R Module 4: Descriptive Spatial Statistics

Tabular and visual explorations of data are always an important first step to understanding the distribution of actual values across a range of possible outcomes. Frequency tables, bar plots, histograms, and box plots are basic tools for this tasks. When we have spatially located data, mapping is an additional crucial tool.

These types of explorations have limits and we will also need actual measures of distribution of values across a range of possible outcomes and across geographic space. For this we rely on measures of central tendency and measures of dispersion.

Let's start again with a shapefile of manufacturing jobs in North Carolina. This shapefile has three different variables about employment: manufacturing jobs in 2000 (MNEM2000), manufacturing jobs in 1990 (MNEM1990), all jobs in 2000(TOTJOB2000), and all jobs in 1990 (TOTJO1990). You can download the files on ASULearn.

```
#let's install the GISTool library which allows us to read in
the shapefile

library(GISTools)
```

```
#part of our efforts will result in a new shapefile
#so next we need to tell R where to save files
#we do this by setting a new working directory with setwd()
#of course you'll have to type in your own pathway
setwd('G:\\Rmod4')
```

```
#read in the file
#we can skip the tedium of typing in pathways by using
file.choose()
#just make sure to select the .shp file
NC=readShapePoly(file.choose())



#make sure the file is read in correctly
summary(NC)
```

Notice that some basic descriptive statistics are provided with the summary(NC) output: mean, median, 1st and 3rd quartiles, and minimum and maximum values. We might like to add such basic descriptive statistics to a histogram of the variables as part of our data exploration.

```
#first create a relative frequency histogram (also called a
density plot)
#I've specified a particular technique for defining the breaks
#To learn more about this break method, review help(hist)
hist(NC$MNEM2000, prob=T, col='red', breaks='fd')

#Now I'll add the smoothed frequency polygon curve to the
histogram
```

```
lines(density(NC$MNEM2000), lwd=4, lty='dashed', col='black')
#Next I’ll add the mean and median as vertical lines using

abline()

abline(v=mean(NC$MNEM2000), lwd=4, lty='dotted', col='blue')
abline(v=median(NC$MNEM2000), lwd=3, lty='twodash',
col='green')

#Last, I’ll add a legend to keep track of all these lines…
legend(x='topright', c('Density plot', 'Mean', 'Median'),
col=c('black', 'blue', 'green'), lwd=c(4, 4, 3), lty=c('dashed',
'dotted', 'twodash'))
```

1. **Create a histogram (using the “FD” breaks method) with frequency curves, mean lines, median lines, legends, appropriate axis labels, and titles for all four employment variables.**

We already know some important things about employment variables from reviewing visual and descriptive measures. For instance, we know that the data class to the right of the mean have a lot observations for manufacturing in 2000. Since we know that this data is summed up by counties, this means the majority of NC’s 100 counties have far fewer manufacturing jobs than the state mean. In other words, we have an initial clue that jobs are distributed unevenly across the 100 counties.

This is a perfect test case for one of the descriptive spatial measures that you were introduced to: the location quotient. Let’s explore how evenly (or unevenly) jobs are distributed across the state’s 100 counties. To do this, we’ll use our shapefiles and introduce a new technique in R by creating a custom function to calculate the LQ for each county in 2000.

Recall from our discussion that LQ is a ratio of the frequency of something within a particular area to the total frequency across a larger set of such areas. Applying this to the NC manufacturing would result in formula like this for each county in 2000: (total manufacturing jobs in county/total all jobs in county)/(total manufacturing jobs in state/total all jobs in state).

The information that we need for the first part of the LQ calculation is already in the shapefile: (total manufacturing jobs in county/total all jobs in county). But we need to do some preparation to calculate the second part.

```
#to make this a little easier, let's first calculate the
statewide ratio
#of manufacturing jobs to total jobs by summing each variable
#we’ll assign this to a new variable so we can call it later

jobrate.2000=sum(NC$MNEM2000)/sum(NC$TOTJOB2000)
#returning this new variable gives us an expected outcome
(0.1144246)
```

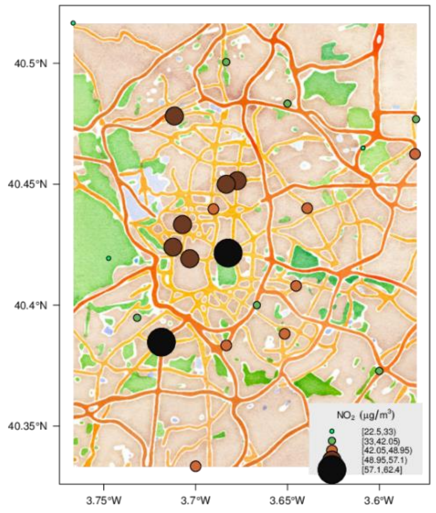
```
#if manufacturing jobs are evenly distributed across the 92
Counties

#manufacturing should comprise roughly 11% of all jobs in a
given county

jobrate.2000
[1] 0.1144246

#we can now calculate the LQ for each county

#but we need to add a new field to the attribute table to hold
the results

#we'll create the field and populate it with the calculated LQ
Values

NC$LQ2000=(NC$MNEM2000/NC$TOTJOB2000)/jobrate.2000

#we can use summary to see that the new field was added and is
#populated
#with our LQ values for the year 2000

summary(NC$LQ2000)

#now I can map the LQ measures for each county
#first I'll customize my data classes and breaks using shading()

shades=shading(c(0.5, 1, 1.5), cols=brewer.pal(4, 'Blues'))

#next I'll make my map and add a legend
choropleth(NC, NC$LQ2000, shades)
choro.legend("bottomleft", sh=shades, title='LQ Manufacturing
(2000)')

#I might want to work with this shapefile later so I should save
it
#to do this, use the writePolyShape() command
#this will save the shapefile to the working directory specified
earlier
#I'm going to call mine new file 'test_RMOD4'

writePolyShape(NC, 'test_RMOD4')
```

2.) Create and provide a map of the LQ for both 2000 and 2012. Provide the code for the 2012 map. Make sure each map has a legend, use the same data classes/break points, and is appropriately titled. Use the writePolyShape() command to save the new shapefile with the LQ measures to a location where you can use it again later.

3.) Use your LQ maps to provide a short description of how jobs are distributed across North Carolina's counties and how this changed (if at all) from 1990 to 2000.

Fun with other packages

Now that we are getting familiar with GISTool package, let's explore other packages. R is not only great at statistics, but can also create maps like this:



Let's also play around with some other packages, first you must install the packages below, and then load the libraries.

```
library(MASS)
library(GISTools)
library(sp)  # classes for spatial data
library(raster)  # grids, rasters
library(rasterVis)  # raster visualisation
library(maptools)
library(rgeos)# and their dependencies
library(rgdal)
library(XML)
library(dismo)
library(googleVis)
```

```
mymap <- gmap("Uzbekistan")  # choose a different country

plot(mymap)

mymap <- gmap("Uzbekistan", type = "satellite", exp=3)  ##Choose
a different map type and zoom level look at documentation
help(gmap)

plot(mymap)

mymap <- gmap("France", type = "satellite", filename =
"France.gmap") #Save map in working directory for future use and
export for report
```

4.) Include map of different country, different google layer type, and different type in your PDF

```
##Now let's draw a map extent with cursor

mymap <- gmap("Europe")

plot(mymap)

select.area <- drawExtent()

# now click 2 times on the map to select your region

mymap <- gmap(select.area)
```

```
plot(mymap)

##You can also plot on lines and points using functions like
##PlotonStaticMap(), for those who feel inspired to try please
##include in report.

#We can also plot data on Google API, use this example

data(Andrew)

M1 <- gvisMap(Andrew, "LatLong", "Tip",

            options=list(showTip=TRUE, showLine=F,
enableScrollWheel=TRUE, mapType='satellite',
useMapTypeControl=TRUE, width=800,height=400))

plot(M1)
```

5.) Create another map of Google API and take a screenshoot. If you cannot use your own file, you can use data available in the googleVIS package. Please refer to documentation to find a data set. (Sidenote: the NC Shapefile will not work, a point shapefile will work with the Google API. You can convert a shapefile to a data frame with the following code NC.df <- data.frame(NC) and then attach(NC.df)).