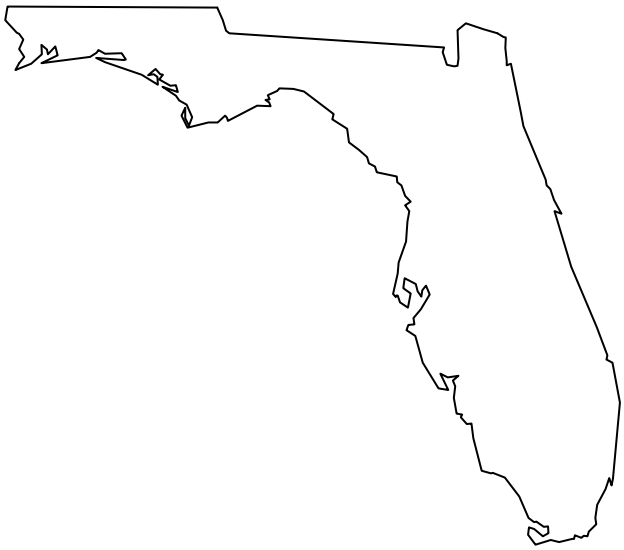
county_map.R

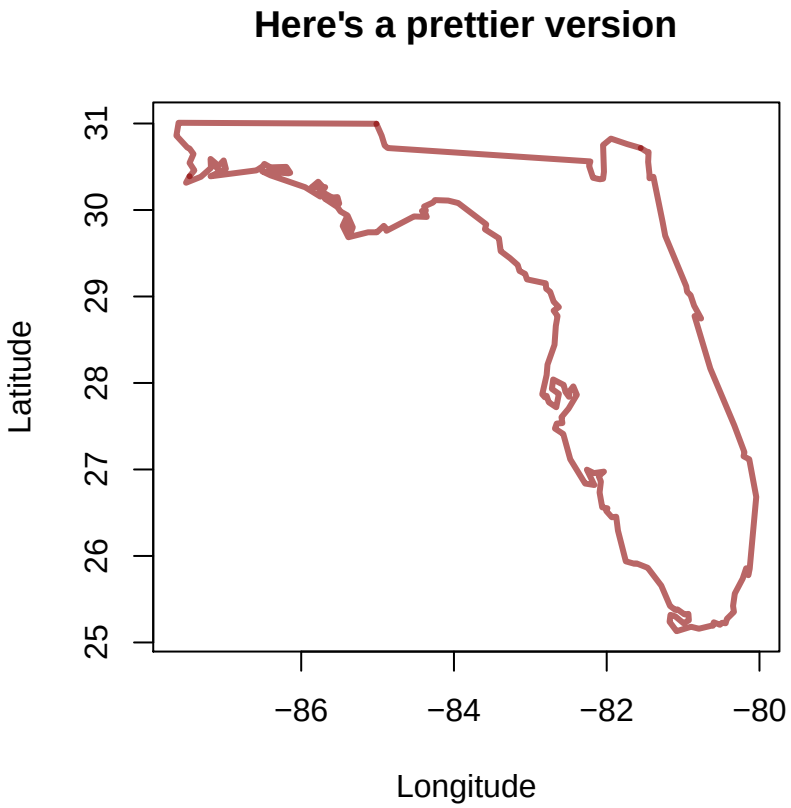
joebrew

Tue Aug 25 09:50:28 2015

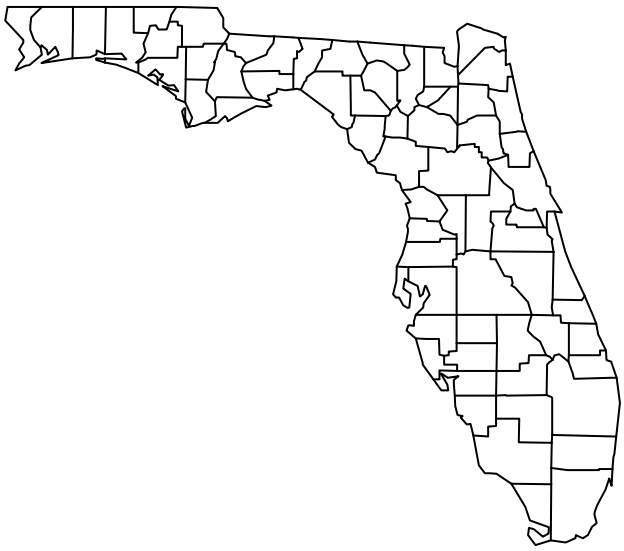
```
#####
# SIMPLE MAPS
#####
library(maps)

### Create a simple map of Florida
florida_state <- map('state', 'fl')
```



```
# Note that this is a "list" object with 4 components:
# --- x (the longitude coordinates)
# --- y (the latitude coordinates)
# --- range (the extremes of the maps in lon/lat)
# --- names (the names of the polygons)

# You can make some variations of this plot
plot(x = florida_state$x,
     y = florida_state$y,
     type = 'l',
     xlab = 'Longitude',
     ylab = 'Latitude',
     lwd = 3,
     col = adjustcolor('darkred', alpha.f = 0.6),
     main = 'Here\'s a prettier version')
```




```
### Create a simple map of Florida with counties
florida_counties <- map('county', 'fl')
```

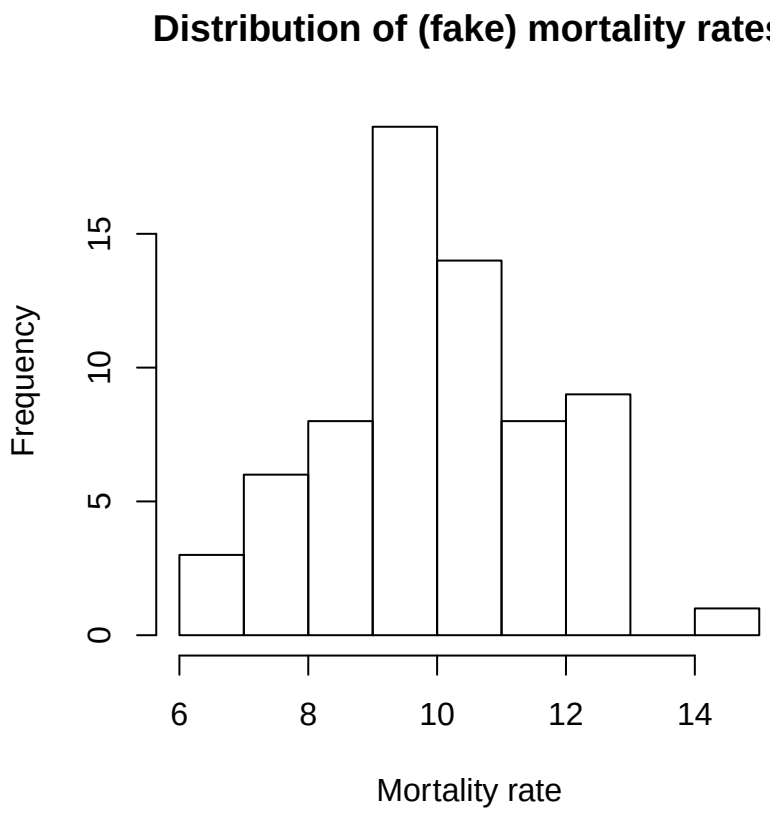


```
# Again, this is a "list" object with 4 components:
# --- x (the longitude coordinates)
# --- y (the latitude coordinates)
# --- range (the extremes of the maps in lon/lat)
# --- names (the names of the polygons)
```

```r
# Note that the names contains the names of all the counties
florida_counties$names
```

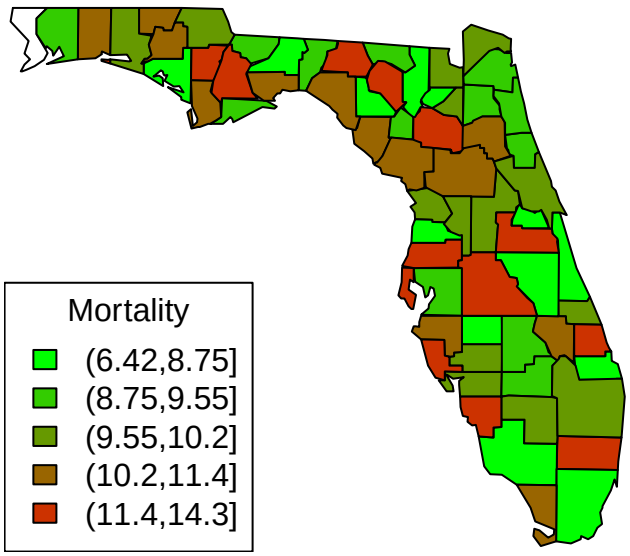
```
##  [1] "florida,alachua"       "florida,baker"
##  [3] "florida,bay"           "florida,bradford"
##  [5] "florida,brevard"       "florida,broward"
##  [7] "florida,calhoun"       "florida,charlotte"
##  [9] "florida,citrus"        "florida,clay"
## [11] "florida,collier"       "florida,columbia"
## [13] "florida,miami-dade"    "florida,de soto"
## [15] "florida,dixie"         "florida,duval"
## [17] "florida,escambia"      "florida,flagler"
## [19] "florida,franklin"      "florida,gadsden"
## [21] "florida,gilchrist"     "florida,glades"
## [23] "florida,gulf"          "florida,hamilton"
## [25] "florida,hardee"        "florida,hendry"
## [27] "florida,hernando"      "florida,highlands"
## [29] "florida,hillsborough"  "florida,holmes"
## [31] "florida,indian river"  "florida,jackson"
## [33] "florida,jefferson"     "florida,lafayette"
## [35] "florida,lake"          "florida,lee"
## [37] "florida,leon"          "florida,levy"
## [39] "florida,liberty"       "florida,madison"
## [41] "florida,manatee"       "florida,marion"
## [43] "florida,martin"        "florida,monroe"
## [45] "florida,nassau"        "florida,okaloosa:main"
## [47] "florida,okaloosa:spit" "florida,okeechobee"
## [49] "florida,orange"        "florida,osceola"
## [51] "florida,palm beach"    "florida,pasco"
## [53] "florida,pinellas"      "florida,polk"
## [55] "florida,putnam"        "florida,st johns"
## [57] "florida,st lucie"      "florida,santa rosa"
## [59] "florida,sarasota"      "florida,seminole"
## [61] "florida,sumter"        "florida,suwannee"
## [63] "florida,taylor"        "florida,union"
## [65] "florida,volusia"       "florida,wakulla"
## [67] "florida,walton"        "florida,washington"
```

```r
# To fill in each county, first make a vector of your values (this could be, say, mortality rates)
mortality_rates <- rnorm(mean = 10, n = length(florida_counties$names), sd = 2)

# Our vector is "mortality rates" and it looks like this:
hist(mortality_rates, main = 'Distribution of (fake) mortality rates',
     xlab = 'Mortality rate')
```




```
# Now we can "bin" our mortality rates into five groups
mortality_buckets <- cut(mortality_rates, breaks = quantile(mortality_rates, prob = seq(0, 1, length = 

# Since we have 6 groups, we need 6 (sequential colors):
mortality_colors <- colorRampPalette(c('green', 'red'))(6)

# Now we can plot with our colors
map('county', 'fl', fill = TRUE, col = mortality_colors[as.numeric(mortality_buckets)])
legend('bottomleft',
       fill = mortality_colors,
       legend = levels(mortality_buckets),
       title = 'Mortality')

#####
# MORE COMPLICATED WAYS
#####

# If you want to work directly with shapefiles
# the raster package provides an easy interface for doing so
library(raster)
```

```
## Loading required package: sp
```

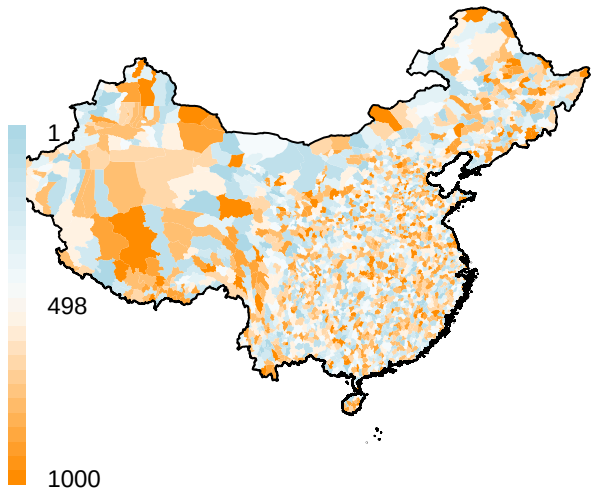



```
china <- getData('GADM', country = 'CHN', level = 3)

# As with the previous example, you can plot a choropleth by
# passing a vector of the same length as the number of polygons
# manually defining the color scheme, etc.

# Alternatively, I've written some code to do this
# more automatically.  It's at
# https://github.com/joebrew/misc/blob/master/functions/functions.R

library(devtools)
source('helpers.R')

# Assign a fake variable
china$fake_var <- sample(1:1000, nrow(china), replace = TRUE)

# Get a boundary
boundary <- collapse_map(china)
```

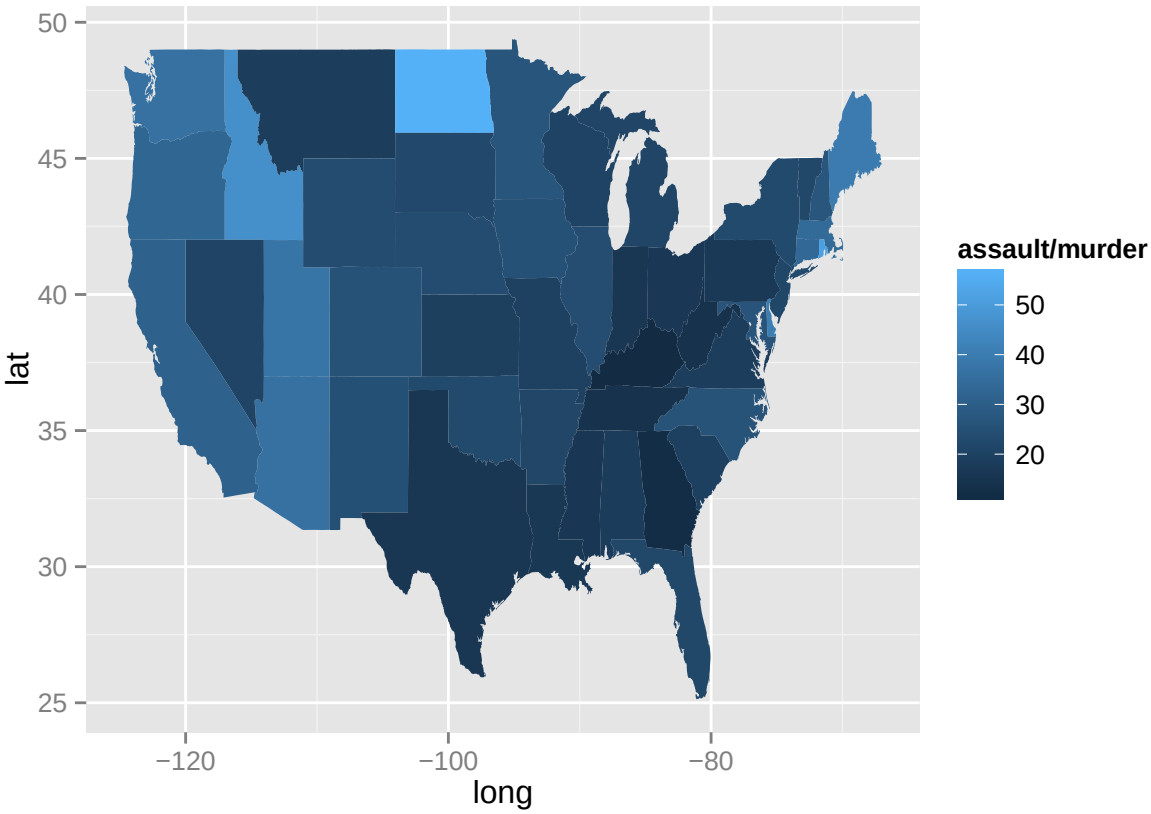
```
## Loading required package: maptools
## Checking rgeos availability: TRUE
```

```
choro(shape = china,
      var = china$fake_var,
      boundary = boundary,
      legend_round = 0,
      border = FALSE)
title(main = 'A fake map of China at the district/county level')
```

A fake map of China at the district/county level




```
#### If you like the ggplot2 framework, mapping is pretty simple in that as well
# (http://docs.ggplot2.org/0.9.3.1/map_data.html)
library(ggplot2)
if (require("maps")) {
  states <- map_data("state")
  arrests <- USArrests
  names(arrests) <- tolower(names(arrests))
  arrests$region <- tolower(rownames(USArrests))

  choro <- merge(states, arrests, sort = FALSE, by = "region")
  choro <- choro[order(choro$order), ]
  qplot(long, lat, data = choro, group = group, fill = assault,
        geom = "polygon")
  qplot(long, lat, data = choro, group = group, fill = assault / murder,
        geom = "polygon")
}
```


50
45
40
35
30
25
lat
−120
−100
−80
long
assault/murder
50
40
30
20