Introduction

These are lecture/lab notes for Chapter 2 in Spatial Regression Models. You will find all necessary information, including \texttt{R}-code, to replicate the figures and maps found in this chapter. A few of the maps were obtained from public repositories (Wiki and others), but what you find here will allow replication of the bulk of chapter 2. This is intended to provide code as well as a didactic repository.

Code appears below as it is in the unadorned code files. Comments are shown with one poundsign; messages from the execution of this code is flagged with two poundsigns.

There are two goals of this material:

1. Provide replication code for the materials in Spatial Regression Models, version 2. This is hosted on \texttt{https://srmbook.com}
2. To elaborate a bit on what the code does to provide instruction on how to construct and work with maps in \texttt{R}.

This was developed on the following platform. Earlier (and later) platforms should work (as should windows and linux flavored platforms). However, some libraries may not be available for all platforms. Such is life.

If you have any feature requests or find bugs herein, please do not hesitate to pass them along to the authors (michael.don.ward at gmail.com and ksg at essex.ac.uk).

Figure 2.2

The following \texttt{R} code illustrates the use of the GIS data on Cholera in London, aka the John Snow map. The shapefile is downloaded and then read with \texttt{rgdal}. Some warnings occur but can be ignored for this exercise.

```r
# Chapter 2 Code for Ward & Gleditsch, Spatial Regression Models
# Sage, 2018.

# R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
# Copyright (C) 2017 The R Foundation for Statistical Computing
# Platform: x86_64-apple-darwin15.6.0 (64-bit)

# Figure 2.2 John Snow Map of 1854 Data on Cholera deaths
# setup
rm(list = ls())
library(maptools)

## Loading required package: sp
## Checking rgeos availability: TRUE
library(rgdal)
```

## rgeos: version: 1.2-8, (SVN revision 663)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 2.1.3, released 2017/20/01
## Path to GDAL shared files: /Library/Frameworks/R.framework/Versions/3.4/Resources/library/rgdal/gdal
## Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ_VERSION: 493]
## Path to PROJ.4 shared files: /Library/Frameworks/R.framework/Versions/3.4/Resources/library/rgdal/proj
## Linking to sp version: 1.2-4

```r
library(rgeos)
```

## rgeos version: 0.3-23, (SVN revision 546)
## GEOS runtime version: 3.6.1-CAPI-1.10.1 r0
## Linking to sp version: 1.2-4
## Polygon checking: TRUE

```r
# Download the zip archive of geographical information
# download.file(url = "http://www.rtwilson.com/downloads/SnowGIS_v2.zip",
#    destfile = "SnowGIS_v2.zip")
# Unzip
# unzip(zipfile = "SnowGIS_v2.zip")
# List files in the unzipped folder
# dir(path = ".!/SnowGIS")
# Load gray scale image
OSMap <- readGDAL("../shapefiles/SnowGIS/OSMap_Grayscale.tif")

## ../shapefiles/SnowGIS/OSMap_Grayscale.tif has GDAL driver GTiff
## and has 1070 rows and 1169 columns

```r
# open png device
#png(filename = "graphics/Fig2.2.png",
#     width = 480, height = 480, units = "px", pointsize = 12,
#     bg = "white")
# Set margins to zero
par(mar = c(0,0,0,0))
# Plot it
image(OSMap, col = grey(1:999/1000))
```

## Load geographical information of sites of cholera deaths
Cholera_Deaths <- readShapePoints(fn = ".!/shapefiles/SnowGIS/Cholera_Deaths.shp")

```r
## Warning: use rgdal::readOGR or sf::st_read
# Plot sites of cholera deaths
plot(Cholera_Deaths, pch = 16, cex = log(Cholera_Deaths$Count), col = "red", add = TRUE)
```

## Load geographical information of sites of pumps
Pumps <- readShapePoints(fn = ".!/shapefiles/SnowGIS/Pumps.shp")

```r
## Warning: use rgdal::readOGR or sf::st_read
# Plot sites of pumps
plot(Pumps, add = TRUE, pch = 25, col = "blue", cex=2)
```

```r
#dev.off() # close png device
```

---

**Figure 2.3**

---

# Chapter 2 Code for Ward & Gleditsch, Spatial Regression Models
# Sage, 2018.

# R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
# Copyright (C) 2017 The R Foundation for Statistical Computing
# Platform: x86_64-apple-darwin15.6.0 (64-bit)

# Figure 2.3 Simple Latitude - Longitude Projection
# setup

---

2
Figure 1: Cholera in 1854, London

```r
rm(list = ls())
library(maptools)
library(rgdal)

wmap <- readOGR(dsn="../shapefiles/110m_cultural", layer="ne_110m_admin_0_countries")

# OGR data source with driver: ESRI Shapefile
# Source: "../shapefiles/110m_cultural", layer: "ne_110m_admin_0_countries"
# with 177 features
# It has 63 fields

#png(filename = "graphics/Fig2.3.png",
#     width = 776, height = 480, units = "px", pointsize = 12,
#     bg = "white")

plot(wmap)

abline(v=180,lwd=4)
abline(v=-180,lwd=4)

#dev.off()
```

Figure 2.4

# Chapter 2 Code for Ward & Gleditsch, Spatial Regression Models
# Sage, 2018.

# R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
# Copyright (C) 2017 The R Foundation for Statistical Computing
# Platform: x86_64-apple-darwin15.6.0 (64-bit)

# Figure 2.4 World Map: Robinson Projection
# setup
Figure 2: Simple Latitude - Longitude Projection

```r
rm(list = ls())

library(maps)
library(maptools)
library(rgdal)
library(rgeos)
library(sp)

# plotting packages
library(ggplot2)
library(ggmap)

wmap <- readOGR(dsn="../shapefiles/110m_cultural", layer="ne_110m_admin_0_countries")

## OGR data source with driver: ESRI Shapefile
## Source: "../shapefiles/110m_cultural", layer: "ne_110m_admin_0_countries"
## with 177 features
## It has 63 fields

wmap <- wmap[wmap@data$name != "Antarctica", ]  # Get Rid of Antarctica
wmap <- wmap[wmap@data$name != "Greenland", ]  # Get Rid of Greenland

# plot map
wmap_robin <- spTransform(wmap, CRS("+proj=robin"))

## Regions defined for each Polygons

ggplot(wmap_robin, aes(long,lat, group=group)) +
  geom_polygon(fill="white") + geom_path(color="black") +
  labs(title="World Map, with Robinson Projection", x="Longitude", y="Latitude") +
  coord_equal() + scale_x_continuous(labels = waiver()) + scale_y_continuous(labels = waiver())
```
World Map, with Robinson Projection

Figure 3: World Map: Robinson Projection

#dev.off()

Figure 2.5

# Chapter 2 Code for Ward & Gleditsch, Spatial Regression Models
# Sage, 2018.

# Figure 2.5 Winkel Tripel Projection
# setup

rm(list = ls())

library(maps)
library(maptools)
library(rgdal)
library(rgeos)
library(sp)

# plotting packages
library(ggplot2)
library(ggmap)

wmap <- readOGR(dsn="../shapefiles/110m_cultural", layer="ne_110m_admin_0_countries")

## OGR data source with driver: ESRI Shapefile
## Source: ".../shapefiles/110m_cultural", layer: "ne_110m_admin_0_countries"
## with 177 features
## It has 63 fields

wmap <- wmap[wmap@data$name!="Antarctica", ] # Get Rid of Antarctica
wmap <- wmap[wmap@data$name!="Greenland", ] # Get Rid of Greenland

# plot map
wmap_WIII <- spTransform(wmap, CRS("+proj=wintri"))
World map, with Winkel Tripel projection

Figure 4: Winkel Tripel Projection

wmap_df_wIII <- (wmap_wIII) # not fortified

#png(filename = "graphics/Fig2.5.png",
# width = 776, height = 480, units = "px", pointsize = 12,
# bg = "white")

ggplot(wmap_df_wIII, aes(long,lat, group=group)) +
geom_polygon(fill="grey60") + geom_path(color="black") +
labs(title="World map, with Winkel Tripel projection", x="Longitude", y="Latitude") +
coord_equal() + scale_x_continuous(breaks=0) + scale_y_continuous(breaks=0)

## Regions defined for each Polygons
# dev.off()

Figure 2.6

# Chapter 2 Code for Ward & Gleditsch, Spatial Regression Models
# Sage, 2018.

# R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
# Copyright (C) 2017 The R Foundation for Statistical Computing
# Platform: x86_64-apple-darwin15.6.0 (64-bit)
# Figure 2.8 Air Pollution from World Bank. 2013 Data.

# Figure 2.6 Winkel Tripel with Graticules and Bounding Box
# setup

rm(list = ls())
library(maps)
library(maptools)
library(rgdal)
library(rgeos)
library(sp)

# plotting packages
library(ggplot2)
library(ggmap)

wmap <- readOGR(dsn="../shapefiles/110m_cultural", layer="ne_110m_admin_0_countries")

## OGR data source with driver: ESRI Shapefile
## Source: "/shapefiles/110m_cultural", layer: "ne_110m_admin_0_countries"
## It has 177 features
## It has 63 fields
wmap <- wmap [wmap$@data$name != "Antarctica", ] # Get Rid of Antarctica
wmap <- wmap [wmap$@data$name != "Greenland", ] # Get Rid of Greenland

wmap_wIII <- spTransform(wmap, CRS("+proj=wintri")) # plot map
wmap_df_wIII <- (wmap_wIII) # not fortified

#png(filename = "graphics/Fig2.6.png",
# width = 776, height = 480, units = "px", pointsize = 12,
# bg = "white")

grat <- readOGR("../shapefiles/ne_110m_graticules_all", layer="ne_110m_graticules_15")

## OGR data source with driver: ESRI Shapefile
## Source: "/shapefiles/ne_110m_graticules_all", layer: "ne_110m_graticules_15"
## It has 35 features
## It has 5 fields
## Integer64 fields read as strings: degrees scalerank
grat_wintri <- spTransform(grat, CRS("+proj=wintri")) # reproject graticule
grat_df_wintri <- (grat_wintri) # not fortified
bbox <- readOGR("../shapefiles/ne_110m_graticules_all", layer="ne_110m_wgs84_bounding_box")

## OGR data source with driver: ESRI Shapefile
## Source: "/shapefiles/ne_110m_graticules_all", layer: "ne_110m_wgs84_bounding_box"
## It has 1 features
## It has 2 fields
bbox_wintri <- spTransform(bbox, CRS("+proj=wintri")) # reproject bounding box
bbox_wintri_df <- (bbox_wintri) # not fortified

ggplot(bbox_wintri_df, aes(long, lat, group=group)) +
   geom_polygon(fill="white") + geom_path(color="red") +
   geom_polygon(data=wmap_df_wIII, aes(long, lat, group=group, fill=id )) +
   geom_path(data=grat_df_wintri, aes(long, lat, group=group, fill="NULL"),
             linetype="solid", color="grey") +
   labs(title="World Map (Winkel Tripel)",x="Longitude",y="Latitude") +
   theme(legend.position="none")

## Regions defined for each Polygons
## Regions defined for each Polygons
## Warning: Ignoring unknown aesthetics: fill
#dev.off()

**Figure 5**: Winkel Tripel with Graticules and Bounding Box

---

**Figure 2.7**

```r
# Chapter 2 Code for Ward & Gleditsch, Spatial Regression Models
# Sage, 2018.

# R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
# Copyright (C) 2017 The R Foundation for Statistical Computing
# Platform: x86_64-apple-darwin15.6.0 (64-bit)

# Figure 2.7 Japanese Population Density
# setup

rm(list = ls())

#set repository; this uses the "hutch"
local({
  r <-getOption("repos")
  r["CRAN"] <- "https://cran.fhcrc.org/"
  options(repos = r)
})

# loading libraries; install only once.
# for example
# install.packages("choroplethr",quiet=TRUE)
```
# spatial packages
library(choroplethr)

## Loading required package: acs
## Loading required package: stringr
## Loading required package: XML
##
## Attaching package: 'acs'
## The following object is masked from 'package:base':
##
## apply

library(choroplethrAdmin1)
library(cshapes)

## Loading required package: plyr
##
## Attaching package: 'plyr'
## The following object is masked from 'package:maps':
##
## ozone

library(maps)
library(maptools)
library(rgdal)
library(rgeos)
library(sp)

## plotting packages
library(ggplot2)
library(ggmap)
library(plotly)

## Attaching package: 'plotly'
## The following objects are masked from 'package:plyr':
##
## arrange, mutate, rename, summarise
## The following object is masked from 'package:ggmap':
##
## wind
## The following object is masked from 'package:ggplot2':
##
## last_plot
## The following object is masked from 'package:stats':
##
## filter
## The following object is masked from 'package:graphics':
##
## layout

library(RColorBrewer)
library(treemap)
Japanese Population Density
per square kilometer
2012

Figure 6: Japanese Population Density
library(cshapes)
library(maps)
library(maptools)
library(rgdal)
library(rgeos)
library(sp)

# plotting packages
library(ggplot2)
library(ggmap)
library(plotly)
library(RColorBrewer)
library(treemap)

# data wrangling packages
library(broom)
library(classInt)
library(devtools)
library(dplyr)

##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:plyr':
##
## arrange, count, desc, failwith, id, mutate, rename, summarise,
## summarize

## The following object is masked from 'package:acs':
##
## combine

## The following objects are masked from 'package:rgeos':
##
## intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
## filter, lag

## The following objects are masked from 'package:base':
##
## intersect, setdiff, setequal, union

library(gridExtra)

##
## Attaching package: 'gridExtra'
This set of code shows how to use point data and a map with boundaries to collect counts of things (here events) that are inside of the boundaries. Two things are noteworthy. One is that there are event that are outside of the
boundaries and these must be excised for mapping and aggregation purposes. The second is that these can be imposed on terrain from other projections.

# Chapter 2 Code for Ward & Gleditsch, Spatial Regression Models
# Sage, 2018.

# R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
# Copyright (C) 2017 The R Foundation for Statistical Computing
# Platform: x86_64-apple-darwin15.6.0 (64-bit)

# Figure 2.9 Conflict Events in Bosnia-Herzegovina
# Figure 2.10 Aggregations of Conflict events in B-H
# Figure 2.11 Conflict Events in B-H with Elevation

rmlist = ls())
# Libraries used
library(sp)
library(maptools)
library(cshapes)
library(RColorBrewer)
library(classInt)

# Read municipalities shapefile
units <- readShapePoly("../shapefiles/bosnia/bosnia_units.shp", proj4string=CRS("+proj=longlat +ellps=WGS84")

## Warning: use rgdal::readOGR or sf::st_read
units$unit <- c(1:dim(units@data)[1])

# Read the ACLED conflict events
events <- readShapePoints("../shapefiles/bosnia/bosnia_twosided.shp", proj4string=CRS("+proj=longlat +ellps=WGS84"))

## Warning: use rgdal::readOGR or sf::st_read

# Overlay analysis (new preferred syntax)
res.overlay <- over(events, geometry(units), returnList = F)

# Add a unit index number to events
events$unit <- res.overlay

# Find the ID of the events outside units
events[is.na(events$unit),]

## coordinates EventType Event_ID Canton_ID Region_ID Type_ID
## 2 (17.99, 45.14) 1 6 13 2 1
## 142 (17.99, 45.14) 1 252 13 2 9
## 143 (17.99, 45.14) 1 253 13 2 9
## 171 (17.99, 45.14) 1 281 13 2 9
## 174 (17.99, 45.14) 2 284 13 2 9
## 330 (19.71, 44.86) 1 455 18 3 9
## 395 (13.39, 43.98) 2 514 9 1 9

## Location Latitude Longitude unit
## 2 Bosanski Brod 45.14 17.99 NA
## 142 Bosanski Brod 45.14 17.99 NA
## 143 Bosanski Brod 45.14 17.99 NA
## 171 Bosanski Brod 45.14 17.99 NA
## 174 Bosanski Brod 45.14 17.99 NA
## 330 Bukovac 44.86 19.71 NA
## 395 Cepiljace 43.98 13.39 NA

## Tag all events outside units
events$inside <- 1
events$inside[is.na(events$unit)] <- 0

# Delete events without a unit index number
events2 <- events[!is.na(events$unit),]

# Plot all events and municipalities
#png(filename = "graphics/Fig2.9.png",
# width = 776, height = 480, units = "px", pointsize = 12,
# bg = "white")
plot(units)
plot(events, add=T, cex=0.5, pch=19)

#dev.off()

# Aggregate by unit index
events2$event <- 1
events.by.unit <- aggregate(event ~ unit, data=events2, sum)
names(events.by.unit)[2] <- "event.count"

# Add to units spatial data frame
units$data = data.frame(units@data, 
  events.by.unit[match(units@data[,"unit"], events.by.unit[,"unit"]),])
units$event.count[is.na(units$event.count)] <- 0
Figure 8: Bosnia-Herzegovina Maps

Conflict events
- under 1
- 1 – 5
- 5 – 10
- 10 – 20
- over 20

Figure 9: Bosnia-Herzegovina Maps

# Plot event counts
#png(filename = "graphics/Fig2.10.png",
# width = 776, height = 480, units = "px", pointsize = 12,
# bg = "white")
breaks <- c(0,1,5,10,20,max(units$event.count))
cols <- brewer.pal(length(breaks), "Greys")
plot(units, bty="n", col=cols[findInterval(units$event.count, breaks, all.inside=T)])
legend(x="bottomleft", legend = leglabs(breaks), fill=cols,
       bty="n", title="Conflict events")

#dev.off()

# Load terrain information
terrain <- readAsciiGrid("../shapefiles/bosnia/bosnia_terrain.asc", proj4string=CRS("+proj=longlat +ellps=WGS84"))
summary(terrain)

## Object of class SpatialGridDataFrame
## Coordinates:
##   min    max

15
## Additional information about mapping

This is included to show that things can go wrong. Easily. This takes a while to run, even on a currently fast platform. So you may want to do this at an appropriate time.

```r
# dev.off()

library(wbstats)
library(cshapes)
library(ggplot2)
```
# Get data from the World Bank

```r
pollution_data <- wb(indicator = "EN.ATM.PM25.MC.M3", startdate = 2013, enddate = 2014)
temp.data <- pollution_data[,c(1:3,5:6)]  # cleanup
```

# generate a map

```r
wmapchoro <- cshep(date = as.Date("2015-01-01"))  # get map from CSHAPES
```

## Warning: use rgdal::readOGR or sf::st_read

```r
wmapchoro@data$id <- as.character(rownames(wmapchoro@data))
wmapchoro.df <- fortify(wmapchoro)
```

## Regions defined for each Polygons

```r
wmapchoro.df <- join(wmapchoro.df, wmapchoro@data, by="id")
```

## issue comes up when using merge

```r
wmap.df_usingMerge <- merge(wmapchoro.df, temp.data, 
    by.x ="ISO1AL2", by.y="iso2c", 
    all.x=T, all.y=F)
```

## instead of using merge, add the variable in with match

```r
wmap.df <- wmapchoro.df
wmap.df$value <- temp.data$value[match(wmap.df$ISO1AL2, temp.data$iso2c)]
```

## Plotting

```r
ggcmap_usingMerge <- ggplot(data=wmap.df_usingMerge, 
    aes(x=long, y=lat, group=group))
ggcmap_usingMerge <- ggcmap_usingMerge + geom_polygon(aes(fill = value))
ggcmap <- ggplot(data=wmap.df, aes(x=long, y=lat, group=group))
ggcmap <- ggcmap + geom_polygon(aes(fill = value))
ggrid.arrange(ggcmap_usingMerge, ggcmap)
```
# Reorder using merge data

```r
wmap.df$orderCol <- paste0(wmap.df$order, wmap.df$id, 
  wmap.df$group, wmap.df$piece, 
  wmap.df$long, wmap.df$lat)

wmap.df_usingMerge$orderCol <- paste0(wmap.df_usingMerge$order, 
  wmap.df_usingMerge$id, 
  wmap.df_usingMerge$group, 
  wmap.df_usingMerge$piece, 
  wmap.df_usingMerge$long, 
  wmap.df_usingMerge$lat)

wmap.df_usingMerge <- wmap.df_usingMerge[match(wmap.df$orderCol, 
  wmap.df_usingMerge$orderCol), ]
```