Package 'Grid2Polygons'

February 9, 2017

Title Convert Spatial Grids to Polygons

Version 0.1.6

Description Converts a spatial object from class SpatialGridDataFrame to SpatialPolygonsDataFrame.

Depends R (>= 2.15.0)

Imports methods, raster, rgeos, sp

Suggests rgdal, roxygen2

License GPL (>= 2)

URL https://github.com/jfisher-usgs/Grid2Polygons

BugReports https://github.com/jfisher-usgs/Grid2Polygons/issues

Encoding UTF-8

LazyData true

ByteCompile true

RoxygenNote 6.0.1

NeedsCompilation yes

Author Jason C. Fisher [aut, cre]

Maintainer Jason C. Fisher <jfisher@usgs.gov>

Repository CRAN

Date/Publication 2017-02-09 00:55:15

R topics documented:

Grid2Polygons *Convert Spatial Grids to Polygons*

Description

Converts **sp** spatial objects from class SpatialGridDataFrame to SpatialPolygonsDataFrame. Spatial polygons can then be transformed to a different projection or datum with spTransform in package **rgdal**. Image files created with spatial polygons are reduced in size and result in a much "cleaner" version of your image.

Usage

```
Grid2Polygons(grd, zcol = 1, level = FALSE, at, cuts = 20,
  pretty = FALSE, xlim = NULL, ylim = NULL, ply = NULL)
```

Arguments

grd	SpatialGridDataFrame. Spatial grid data frame
zcol	character or integer. Attribute name or column number in attribute table.
level	logical. If true, a set of levels is used to partition the range of z, its default is false.
at	numeric. A vector giving breakpoints along the range of z.
cuts	integer. Number of levels the range of z would be divided into.
pretty	logical. Whether to use pretty cut locations.
xlim	numeric. Vector of length 2 giving left and right limits of the spatial grid, data outside these limits is excluded.
ylim	numeric. Vector of length 2 giving lower and upper limits of the spatial grid, data outside these limits is excluded.
ply	SpatialPolygons, or SpatialGridDataFrame. Cropping polygon

Value

Returns an object of SpatialPolygonsDataFrame. The objects data slot is a data frame, number of rows equal to the number of Polygons objects and a single column containing values of z. If level is true, z values are set equal to the midpoint between breakpoints. The status of the polygon as a hole or an island is taken from the ring direction, with clockwise meaning island, and counter-clockwise meaning hole.

Note

The traditional R graphics model does not draw polygon holes correctly, holes overpaint their containing Polygon object using a user defined background color (white by default). Polygon holes are now rendered correctly using the plot method for spatial polygons (SpatialPolygons-class), see polypath for more details. The Trellis graphics model appears to rely on the traditional method so use caution when plotting with spplot.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

References

A general explanation of the algorithm provided here; inspiration provided here.

See Also

SpatialPolygons

Examples

```
# Example 1

z <- c(1.1,  1.5,  4.2,  4.1,  4.3,  4.7,
       1.2,  1.4,  4.8,  4.8,   NA,  4.1,
       1.7,  4.2,  1.4,  4.8,  4.0,  4.4,
       1.1,  1.3,  1.2,  4.8,  1.6,   NA,
       3.3,  2.9,   NA,  4.1,  1.0,  4.0)
m <- 5
n <- 6
x <- rep(0:n, m + 1)
y <- rep(0:m, each = n + 1)
xc <- c(rep(seq(0.5, n - 0.5, by = 1), m))
yc <- rep(rev(seq(0.5, m - 0.5, by = 1)), each = n)
grd <- data.frame(z = z, xc = xc, yc = yc)
sp::coordinates(grd) <- ~ xc + yc
sp::gridded(grd) <- TRUE
grd <- as(grd, "SpatialGridDataFrame")
image(grd, col = gray.colors(30), axes = TRUE)
grid(col = "black", lty = 1)
points(x = x, y = y, pch = 16)
text(cbind(xc, yc), labels = z)
text(cbind(x = x + 0.1, y = rev(y + 0.1)), labels = 1:((m + 1) * (n + 1)), cex = 0.6)

at <- 1:ceiling(max(z, na.rm = TRUE))
plys <- Grid2Polygons(grd, level = TRUE, at = at)
cols <- rainbow(length(plys), alpha = 0.3)
sp::plot(plys, add = TRUE, col = cols)
zz <- plys[[1]]
legend("top", legend = zz, fill = cols, bty = "n", xpd = TRUE,
       inset = c(0, -0.1), ncol = length(plys))

p1 <- sp::Polygon(rbind(c(1.2, 0.5), c(5.8, 1.7), c(2.5, 5.1), c(1.2, 0.5)),
                  hole = FALSE)
p2 <- sp::Polygon(rbind(c(2.5, 2.5), c(3.4, 1.8), c(3.7, 3.1), c(2.5, 2.5)),
                  hole = TRUE)
p3 <- sp::Polygon(rbind(c(-0.3, 3.3), c(1.7, 5.1), c(-1.0, 7.0), c(-0.3, 3.3)),
                  hole = FALSE)
p <- sp::SpatialPolygons(list(sp::Polygons(list(p1, p2, p3), 1)))
plys <- Grid2Polygons(grd, level = TRUE, at = at, ply = p)
```

```
cols <- rainbow(length(zz), alpha = 0.6)[zz %in% plys[[1]]]
sp::plot(plys, col = cols, add = TRUE)

# Example 2

data(meuse.grid, package = "sp")
sp::coordinates(meuse.grid) <- ~ x + y
sp::gridded(meuse.grid) <- TRUE
meuse.grid <- as(meuse.grid, "SpatialGridDataFrame")
meuse.plys <- Grid2Polygons(meuse.grid, "dist", level = FALSE)
op <- par(mfrow = c(1, 2), oma = rep(0, 4), mar = rep(0, 4))
sp::plot(meuse.plys, col = heat.colors(length(meuse.plys)))
title("level = FALSE", line = -7)

meuse.plys.lev <- Grid2Polygons(meuse.grid, "dist", level = TRUE)
sp::plot(meuse.plys.lev, col = heat.colors(length(meuse.plys.lev)))
title("level = TRUE", line = -7)
par(op)
```

Index