**Exercise 3: Choropleth Maps with R**

**3.1 Plot shapefiles**

Install and load the library *rgdal*. If you don’t remember how to do that, check your code of Exercise 2. Check out the *rgdal* documentation [here](https://www.rdocumentation.org/packages/rgdal/versions/1.2-8). It allows you to read shapefiles.

Read the Palo Alto shapefile, using the readOGR function which *rgdal* provides.

palo\_alto <- readOGR("[path-to-file]", "palo\_alto")

Check out the basic info about the shapefile which this returned in the console.

*Q1: How many features and fields does the Palo Alto shapefile have? (1pt)*

Take a look at the shapefile spatially:

plot(palo\_alto)

Modify the plot to add some basic options: red lines and title.

plot(palo\_alto, border = "red")

title(main = "Palo Alto", sub = "By Census Tracts")

*Q2: What spatial units do the polygons represent? (1pt)*

You also have a shapefile about freeways:

freeways <- readOGR("[path-to-file]", "palo\_alto\_freeways")

When plotting lines, you can also add basic options: Use the lwd and col options to set the width and color of lines.

par(mfrow = c(1, 2))

plot(freeways, col = "red", bg = "blue")

plot(freeways, lwd = 10, col = "green")

*Q3: What does the par() function do? (1pt)*

**3.2 Multiple layers**

It’s easy to overlay multiple shapefiles using plot and add option. But first, clean the current plot:

dev.off()

Make sure both layers are in the same projection, then overlay them by plot command.

stopifnot(proj4string(palo\_alto) == proj4string(freeways))

plot(palo\_alto)

plot(freeways, col = "blue", add = T)

*Q4: Btw, where is Palo Alto? (1pt)*

**3.3 spplot**

spplot is an extension of plot specifically for making choropleth maps. Just pass a *Spatial\*DataFrame* object and the name of columns you want to plot (if you don’t pass specific column names, a separate figure will be created for each column.)

spplot(palo\_alto, "PrCpInc", main = "Palo Alto Demographics", sub = "Average Per Capita Income", col = "transparent")

*Q5: What is “PrCpInc”? (1pt)*

**3.4 Color Brewer**

Install and load the package *RColorBrewer*. If you do it in the console, be aware that R is case sensitive (*rcolorbrewer* does not exist in R)! Check the Color Brewer website [here](http://colorbrewer2.org/#type=sequential&scheme=BuGn&n=3).

*Q6: What is Color Brewer? (1pt)*

Display the list of color palettes:

display.brewer.all()

Create your own palette object (called *my.palette*), where n is the number of colors, and name is the name of your color ramp.

my.palette <- brewer.pal(n = 7, name = "OrRd")

Then pass palette to spplot, make sure the cuts parameter is set to the number of colors minus one.

spplot(palo\_alto, "PrCpInc", col.regions = my.palette, cuts = 6, col = "transparent")

**3.4 Controlling the color class breaks**

You can customize your color class breaks using the *classInt* library. Install and load *classInt* and execute the following code:

breaks.qt <- classIntervals(palo\_alto$PrCpInc, n = 6, style = "quantile", intervalClosure = "right")

spplot(palo\_alto, "PrCpInc", col = "transparent", col.regions = my.palette, at = breaks.qt$brks)

Task 1: Pick a city to plot. Three cities are included in RGIS3\_Data, or find your own data. Read-in the city polygons. Use spplot to plot average incomes (or whatever variable) for your city. Use *RColorBrewer* to map income using your own custom color scheme. The code above uses name = "OrRd" to determine the color ramp in the brewer.pal() function. Also, the color class breaks are deterimed by style = “quantiles” in the classIntervals() function. Use a different color ramp and find the *classInt* [documentation](https://cran.r-project.org/web/packages/classInt/classInt.pdf) to choose other styles for determining color breaks. Submit your code, along with a screenshot of the resulting map(s). *(3pts)*

**3.5 Basemaps**

We did some work with basemaps in Exercise 2. Let’s try again and overlay the Palo Alto shapefile with a basemap. We use the libraries *dismo* and *raster*. Install and load.

base.map <- gmap(palo\_alto, type = "terrain")

reprojected.palo\_alto <- spTransform(palo\_alto, base.map@crs)

plot(base.map)

plot(reprojected.palo\_alto, add = T, border = "red", col = "transparent")

Now you see that the shapefile goes way beyond Palo Alto, and covers neighboring cities as well.

Bonus Task: If you’re able to overlay a choropleth map on a base map using your own data, adjust transparencies to create a visually pleasing map product, you are awarded 3 bonus points. Submit your code, together with a screenshot of your map. *(3pts)*