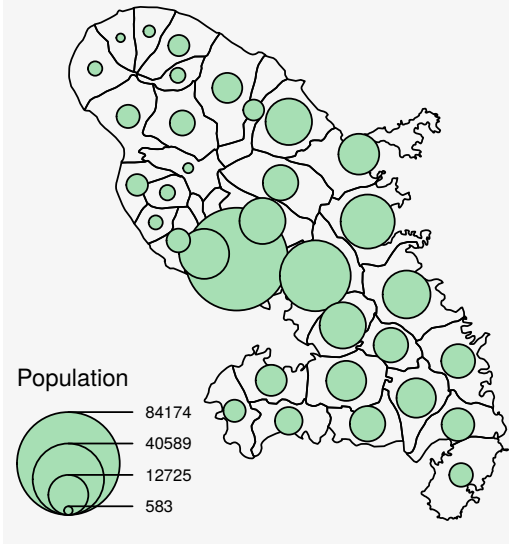


Thematic maps with cartography : : CHEAT SHEET

Use cartography with spatial objects from sf or sp packages to create thematic maps.

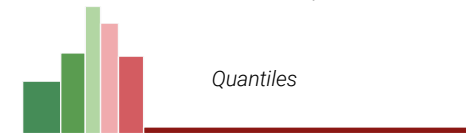
```
library(cartography)
library(sf)
mtq <- st_read("martinique.shp")
plot(st_geometry(mtg))
propSymbolsLayer(x = mtq, var = "P13_POP",
  legend.title.txt = "Population",
  col = "#a7dfb4")
```



Classification

Available methods are: quantile, equal, q6, fisher-jenks, mean-sd, sd, geometric progression...

```
bks1 <- getBreaks(v = var, nclass = 6,
  method = "quantile")
bks2 <- getBreaks(v = var, nclass = 6,
  method = "fisher-jenks")
pal <- carto.pal("green.pal", 3, "wine.pal", 3)
hist(var, breaks = bks1, col = pal)
```

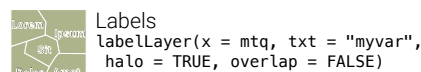
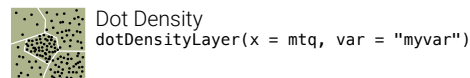
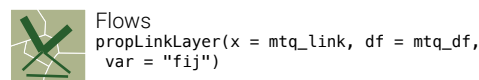
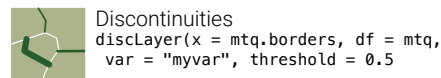
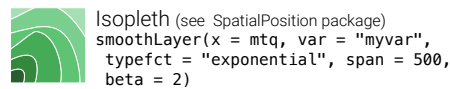
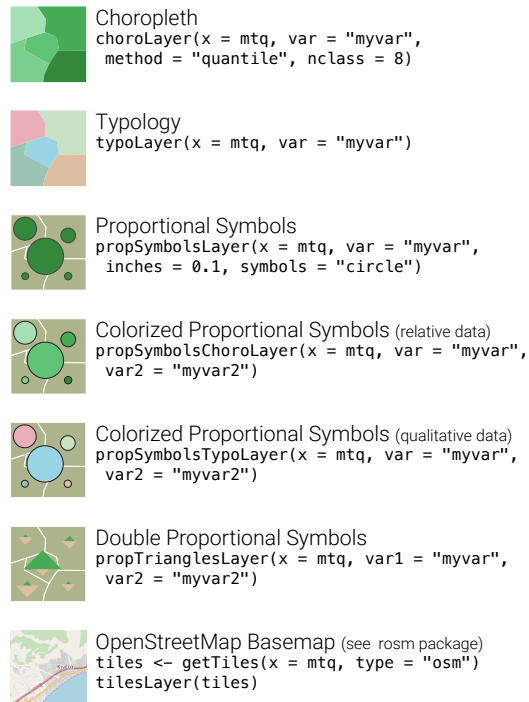


```
hist(var, breaks = bks2, col = pal)
```



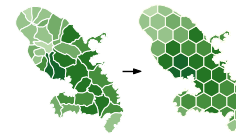
Symbology

In most functions the x argument should be an sf object. sp objects are handled through spdf and df arguments.



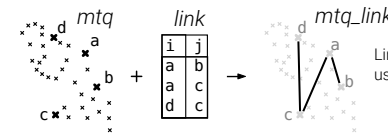
Transformations

Polygons to Grid
mtq_grid <- getGridLayer(x = mtq, cellsize = 3.6e+07,
type = "hexagonal", var = "myvar")



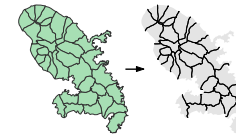
Grids layers can be used by
choroLayer() or propSymbolsLayer().

Points to Links
mtq_link <- getLinkLayer(x = mtq, df = link)



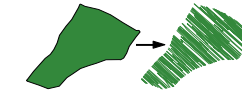
Links layers can be
used by *LinkLayer().

Polygons to Borders
mtq_border <- getBorders(x = mtq)



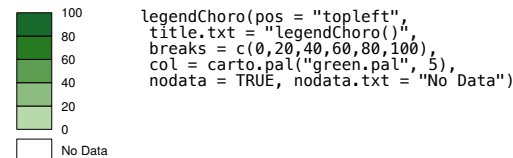
Borders layers can be used by
disclayer() function

Polygons to Pencil Lines
mtq_pen <- getPencilLayer(x = mtq)

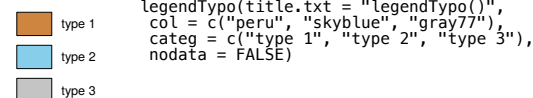


Legends

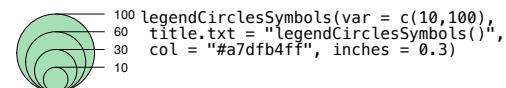
legendChoro()



legendTypo()



legendCirclesSymbols()



See also legendSquaresSymbols(), legendBarsSymbols(),
legendGradLines(), legendPropLines() and legendPropTriangles().

Map Layout

North Arrow:
north(pos = "topright")

Scale Bar:
barscale(size = 5)

Full Layout:
layoutLayer(
title = "Martinique",
tabtitle = TRUE,
frame = TRUE,
author = "Author",
sources = "Sources",
north = TRUE,
scale = 5)

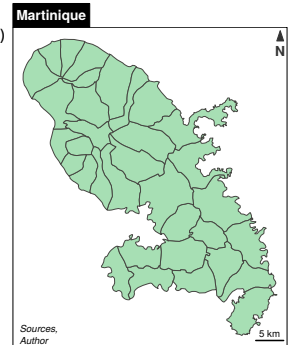
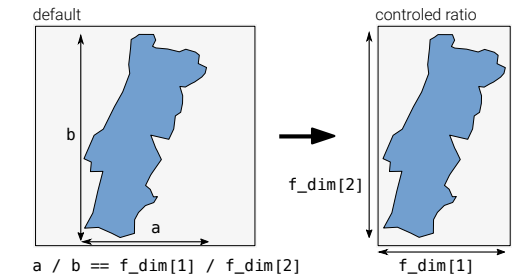


Figure Dimensions

Get figure dimensions based on the dimension ratio of a spatial object,
figure margins and output resolution.

```
f_dim <- getFigDim(x = sf_obj, width = 500,  
mar = c(0,0,0,0))  
png("fig.png", width = 500, height = f_dim[2])  
par(mar = c(0,0,0,0))  
plot(sf_obj, col = "#729fcf")  
dev.off()
```



Color Palettes

carto.pal(pal1 = "blue.pal", n1 = 5,
pal2 = sand.pal, n2 = 3)

display.carto.all(n = 8)

