Package 'grainscape'

July 7, 2026

Type Package

Title Landscape Connectivity, Habitat, and Protected Area Networks

Description Given a landscape resistance surface, creates minimum planar graph (Fall et al. (2007) <doi:10.1007/s10021-007-9038-7>) and grains of connectivity (Galpern et al. (2012) <doi:10.1111/j.1365-294X.2012.05677.x>) models that can be used to calculate effective distances for landscape connectivity at multiple scales. Documentation is provided by several vignettes, and a paper (Chubaty, Galpern & Doctolero (2020) <doi:10.1111/2041-210X.13350>).

URL https://www.alexchubaty.com/grainscape/, https://github.com/achubaty/grainscape

Version 1.0.0

License GPL (>= 2)

Depends R (>= 4.2)

Imports ggplot2, graphics, grDevices, igraph, methods, Rcpp (>= 0.12.11.4), sf, stats, terra, utils

Suggests covr, cowplot, DiagrammeR, DiagrammeRsvg, dplyr, ggthemes, knitr, parallel, rmarkdown, rsvg, spelling, svglite, testthat, vdiffr, withr

LinkingTo Rcpp

VignetteBuilder knitr, rmarkdown

BugReports https://github.com/achubaty/grainscape/issues

ByteCompile yes

Collate 'grainscape-package.R' 'classes.R' 'GOC.R' 'MPG.R' 'RcppExports.R' 'grain.R' 'corridor.R' 'distance.R' 'export.R' 'extract.R' 'ggGS.R' 'grainscape-defunct.R' 'graphdf.R' 'habitatConnectivityEngine.R' 'patchFilter.R' 'plot.R' 'plotResistance.R' 'point.R' 'theme_grainscape.R' 'threshold.R' 'zzz.R'

Encoding UTF-8

Language en-CA

Config/Needs/website pkgdown

Config/roxygen2/version 8.0.0

Config/testthat/edition 3

Config/testthat/parallel true

NeedsCompilation yes

Author Paul Galpern [aut, cph] (ORCID:
<https://orcid.org/0000-0003-0099-3981>),
Sam Doctolero [aut],
Alex M Chubaty [aut, cre] (ORCID:
<https://orcid.org/0000-0001-7146-8135>)

Maintainer Alex M Chubaty <achubaty@for-cast.ca>

Repository CRAN

Date/Publication 2026-07-07 02:30:02 UTC

Contents

grainscape-package . . . 3
corridor . . . 4
corridor-class . . . 6
distance . . . 6
export . . . 8
ggGS . . . 12
GOC . . . 14
goc-class . . . 17
grain . . . 17
grain-class . . . 19
grainscape-maps . . . 19
graphdf . . . 20
MPG . . . 22
mpg-class . . . 24
patchFilter . . . 25
plot,corridor,ANY-method . . . 26
plotResistance . . . 29
plotWithResistance . . . 30
point . . . 31
show,goc-method . . . 32
theme_grainscape . . . 33
threshold . . . 34
$ . . . 36

Index **37**

`grainscape-package` *Landscape Connectivity, Habitat, and Protected Area Networks*

Description

Given a landscape resistance surface, creates minimum planar graph and grains of connectivity models that can be used to calculate effective distances for landscape connectivity at multiple scales.

Details

Landscape connectivity modelling to understand the movement and dispersal of organisms has been done using raster resistance surfaces and landscape graph methods. Grains of connectivity (GOC) models combine elements of both approaches to produce a continuous and scalable tool that can be applied in a variety of study systems. The purpose of this package is to implement grains of connectivity analyses. Routines accept raster-based resistance surfaces as input and return raster, vector and graph-based data structures to represent connectivity at multiple scales. Effective distances describing connectivity between geographic locations can be determined at multiple scales. Analyses of this sort can contribute to corridor identification, landscape genetics, as well as other connectivity assessments. Minimum planar graph (MPG; Fall *et al.*, 2007) models of resource patches on landscapes can also be generated using the software.

MPG calculations and generalization of the Voronoi tessellation used in GOC models is based on the routines in SELES software (Fall and Fall, 2001). Routines depend on the **terra** (Hijmans, 2023), **sf** (Pebesma, 2018), and **igraph** (Csardi and Nepusz, 2006) packages.

A paper describing the use of this package for landscape connectivity modelling is available at doi:10.1111/2041210X.13350.

A detailed tutorial is available as a vignette (see `browseVignettes('grainscape')`).

Author(s)

Maintainer: Alex M Chubaty <achubaty@for-cast.ca> (ORCID)

Authors:

- Alex M Chubaty <achubaty@for-cast.ca> (ORCID)
- Paul Galpern <pgalpern@gmail.com> (ORCID) [copyright holder]
- Sam Doctolero <sam.doctolero@gmail.com>

References

Csardi, G. and T. Nepusz. (2006). The igraph software package for complex network research. InterJournal Complex Systems 1695. https://igraph.org.

Fall, A. and J. Fall. (2001). A domain-specific language for models of landscape dynamics. Ecological Modelling 141:1-18.

Fall, A., M.-J. Fortin, M. Manseau, D. O'Brien. (2007) Spatial graphs: Principles and applications for habitat connectivity. Ecosystems 10:448:461.

Galpern, P., M. Manseau. (2013a) Finding the functional grain: comparing methods for scaling resistance surfaces. Landscape Ecology 28:1269-1291.

Galpern, P., M. Manseau. (2013b) Modelling the influence of landscape connectivity on animal distribution: a functional grain approach. Ecography 36:1004-1016.

Galpern, P., M. Manseau, A. Fall. (2011) Patch-based graphs of landscape connectivity: A guide to construction, analysis and application for conservation. Biological Conservation 144:44-55.

Galpern, P., M. Manseau, P.J. Wilson. (2012) Grains of connectivity: analysis at multiple spatial scales in landscape genetics. Molecular Ecology 21:3996-4009.

Hijmans, R.J. (2023). terra: Spatial Data Analysis. R package, https://CRAN.R-project.org/package=terra.

Pebesma, E.J. (2018). Simple features for R: Standardized support for spatial vector data. The R Journal 10:1, 439-446, doi:10.32614/RJ2018009.

See Also

Useful links:

- https://www.alexchubaty.com/grainscape/
- https://github.com/achubaty/grainscape
- Report bugs at https://github.com/achubaty/grainscape/issues

---

`corridor` *Visualize corridors between two points using a grains of connectivity (GOC)*

Description

Given a series of GOC models built at different scales, visualize the corridor (or shortest path) between two points using one of the tessellations (i.e., scales) in these models.

Usage

```
corridor(x, ...)

## S4 method for signature 'goc'
corridor(x, whichThresh, coords, weight = "meanWeight", ...)
```

Arguments

`x`	A goc object created by `GOC()`.
`...`	Additional arguments (not used).
`whichThresh`	Integer giving the index of the threshold to visualize.
`coords`	A two-column matrix or an `sf` (POINT) object giving coordinates at the end points of the corridor.
`weight`	The GOC graph link weight to use in calculating the distance. Please see details in `distance()`.

Value

An object of class corridor.

Author(s)

Paul Galpern and Alex Chubaty

References

Fall, A., M.-J. Fortin, M. Manseau, D. O'Brien. (2007) Spatial graphs: Principles and applications for habitat connectivity. Ecosystems 10:448:461.

Galpern, P., M. Manseau. (2013a) Finding the functional grain: comparing methods for scaling resistance surfaces. Landscape Ecology 28:1269-1291.

Galpern, P., M. Manseau. (2013b) Modelling the influence of landscape connectivity on animal distribution: a functional grain approach. Ecography 36:1004-1016.

Galpern, P., M. Manseau, A. Fall. (2011) Patch-based graphs of landscape connectivity: a guide to construction, analysis, and application for conservation. Biological Conservation 144:44-55.

Galpern, P., M. Manseau, P.J. Wilson. (2012) Grains of connectivity: analysis at multiple spatial scales in landscape genetics. Molecular Ecology 21:3996-4009.

See Also

GOC(), grain()

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
## Quick visualization of a corridor
corridorStartEnd <- rbind(c(10, 10), c(90, 90))
tinyPatchCorridor <- corridor(tinyPatchGOC, whichThresh = 3, coords = corridorStartEnd)
if (interactive()) {
  plot(tinyPatchCorridor)
}

## More control over a corridor visualization
if (interactive()) {
  plot(tinyPatchCorridor@voronoi, col = "lightgrey", lwd = 2)
  plot(tinyPatchCorridor@linksSP, col = "darkred", lty = "dashed", add = TRUE)
  plot(tinyPatchCorridor@nodesSP, col = "darkred", pch = 21, bg = "white", add = TRUE)
```

```
 plot(tinyPatchCorridor@shortestLinksSP, col = "darkred", lty = "solid", lwd = 2, add = TRUE)
 plot(tinyPatchCorridor@shortestNodesSP, col = "darkred", pch = 21, bg = "darkred", add = TRUE)
  mtext(
    paste(
      "Corridor shortest path length:",
      round(tinyPatchCorridor@corridorLength, 2),
      "resistance units"
    ),
    side = 1
  )
}
```

`corridor-class` *The* `corridor` *class*

Description

The `corridor` class

Slots

`voronoi` A `SpatRaster` representation of the boundaries of the voronoi polygons.

`linksSP` An `sf` representation of links in the grains of connectivity graph.

`nodesSP` An `sf` representation of the nodes in the grains of connectivity graph

`shortestLinksSP` An `sf` representation of the links in the shortest path between coordinates

`shortestNodesSP` An `sf` representation of the nodes in the shortest path between coordinates

`corridorLength` A `numeric` of length 1 giving the length of the shortest path between coordinates in accumulated resistance units.

See `corridor()` for more information.

Author(s)

Alex Chubaty and Paul Galpern

`distance` *Find the grains of connectivity network distance*

Description

Find the shortest network distance between pairs of points using the GOC graph. This can be used as an effective distance for landscape connectivity assessments.

Usage

```
distance(x, y, ...)

## S4 method for signature 'goc,sf'
distance(x, y, weight = "meanWeight", ...)

## S4 method for signature 'goc,matrix'
distance(x, y, weight = "meanWeight", ...)

## S4 method for signature 'goc,numeric'
distance(x, y, weight = "meanWeight", ...)
```

Arguments

x	A goc object produced by GOC().
y	A two-column matrix or an sf (POINT) object giving the coordinates of points of interest.
...	Additional arguments (not used).
weight	The GOC graph link weight to use in calculating the distance. Please see Details for explanation.

Value

A list object giving a distance matrix for each threshold in the GOC object. Distance matrices give the pairwise grains of connectivity network distances between sampling locations. Matrix indices correspond to rows in the coordinates matrix (y).

Author(s)

Paul Galpern and Alex Chubaty

References

Fall, A., M.-J. Fortin, M. Manseau, D. O'Brien. (2007) Spatial graphs: Principles and applications for habitat connectivity. Ecosystems 10:448:461.

Galpern, P., M. Manseau. (2013a) Finding the functional grain: comparing methods for scaling resistance surfaces. Landscape Ecology 28:1269-1291.

Galpern, P., M. Manseau. (2013b) Modelling the influence of landscape connectivity on animal distribution: a functional grain approach. Ecography 36:1004-1016.

Galpern, P., M. Manseau, A. Fall. (2011) Patch-based graphs of landscape connectivity: a guide to construction, analysis, and application for conservation. Biological Conservation 144:44-55.

Galpern, P., M. Manseau, P.J. Wilson. (2012) Grains of connectivity: analysis at multiple spatial scales in landscape genetics. Molecular Ecology 21:3996-4009.

See Also

GOC(), point()

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
## Three sets of coordinates in the study area
loc <- cbind(c(30, 60, 90), c(30, 60, 90))

## Find the GOC network distance matrices between these points
## for each of the 5 grains of connectivity
tinyDist <- grainscape::distance(tinyPatchGOC, loc)
```

---

export *Export spatial data from MPG and GOC models*

Description

This function automates the export of raster and vector spatial data from `mpg` and `grain` objects. By default it places them in a new directory, unless an existing one is specified with `overwrite = TRUE`.

It can also be used to process `mpg` and `grain` objects to produce R spatial objects that are convenient for plotting or analysis within R. Use `R = TRUE` in which case all parameters related to file export are ignored. (Default `R = FALSE`)

The `terra::writeRaster()` function is used for rasters, and `sf::st_write()` is used to export ESRI compatible shape files.

Usage

```
export(
  x,
  dirname = NULL,
  path = ".",
  rasterFormat = "GTiff",
  overwrite = FALSE,
  R = FALSE,
  vorBound = FALSE,
  ...
)

## S4 method for signature 'mpg'
```

```
export(
  x,
  dirname = NULL,
  path = ".",
  rasterFormat = "GTiff",
  overwrite = FALSE,
  R = FALSE,
  vorBound = FALSE,
  ...
)

## S4 method for signature 'grain'
export(
  x,
  dirname = NULL,
  path = ".",
  rasterFormat = "GTiff",
  overwrite = FALSE,
  R = FALSE,
  vorBound = FALSE,
  ...
)

## S4 method for signature 'goc'
export(
  x,
  dirname = NULL,
  path = ".",
  rasterFormat = "GTiff",
  overwrite = FALSE,
  R = FALSE,
  vorBound = FALSE,
  ...
)
```

Arguments

x	A mpg or grain object
dirname	The name of a new directory to create. If NULL a directory with a name containing the date and time will be created.
path	A path to where this new directory dirname should be created. Defaults to the working directory.
rasterFormat	The format for exported rasters. See terra::writeRaster() for options. Defaults to GeoTiff (rasterFormat='GTiff').
overwrite	If directory already exists will overwrite existing files inside. Defaults to FALSE.
R	If TRUE, return the spatial objects that would be written to files. Do not write these files and ignore dirname, path, rasterFormat, overwrite parameters.

This is useful for visualization using R plotting functions, or spatial analysis within R. Defaults to FALSE

vorBound — Specify whether to create a raster with the boundaries of the Voronoi polygons =1 and the remainder =NA. This may be useful for visualizing relationships among polygons in a grain of connectivity. This can add time to the export on very large rasters. Defaults to FALSE.

... — Additional arguments (not used).

Value

Invisibly returns the path to the folder created.

Side effect of exporting files representing raster and vector spatial data in the object.

Please note that for vector data export the attribute name is limited to 8 characters in shape files. See the tables below for the abbreviations used and their meaning.

Exported from mpg objects:

nodes, linksCentroid, linksPerim are shape files giving the locations of the patch centroids, links among centroids, and links among perimeters of patches respectively. patchId, voronoi are rasters giving the patch identifier of the patch, or of the patch that the Voronoi polygon refers to. lcpPerimWeight, lcpLinkId give the weight in cost surface units of the shortest paths between perimeters, and the identifiers of those links respectively. vorBound gives the boundaries of the Voronoi polygons (if specified).

Description of node (vertex) and link (edge) weights in mpg objects and their corresponding attribute names in the shape files created.

type	MPG name	SHP name	Description
node	patchId	patchId	Patch ID from patchId raster
node	patchArea	patchA	Area of patch
node	patchEdgeArea	patchEA	Edge area of patch
node	coreArea	coreA	Area excluding edge of patch
node	centroidX	ctrX	Centroid of the patch (X)
node	centroidY	ctrY	Centroid of the patch (Y)
link	e1	e1	Id of first patch at end of link
link	e2	e2	Id of second patch at end of link
link	linkId	linkId	Link ID from lcpLinkId raster
link	lcPerimWeight	lcpPerWt	Cost length of link from patch perimeters
link	startPerimX	strtPerX	Coordinate of link endpoint on first patch (X)
link	startPerimY	strtPerY	Coordinate of link endpoint on first patch (Y)
link	endPerimX	endPerX	Coordinate of link endpoint on second patch (X)
link	endPerimY	endPerY	Coordinate of link endpoint on second patch (Y)

Exported from grain objects:

nodes, linksCentroid are shape files giving the locations of the Voronoi polygon centroids and links among them respectively. voronoi are rasters gives the polygon identifier of each cluster of patches. vorBound gives the boundaries of the Voronoi polygons (if specified).

Description of node (vertex) and link (edge) weights in `grain` objects and their corresponding attribute names in the shape files created.

Type	GOC name	SHP name	Description
node	`polygonId`	`polyId`	Polygon ID from grain voronoi raster
node	`polygonArea`	`polyA`	Area of polygon from grain voronoi raster
node	`totalPatchArea`	`patchA`	Total area of all patches in polygon
node	`totalPatchEdgeArea`	`patchEA`	Total area of all patch edges in polygon
node	`totalCoreArea`	`coreA`	Total area of patches in polygon excluding edges
node	`centroidX`	`ctrX`	Centroid of the polygon (X)
node	`centroidY`	`ctrY`	Centroid of the polygon (Y)
link	`e1`	`e1`	ID of first patch at end of link
link	`e2`	`e2`	ID of second patch at end of link
link	`maxWeight`	`maxWt`	The maximum weight of all links connecting patches between polygons
link	`linkIdMaxWeight`	`maxWt`	The link id of that maximum weight `link (lcpLinkId)`
link	`minWeight`	`min`	The minimum weight of all links connecting patches between polygons
link	`linkIdMinWeight`	`minWt`	The link id of that minimum weight `link (lcpLinkId)`
link	`medianWeight`	`medWt`	The median weight of all links connecting patches between polygons
link	`meanWeight`	`meanWT`	The minimum weight of all links connecting patches between polygons
link	`numlinksWeight`	`numEWt`	The number of links connecting patches between polygons
link	`eucCentroidWeight`	`eucCtrWt`	The Euclidean distance between centroids of polygons

Author(s)

Paul Galpern and Alex Chubaty

See Also

`MPG()`, `GOC()`, `grain()`

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
## Export rasters and vectors and place in an R object
sp_tinyPatchGOC <- export(grain(tinyPatchGOC, 2), R = TRUE)
sp_tinyPatchMPG <- export(tinyPatchMPG, R = TRUE)

## Export raster and vectors to a specified directory
exportPath <- tempdir()
export(grain(tinyPatchGOC, 2), dirname = "tiny_goc_thresh2", path = exportPath)
```

```
export(tinyPatchMPG, dirname = "tiny_mpg", path = exportPath, vorBound = TRUE)

## clean up
unlink(file.path(exportPath, "tiny_goc_thresh2"), recursive = TRUE)
unlink(file.path(exportPath, "tiny_mpg"), recursive = TRUE)
```

---

ggGS *Prepare data in* MPG *and* grain *objects for use with* **ggplot2**

Description

This is an informal fortify-type method that prepares either SpatRaster or igraph objects contained as slots within MPG or grain objects for easy plotting with ggplot2::ggplot().

Usage

```
ggGS(x, type = NULL, ...)

## S4 method for signature 'SpatRaster'
ggGS(x, type = NULL, ...)

## S4 method for signature 'list'
ggGS(x, type = NULL, ...)

## S4 method for signature 'mpg'
ggGS(x, type = NULL, ...)

## S4 method for signature 'grain'
ggGS(x, type = NULL, ...)

## S4 method for signature 'goc'
ggGS(x, type = NULL, ...)
```

Arguments

x	A mpg, grain, or SpatRaster object.
type	If a mpg or grain object is supplied, this gives the name of the slot to prepare for plotting. Options are discussed below. Not required if a SpatRaster is supplied.
...	Additional arguments (not used).

Value

A data.frame suitable for plotting with ggplot2::ggplot().

Where type is a raster the data.frame will have the following columns:

value the value of the raster cell

x the x coordinate of the centre of the raster cell

`y` the y coordinate of the centre of the raster cell

Where `type = 'nodes'` the `data.frame` will have the following columns:

`x` the x coordinate of the node

`y` the y coordinate of the node

`...` other attributes associated with the network nodes

Where `type = 'links'` the `data.frame` will have the following columns:

`x1` the x coordinate of the first node

`y1` the y coordinate of the first node

`x2` the x coordinate of the second node

`y2` the y coordinate of the second node

`x1p` the x coordinate at the perimeter of the first node

`y1p` the y coordinate at the perimeter of the first node

`x2p` the x coordinate at the perimeter of the second node

`y2p` the y coordinate at the perimeter of the second node

`...` other attributes associated with the network links

Note

Options for `type` parameter

If a `SpatRaster` is supplied `type` is optional.

For `mpg` `type` options are `"node"` or `"links"`. This prepares the nodes and links of the minimum planar graph network for plotting, Also `"patchId"`, `"voronoi"`, `"lcpPerimWeight"`, `"lcpLinkId"`, `"mpgPlot"` will prepare rasters for plotting.

For `grain` objects `type` options are `"nodes"` or`"links"` to prepare the nodes and links of the grains of connectivity network for plotting. Also `"voronoi"` will prepare the grains of connectivity Voronoi polygons raster for plotting.

For either `mpg` or `grain` objects `type = "vorBound"` will identify the boundaries of the Voronoi polygons for plotting. This is potentially time consuming for large rasters.

Author(s)

Paul Galpern and Alex Chubaty

See Also

`MPG()`, `GOC()`

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
if (interactive()) {
  library(ggplot2)

  ## Plot the patches in a minimum planar graph
  theme_set(theme_grainscape())
  ggplot() +
    geom_tile(data = ggGS(tinyPatchMPG, "patchId"), aes(x = x, y = y, fill = value))

  ## Plot the grain polygons in a grain of connectivity
  ggplot() +
   geom_tile(data = ggGS(grain(tinyPatchGOC, 3), "voronoi"), aes(x = x, y = y, fill = value))

  ## Plot the grain polygon boundaries
  ggplot() +
   geom_tile(data = ggGS(grain(tinyPatchGOC, 3), "vorBound"), aes(x = x, y = y, fill = value))

  ## Plot the patches and perimeter links of a minimum planar graph
  ggplot() +
    geom_tile(data = ggGS(tinyPatchMPG, "patchId"), aes(x = x, y = y, fill = value)) +
   geom_segment(data = ggGS(tinyPatchMPG, "links"), aes(x = x1p, y = y1p, xend = x2p, yend = y2p))

  ## Plot the patches and linear representations of the perimeter links
  ## of a minimum planar graph
  ggplot() +
    geom_tile(data = ggGS(tinyPatchMPG, "patchId"), aes(x = x, y = y, fill = value)) +
   geom_segment(data = ggGS(tinyPatchMPG, "links"), aes(x = x1p, y = y1p, xend = x2p, yend = y2p))

  ## Plot the nodes and links of a grains of connectivity network
  ## superimposed over the grain polygons
  focalGrain <- grain(tinyPatchGOC, 3)
  ggplot() +
    geom_tile(data = ggGS(focalGrain, "vorBound"), aes(x = x, y = y, fill = value)) +
    geom_point(data = ggGS(focalGrain, "nodes"), aes(x = x, y = y)) +
   geom_segment(data = ggGS(focalGrain, "links"), aes(x = x1, y = y1, xend = x2, yend = y2))
}
```

GOC *Produce a grains of connectivity model at multiple scales (patch-based or lattice GOC)*

Description

Produce a grains of connectivity (GOC) model at multiple scales (resistance thresholds) by scalar analysis. Patch-based or lattice GOC modelling can be done with this function.

Usage

```
GOC(x, ...)

## S4 method for signature 'mpg'
GOC(
  x,
  nThresh = NULL,
  doThresh = NULL,
  weight = "lcpPerimWeight",
  verbose = 0,
  ...
)
```

Arguments

x	A mpg object produced by MPG(). For lattice GOC MPG must be run with patch set as an integer value.
...	Additional arguments (not used).
nThresh	Optional. An integer giving the number of thresholds (or scales) at which to create GOC models. Thresholds are selected to produce a maximum number of unique grains (i.e., models). nThresh thresholds are also approximately evenly spread between 0 and the threshold at which all patches or focal points on the landscape are connected. This is a simple way to get a representative subset of all possible GOC models. Provide either nThresh or doThresh not both.
doThresh	Optional. A vector giving the link thresholds at which to create GOC models. Use threshold() to identify thresholds of interest. Provide either nThresh or doThresh not both.
weight	A string giving the link weight or attribute to use for threshold. "lcpPerimWeight" uses the accumulated resistance or least-cost path distance from the perimeters of patches as the link weight.
verbose	Set verbose=0 for no progress information to console.

Details

Grain or scalar analysis of connectivity may be appropriate for a variety of purposes, not limited to visualization and improving connectivity estimates for highly-mobile organisms. See Galpern *et al.* (2012), Galpern & Manseau (2013a, 2013b) for applications and review of these capabilities.

Value

A goc() object.

Note

Researchers should consider whether the use of a patch-based GOC or a lattice GOC model is appropriate based on the patch-dependency of the organism under study. Patch-based models make most sense when animals are restricted to, or dependent on, a resource patch. Lattice models can be used as a generalized and functional approach to scaling resistance surfaces.

See `MPG()` for warning related to areal measurements.

Author(s)

Paul Galpern

References

Fall, A., M.-J. Fortin, M. Manseau, D. O'Brien. (2007) Spatial graphs: Principles and applications for habitat connectivity. Ecosystems 10:448:461.

Galpern, P., M. Manseau. (2013a) Finding the functional grain: comparing methods for scaling resistance surfaces. Landscape Ecology 28:1269-1291.

Galpern, P., M. Manseau. (2013b) Modelling the influence of landscape connectivity on animal distribution: a functional grain approach. Ecography 36:1004-1016.

Galpern, P., M. Manseau, A. Fall. (2011) Patch-based graphs of landscape connectivity: a guide to construction, analysis, and application for conservation. Biological Conservation 144:44-55.

Galpern, P., M. Manseau, P.J. Wilson. (2012) Grains of connectivity: analysis at multiple spatial scales in landscape genetics. Molecular Ecology 21:3996-4009.

See Also

`MPG()`, `grain()`, `distance()`, `point()`

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
## Examine the properties of the GOC graph of grain 3 of 5
graphdf(grain(tinyPatchGOC, whichThresh = 3))

## Extract grains of connectivity
## representation of the finest grain and three others
## by giving thresholds in link weights (doThresh)
tinyPatchGOC <- GOC(tinyPatchMPG, doThresh = c(0, 20, 40))
```

goc-class *The* goc *class*

Description

The goc class

Slots

`voronoi` A `SpatRaster` describing the regions of proximity in resistance units around the focal patches or points.

`summary` A summary of the the grains of connectivity generated and their properties.

`th` A list giving the GOC graph at each threshold.

Each element of `th` contains a goc object giving the GOC graph as class `igraph::igraph()`. Vertex attributes describes qualities of each polygon including the coordinates of each polygon centroid, the area of these polygons, and the original patch IDs in the MPG that are included in each polygon. All areal measurements are given as raster cell counts. A variety of edge attributes are also given in the GOC graph. See `distance()` for more information.

Author(s)

Alex Chubaty and Paul Galpern

grain *Extract a grain of connectivity (GOC) tessellation at a given scale*

Description

Extract a grain (i.e. a scaled version of a Voronoi tessellation) from a GOC model.

Usage

```
grain(x, ...)

## S4 method for signature 'goc'
grain(x, whichThresh, ...)
```

Arguments

`x`	A goc object created by `GOC()`.
`...`	Additional arguments (not used).
`whichThresh`	Integer giving the grain threshold to extract. This is the index of the threshold extracted by `GOC()`.

Value

A list object containing the following elements:

`summary` gives the properties of the specified scale/grain `whichThresh` of the GOC model;

`voronoi` a `SpatRaster` giving the Voronoi tessellation the specified scale/grain `whichThresh` of the GOC model;

`centroids` an `sf` object giving the centroids of the polygons in the Voronoi tessellation at the specified scale/grain `whichThresh`;

`th` a `igraph` object giving the graph describing the relationship among the polygons at the specified scale/grain `whichThresh`

Author(s)

Paul Galpern and Alex Chubaty

References

Fall, A., M.-J. Fortin, M. Manseau, D. O'Brien. (2007) Spatial graphs: Principles and applications for habitat connectivity. Ecosystems 10:448:461.

Galpern, P., M. Manseau. (2013a) Finding the functional grain: comparing methods for scaling resistance surfaces. Landscape Ecology 28:1269-1291.

Galpern, P., M. Manseau. (2013b) Modelling the influence of landscape connectivity on animal distribution: a functional grain approach. Ecography 36:1004-1016.

Galpern, P., M. Manseau, A. Fall. (2011) Patch-based graphs of landscape connectivity: a guide to construction, analysis, and application for conservation. Biological Conservation 144:44-55.

Galpern, P., M. Manseau, P.J. Wilson. (2012) Grains of connectivity: analysis at multiple spatial scales in landscape genetics. Molecular Ecology 21:3996-4009.

See Also

`GOC()`

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
## Very quick visualization at the finest scale/grain/threshold
tinyPatchGOCgrain <- grain(tinyPatchGOC, whichThresh = 1)
if (interactive()) {
```

```
  plot(tinyPatchGOCgrain, col = topo.colors(10))
}

## Visualize the model at the finest scale/grain/threshold
## Manual control of plotting
if (interactive()) {
  plot(
    grain(tinyPatchGOC, whichThresh = 1)@voronoi,
    col = sample(rainbow(100)),
    legend = FALSE,
    main = "Threshold 1"
  )
}
```

`grain-class` *The* `grain` *class*

Description

The `grain` class

Slots

`voronoi` A `SpatRaster` describing the regions of proximity in resistance units around the focal patches or points.

`summary` A summary of the the grains of connectivity generated and their properties.

`centroids` An `sf` object indicating the grain's polygon centroids.

`th` A list of `igraph` objects giving the graphs describing the relationship among the polygons in that grain

See `grain()` for more information.

Author(s)

Alex Chubaty and Paul Galpern

`grainscape-maps` *Test maps included with* `grainscape`

Description

Simple, artificial land cover maps intended for users to explore the functionality of the package. Each map has discrete integer land cover classes representing distinct land cover types. Typical analyses begin by reclassifying these to reflect resistance to movement (see the examples below and the package vignette).

Format

Three *ESRI ArcASCII* (`.asc`) raster grids stored in `extdata` and read with `terra::rast()`: `patchy.asc` and `fragmented.asc` are 400 x 400 cells, `tiny.asc` is 100 x 100 cells. Cell values are integer land cover classes (1-5 for `patchy.asc`, 1-4 for `fragmented.asc` and `tiny.asc`), with no associated coordinate reference system.

Details

The maps are distributed as *ESRI ArcASCII* (`.asc`) grids in the package's `extdata` directory and are read with `terra::rast()`. They use an arbitrary unprojected coordinate system (no CRS) with a unit cell size.

`patchy.asc` A caricatured map of five land cover classes (integer values 1-5), where patches are large, easy to identify polygonal regions for heuristic purposes. This unrealistic map can be used to illustrate the method and understand how it works. The map also serves a similar heuristic purpose in a review of graph-based connectivity methods (Galpern *et al.*, 2011). (400 x 400 raster cells.)

`fragmented.asc` A simulated land cover map with four land cover classes (integer values 1-4), produced using an algorithm that generates fragmentation. (400 x 400 raster cells.)

`tiny.asc` Similar to `fragmented.asc` (four land cover classes, integer values 1-4) but smaller in extent for lightning-fast computation and experimental use. (100 x 100 raster cells.)

Source

Simulated, artificial land cover maps created for demonstrating `grainscape`. `patchy.asc` also illustrates graph-based connectivity methods in Galpern *et al.* (2011) (see references in the package documentation, grainscape-package).

Examples

```
## load a bundled map and reclassify it into a resistance surface
patchy <- terra::rast(
  system.file("extdata", "patchy.asc", package = "grainscape", mustWork = TRUE)
)
patchyCost <- terra::classify(patchy, rcl = cbind(1:5, c(1, 10, 8, 3, 6)))
terra::plot(patchyCost)
```

---

graphdf — *Produce a* `data.frame` *containing the structure and associated attributes*

Description

Produce a `data.frame` containing the node (vertex) and link (edge) structure as well as the associated attributes for these. This provides an easy way to create data tables describing graphs, particularly helpful for users unfamiliar with the structure of `igraph` objects.

Usage

```
graphdf(x, ...)

## S4 method for signature 'list'
graphdf(x, ...)

## S4 method for signature 'goc'
graphdf(x, ...)

## S4 method for signature 'grain'
graphdf(x, ...)

## S4 method for signature 'mpg'
graphdf(x, ...)

## S4 method for signature 'igraph'
graphdf(x, ...)
```

Arguments

x	A goc, mpg, igraph, or list object.
...	Additional arguments (not used).

Value

A list object containing:

v node (vertex) names and associated attributes;

e link (edge) lists and associated attributes.

Please see MPG() and GOC() for details about the attributes.

For GOC() objects which typically contain multiple thresholds, an enumerated list of the same length as the number of thresholds is returned each containing v and e elements.

Author(s)

Paul Galpern and Alex Chubaty

See Also

MPG(), GOC()

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)
```

```
## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
## Create a data.frame with the structure and attributes of a MPG object
tinyPatchMPG_df <- graphdf(tinyPatchMPG)

## Create a data.frame with the structure and attributes of a GOC object
tinyPatchGOC_df <- graphdf(tinyPatchGOC)

## Create a data.frame with the structure and attributes of any igraph object
graphdf(tinyPatchGOC@th[[1]]$goc)
```

`MPG` *Extract a minimum planar graph (MPG) model from a landscape resistance surface*

Description

Extracts a minimum planar graph (MPG) and is also the first step in grains of connectivity (GOC) modelling. Both patch-based and lattice MPGs can be extracted.

Usage

```
MPG(cost, patch, ...)

## S4 method for signature 'SpatRaster,SpatRaster'
MPG(cost, patch, ...)

## S4 method for signature 'SpatRaster,numeric'
MPG(cost, patch, ...)
```

Arguments

`cost`	A `SpatRaster` giving a landscape resistance surface, where the values of each raster cell are proportional to the resistance to movement, dispersal, or gene flow for an organism in the landscape feature they represent. Missing values `NA` are acceptable (but see below). Negative values are not. To extract an MPG with Euclidean links (i.e., and not least-cost path links) set `cost[] <- 1`.
`patch`	A raster of class `SpatRaster` for a patch-based analysis OR an integer for a lattice analysis. If a raster is given it must be of the same extent, origin and projection as `cost` and be binary, without missing values, where patches=1 and non-patches=0. For lattice analyses, an integer gives the spacing in raster cells between focal points in the lattice.
`...`	Additional arguments (not used).

Details

Use this function to create a minimum planar graph (MPG) that can be further analyzed using `igraph::igraph()` routines. It is also the first step in grains of connectivity (GOC) modelling.

Value

A `mpg()` object.

Note

Researchers should consider whether the use of a patch-based MPG or a lattice MPG model is appropriate based on the patch-dependency of the organism under study. Patch-based models make most sense when animals are restricted to, or dependent on, a resource patch. Lattice models can be used as a generalized and functional approach to scaling resistance surfaces.

Rasters should be projected and not in geographic coordinates (i.e. `terra::crs(cost)` should not contain `"longlat"`) or the function will issue a warning. In unprojected cases consider using `terra::project()` to change to an appropriate coordinate system for the location and extent of interest that balances both distance and areal accuracy. See https://spatialreference.org/ for location-specific suggestions. Use of geographic coordinates will result in inaccurate areal and distance measurements, rendering the models themselves inaccurate.

Author(s)

Paul Galpern, Sam Doctolero, Alex Chubaty

References

Fall, A., M.-J. Fortin, M. Manseau, D. O'Brien. (2007) Spatial graphs: Principles and applications for habitat connectivity. Ecosystems 10:448:461.

Galpern, P., M. Manseau. (2013a) Finding the functional grain: comparing methods for scaling resistance surfaces. Landscape Ecology 28:1269-1291.

Galpern, P., M. Manseau. (2013b) Modelling the influence of landscape connectivity on animal distribution: a functional grain approach. Ecography 36:1004-1016.

Galpern, P., M. Manseau, A. Fall. (2011) Patch-based graphs of landscape connectivity: a guide to construction, analysis, and application for conservation. Biological Conservation 144:44-55.

Galpern, P., M. Manseau, P.J. Wilson. (2012) Grains of connectivity: analysis at multiple spatial scales in landscape genetics. Molecular Ecology 21:3996-4009.

See Also

`GOC()`, `threshold()`

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)
```

```
## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Explore the graph structure and node/link attributes
graphdf(tinyPatchMPG)

## Find the mean patch area (see igraph manual for use of V() and E())
mean(igraph::V(tinyPatchMPG@mpg)$patchArea)

## Quick visualization of the MPG
if (interactive()) {
  plot(tinyPatchMPG, col = c("grey", "black"), legend = FALSE)
}

## Additional graph extraction scenarios
## Produce a lattice MPG where focal points are spaced 10 cells apart
tinyLatticeMPG <- MPG(cost = tinyCost, patch = 10)
if (interactive()) {
  plot(tinyLatticeMPG)
}
```

`mpg-class` *The* `mpg` *class*

Description

The `mpg` class

Slots

`mpg` The minimum planar graph as class `igraph`.

`patchId` The input `patch` raster with patch cells assigned to their id (`SpatRaster`).

`voronoi` The Voronoi tessellation of the patches and resistance surface (`SpatRaster`).

`lcpPerimWeight` The paths of the links between patches and their accumulated costs (`SpatRaster`).

`lcpLinkId` The paths of the links between patches and their id (`SpatRaster`).

`mpgPlot` A `SpatRaster` version of the `mpg`, which can be easily plotted to visualize the MPG.

The `mpg` slot contains useful vertex and edge attributes. Vertex attributes give attributes of patches including patch area, the area of patch edges, the core area of each patch, and the coordinates of the patch centroid. All areal measurements are given as raster cell counts. Edge attributes give attributes of the graph links including link weights giving accumulated resistance/least-cost path distance, Euclidean distance, and the start and end coordinates of each link.

Author(s)

Alex Chubaty and Paul Galpern

patchFilter *Filter out patches smaller than a specified area*

Description

Pre-process patch rasters prior to their use with MPG().

Usage

```
patchFilter(x, cells = NULL, area = NULL, ...)

## S4 method for signature 'SpatRaster'
patchFilter(x, cells = NULL, area = NULL, ...)
```

Arguments

x	A binary raster (i.e. consisting of 0, 1, or NA cells), where cells =1 represent patches
cells	The minimum number of cells that constitute a patch. Default NULL. Only one of cells or area may be specified.
area	The minimum area that constitutes a patch (where area is calculated in the coordinate reference system of the raster by multiplying the count of cells in patch by the x and y resolution of a raster cell). Default NULL. Only one of cells or area may be specified.
...	Additional arguments passed to terra::patches(). For example, directions = 4 may be used to be more conservative about which cells constitute a patch.

Details

It examines a binary raster to identify all patches or clumps of cells with values =1, determines their area, and returns a binary raster where only patches of area greater than or equal to the specified amount are represented.

This is helpful when analyzing habitat connectivity models where patches represent a land cover or habitat type. For example, a raster may have patches of a certain habitat type of insufficient area to support the ecological process of interest. Another use case is remote sensing classification errors that have introduced artifacts. Filtering can help in both cases.

Value

A binary raster where all patches (i.e. clumped areas =1) are greater than the specified area.

Author(s)

Paul Galpern and Alex Chubaty

See Also

MPG()

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features = 10
## and all patches are greater than or equal to 2 cells in size
filteredPatch <- patchFilter(tinyCost == 10, cells = 2)
tinyPatchMPG <- MPG(cost = tinyCost, patch = filteredPatch)
if (interactive()) {
  plot(tinyPatchMPG)
}

## Compare to removal of patches greater than or equal to 40 cells in size!
filteredPatch <- patchFilter(tinyCost == 10, cells = 40)
tinyPatchMPG <- MPG(cost = tinyCost, patch = filteredPatch)
if (interactive()) {
  plot(tinyPatchMPG)
}

## Use a rook/castle 4-direction case rather than the queen 8-direction case
## to identify neighbouring cells in a patch
filteredPatch <- patchFilter(tinyCost == 10, cells = 40, directions = 4)
tinyPatchMPG <- MPG(cost = tinyCost, patch = filteredPatch)
if (interactive()) {
  plot(tinyPatchMPG)
}
```

---

plot,corridor,ANY-method

Plot quick visualizations of grainscape *objects*

Description

Plot quick visualizations of mpg, grain, and corridor objects.

This function is intended to get a quick look at the state of a grainscape object by rendering what are likely to be the most universally useful visualizations of the spatial data within these objects.

Much more control is available using ggGS() with ggplot2::ggplot() enabling the layering of different different analytical products, and the visualization of node and link attributes.

For high-resolution visualization and the greatest level of control use export() to export spatial objects for cartographic representation in a geographic information system (GIS).

Usage

```
## S4 method for signature 'corridor,ANY'
plot(x, y, quick = NULL, print = TRUE, theme = TRUE, ...)

## S4 method for signature 'grain,ANY'
plot(x, y, quick = NULL, print = TRUE, theme = TRUE, ...)

## S4 method for signature 'mpg,ANY'
plot(x, y, quick = NULL, print = TRUE, theme = TRUE, ...)
```

Arguments

x	A grainscape object (corridor, grain, or mpg).
y	Ignored.
quick	If NULL (the default) it will plot the most useful quick visualization for the supplied object type. See below for a description of the available quick plots, and the defaults.
print	Render the ggplot on the default graphics device. Default is TRUE.
theme	Apply grainscape theme and scale aesthetics. Default is TRUE.
...	Additional arguments (not used).

Value

```
Invisibly, a `ggplot2` object to which additional `ggplot`
        geoms and adjustments can be applied. Has the side effect of
        rendering the plot, unless `print = FALSE`.
```

Types of visualization available with the quick parameter

"mpgPerimPlot" gives a a vector rendering of the minimum planar graph with vector links connecting the perimeters of the patches. This doesn't accurately represent the sinuosity of paths of the links between patches but offers a good approximation that renders better at large extents. Default for mpg objects. Not available for other objects.

"mpgPlot" gives a raster-only rendering of the minimum planar graph where patchId are positive integers, and linkId are negative integers showing the shortest paths between patches Only available for mpg objects.

"network" gives a vector rendering of the minimum planar graph or the grains of connectivity network with nodes and links plotted at the patch or polygon centroid locations. Available for mpg and grain objects. Default for grain objects.

"grainPlot" gives a raster and vector rendering of the grains of connectivity network with nodes and links plotted at polygon centroid locations, superimposed over the boundaries of the Voronoi polygons. Can be time consuming on large rasters due to the Voronoi boundary extraction. Only available for grain objects.

"corridorPlot" renders the output of a corridor() analysis. It is the only option available with corridor objects and the default.

Author(s)

Alex Chubaty and Paul Galpern

See Also

ggGS(), export(), corridor, grain, mpg

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
if (interactive()) {
  library(ggplot2)

  ## MPG and showing simplified links among the perimeters of patches
  plot(tinyPatchMPG)

  ## MPG showing links among the nodes of connected patches
  plot(tinyPatchMPG, quick = "network")

  ## MPG showing the shortest paths between patches actually used to
  ## to calculate link weight values
  plot(tinyPatchMPG, quick = "mpgPlot")

  ## A grain of connectivity network plot with Voronoi boundaries
  plot(grain(tinyPatchGOC, 3), quick = "grainPlot")

  ## Capture plot output for further processing with ggplot
  g <- plot(tinyPatchMPG, print = FALSE, theme = FALSE)
  g <- g +
    theme_minimal() +
    ggtitle("Minimum planar graph") +
    theme(plot.title = element_text(size = 20, hjust = 0.5)) +
    theme(legend.position = "none") +
    xlab("Easting") +
    ylab("Northing")
  g

  ## To change aesthetics it is best to build the plot from scratch
  ## using grainscape::ggGS(). See examples therein.
}
```

plotResistance *Plot a resistance surface with a consistent colour scheme*

Description

Draws a resistance SpatRaster using a shared colour scheme (green for low resistance through red for high resistance) with a discrete-swatch legend placed to the left of the panel. Each resistance value is mapped to a fixed colour so that the same value appears identical across figures, which is useful when comparing several landscapes or when pairing a resistance surface with a network plot (see plotWithResistance()).

Usage

```
plotResistance(x, maxResistance = 12L)
```

Arguments

x	A resistance SpatRaster, e.g. the cost surface passed to MPG().
maxResistance	Integer. The resistance value mapped to the top (red) end of the colour ramp; values from 1 to maxResistance receive fixed colours so the scheme stays consistent across surfaces. Defaults to 12, and is automatically raised to the largest value present in x if that exceeds it.

Value

A ggplot2::ggplot() object.

Author(s)

Alex M Chubaty

See Also

plotWithResistance(), ggGS()

Examples

```
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
plotResistance(tinyCost)
```

`plotWithResistance` *Show a resistance surface beside a network plot*

Description

Places a resistance-surface reference panel (see `plotResistance()`) to the left of a network plot – for example one produced by `plot()` on an `mpg` or `grain` object, or any **ggplot2** plot – so the two can be compared at the same size as a visual sanity check. Both panels are drawn at equal size (using `cowplot::plot_grid()` with `align = "h"` and equal coordinates) and the resistance legend is placed at the far left.

Requires the **cowplot** package.

Usage

```
plotWithResistance(x, gg, maxResistance = 12L)
```

Arguments

`x`	A resistance `SpatRaster` shown as the reference panel on the left (passed to `plotResistance()`).
`gg`	A plot shown on the right, e.g. the result of `plot()` on an `mpg` or `grain` object (typically with `theme = FALSE`), or any `ggplot2::ggplot()` object.
`maxResistance`	Passed to `plotResistance()`.

Value

A two-panel plot grid produced by `cowplot::plot_grid()`.

Author(s)

Alex M Chubaty

See Also

`plotResistance()`

Examples

```
if (requireNamespace("cowplot", quietly = TRUE)) {
  tiny <- terra::rast(
    system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
  )
  tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
  tinyMPG <- MPG(tinyCost, patch = tinyCost == 1)
  plotWithResistance(tinyCost, plot(tinyMPG, quick = "mpgPlot", theme = FALSE))
}
```

point *Identify the polygons containing locations in grains of connectivity (GOC) tessellations*

Description

Identify the polygon containing a location at multiple scales.

Usage

```
point(x, ...)

## S4 method for signature 'goc'
point(x, coords, ...)
```

Arguments

x	A goc object produced by GOC().
...	Additional arguments (not used).
coords	A two-column matrix or an sf (POINT) object giving the coordinates of points of interest.

Value

A list with elements:

pointPolygon a matrix with elements giving the id of the polygon from the goc, where rows give points of interest and columns give thresholds;

pointTotalPatchArea is a matrix with elements giving the area of patches in a polygon (in cell counts), where rows give points of and columns give thresholds;

pointTotalCoreArea the same for core area of patches;

pointECS gives the patch area (in cell counts) averaged for all points of interest (defined by O'Brien *et al.*, 2006);

pointECSCore is the same for the core area of patches.

Note

See MPG() for warning related to areal measurements.

Author(s)

Paul Galpern and Alex Chubaty

References

Fall, A., M.-J. Fortin, M. Manseau, D. O'Brien. (2007) Spatial graphs: Principles and applications for habitat connectivity. Ecosystems 10:448:461.

Galpern, P., M. Manseau. (2013a) Finding the functional grain: comparing methods for scaling resistance surfaces. Landscape Ecology 28:1269-1291.

Galpern, P., M. Manseau. (2013b) Modelling the influence of landscape connectivity on animal distribution: a functional grain approach. Ecography 36:1004-1016.

Galpern, P., M. Manseau, A. Fall. (2011) Patch-based graphs of landscape connectivity: a guide to construction, analysis, and application for conservation. Biological Conservation 144:44-55.

Galpern, P., M. Manseau, P.J. Wilson. (2012) Grains of connectivity: analysis at multiple spatial scales in landscape genetics. Molecular Ecology 21:3996-4009.

O'Brien, D., M. Manseau, A. Fall, and M.-J. Fortin. (2006) Testing the importance of spatial configuration of winter habitat for woodland caribou: An application of graph theory. Biological Conservation 130:70-83.

See Also

GOC(), distance()

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Extract a representative subset of 5 grains of connectivity
tinyPatchGOC <- GOC(tinyPatchMPG, nThresh = 5)
## Three sets of coordinates in the study area
loc <- cbind(c(30, 60, 90), c(30, 60, 90))

## Find the GOC polygon containing these three locations
## for each of the 5 grains of connectivity
tinyPts <- point(tinyPatchGOC, loc)
```

---

show,goc-method *Show a* grainscape *object*

Description

Custom show method to safely print the contents of a goc or grain object.

Usage

```
## S4 method for signature 'goc'
show(object)

## S4 method for signature 'grain'
show(object)

## S4 method for signature 'corridor'
show(object)
```

Arguments

object	A goc() or grain() object.

---

theme_grainscape *A* ggplot2 *theme for* grainscape

Description

A **ggplot2** theme designed for grainscape based on the ggthemes::theme_map() theme, with several modifications.

Usage

```
theme_grainscape(base_size = 9, base_family = "")
```

Arguments

base_size	Base font size
base_family	Base font family

Value

A theme suitable for use with ggplot2::ggplot()

Author(s)

Paul Galpern and Alex Chubaty

See Also

ggGS(), plot(), ggthemes::theme_map()

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
if (interactive()) {
  library(ggplot2)

  ## Plot the patches in a minimum planar graph
  theme_set(theme_grainscape())
  plot(tinyPatchMPG, quick = "mpgPlot")
}
```

---

threshold *Produce a minimum planar graph (MPG) at multiple scales*

Description

Perform a scalar analysis of a minimum planar graph (MPG) by building the graph at a series of link thresholds. As the threshold value increases more nodes in the graph become connected, forming increasingly fewer components, until the graph becomes connected (e.g., Brooks, 2003). N.B. Grains of connectivity (GOC) done by `GOC()` is also a scalar analysis using Voronoi tessellations rather than patches (see Galpern *et al.*, 2012).

Usage

```
threshold(x, ...)

## S4 method for signature 'mpg'
threshold(x, weight = "lcpPerimWeight", nThresh = NULL, doThresh = NULL, ...)
```

Arguments

`x`	A `mpg` object produced by `MPG()`.
`...`	Additional arguments (not used).
`weight`	A string giving the link weight or attribute to use for threshold. `"lcpPerimWeight"` uses the accumulated resistance or least-cost path distance from the perimeters of patches as the link weight.
`nThresh`	Optional. An integer giving the number of thresholds (or scales) at which to create GOC models. Thresholds are selected to produce a maximum number of unique grains (i.e., models). `nThresh` thresholds are also approximately evenly spread between 0 and the threshold at which all patches or focal points on the

landscape are connected. This is a simple way to get a representative subset of all possible GOC models. Provide either `nThresh` or `doThresh` not both.

`doThresh` Optional. A vector giving the link thresholds at which to create GOC models. Use `threshold()` to identify thresholds of interest. Provide either `nThresh` or `doThresh` not both.

Value

A list object with the following elements:

`summary` summarizes the thresholded graphs generated and their properties;

`th` a list of length `nThresh` or `length(doThresh)` giving the thresholded graph (class `igraph`) at each threshold.

Note

See `MPG()` for warning related to areal measurements.

Author(s)

Paul Galpern and Alex Chubaty

References

Brooks, C.P. (2003) A scalar analysis of landscape connectivity. Oikos 102:433-439.

Fall, A., M.-J. Fortin, M. Manseau, D. O'Brien. (2007) Spatial graphs: Principles and applications for habitat connectivity. Ecosystems 10:448:461.

Galpern, P., M. Manseau. (2013a) Finding the functional grain: comparing methods for scaling resistance surfaces. Landscape Ecology 28:1269-1291.

Galpern, P., M. Manseau. (2013b) Modelling the influence of landscape connectivity on animal distribution: a functional grain approach. Ecography 36:1004-1016.

Galpern, P., M. Manseau, A. Fall. (2011) Patch-based graphs of landscape connectivity: a guide to construction, analysis, and application for conservation. Biological Conservation 144:44-55.

Galpern, P., M. Manseau, P.J. Wilson. (2012) Grains of connectivity: analysis at multiple spatial scales in landscape genetics. Molecular Ecology 21:3996-4009.

See Also

`MPG()`

Examples

```
## Load raster landscape
tiny <- terra::rast(
  system.file("extdata", "tiny.asc", package = "grainscape", mustWork = TRUE)
)

## Create a resistance surface from a raster using an is-becomes reclassification
```

```
tinyCost <- terra::classify(tiny, rcl = cbind(c(1, 2, 3, 4), c(1, 5, 10, 12)))
## Produce a patch-based MPG where patches are resistance features=1
tinyPatchMPG <- MPG(cost = tinyCost, patch = tinyCost == 1)
## Threshold this graph at a representative subset of 10 thresholds
tinyThresh <- threshold(tinyPatchMPG, nThresh = 10)

## Examine the properties of one of these threshold graphs
print(tinyThresh$th[[7]], vertex.attributes = TRUE, edge.attributes = TRUE)
```

$ *Extract or Replace Parts of an Object*

Description

Extract or Replace Parts of an Object

Usage

```
## S4 method for signature 'goc'
x$name

## S4 replacement method for signature 'goc'
x$name <- value

## S4 method for signature 'mpg'
x$name

## S4 replacement method for signature 'mpg'
x$name <- value
```

Arguments

x	A simList object from which to extract element(s) or in which to replace element(s).
name	A literal character string or a name.
value	Any R object.

Index

∗ **connectivity**
grainscape-package, 3
∗ **graph**
grainscape-package, 3
∗ **maps**
grainscape-maps, 19
∗ **minimum**
grainscape-package, 3
∗ **planar**
grainscape-package, 3
∗ **spatial**
grainscape-package, 3
$, 36
$,goc-method *($)*, 36
$,mpg-method *($)*, 36
$<- *($)*, 36
$<-,goc-method *($)*, 36
$<-,mpg-method *($)*, 36

corridor, 4, *5*, *28*
corridor(), *6*, *27*
corridor,goc-method *(corridor)*, 4
corridor-class, 6

distance, 6
distance(), *4*, *16, 17*, *32*
distance,goc,matrix-method *(distance)*, 6
distance,goc,numeric-method *(distance)*, 6
distance,goc,sf-method *(distance)*, 6

export, 8
export(), *26*, *28*
export,goc-method *(export)*, 8
export,grain-method *(export)*, 8
export,mpg-method *(export)*, 8

ggGS, 12
ggGS(), *26*, *28, 29*, *33*
ggGS,goc-method *(ggGS)*, 12
ggGS,grain-method *(ggGS)*, 12
ggGS,list-method *(ggGS)*, 12
ggGS,mpg-method *(ggGS)*, 12
ggGS,SpatRaster-method *(ggGS)*, 12
ggplot2::ggplot(), *12*, *26*, *29, 30*, *33*
ggthemes::theme_map(), *33*
GOC, 14
goc, *4*, *31*
GOC(), *4, 5*, *7*, *11*, *13*, *17, 18*, *21*, *23*, *31, 32*, *34*
goc(), *15*, *33*
GOC,mpg-method *(GOC)*, 15
goc-class, 17
grain, 17, *28*
grain(), *5*, *11*, *16*, *19*, *33*
grain,goc-method *(grain)*, 17
grain-class, 19
grainscape *(grainscape-package)*, 3
grainscape-maps, 19
grainscape-package, 3, *20*
graphdf, 20
graphdf,goc-method *(graphdf)*, 20
graphdf,grain-method *(graphdf)*, 20
graphdf,igraph-method *(graphdf)*, 20
graphdf,list-method *(graphdf)*, 20
graphdf,mpg-method *(graphdf)*, 20

igraph::igraph(), *17*, *23*

MPG, 22
mpg, *28*
MPG(), *11*, *13*, *15, 16*, *21*, *25, 26*, *29*, *31*, *34, 35*
mpg(), *23*
MPG,SpatRaster,numeric-method *(MPG)*, 22
MPG,SpatRaster,SpatRaster-method *(MPG)*, 22
mpg-class, 24

name, *36*

patchFilter, 25

patchFilter,SpatRaster-method
(patchFilter), 25
plot(), *30*, *33*
plot,corridor,ANY-method, 26
plot,grain,ANY-method
(plot,corridor,ANY-method), 26
plot,mpg,ANY-method
(plot,corridor,ANY-method), 26
plotResistance, 29
plotResistance(), *30*
plotWithResistance, 30
plotWithResistance(), *29*
point, 31
point(), *7*, *16*
point,goc-method (point), 31

sf::st_write(), *8*
show,corridor-method (show,goc-method),
32
show,goc-method, 32
show,grain-method (show,goc-method), 32

terra::patches(), *25*
terra::project(), *23*
terra::rast(), *20*
terra::writeRaster(), *8*, *9*
theme_grainscape, 33
threshold, 34
threshold(), *15*, *23*, *35*
threshold,mpg-method (threshold), 34