for testing the `porosity` function.

Details

Details of the individual objects are best obtained using `str()`.

Source

These pond perimeters were digitized in-house by Connie Ko at York University; other materials were produced by Tarmo K. Remmel.

References

There are no references for these data elements.

Examples

```
data(data)
plot(data$p4no)
plot(data$p4is)
print(data$DIFF50)
image(data$demoimage1)
image(data$demoimage2)
plot(data$rst)
```

`doubleplotter` *A comparison of two maps based on their empirical (null) distributions for a class-focused pattern metric*

Description

This function produces boxplots comparing the expected values (empirical, null distributions) for a given class-focused pattern metric between two maps.

Usage

```
doubleplotter(data1 = data$result1,
  data2 = data$result2,
  metric = 5)
```

Arguments

`data1`	This is the result object for the first map to be compared, where the output is returned by the function `singlemap`.
`data2`	This is the result object for the second map to be compared, where the output is returned by the function `singlemap`.
`metric`	This is an integer (1-110) indicating which metric the comparison boxplots will be drawn for. A list of these metrics is given in the header of the function's source code.

Details

No additional details at this time.

Value

The output is a double boxplot drawn on the graphics device.

Note

No additional notes at this time.

Author(s)

Tarmo K. Remmel

References

No references at this time.

See Also

See Also singlemap, singleplotter, and imaks.

Examples

```
# EXAMPLE USES PREVIOUSLY PRODUCED RESULTS TO SPEED-UP THE EXAMPLE,
# BUT THE EXAMPLE FROM singlemap() SHOULD BE CALLED FIRST, ONCE FOR EACH
# MAP TO BE COMPARED (FOR EXAMPLE)
#result1 <- singlemap(IMG = data$demoimage1, VERBOSE = TRUE, reps = 5, LEVEL=6)
#result2 <- singlemap(IMG = data$demoimage2, VERBOSE = TRUE, reps = 5, LEVEL=6)
doubleplotter(data1 = data$result1, data2 = data$result2, metric = 5)
```

---

findcol *An internal utility that identifies that identifies the appropriate column of the Whittle correction matrix.*

Description

This is an internal utility function for identifying a column index value in the Whittle correction matrix and returns it to the calling function for use in further processing. This function is not intended for use directly by users.

Usage

```
findcol(prop = 0.32, DIFFMAT=data$DIFF50, VERBOSE=FALSE)
```

Arguments

prop	This is a numeric argument providing the proportion of white pixels in the input binary map.
DIFFMAT	This is a matrix object containing the Whittle estimations (corrections) for spatial autocorrelation and generally constructed by build.lut.
VERBOSE	A Boolean argument that indicates whether the function should run in verbose mode or not.

Details

This tool is only used internally.

Value

The result is a an index to be used as a pointer.

Note

This function is not to be used directly by users of this package.

Author(s)

Tarmo K. Remmel

References

No references currently.

See Also

Currently none.

Examples

```
# No example.
```

---

findrow *An internal utility that identifies that identifies the appropriate row of the Whittle correction matrix.*

Description

This is an internal utility function for identifying a row index value in the Whittle correction matrix and returns it to the calling function for use in further processing. This function is not intended for use directly by users.

Usage

```
findrow(autocorr = 0.2, DIFFMAT=data$DIFF50, VERBOSE=FALSE)
```

Arguments

autocorr	This is a numeric argument providing the level of spatial autocorrelation.
DIFFMAT	This is a matrix object containing the Whittle estimations (corrections) for spatial autocorrelation and generally constructed by build.lut.
VERBOSE	A Boolean argument that indicates whether the function should run in verbose mode or not.

Details

This tool is only used internally.

Value

The result is a an index to be used as a pointer.

Note

This function is not to be used directly by users of this package.

Author(s)

Tarmo K. Remmel

References

No references currently.

See Also

Currently none.

Examples

```
# No example.
```

imaks *Draws a matrix in the proper orientation, as it it were a raster landscape.*

Description

A simple drawing function for matrices that are actually representing raster landcape maps. This function also controls the use of colour for nominal maps. This is a generic function for drawing any raster image that should not be drawn as a matrix (where the origin is not at the lower-left corner, but rather the upper-left corner. This function draws a matrix as a map using the proper positioning of the origin and gridded values.

Usage

```
imaks(BE = data$demoimage1, numcol = NULL, LENG = 4, colour = FALSE)
```

Arguments

BE	The input landscape map as a matrix object.
numcol	A numeric value indicating the number of total colours on the map.
LENG	An argument that is depreciating. Do not adjust this value.
colour	Should a special colour scheme be applied (TRUE) or not (FALSE)

Details

In a future release, this function may become obsolete as we migrate to using the raster library and raster objects.

Value

The result is a graphic plot of a raster landscape as read from a matrix representation.

Note

This function was originally released in the package: hdeco

Author(s)

Sandor Kabos (modified by Tarmo K. Remmel)

References

No references currently.

See Also

Currently none.

Examples

```
data(demoimage1)
imaks(BE = data$demoimage1, numcol = NULL, LENG = 4, colour = FALSE)
```

KLPQ *Computes and returns the Kullback-Leibler divergence between two probability distributions*

Description

Given two probability distributions (vectors) of the same length, this function computes the Kullback-Leibler divergence (relative entropy) between them. If any entries in the distribution are 0, then the argument JITTER can be used to add a tiny offset.

Usage

```
KLPQ(P = c(0.36, 0.48, 0.16), Q = c(0.333, 0.333, 0.333), JITTER = 0.0)
```

Arguments

P	A first probability distribution vector.
Q	A second probability distribution vector.
JITTER	An optional tiny value to be added to the probabilities to avoid non-zero entries (e.g., 0.000000001).

Value

The function returns a numeric value that is the Kullback-Leibler divergence. If this value is 0 (zero), then the two distributions are identical.

Note

Used to provide a uni-directional divergence measure. Note that P||Q does not necessarily equal Q||P.

Author(s)

Tarmo K. Remmel

References

None currently.

See Also

See Also patternbits

Examples

```
KLPQ(P = c(0.36, 0.48, 0.16), Q = c(0.333, 0.333, 0.333), JITTER = 0.0)
```

---

patternbits *Decompose a binary landscape (grid) to count its pattern elements*

Description

Given an input binary (0,1) pattern as a .csv file or matrix object, this function shifts that pattern in each of the 4 cardinal directions by one location (cell) and then counts the frequency for each of the 32 hyper-local neighbourhood configurations (CODE). The frequencies are provided for each CODE along with their PROBABILITY. Finally an object that maps the CODE into a matrix that can be drawn as a map is also provided as part of the ouput list object. Note that the minimum number of grid rows or columns are 3 each if processing a non-toroidal surface. The minimum grid size can be [2,2], but in reality, to ensure sufficient opportunities for cases to present themselves, the total number of cells should be much greater than 32. Grids do not have to have an equal number of rows and columns and the option to select a toroidal versus non-toroidal surface are now operational too.

Usage

```
patternbits(IMGCSV="~/Desktop/Book2.csv",
            OBJ=TRUE,
            OBJname=matrix(c(1,1,0,0,0,1,1,1,0), nrow=3, ncol=3),
            OBJtxt="NAMEhere",
            DRAW=TRUE,
            VERBOSE=FALSE,
            TORUS=TRUE,
            rho=0,
            proportion=0.5)
```

Arguments

IMGCSV	This is a .csv (comma separated values) textfile format of an input binary grid. It is read into a matrix object and needs to have an equal number of rows and columns. No header is permitted.
OBJ	This is a Boolean flag that determines whether the input grid is available as on object (TRUE) or whether it will be read from a .csv file (FALSE).
OBJname	If the input data is an object, the name of that object is provided here. If OBJ is FALSE, whatever is entered here is ignored.
OBJtxt	A character string that can be used to describe the input data or to encode a batch number, simulation identifier or other code that will be included in the output list object for later reference.
DRAW	This is a Boolean flag that is used to turn the drawing function on (TRUE) or off (FALSE). It can be useful to see the intermediate shifting layers used by the algorithm. This is often used to learn how the function works, rather than in production mode.

VERBOSE	This is a Boolean flag that is used to control the display of additional state feeback during operation (TRUE), or to suppress it (FALSE). Generally, this is used in the default FALSE mode, but since it was useful during development, it has been left in for now).
TORUS	This is a Boolean flag that determines whether the data is considered to be toroidal (continuous) by having the edges wrap around to the opposite side (TRUE), or not (FALSE). If FALSE, the outer edge of result cells are omitted and the output is 2 rows and 2 columns smaller than the input. This means that an imput must have at least 3 rows and 3 columns for a non-toroidal case to not provide an error message.
rho	A numeric element that serves the purpose of passing this pattern parameter to the function, allowng it to be added to the output list object as reference. It is not actually used to control processing. This value describes the degree of spatial autocorrelation in the input object.
proportion	A numeric element that serves the purpose of passing this pattern parameter to the function, allowng it to be added to the output list object as reference. It is not actually used to control processing. This value describes the proportion of black to white cells in the input object.

Value

The output is a list object that contains PROCESSINGDATE (when the code was run), IMAGE (the name of the image, if supplied), NCELLS (the total number of cells in the grid), TORUS (Boolean indicator of whether processing as a torus or not), RHO (spatial autocorrelation value of grid), PROPORTION (proportion value of grid categories), RESULTS (output data.frame with all frequencies and probabilities), and JOINT (the output matrix of all pattern element codes). Note that C = centre, R = right, A = above, L = left, and B = below.

Note

TBA

Author(s)

Tarmo K. Remmel

References

None currently.

See Also

See Also CARsimu.

Examples

```
elements <- patternbits(OBJ=TRUE, OBJtxt="Demo")
str(elements)
```

porosity *Compute the porosity of raster zones (patches)*

Description

This function scans an input binary raster to assess zones and then computes the porosity of each unique zone.

Usage

```
porosity(IN=data$rst,
            PLOT=TRUE,
            NEIGH=4)
```

Arguments

IN	This is a binary raster input object. This raster will be clumped (i.e., like Esri's ArcGIS 'regiongroup' functionality) to produce zones of contiguous pixels. Each contiguous zone will be assessed for its porosity, or deviation from a theoretically maximally compact shape.
PLOT	This is a Boolean flag that controls whether plots will be provided as output during processing.
NEIGH	This is an integer, either 4 or 8 that controls whether a Rook's case or Queen's case neighbourhood contiguity is used to assess the zones.

Value

The returned object is a dataframe with one row for each zone, and columns: Zone, N, Jact, Jmax, and Porosity. Here Zone is the unique ID for a specific raster zone of contiguous cells, N is the nuber of cells comprising that zone, Jact stores the number of actual internal edge-edge joins within the zone, Jmax is the theoretical maximum number of edge-edge joins based on N, and Porosity is the computed porosity value.

Note

For zones with only 1 cell, there will be no internal joins possible; thus, Jmax and Jact will both be 0 (zero) and hence the Porosity will appear as NaN in the output dataframe. This function works fully in R, unlike the version used for publishing the paper referred to below that used a combination of python scripts written for ArcGIS that pass results to to R. This version is much easier to implement and use generically.

Author(s)

Tarmo K. Remmel

References

Remmel, T.K. 2018. An incremental and philosophically different approach to measuring raster patch porosity. Sustainability 10:3413.

See Also

See also the package 'raster'.

Examples

```
porosity(IN=data$rst, PLOT=TRUE, NEIGH=4)
```

`singlemap` *Process a single binary map*

Description

This function estimates the composition and configuration parameters for a binary map having 64x64 pixel extent. The 64x64 extent is a current limitation and will be expanded upon in subsequent updates. The parameter estimates are corrected for bias introduced by the Whittle estimation and then these parameters are used to simulate a series of statistically similar landscapes. Each of these simulated landscapes are processed with SDMTools to compute class-focused pattern metrics that are saved to produce empirical (null) distributions for each metric, given the specified level of spatial autocorrelation (configuration) and class proportion (composition). The results can be saved to a new object that is later used for producing graphical and statistical outputs.

Usage

```
singlemap(IMG = data$demoimage1,
  CORRECTIONMAT = data$DIFF50,
  VERBOSE = TRUE,
  reps = 1,
  LEVEL=6)
```

Arguments

`IMG`	This is a numeric, binary matrix that represents a raster landscape. Internally, for computing landscape metrics, this is handled as a raster class object, but input must be a two-valued matrix. There is no implementation for the spatial resolution of each cell as the computed metrics do not require this value. It assumes that the spatial resolution is consistent in both dimensions and across the entire scene. The image is also assumed to be a graphic representation resulting from a stationary spatial process that is 64x64 pixels in extent. The extent limitation will be expanded in a future release but for stationary and isotropic landscape patterns, this is deemed sufficient for now and could be used in a landscape sampling context.

CORRECTIONMAT — This is the name of a numeric matrix that provides the correction factors based on Whittle's estimation. The defalut matrix DIFF50 is provided in the WhittleData environment that is supplied with this package. However, if image sizes differ (i.e., not the 64x64 demos that are provided), this matrix needs to be recreated using `build.lut` and the result provided here. The code, by default, will search for this name in the data environment unless you specifiy a custom matrix that you have created.

VERBOSE — This argument is set to TRUE if you desire extensive on-screen feedback as to what the program is doing. If set to FALSE, much less information is written to screen. It is useful to set to TRUE if `reps` is large, since processing times can be long; this gives an idea as to how much time is remaining to complete processing the data.

reps — This argument controls the number of replicate landscapes that will be simulated and used to produce the empirical (null) distributions for the class-focused pattern metrics. The default value is set at 1 to permit a quick check that the code is loaded and working correctly, but should be run with a value of 100-1000 or larger.

LEVEL — This is a measure of the image size being processed. It is necessary for preparing the necessary arrays used for extimating rho. The value must be an integer conforming to the image size, such that LEVEL = N, where N is 2^N x 2^N for the image dimensions. Thus, for a 64x64 pixel map, LEVEL = 6.

Details

When analyzing multiple maps, this function should be called independently for each map, with the results stored to unique objects that become input to either `singleplotter` or `doubleplotter`.

Value

This returns an object containing the empirical (null) distributions for a suite of class-focused pattern metrics. The rows are observations for each replicate and the columns are the class metrics. Column names identify the actual metrics that are defined in the landscapemetrics package.

Note

Implementation is currently for 64x64 binary landscapes, provided as arguments in integer matrix format. The size limitation will be relaxed in a future release when some of the internal coding structure is streamlined. The result of this function call should be saved to an object that can subsequently be used in calls by either `singleplotter` of `doubleplotter`.

Author(s)

Tarmo K. Remmel

References

Remmel, T.K. and F. Csillag. 2003. When are two landscape pattern indices significantly different? Journal of Geographical Systems 5(4):331-351

Remmel, T.K. and M.-J. Fortin. 2013. Categorical class map patterns: characterization and comparison. Landscape Ecology. DOI: 10.1007/s/10980-013-9905-x.

Remmel, T.K. and M.-J. Fortin. What constitutes a significant difference in landscape pattern? (using R). 2016. In Gergel, S.E. and M.G. Turner. Learning landscape ecology: concepts and techniques for a sustainable world (2nd ed.). New York: Springer.

See Also

See Also `singleplotter`, and `doubleplotter`.

Examples

```
result1 <- singlemap(IMG = data$demoimage1, CORRECTIONMAT = data$DIFF50,
  VERBOSE = TRUE, reps = 1, LEVEL=6)
```

---

singleplotter *Depict the empirical (null) distribution for class-focused pattern metrics.*

Description

Produce a series of one or more boxplots depicting the empirical (null) distribuitons of class-focused pattern metrics, as computed for a single landscape map. The code has been updated as of October 24 2016 to permit boxplots to be drawn even if landscape metrics have NA in them (those values are simply ignored). As of 21 January 2020, the code now relies on landscapemetrics and raster packages rather than the diminished SDMTools.

Usage

```
singleplotter(dat = data$result1,
  img = data$demoimage1,
  metrics = c(1, 5, 10),
  rows = 1,
  cols = 3,
  addactual = TRUE,
  colour = TRUE)
```

Arguments

`dat`	This is the output object produced by `singlemap`. It contains the empirical (null) distributions for class-focused pattern metrics computed for a single binary landscape map.
`img`	This is the binary integer matrix landscape map that corresponds to the `data` argument.
`metrics`	A vector listing the integers, corresponding to specific metrics, that you want to plot. There are 55 unique class-level metrics computed for each of 2 classes; thus, the specific metric values in the specified vector can range from 1-110.

rows	This argument controls the number of rows (of plots) on the output graphics device. The total number of graphs that can be displayed simultaneously will be (rows * cols). These values should be assigned to correspond with the number of metrics that you want to view. Trying to display too many will make them very small and not very informative. It may be more effective to simply call this function multiple times with fewer metrics.
cols	This argument controls the number of columns (of plots) on the output graphics device. The total number of graphs that can be displayed simultaneously will be (rows * cols). These values should be assigned to correspond with the number of metrics that you want to view. Trying to display too many will make them very small and not very informative. It may be more effective to simply call this function multiple times with fewer metrics.
addactual	If TRUE, then the landscape's actual class-focused pattern metric will be added to the plot to depict its relative position within the empirical (null) distribution.
colour	If TRUE, the actual landscape's class-focused pattern metric, if added, will be drawn in red. If FALSE, the plot will be without colour (hollow circle).

Details

This function requires as input, both the original image object (img) and output object from singlemap (data) such that the proper metric values can be plotted along with their expectations (distributions). This is the work-horse plotting function for metric values and distributions for any specific landscape object. Currently, implementaiton is for 64x64 pixel subset landscape maps. Future development will expand on this limiatation; however, our work has shown that this extent is sufficient for adequate pattern characterization and comparison purposes.

Value

The output is graphical.

Note

No further notes at this time.

Author(s)

Tarmo K. Remmel

References

Remmel, T.K. and M.-J. Fortin. 2013. Categorical class map patterns: characterization and comparison. Landscape Ecology. DOI: 10.1007/s/10980-013-9905-x.

Remmel, T.K. and M.-J. Fortin. What constitutes a significant difference in landscape pattern? (using R). 2016. In Gergel, S.E. and M.G. Turner. Learning landscape ecology: concepts and techniques for a sustainable world (2nd ed.). New York: Springer.

See Also

See Also singlemap, and doubleplotter.

Examples

```
# EXAMPLE USES PREVIOUSLY PRODUCED RESULTS TO SPEED-UP THE EXAMPLE,
# BUT THE EXAMPLE FROM singlemap() SHOULD BE CALLED FIRST
singleplotter(dat = data$result1, img = data$demoimage1, metrics = c(1, 5, 10))
```

`ssr` *Decompose a planar shape (polygon) to produce area, perimeter, and number of parts spectra*

Description

Given a single- or multi-part polygon (imported from a shapefile), ssr decomposes the shape by iteratively shrinking it by a specified distance until it becomes extinct. At each iteration of shrinking, the area, perimeter, and number of parts forming the resultant polygon are stored. Graphic plots and maps along with a tabular result are returned. This version works fully within R and no longer requires RSAGA GIS installed, making it more streamlined and faster.

Usage

```
ssr(DIST = 25, shp = NULL, colours = c("LightGreen", "Tan"), PLOT = TRUE)
```

Arguments

`DIST`	The specified distance (in meters) by which to incrementally shrink the shape internally.
`shp`	The shape (polygon) to be processed by ssr. This must be a single shape (although it can be multi-part and contain holes). This object needs to be imported by readOGR and saved as an object prior to calling ssr.
`colours`	An option to provide a vector of two colours that will be used alternately when creating graphic outputs. This is implemented only if PLOT = TRUE.
`PLOT`	This is a Boolean (TRUE | FALSE) flag that governs whether graphic output is produced and presented at the conclusion of the decomposition. When ssr is called in batch mode (with batchssr), it is advised that this be set to FALSE as it dramatically slows processing and will continue to overplot itself.

Value

If PLOT = TRUE, the function returns to the display maps of the decomposed polygon shape along with plots of the area, perimeter, and number of parts spectra. The function by default returns the tabular data from which the spectra can be plotted at any time.

Note

The shapefile must already be imported and have only one unique shape (although it may be multi-part and contain holes). The projection must be rectangular with units in meters (not decimal degrees or other angular unit). Code adjustments were provided by Robert J. Hijmans to allow smooth operation with the package 'terra' modifications.

Author(s)

Tarmo K. Remmel

References

Remmel, T.K. 2015. ShrinkShape2: a FOSS toolbox for computing rotation-invariant shape spectra for characterizing and comparing polygons. The Canadian Geographer 59(4):532-547.

See Also

None.

Examples

```
# ALTERNATE DEMO DATASETS
# shpfileobj <- terra::vect(data$p4no)
# shpfileobj <- terra::vect(data$p4is)

# PATH TO DEMO SHAPEFILE
f <- system.file("ex/lux.shp", package="terra")
# IMPORT SHAPEFILE TO SpatVect FORMAT WITH terra PACKAGE
v <- terra::vect(f)
# REPROJECT TO A VALID PROEJCTION
shpfileobj <- terra::project(v, "EPSG:2150")
# CALL ssr ON FIRST OF THE POLYGONS WITHIN THE SpatVect
out <- ssr(DIST=500, shp=shpfileobj[1], colours=c("LightGreen", "Tan"), PLOT=TRUE)
print(out)
```

---

surfplot — *Produce graphic plots providing class-focused pattern metric context for a landcape map*

Description

This function produces three separate plots on a 1 row by 3 column output view. The first plot is the expected mean surface for a selected class-focused pattern metric across the possible range of composition and configuration. A dropped marker can be added to the plot that depicts the position of any combination of composition or configuration parameterization. The second and thrid plots provide series of boxplots representing orthogonal cross-sections of the surface in the first plot, intersecting at the specified location. This provides a sense of how the selected metric's expected value and variability will change as the composition and/or configuration parameter are altered.

Usage

```
surfplot(metric = 9, prop = 0.7, rho = 0.2, colour = TRUE, drop = TRUE,
cross = TRUE, dat=data$surfaces)
```

Arguments

metric	An integer identifying the class-focused pattern metric that the plots will represent. These integers are listed in the source code for singleplotter.
prop	A real value between zero (0) and one (1) that defines the proportion of the focal land cover category. This value along with rho define where the marker will be drawn on the first plot and through which the orthogonal cross-sections will be drawn.
rho	A real value between zero (0) and one (1) that defines the level of spatial autocorrelation for the focal land cover category. This value along with prop define where the marker will be drawn on the first plot and through which the orthogonal cross-sections will be drawn.
colour	A Boolean indication as to whether the plot should contain colour (TRUE) or not (FALSE).
drop	A Boolean indication as to whether the specified point based on composition (prop) and configuration (rho) should be added to the plot (TRUE) or not (FALSE)
cross	A Boolean indication as to whether the cross-sectional boxplots should be drawn (TRUE) or not (FALSE).
dat	A numeric array object created by buildsurfs that has dimensions [38,9,11,replicates]. The version provided is produced with only 5 replicates to save space; it is highly recommended to produce a dat object, such as surfaces that is produced with 1000 replicates.

Details

This function requires a valid result from the function buildsurfs to exist. That result is an array of simulated maps for a series of replicates produced for all pair-wise paramterizations of composition and configuration. This object can be quite large; however, needs to only be produced once. All polts thereafter can be produced (for any class-focused metric) from this stored object. To save processing time, the PatternClass package is provided with these reference (lookup) surfaces already produced for 64x64 images. Future developments will permit these layers to be produced at a wider range of extents.

Value

This function returns a graphic plot.

Note

No further notes at this time.

Author(s)

Tarmo K. Remmel

References

Remmel, T.K. and M.-J. Fortin. What constitutes a significant difference in landscape pattern? (using R). 2016. In Gergel, S.E. and M.G. Turner. Learning landscape ecology: concepts and techniques for a sustainable world (2nd ed.). New York: Springer.

See Also

See Also singlemap, singleplotter, buildsurfs, and doubleplotter.

Examples

```
surfplot(metric = 9, prop = 0.7, rho = 0.2, colour = TRUE, drop = TRUE,
cross = TRUE, dat=data$surfaces)
```

---

wi *An internal utility that helps with the Whittle estimation.*

Description

This is an internal utility function for helping with the Whittle estimation. This function is not intended for use directly by users.

Usage

```
wi(BE=data$demoimage1, CONTROL=FALSE, SIZE=6)
```

Arguments

BE	This is a binary input image.
CONTROL	This is a Boolean flag.
SIZE	Controls the output matrix size.

Details

This tool is only used internally.

Value

The result is a value.

Note

This function is not to be used directly by users of this package.

Author(s)

Ferko Csillag, Sandor Kabos, Tarmo K. Remmel

References

No references currently.

See Also

Currently none.

Examples

```
# No example.
```

`wtest.run` *Whittle estimation for binary map*

Description

This function is the workhorse for estimating the Whittle correction for a binary map. This function is generally called by a wrapping function to facilitate its use (e.g., `singlemap`).

Usage

```
wtest.run(LEVEL=6, REPSIM=20, RHO=0.2499999, CPROP=0.5, RAJZ=TRUE, CIM="CIM", ENV="data")
```

Arguments

`LEVEL`	This is a numeric, binary matrix that represents a raster landscape. In future updates, this argumnent may migrate to be a raster object, but for now, it must be a two-valued matrix. There is no implementation for the spatial resolution of each cell as the computed metrics do not require this value. It assumes that the spatial resolution is consistent in both dimensions and across the entire scene. The image is also assumed to be a graphic representation resulting from a stationary spatial process.
`REPSIM`	This is a numeric matrix that provides the correction factors based on Whittle's estimation. The defalut matrix DIFF50 is provided in the data environment that is supplied with this package. However, if image sizes differ (i.e., not the 64x64 demos that are provided), this matrix needs to be recreated using `build.lut` and the result provided here.
`RHO`	This is a numeric argument recording the spatial autocorrelation parameter. Note that this is divided into 4 and thus 0.2499999 approaches the theoretical maximum value of 1.
`CPROP`	This argument controls the proportion of white to black pixels.
`RAJZ`	This Boolean argument controls whether outputs of the simulation are drawn on the screen.
`CIM`	A parameter that can control the naming of outputs.
`ENV`	A parameter that can controls the name of the environment where temporary objects are stored.

Details

This code is generally not called directly by users.

Value

Not for users.

Note

This is for internal use. Use `singlemap` instead.

Author(s)

Tarmo K. Remmel

References

Remmel, T.K. and F. Csillag. 2003. When are two landscape pattern indices significantly different? Journal of Geographical Systems 5(4):331-351

Remmel, T.K. and M.-J. Fortin. 2013. Categorical class map patterns: characterization and comparison. Landscape Ecology. DOI: 10.1007/s/10980-013-9905-x.

Remmel, T.K. and M.-J. Fortin. What constitutes a significant difference in landscape pattern? (using R). 2016. In Gergel, S.E. and M.G. Turner. Learning landscape ecology: concepts and techniques for a sustainable world (2nd ed.). New York: Springer.

See Also

See Also `singlemap`.

Examples

```
# No example.
```

Index