Map building using `mapsf` package in `R`

Wyclife Agumba Oluoch

2021-05-26 15:53:11

Background

With the steady growth in spatial analysis capabilities in `R`, there has been a number of attempts to make optimized visualizations of outputs comparable or better than traditional GUI based interfaces such as QGIS.In this process, several packages have been developed including tmap, mapvis, geovis among others. The list has not left our ggplot2 which is the bread-and-butter of visualization in R. When `sf` package was built for handling spatial data as regular dataframes in R, the spatial community appreciated the analysis capabilities but were not fullt satisfied with visualization of the outputs. This is the niche that has been filled by the development of `mapsf` package. In this tutorial, we are going to see how to use `mapsf` to visualize several vector map types and raster as well.

Making maps with mapsf

`mapsf` is a package which is enabling creating maps of simple features in R. For more information on the package, see.

You can install and load the package with the following code:

```
# install.packages('mapsf')
library(mapsf)
```

```
## Loading required package: sf
```

```
## Linking to GEOS 3.9.0, GDAL 3.2.1, PROJ 7.2.1
```

Since it is mapping simple features, loading the library will show *Loading required package: sf* and of course link to GEOS, GDAL, and PROJ.

Loading data

The package is coming with an inbuilt data-set for Martinique island in France. We can access the data-set and save it to an object called mtq using the following simple code:

```
mtq <- mf_get_mtq() # Most of the functions in mapsf package start with mf_
```

We can then go ahead and initialize the mapping process. You can think of this as designing the layout pane on which to display the map in QGIS.

```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk') # This is the layout to map on
```


There are several themes which one can pick from to use in the mapping process. Just before that, let us look at what is in the mtq object.

```
head(mtq) # multipolygon, WGS84 projection zone 20Nm 6 features and 7 fields plus bounding box. geom ho
```

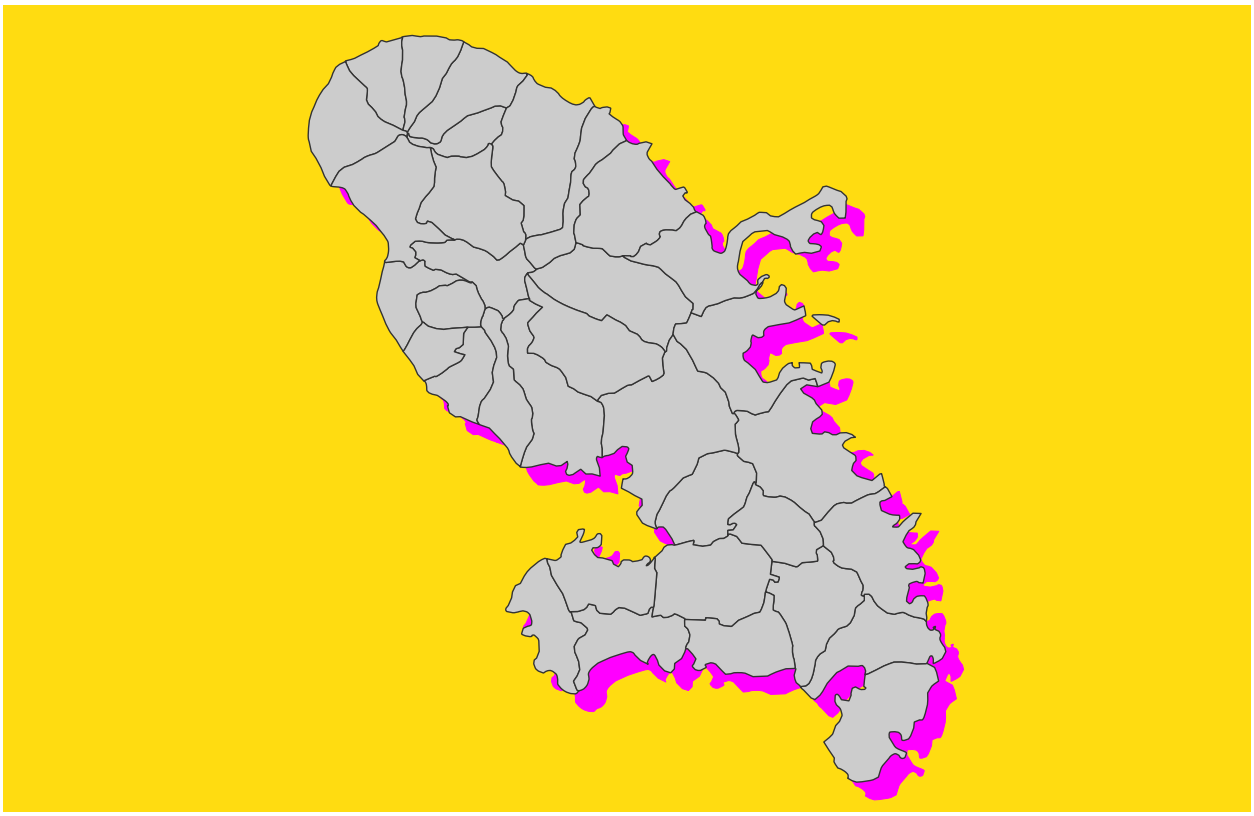
```
## Simple feature collection with 6 features and 7 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 695444 ymin: 1598818 xmax: 717731 ymax: 1645182
## Projected CRS: WGS 84 / UTM zone 20N
##   INSEE_COM              STATUS            LIBGEO  POP   MED CHOM  ACT
## 1     97201 Simple municipality L'Ajoupa-Bouillon 1902 13633  254  801
## 2     97202 Simple municipality Les Anses-d'Arlet 3737 14558  425 1659
## 3     97203 Simple municipality      Basse-Pointe 3357 14456  400 1395
## 4     97204 Simple municipality         Le Carbet 3683 18847  285 1568
## 5     97205 Simple municipality       Case-Pilote 4458 21005  347 2096
## 6     97206 Simple municipality        Le Diamant 5976 19704  430 2654
##                            geom
## 1 MULTIPOLYGON (((699261 1637...
## 2 MULTIPOLYGON (((709840 1599...
## 3 MULTIPOLYGON (((697602 1638...
## 4 MULTIPOLYGON (((702229 1628...
## 5 MULTIPOLYGON (((698805 1621...
## 6 MULTIPOLYGON (((709840 1599...
```

There is a beautiful shadow feature in mapsf which can be plotted just before overlaying it with the real map. We can go ahead and plot the shadow using mf_shadow as follows:

```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_shadow(mtq, col = 'purple', cex = 2, add = TRUE) # Now we have shadow on the layout added
```

The next step now is to plot the map itself.

```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_shadow(mtq, col = 'magenta', cex = 2, add = TRUE)
mf_map(mtq, type = 'base', add = TRUE) # Awesome
```



In as much as the map layout looks fine, it is still way below what mapsf can do. Let us use the mf_layout() function to create a layout more suitable for map creation. The layout will capture north arrow, scale, Source, author, package and version used, and title of the map.

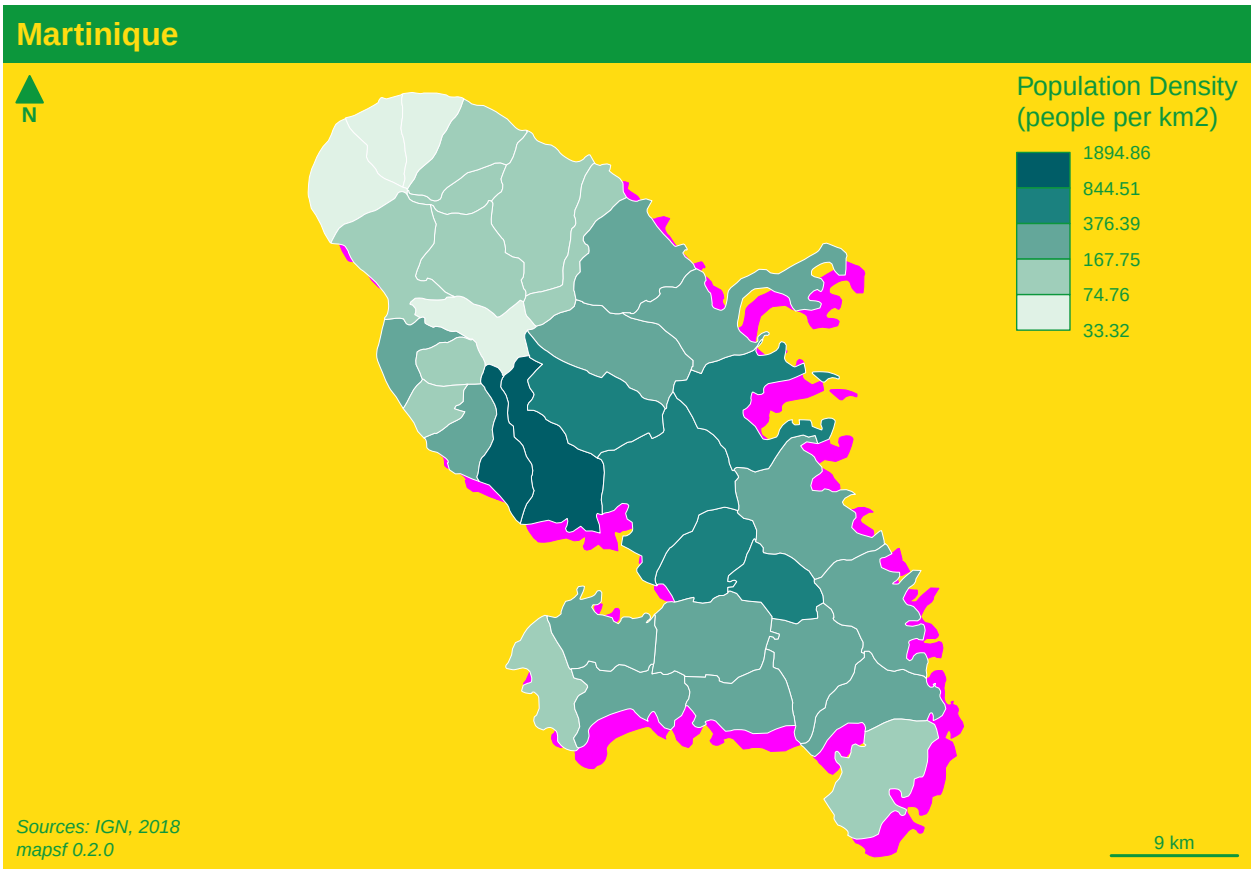
```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_layout(title = 'Martinique',
          credits = paste0('Sources: IGN, 2018\n',
          'mapsf ',
          packageVersion('mapsf')))
```

Now we want to show a map of the population since POP is a field in the mtq data per region. We want to represent this as graduated symbol so that we have larger symbols for those regions with higher population values.

```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_shadow(mtq, col = 'magenta', cex = 2, add = TRUE)
mf_layout(title = 'Martinique',
          credits = paste0('Sources: IGN, 2018\n',
          'mapsf ',
          packageVersion('mapsf')))
mf_map(mtq, add = TRUE)
mf_map(x = mtq, var = 'POP', type = 'prop', inches = 0.25,
       col = 'purple', leg_pos = 'bottomleft2', leg_title = 'Total population'
)
```

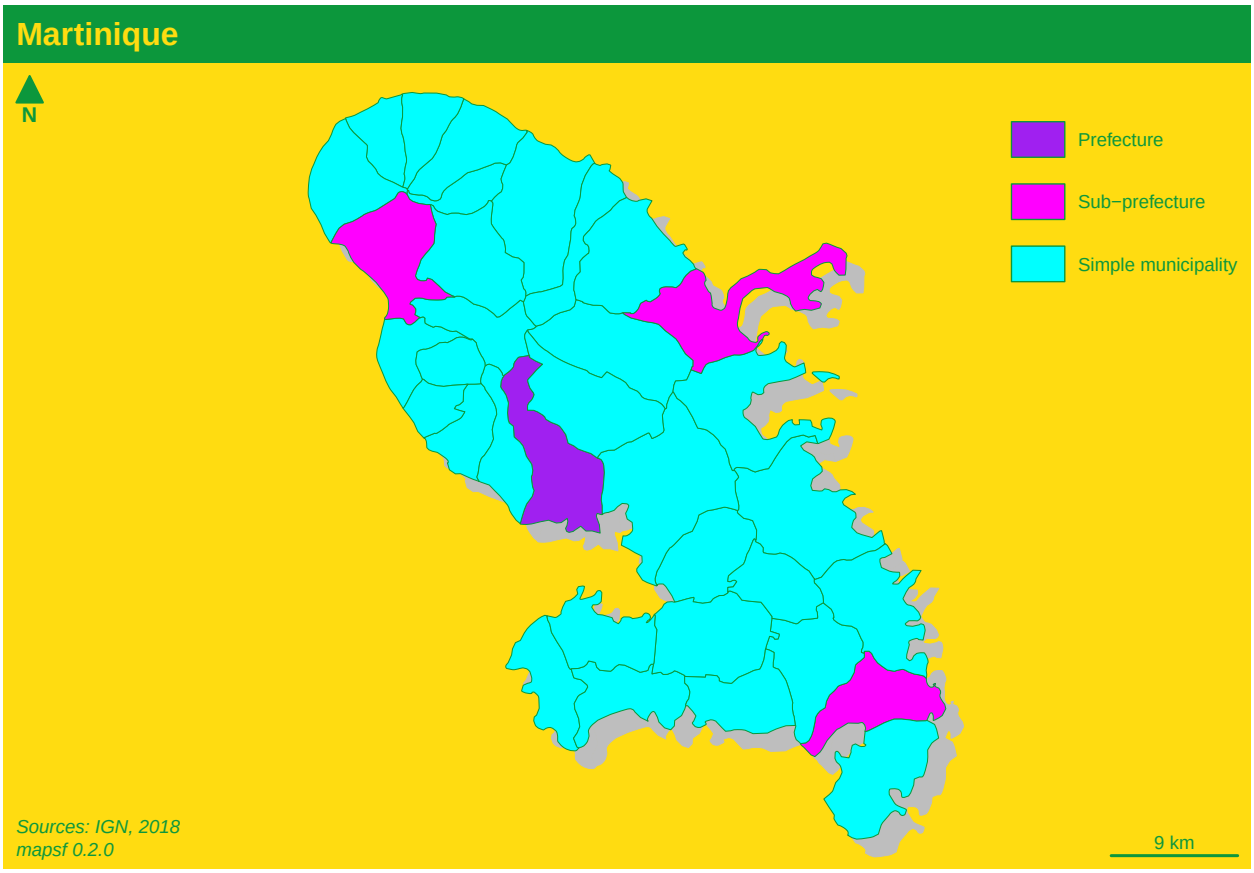
Awesome so far. Here we make a step to the choropleth maps. Since regions are polygons, we can calculate area of each polygon using the st_area() function. Then we can use the population column and the created area values to create a new column called population density. This is given by dividing the population by area.

```
mtq$POPDENS <- 1e6 * mtq$POP / st_area(mtq)
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_shadow(mtq, col = 'magenta', cex = 2, add = TRUE)
mf_layout(title = 'Martinique',
          credits = paste0('Sources: IGN, 2018\n',
          'mapsf ',
          packageVersion('mapsf')))
mf_map(x = mtq, var = "POPDENS", type = "choro", breaks = "geom", nbreaks = 5,
  col = "Mint", border = "white", lwd = 0.5, leg_pos = "topright",
  leg_title = "Population Density\n(people per km2)", add = TRUE
)
```



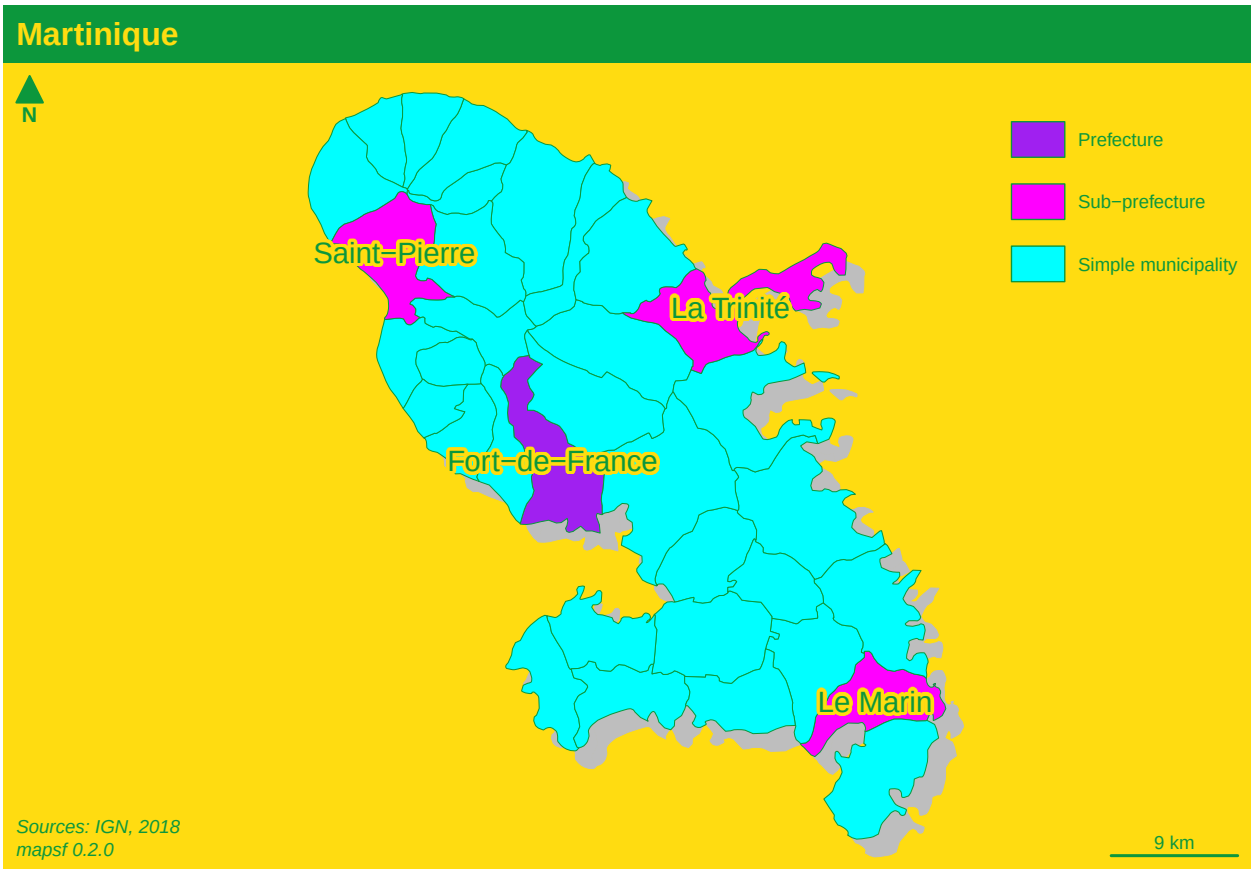
We can also plot a map which is showing the character variable in the fields such as the STATUS field. Then we color the map based on individual categories, so all regions having same category will have the same color.

```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_shadow(mtq, col = 'gray', cex = 2, add = TRUE)
mf_layout(title = 'Martinique',
          credits = paste0('Sources: IGN, 2018\n',
          'mapsf ',
          packageVersion('mapsf')))
mf_map(x = mtq, var = "STATUS", type = "typo", pal = c("purple", "magenta", "cyan"), lwd = .5, val_orde
"Simple municipality"), leg_pos = "topright", leg_title = "", add = TRUE)
```



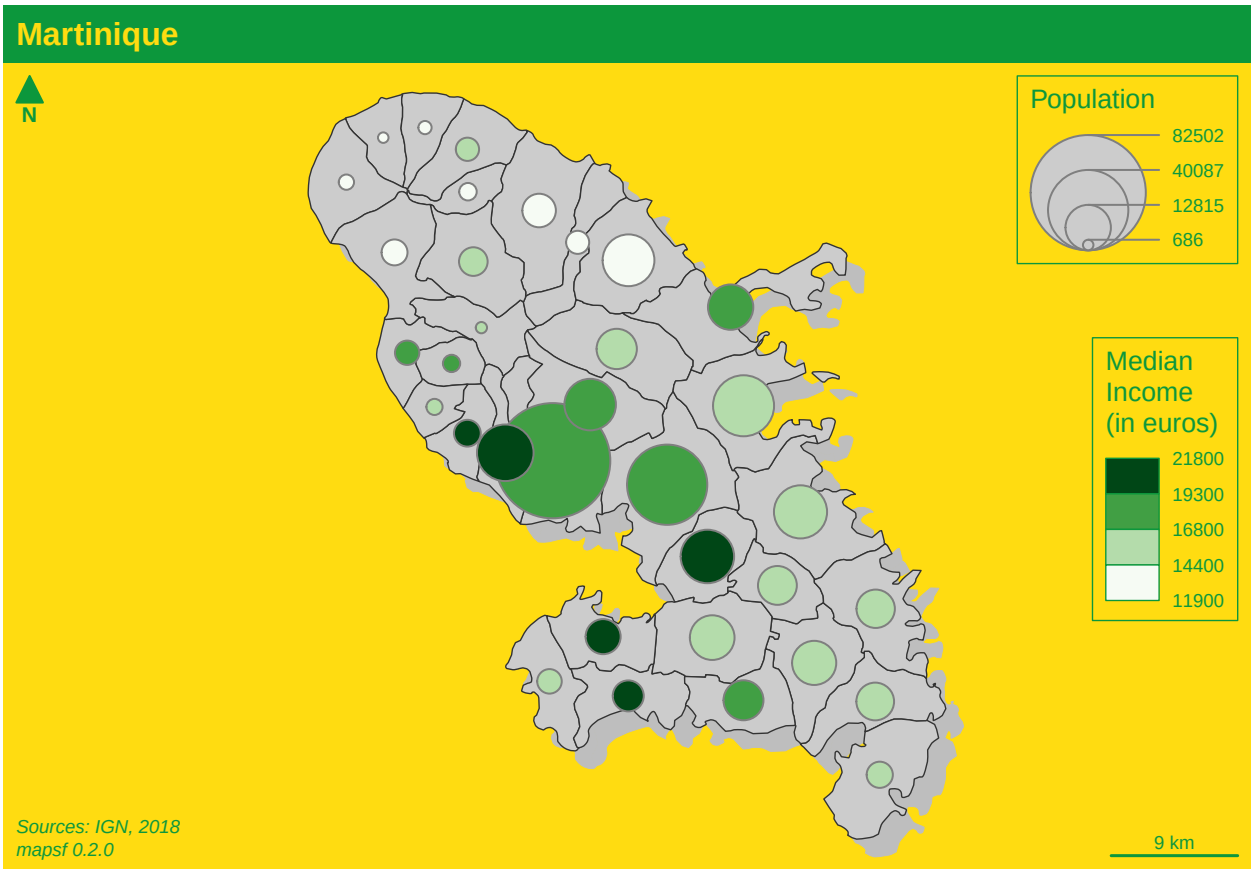
I may need to label some of the regions. This is achievable by using the mf_label() function as follows:

```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_shadow(mtq, col = 'gray', cex = 2, add = TRUE)
mf_layout(title = 'Martinique',
          credits = paste0('Sources: IGN, 2018\n',
          'mapsf ',
          packageVersion('mapsf')))
mf_map(x = mtq, var = "STATUS", type = "typo", pal = c("purple", "magenta", "cyan"), lwd = .5, val_orde
"Simple municipality"), leg_pos = "topright", leg_title = "", add = TRUE)
mf_label(x = mtq[mtq$STATUS != "Simple municipality", ], var = "LIBGEO",
         cex = 0.9, halo = TRUE, r = 0.15)
```



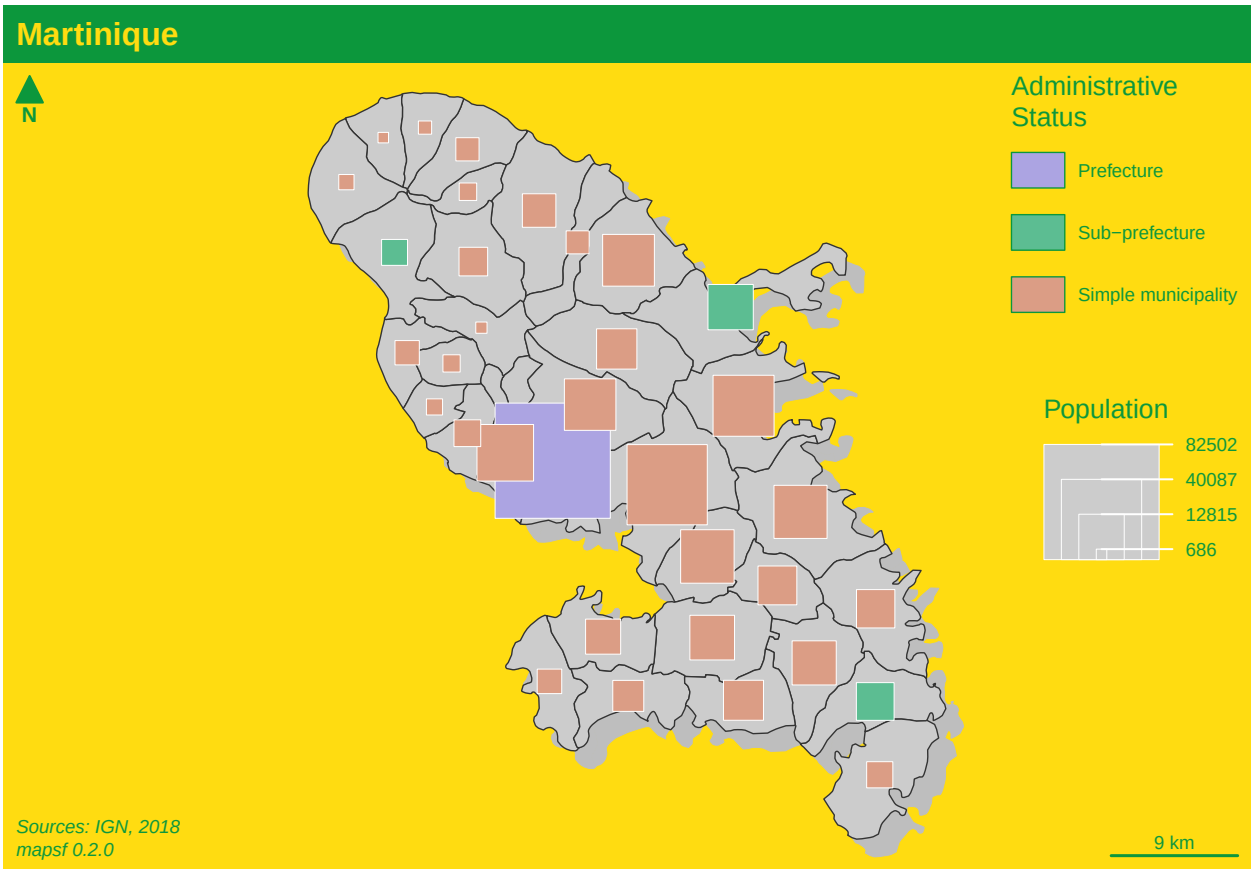
Both the proportional map can be combined with choropleth coloraion. The size of circles and the color density represent different variables. This is achievable with the following code:

```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_shadow(mtq, col = 'gray', cex = 2, add = TRUE)
mf_layout(title = 'Martinique',
          credits = paste0('Sources: IGN, 2018\n',
          'mapsf ',
          packageVersion('mapsf')))
mf_map(mtq, add = TRUE)
mf_map(x = mtq, var = c("POP", "MED"), type = "prop_choro", border = "grey50",
  lwd = 1, leg_pos = c("topright", "right"), leg_title = c("Population","Median\nIncome\n(in euros)"), l
  pal = "Greens", leg_val_rnd = c(0, -2), leg_frame = c(TRUE, TRUE)
)
```



Proportional map and typology in one map:

```
mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
mf_shadow(mtq, col = 'gray', cex = 2, add = TRUE)
mf_layout(title = 'Martinique',
          credits = paste0('Sources: IGN, 2018\n',
          'mapsf ',
          packageVersion('mapsf')))
mf_map(mtq, add = TRUE)
mf_map(x = mtq, var = c("POP","STATUS"), type = "prop_typo", symbol = "square",
  border = "white", lwd = .5, leg_pos = c("right",  "topright"),
  leg_title = c("Population", "Administrative\nStatus"),
  val_order = c("Prefecture", "Sub-prefecture", "Simple municipality"))
```

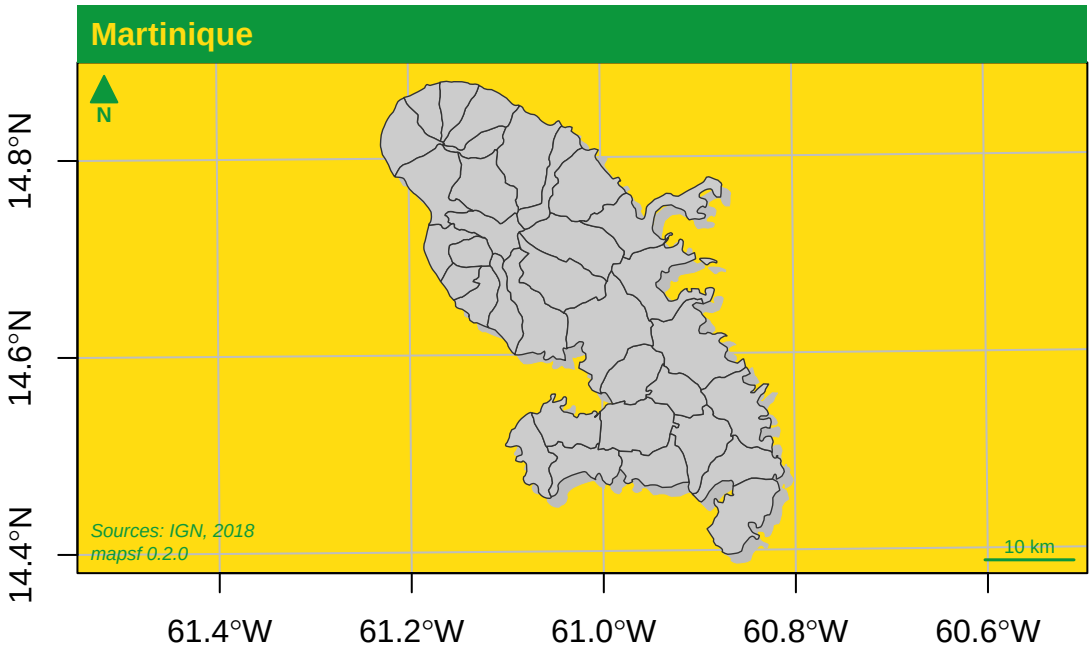


Other functions such as `mf_arrow`, `mf_scale`,and `mf_credits` can be used to alter the position of the map aesthetics.

Adding graticule on the map

It may sometimes, not always, be necessary to add graticule on the map. That can be achieved with the following:

```
#mf_init(mtq, expandBB = rep(0, 4), theme = 'jsk')
#mf_theme(mar = c(2, 2, 1.2, 0)+5)
plot(st_geometry(mtq), border = NA, col = NA, graticule = st_crs(4326),
     axes = TRUE, bg = mf_theme(mar = c(2, 2, 1.2, 0)+2)$bg,
     lon = seq(-62, -60, by = 0.2), lat = seq(14, 15, by = 0.2))
mf_layout(title = 'Martinique',
          credits = paste0('Sources: IGN, 2018\n',
          'mapsf ',
          packageVersion('mapsf')))
mf_shadow(mtq, col = 'gray', cex = 1, add = TRUE)
mf_map(mtq, add = TRUE)
```


Martinique
N
14.8°N
14.6°N
14.4°N
61.4°W
61.2°W
61.0°W
60.8°W
60.6°W
Sources: IGN, 2018
mapsf 0.2.0
10 km