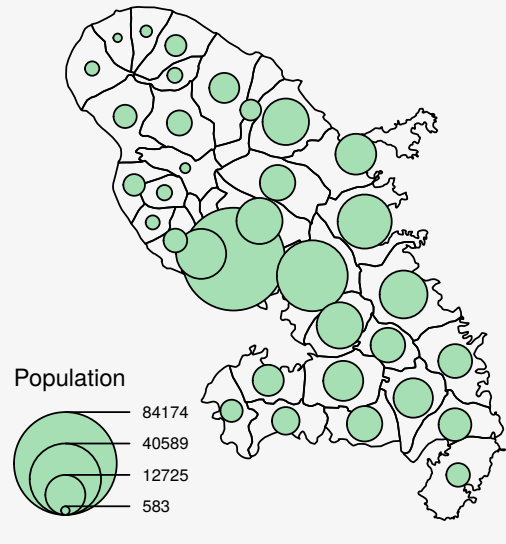


# Thematic maps with cartography : : CHEAT SHEET



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

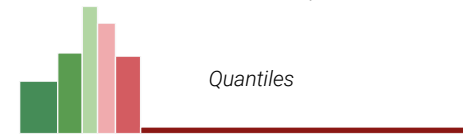
```
library(cartography)
library(sf)
mtq <- st_read("martinique.shp")
plot(st_geometry(mtg))
propSymbolsLayer(x = mtq, var = "P13_POP",
  legend.title.txt = "Population",
  col = "#a7dfb4")
```



## 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

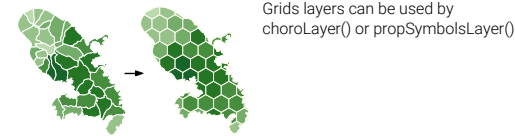
In most functions the x argument should be an sf object. sp objects are handled through spdf and ddf arguments.

- Choropleth**  
choroLayer(x = mtq, var = "myvar", method = "quantile", nclass = 8)
- Typology**  
typoLayer(x = mtq, var = "myvar")
- Proportional Symbols**  
propSymbolsLayer(x = mtq, var = "myvar", inches = 0.1, symbols = "circle")
- Colorized Proportional Symbols (relative data)**  
propSymbolsChoroLayer(x = mtq, var = "myvar", var2 = "myvar2")
- Colorized Proportional Symbols (qualitative data)**  
propSymbolsTypoLayer(x = mtq, var = "myvar", var2 = "myvar2")
- Double Proportional Symbols**  
propTrianglesLayer(x = mtq, var1 = "myvar", var2 = "myvar2")
- OpenStreetMap Basemap** (see rosm package)  
tiles <- getTiles(x = mtq, type = "osm")  
tilesLayer(tiles)

- Isopleth** (see SpatialPosition package)  
smoothLayer(x = mtq, var = "myvar", typefct = "exponential", span = 500, beta = 2)
- Discontinuities**  
disclayer(x = mtq.borders, df = mtq, var = "myvar", threshold = 0.5)
- Flows**  
propLinkLayer(x = mtq\_link, df = mtq\_df, var = "fij")
- Dot Density**  
dotDensityLayer(x = mtq, var = "myvar")
- Labels**  
labelLayer(x = mtq, txt = "myvar", halo = TRUE, overlap = FALSE)

## Transformations

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



**Points to Links**  
mtq\_link <- getLinkLayer(x = mtq, df = link)

**Polygons to Borders**  
mtq\_border <- getBorders(x = mtq)

**Polygons to Pencil Lines**  
mtq\_pen <- getPencilLayer(x = mtq)

## Legends

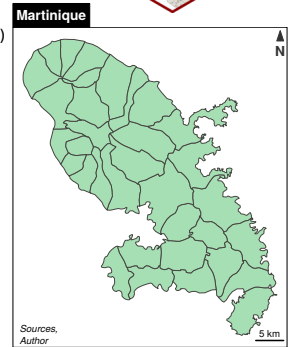
- legendChoro()**  
legendChoro(pos = "topleft", title.txt = "legendChoro()", breaks = c(0,20,40,60,80,100), col = carto.pal("green.pal", 5), nodata = TRUE, nodata.txt = "No Data")
  - legendTypo()**  
legendTypo(title.txt = "legendTypo()", col = c("peru", "skyblue", "gray77"), categ = c("type 1", "type 2", "type 3"), nodata = FALSE)
  - legendCirclesSymbols()**  
legendCirclesSymbols(var = c(10,100), title.txt = "legendCirclesSymbols()", col = "#a7dfb4ff", inches = 0.3)
- See also legendSquaresSymbols(), legendBarsSymbols(), legendGradLines(), legendPropLines() and legendPropTriangles().

## Map Layout

**North Arrow:**  
north(pos = "topright")

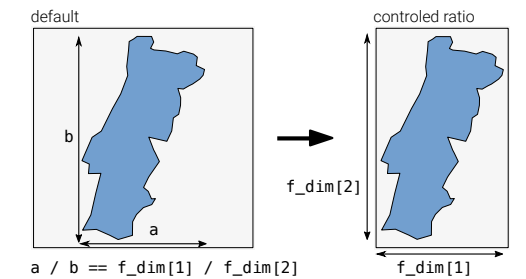
**Scale Bar:**  
barscale(size = 5)

**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)

