Package ‘isoband’

April 7, 2019

Title  Generate Isolines and Isobands from Regularly Spaced Elevation Grids

Version  0.2.0

Description  A fast C++ implementation to generate contour lines (isolines) and contour polygons (isobands) from regularly spaced grids containing elevation data.

URL  https://github.com/wilkelab/isoband

BugReports  https://github.com/wilkelab/isoband/issues

License  MIT + file LICENSE

Encoding  UTF-8

LazyData  true

LinkingTo  Rcpp, testthat

Imports  Rcpp, grid, utils

RoxygenNote  6.1.1

Suggests  covr, ggplot2, knitr, lwgeom, magick, microbenchmark, rmarkdown, sf, testthat

SystemRequirements  C++11

VignetteBuilder  knitr

NeedsCompilation  yes

Author  Claus Wilke [aut, cre]

Maintainer  Claus Wilke <wilke@austin.utexas.edu>

Repository  CRAN

Date/Publication  2019-04-06 22:20:03 UTC

R topics documented:

| angle_halfcircle_bottom                     | 2 |
| clip_lines                                  | 3 |
| isoband                                    | 3 |
| isobands                                   | 3 |
angle_halfcIRCLE_bottom

Standardize label angles

Description

Function factories that return functions to standardize rotation angles to specific angle ranges.

Usage

angle_halfcIRCLE_bottom()

angle_halfcIRCLE_right()

angle_fixed(theta = 0)

angle_identity()

Arguments

theta Fixed angle, in radians.

Details

angle_halfcIRCLE_bottom() standardizes angles to (-pi/2, pi/2].

angle_halfcIRCLE_right() standardizes angles to (0, pi].

angle_fixed() sets all angles to a fixed value (0 by default).

angle_identity() does not modify any angles.
**clip_lines**

*Clip lines so they don’t run into a set of boxes.*

**Description**

Clip lines so they don’t run into a set of boxes. Useful for labeling isolines, as it allows removal of line segments that would run into any text labels.

**Usage**

```r
clip_lines(x, y, id, clip_boxes, asp = 1)
```

**Arguments**

- `x`: Numeric vector of x coordinates
- `y`: Numeric vector of y coordinates
- `id`: Integer vector of id numbers indicating which lines are connected
- `clip_boxes`: Data frame specifying the locations of boxes to clip to. Should have five columns, named `x`, `y`, `width`, `height`, `theta`, which specify the x and y positions of each box midpoint, as well as the box width, box height, and box angle in radians. Each box is specified by one data row.
- `asp`: Aspect ratio (width/height) of the target canvas. This is used to convert widths to heights and vice versa for rotated boxes

**isoband**

*Generate isolines and isobands*

**Description**

Generate isolines and isobands

**isobands**

*Efficient calculation of isolines and isobands from elevation grid*

**Description**

Efficient calculation of isolines and isobands from elevation grid

**Usage**

```r
isobands(x, y, z, levels_low, levels_high)
```

```r
isolines(x, y, z, levels)
```
isobands

Arguments

x  Numeric vector specifying the x locations of the grid points.
y  Numeric vector specifying the y locations of the grid points.
z  Numeric matrix specifying the elevation values for each grid point.
levels_low, levels_high
Numeric vectors of minimum/maximum z values for which isobands should be
generated. Any z values that are exactly equal to a value in levels_low are
considered part of the corresponding isoband, but any z values that are exactly
equal to a value in levels_high are not considered part of the corresponding
isoband. In other words, the intervals specifying isobands are closed at their
lower boundary and open at their upper boundary.

levels  Numeric vector of z values for which isolines should be generated.

See Also

plot_iso

Examples

library(grid)

#' # one simple connected shape
m <- matrix(c(0, 0, 0, 0, 0, 0,
             0, 1, 1, 0, 0, 0,
             0, 0, 0, 1, 1, 0,
             0, 0, 0, 1, 0, 0,
             0, 0, 0, 0, 0, 1,
             0, 0, 0, 0, 0, 0), 6, 6, byrow = TRUE)
df_bands <- isobands((1:ncol(m))/(ncol(m)+1), (nrow(m):1)/(nrow(m)+1), m, 0.5, 1.5)[[1]]
df_lines <- isolines((1:ncol(m))/(ncol(m)+1), (nrow(m):1)/(nrow(m)+1), m, 0.5)[[1]]
g <- expand.grid(x = (1:ncol(m))/(ncol(m)+1), y = (nrow(m):1)/(nrow(m)+1))
grid.newpage()
grid.points(g$x, g$y, default.units = "npc", pch = 19, size = unit(0.5, "char"))
grid.path(df_bands$x, df_bands$y, df_bands$id, gp = gpar(fill = "cornsilk", col = NA))
grid.polyline(df_lines$x, df_lines$y, df_lines$id)

# a similar plot can be generated with the plot_iso() function,
# which is useful for exploring how the algorithm works
plot_iso(m, 0.5, 1.5)

# NAs are ignored
m <- matrix(c(NA, NA, NA, 0, 0, 0,
             NA, NA, NA, 1, 1, 0,
             0, 0, 1, 1, 1, 0,
             0, 1, 1, 0, 0, 0,
             0, 0, 0, 1, 0, 0,
             0, 0, 0, 0, 0, 0), 6, 6, byrow = TRUE)
plot_iso(m, 0.5, 1.5)
# two separate shapes
m <- matrix(c(0, 0, 1, 1, 
0, 1, 1, 1, 
1, 1, 0, 0, 
0, 0, 0.8, 0), 4, 4, byrow = TRUE)
plot_iso(m, 0.5, 1.5)

# shape with hole
m <- matrix(c(0, 0, 0, 0, 0, 
0, 1, 1, 1, 1, 0, 
0, 2, 2, 1, 0, 
0, 1, 1, 1, 0, 
0, 0, 0, 0, 0), 6, 6, byrow = TRUE)
plot_iso(m, 0.5, 1.5)

## isobands_grob

### Description

This function generates a grid grob that represents isobands.

### Usage

```r
isobands_grob(bands, gp = gpar(), units = "npc")
```

### Arguments

- **bands**: Isobands, as produced by the `isobands()` function.
- **gp**: Grid graphical parameters. Parameters are recycled among the total number of bands drawn.
- **units**: A character string specifying the units in which to interpret the isobands coordinates. Defaults to "npc".

### See Also

See `isolines_grob()` for drawing of isolines.

### Examples

```r
library(grid)

viridis_pal <- colorRampPalette(
c("#440154", "#414487", "#2A788E", "#22A884", "#7AD151", "#FDE725"),
space = "Lab"
)
x <- (1:ncol(volcano))/(ncol(volcano)+1)
```
y <- (nrow(volcano):1)/(nrow(volcano)+1)
bands <- isobands(x, y, volcano, 5*(18:38), 5*(19:39))

b <- isobands_grob(bands,
    gp = gpar(col = "black", fill = viridis_pal(21), alpha = 0.5)
)

grid.newpage()
grid.draw(b)

---

**isolines_grob**

*Render labeled isolines*

**Description**

This function generates a grid grob that represents labeled isolines.

**Usage**

```r
isolines_grob(lines, gp = gpar(), breaks = NULL, labels = NULL, margin = unit(c(1, 1, 1, 1), "pt"), label_col = NULL, label_alpha = NULL, label_placer = label_placer_minmax(), units = "npc")
```

**Arguments**

- **lines**: Isolines, as produced by the `isolines()` function.
- **gp**: Grid graphical parameters. Parameters applying to lines (such as col, lwd, lty, etc.) are recycled among the total number of lines drawn. Parameters applying only to labels (such as fontfamily, fontsize) are recycled among the specified breaks only. The two parameters col and alpha are also applied to labels, unless overridden (see `label_col` and `label_alpha`), but are matched to the corresponding lines.
- **breaks**: Character vector specifying the isolines that should be labeled. If NULL, labels all isolines.
- **labels**: Character vector specifying the labels for each break. If NULL, uses the breaks as labels. The number of labels provided must match the number of breaks provided.
- **margin**: Unit object of length 4 specifying the top, right, bottom, and left margins around each text label. The same margins are applied to all labels.
- **label_col**: Color applied to labels. Can be used to override the color provided in gp, in case labels and lines should have different colors.
- **label_alpha**: Alpha applied to labels. Can be used to override the alpha value provided in gp, in case labels and lines should have different alpha values.
iso_to_sfg

Convert isolines or isobands to sfg object

Description

Convert isolines or isobands to an sf geometry collection (sfg) object. Further downstream processing needs to happen via the sf package.

Usage

iso_to_sfg(x)
Arguments

x  The object to convert.

Details

The function `iso_to_sfg()` is a generic that takes an object created by either `isolines()` or `isobands()` and turns it into a simple features (sf) geometry collection. Importantly, the isobanding algorithm can produce polygons that do not represent valid simple features. This happens usually when the lower limit of an isoband is exactly equal to some data values (see examples for a demonstration). This can be worked around either by slightly shifting the data or band limits (e.g., round all data values and then shift them by a value smaller than the rounding error) or by fixing the geometries using the function `lwgeom::st_make_valid()`.

Examples

```
library(sf)
library(ggplot2)

# Example 1: simple 5x5 matrix
m <- matrix(c(0, 2, 2, 2, 0,
             0, 1, 0, 1, 0,
             1, 0, 0, 0, 0,
             0, 1, 0, 1, 0,
             0, 0, 0, 0, 0), 5, 5, byrow = TRUE)

z <- isolines(1:ncol(m), nrow(m):1, m, c(0.5, 1.5))
lines <- iso_to_sfg(z)
x <- st_sf(level = names(lines), geometry = st_sfc(lines))
ggplot(x) + geom_sf(aes(color = level))

# Example 2: volcano dataset
m <- volcano
b <- isobands((1:ncol(m))/(ncol(m)+1), (nrow(m):1)/(nrow(m)+1), m,
                10*9:19, 10*10:20)
bands <- iso_to_sfg(b)
x <- st_sf(level = as.numeric(sub("\.:", ",", names(bands))), geometry = st_sfc(bands))
ggplot(x) + geom_sf(aes(color = level, fill = level))

# Example 3: invalid simple features
m <- matrix(c(1.5, 1.5, 1.5, 1.5, 0.6,
             0.5, 1.5, 1.5, 0, 0,
             0, 1, 0, 1, 1,
             0, 1, 0, 0.7, 0,
             0.9, 1.3, 1.8, 1.4, 0.4), 5, 5, byrow = TRUE)

raw <- isobands(1:5, 5:1, m, levels_low = 0:1, levels_high = 1:2)
bands <- iso_to_sfg(raw)

iso <- st_sf(
    id = factor(1:length(bands)),
    geometry = st_sfc(bands)
)
```
label_placer_minmax

Set up a label placement strategy

Description

These functions set up various label placement strategies.

Usage

label_placer_minmax(placement = "tb",
                   rot_adjuster = angle_halfcircle_bottom(), n = 2)

label_placer_none()

label_placer_manual(breaks, x, y, theta)

Arguments

placement  String consisting of any combination of the letters "t", "r", "b", "l" indicating the placement of labels at the top, to the right, at the bottom, to the left of the isoline.

rot_adjuster  Function that standardizes the rotation angles of the labels. See e.g. angle_halfcircle_bottom().

n  Size of the point neighborhood over which the rotation angle should be calculated.

breaks  Character vector specifying the isolines to be labeled, as in isolines_grob().

x, y, theta  Numeric vectors specifying the x and y positions and angles (in radians) for each label corresponding to each break.
Details

`label_placer_minmax()` places labels at the horizontal or vertical minima or maxima of the respective isolines.

`label_placer_none()` places no labels at all.

`label_placer_manual()` places labels at manually defined locations.

---

**plot_iso**

*Visualize a single isoband*

---

**Description**

This function visualizes a single isoband calculated from a matrix. It is mainly useful for debugging and visualizing the isobanding algorithm. See `isobands()` for more examples.

**Usage**

```r
plot_iso(m, vlo, vhi, fill_lo = "gray95", fill_mid = "gray50", 
         fill_hi = "black", fill_band = "cornsilk", col_lo = "black", 
         col_hi = "black", newpage = TRUE)
```

**Arguments**

- `m` input matrix
- `vlo` lower cutoff for isobanding
- `vhi` higher cutoff for isobanding
- `fill_lo` fill color for points below the lower cutoff
- `fill_mid` fill color for points between the two cutoffs
- `fill_hi` fill color for points above the higher cutoff
- `fill_band` fill color for the isoband
- `col_lo` line color for lower cutoff
- `col_hi` line color for higher cutoff
- `newpage` boolean, indicating whether `grid.newpage()` should be called or not

**Examples**

```r
m <- matrix(c(0, 0, 0, 0, 0, 0, 
              0, 2, 2, 2, 2, 0, 
              0, 0, 0, 2, 0, 0, 
              0, 2, 0, 0, 2, 0, 
              0, 2, 2, 2, 2, 0, 
              0, 0, 0, 0, 0, 0), 6, 6, byrow = TRUE)

plot_iso(m, 0.5, 1.5)
```
Index

angle_fixed(angle_halfcircle_bottom), 2
angle_halfcircle_bottom, 2
angle_halfcircle_bottom(), 9
angle_halfcircle_right
  (angle_halfcircle_bottom), 2
angle_identity
  (angle_halfcircle_bottom), 2

clip_lines, 3

iso_to_sfg, 7

isoband, 3

isoband-package (isoband), 3

isobands, 3

isobands(), 5, 8, 10

isobands_grob, 5

isobands_grob(), 7

isolines (isobands), 3

isolines(), 6, 8

isolines_grob, 6

isolines_grob(), 5, 9

label_placer_manual
  (label_placer_minmax), 9

label_placer_minmax, 9

label_placer_minmax(), 7

label_placer_none
  (label_placer_minmax), 9

lwgeom::st_make_valid(), 8

plot_iso, 4, 10