Package ‘habCluster’

October 13, 2022

Type Package

Title Detecting Spatial Clustering Based on Connection Cost Between Grids

Version 1.0.5

Date 2022-05-24

Author Qiang Dai

Maintainer Qiang Dai <daiqiang@cib.ac.cn>

Description Based on landscape connectivity, spatial boundaries were identified using community detection algorithm at grid level. Methods using raster as input and the value of each cell of the raster is the ``smoothness" to indicate how easy the cell connecting with neighbor cells. Details about the 'habCluster' package methods can be found in Zhang et al. <bioRxiv:2022.05.06.490926>.

License GPL (>= 3)

Depends R (>= 4.0.0), igraph (>= 1.3.0), stars (>= 0.5-0), sf (>= 1.0.0), methods

Imports Rcpp, raster

Suggests knitr, rmarkdown, testthat (>= 3.1.0), spelling

Config/testthat/edition 3

LinkingTo Rcpp

Encoding UTF-8

RoxygenNote 7.1.2

VignetteBuilder knitr

Language en-US

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-05-25 15:30:02 UTC

R topics documented:

cluster 2
raster2Graph 4

Index **5**

`cluster` *Clustering cells from a raster by Community Detection Algorithm according to the connections between them and return a cluster map*

Description

This function use Community Detection Algorithm to find structure of raster and return a polygon representing the boundary of the clusters.

Usage

```
cluster(
  r = NULL,
  method = igraph::cluster_fast_greedy,
  cellsize = NULL,
  relative.distance = TRUE,
  silent = TRUE,
  ...
)
```

Arguments

`r`	An object of stars or RasterLayer. The value of each cell of the raster is the 'smoothness' to indicate how easy the cell connecting with neighbor cells.
`method`	method from package igraph used to finding community structure. (see details below).
`cellsize`	Numeric. Re-sample the input raster to given resolution and use the resampled raster to find community structure. Set this to NULL if using the original resolution of of the input raster,given the parameter r is an object of raster.
`relative.distance`	Boolean. If FALSE, absolute distance between cells is used to compute the edge weight; otherwise, relative distance between cells is used . Default is TRUE.
`silent`	Boolean. A logical indicating if some "progress report" should be given. Default is TRUE.
`...`	Optional arguments to method. For example, can set resolution_parameter for cluster_leiden, or resolution for cluster_louvain.(see details below).

Details

Choice of the method used to finding community structure(see Mukerjee, 2021). The default method is cluster_fast_greedy, but could also be methods like cluster_leiden, cluster_walktrap, or cluster_louvain. If cluster_leiden is chosen, then we can use resolution_parameter to control the size of clusters. Higher resolution_parameter lead to more smaller clusters, while lower resolution_parameter lead to fewer larger clusters. The parameter of resolution for cluster_louvain is similar. More details about those methods can be found in the document for package "igraph".

Value

A polygon of sf object for boundaries of habitat clusters, and an object of communities defined in package igraph.

References

Mukerjee, S. (2021). A systematic comparison of community detection algorithms for measuring selective exposure in co-exposure networks. Scientific reports 11, 15218. https://doi.org/10.1038/s41598-021-94724-1
Traag, V. A., Waltman, L., & van Eck, N. J. (2019). From Louvain to Leiden: guaranteeing well-connected communities. Scientific reports, 9(1), 5233. doi: 10.1038/s41598-019-41695-z

Examples

```
library(sf)
library(stars)

# read in habitat suitability data of wolf in Europe
hsi.file = system.file("extdata","wolf3_int.tif",package="habCluster")
wolf = read_stars(hsi.file)

# rescale raster value to 0 - 1
wolf = wolf / 100

# find habitat cluster using Fast Greedy Algorithm.
# Raster will be resampled to 40 km, to cluser at the scale of 40 km and reduce calculation amount.
clst = cluster(wolf, method = cluster_fast_greedy, cellsize = 40000)

# plot the results
image(wolf,col=terrain.colors(100,rev = TRUE),asp = 1)
boundary = clst$boundary
plot( boundary$geometry, add=TRUE, asp=1, border = "lightseagreen")

# discard patches smaller than 1600 sqkm
boundary$area = as.numeric(st_area(boundary))
boundary = boundary[boundary$area > 40000*40000,]
image(wolf,col=terrain.colors(100,rev = TRUE),asp = 1)
plot( boundary$geometry, add=TRUE, asp=1, border = "lightseagreen")

# can also use RasterLayer object#
library(raster)
```

```
wolf = read_stars(hsi.file)
wolf = wolf / 100
clst = cluster(wolf, method = cluster_leiden, cellsize = 40000, resolution_parameter = 0.0002)
```

`raster2Graph` *Create a graph from an raster according the connection between cells*

Description

Create a graph from an raster according the connection between cells

Usage

```
raster2Graph(r, cellsize = NULL, relative.distance = TRUE, silent = TRUE)
```

Arguments

`r`	An object of stars or RasterLayer. The value of each cell of the raster is the 'smoothness' to indicate how easy the cell connecting with neighbor cells.
`cellsize`	Numeric. Re-sample the input raster to given resolution and use the re-sampled raster to build graph. Set this to NULL if using the original resolution of of the input raster.
`relative.distance`	Boolean. If fasle, absolute distance between cells is used to compute the edge weight; otherwise, relative distance between cells is used. Default is true
`silent`	Boolean. A logical indicating if some "progress report" should be given. Default is TRUE.

Value

a list with an graph and the re-sampled raster (a object of stars). The graph is igraph object, with cells as node and connections as weight.

Examples

```
# read in habitat suitability data of wolf in Europe
library(stars)
hsi.file = system.file("extdata","wolf3_int.tif",package="habCluster")
wolf = read_stars(hsi.file)
# build graph from raster
g = raster2Graph(wolf, 40000)
```

Index

cluster, 2

raster2Graph, 4