Package 'isoband'

December 20, 2022

Title Generate Isolines and Isobands from Regularly Spaced Elevation Grids

Version 0.2.7

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

License MIT + file LICENSE

URL <https://isoband.r-lib.org>

BugReports <https://github.com/r-lib/isoband/issues>

Imports grid, utils

Suggests covr, ggplot2, knitr, magick, microbenchmark, rmarkdown, sf, testthat, xml2

VignetteBuilder knitr

Config/Needs/website tidyverse/tidytemplate

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.2.3

SystemRequirements C++11

NeedsCompilation yes

Author Hadley Wickham [aut, cre] (<https://orcid.org/0000-0003-4757-117X>),
Claus O. Wilke [aut] (Original author,
<https://orcid.org/0000-0002-7470-9261>),
Thomas Lin Pedersen [aut] (<https://orcid.org/0000-0002-5147-4711>)

Maintainer Hadley Wickham <hadley@posit.co>

Repository CRAN

Date/Publication 2022-12-20 10:00:13 UTC

R **topics documented:**

---

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.

Description

Efficient calculation of isolines and isobands from elevation grid

Usage

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

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

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, 0, 0, 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)

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,
              0, 1, 1, 1, 1, 0,
              0, 1, 2, 2, 1, 0,
              0, 1, 2, 2, 1, 0,
              0, 1, 1, 1, 1, 0,
              0, 0, 0, 0, 0, 0), 6, 6, byrow = TRUE)
plot_iso(m, 0.5, 1.5)
```

---

isobands_grob   *Render isobands*

Description

This function generates a grid grob that represents isobands.

Usage

```
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

```
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

```
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.
label_placer	Function that controls how labels are placed along the isolines. Uses [label_placer_minmax()](#) by default.
units	A character string specifying the units in which to interpret the isolines coordinates. Defaults to "npc".

See Also

See [isobands_grob()](#) for drawing of isobands. See [label_placer_minmax()](#) for label placement strategies.

Examples

```
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)
lines <- isolines(x, y, volcano, 5*(19:38))
bands <- isobands(x, y, volcano, 5*(18:38), 5*(19:39))

b <- isobands_grob(
  bands,
  gp = gpar(col = NA, fill = viridis_pal(21), alpha = 0.4)
)
```

```
l <- isolines_grob(
  lines, breaks = 20*(5:10),
  gp = gpar(
    lwd = c(.3, 1, .3, .3)
  )
)

grid.newpage()
grid.draw(b)
grid.draw(l)
```

---

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 st_make_valid().

Examples

```
if (requireNamespace("sf", quietly = TRUE)) {
library(sf)
library(ggplot2)

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

# the geometries are not valid
st_is_valid(iso, reason = TRUE)
# this doesn't prevent us from plotting them
ggplot(iso, aes(fill = id)) + geom_sf()

# make all geometries valid, requires GEOS >= 3.8.0
if (sf_extSoftVersion()["GEOS"] >= "3.8.0") {
  iso2 <- st_make_valid(iso)
  st_is_valid(iso2, reason=TRUE)
  # the plot should be unchanged
  ggplot(iso2, aes(fill = id)) + geom_sf()
}

# alternatively, if we shift all data values by a tiny
# amount (here, 1e-10) so they don't coincide with the band
# limits, no invalid geometries are generated.
raw <- isobands(1:5, 5:1, m + 1e-10, 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))
st_is_valid(iso, reason = TRUE)
}
```

---

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)

label_placer_middle(rot_adjuster = angle_halfcircle_bottom())
```

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.

label_placer_middle() places labels at the middle of each isoline.

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

```
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

```
m <- matrix(c(0, 0, 0, 0, 0, 0,
              0, 2, 2, 2, 2, 0,
              0, 2, 0, 0, 2, 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