Waller SISMID 2023: Houston Crime Slippery Slopes

Lance A. Waller

7/17/2023

Houston Crime Geographically Weighted Regression Example

This file reads in data from a shapefile of Houston census tracts (using the same code in the markdown file we used for Lecture 2).

Next we will use "spgwr" to fit geographically weighted regression.

Same code as in GIS example to open libraries, set working directory, and read in the Houston shapefile.

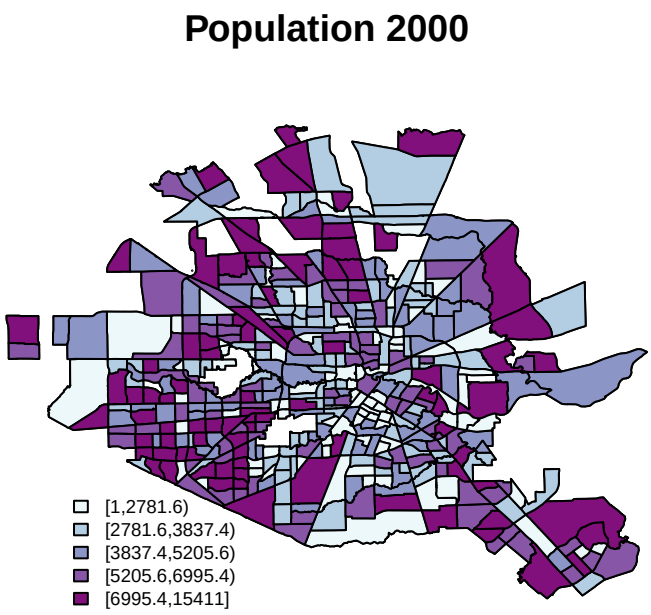
```
##Load libraries

library(maptools)     # loads sp library too
library(RColorBrewer) # creates nice color schemes
library(classInt)     # finds class intervals for continuous variables
library(spgwr)        # Adds the geographically weighted regression functions
library(rgdal)        # Provides tool for reading in shapefiles
library(here)

# Set my working directory (Lance's here for example)
# setwd("~/OneDrive - Emory University/meetings/SISMID.2022/SISMID.2022.Waller.Rcode")

##Read in shapefile - Houston Census Tracts
houston = readOGR(dsn = here("data") , layer = "HoustonENAR2012final")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "/Users/lwaller/Downloads/2022-SISMID-main/data", layer: "HoustonENAR2012final"
## with 439 features
## It has 133 fields
```

Now to plot the map outlines (same code as before)

```
##Map the tracts using plot...old school
# Here are the outlines

plot(houston)

## Making choropleth maps
# Define the variable (attribute) to shade tracts by
pop2000 <- houston@data$POP2000
```

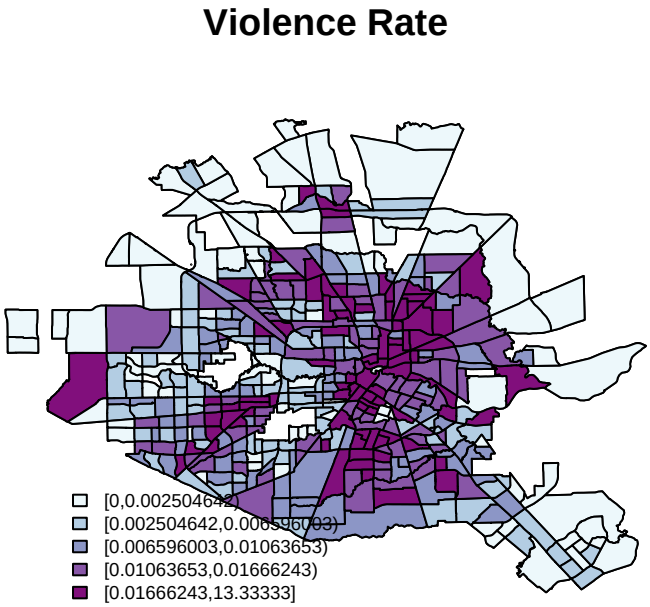
```
# Define the number of classes
nclr <- 5  # quintiles

# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(pop2000, nclr, style="quantile")
colcode <- findColours(class, plotclr)

#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="Population 2000",
    sub="Quantile (Equal-Frequency) Class Intervals")

#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```



Now set up the covariates and map the data (same code as Lecture 2)

```
# The data table has a lot of census data and various transformations
# of the violent crime, alcohol sales, and drug arrest data.  The next
# section pulls the values we want.

# Outcome:  Number of violent crimes by tract
violence = houston@data$violence_2
```

```r
# Divide by the 2000 population to get the rate
violence.rate = violence/houston@data$tot_pop

# Covariate 1 (log standardized total alcohol sales)
Z.log.total = houston@data$Zl_total

# Covariate 2 (log standardized illegal drug arrests)
Z.log.drug = houston@data$Zl_drug

# set up maps 2 rows of 2 plots each.
#par(mfrow=c(2,2))

# Plot Outcome first
plot(houston)
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(violence.rate, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="Violence Rate",
    sub="Quantile (Equal-Frequency) Class Intervals")
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```

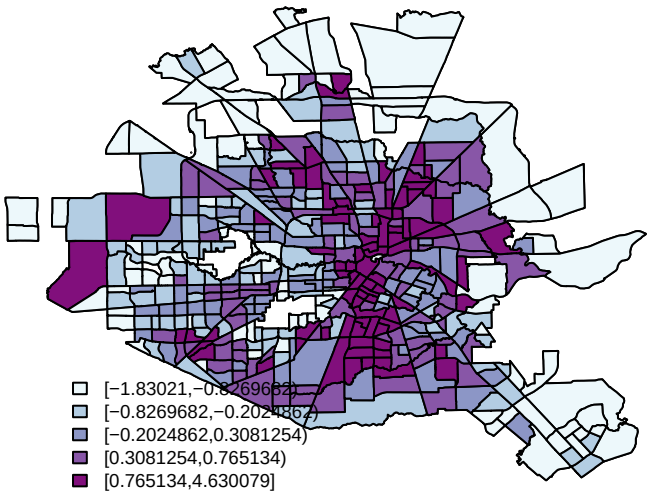


Quantile (Equal−Frequency) Class Intervals

```
# Next plot stdized log total alcohol sales
plot(houston)
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(Z.log.total, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="Std log total sales",
    sub="Quantile (Equal-Frequency) Class Intervals")
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```

Quantile (Equal−Frequency) Class Intervals

```
# Next plot stdized log illegal arrests
plot(houston)
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(Z.log.drug, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="Std log drug arrests",
    sub="Quantile (Equal-Frequency) Class Intervals")
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```




```
# This matches Figure 1 in Waller et al. 2007
```

Now we are ready to fit *geographically weighted regression* (GWR). GWR uses weighting to fit local regression values to allow the slopes associated with each covariate to change (smoothly) over space. (See Lecture notes for details). To estimate the association $\beta(s)$ for a location s, GWR uses kernel weights (similar to our intensity estimation for point process data) to weight observations (outcomes and covariates) near s more to give an estimate of β associated with location s. If you move over, the weights change and the estimated association (β). The kernels make sure that if you don't move far, the estimate doesn't change much so we get a smooth surface for β.

We do this in two steps.

First, we estimate the bandwidth for the kernel using cross validation (CV).

Next, we use the estimated bandwidth to fit the Poisson GWR model.

To get the bandwidth, "ggwr.sel" will test the cross validation score to find a minimum.

```
### Now to fit Poisson GWR!
# The function 'ggwr' in the 'spgwr' package uses syntax similar to 'glm'
# (like we would use in a standard Poisson regression).
# 'longlat' tells the function that our coordinates are in longitude and
# latitude coordinates.
# 'ggwr.sel' selects the bandwidth for GWR based on the data, the model,
# and cross-validation
```

```
houston.bw = ggwr.sel(violence ~ Z.log.total + Z.log.drug + offset(log(pop2000)),
                      data = houston,
                      #coords,
                      adapt = FALSE,
                      gweight = gwr.Gauss,
                      family = poisson,
                      verbose = TRUE,
                      longlat = TRUE,
                      RMSE=FALSE,
                      tol=.Machine$double.eps^0.25)
```

```
## Bandwidth: 31.46106 CV score: 1210981
## Bandwidth: 50.85425 CV score: 1225360
## Bandwidth: 19.47542 CV score: 1183007
## Bandwidth: 12.06788 CV score: 1146373
## Bandwidth: 7.48977 CV score: 1114997
## Bandwidth: 4.660343 CV score: 1080648
## Bandwidth: 2.91166 CV score: 19314054
## Bandwidth: 5.741088 CV score: 1096956
## Bandwidth: 3.992405 CV score: 1068727
## Bandwidth: 3.579598 CV score: 1063343
## Bandwidth: 3.324468 CV score: 1118224
## Bandwidth: 3.737276 CV score: 1064610
## Bandwidth: 3.453874 CV score: 1067964
## Bandwidth: 3.633023 CV score: 1063485
## Bandwidth: 3.58037 CV score: 1063341
## Bandwidth: 3.592384 CV score: 1063330
## Bandwidth: 3.607907 CV score: 1063359
## Bandwidth: 3.590323 CV score: 1063330
## Bandwidth: 3.59054 CV score: 1063330
## Bandwidth: 3.590497 CV score: 1063330
## Bandwidth: 3.590457 CV score: 1063330
## Bandwidth: 3.590497 CV score: 1063330
```

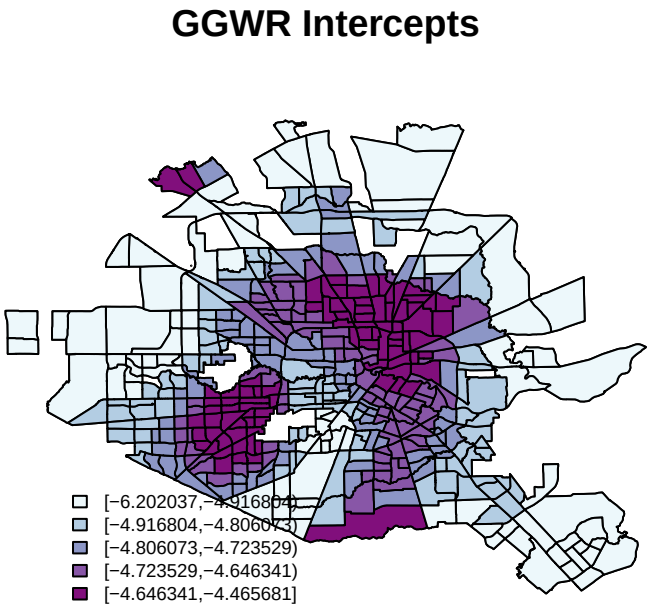
```
houston.ggwr = ggwr(violence ~ Z.log.total + Z.log.drug + offset(log(pop2000)),
                    data = houston,
                    family=poisson,
                    bandwidth = houston.bw, longlat=TRUE)
```

Now to map the results.

The houston.ggwr object contains an element called SDF which contains the spatial data frame (SDF) of the outcomes. We will want to map the (spatially varying) intercept and the parameters associated with each of our two covariates. The notation is tricky since the spatial data frame is within the houston.ggwr object.

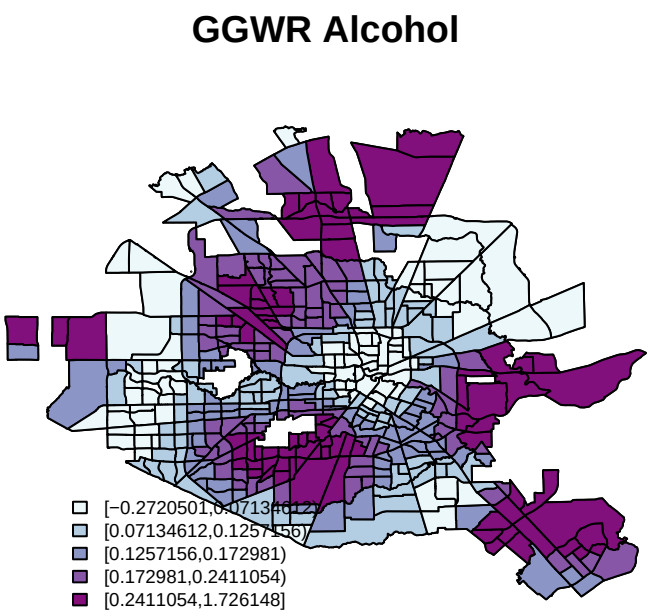
```
intercepts = houston.ggwr$SDF@data$X.Intercept
alcohol.effects = houston.ggwr$SDF@data$Z.log.total
drug.effects = houston.ggwr$SDF@data$Z.log.drug
```

```
# Plot the outlines first
plot(houston)
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(intercepts, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="GGWR Intercepts",
    sub="Quantile (Equal-Frequency) Class Intervals")
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```

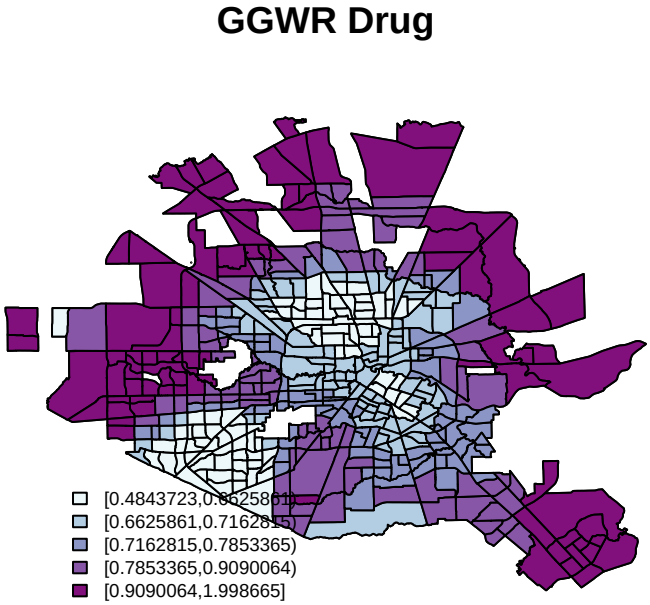



```
# Plot the outlines first
plot(houston)
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(alcohol.effects, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
```

```
title(main="GGWR Alcohol",
    sub="Quantile (Equal-Frequency) Class Intervals")
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```



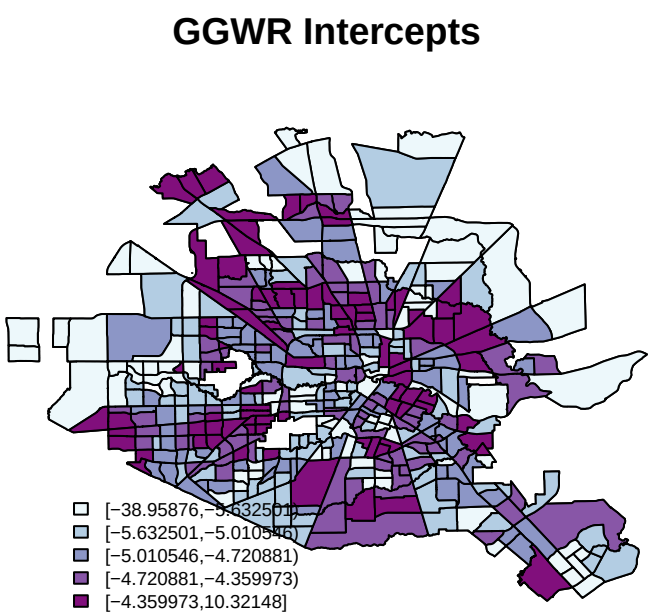

```
# Plot the outlines first
plot(houston)
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(drug.effects, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="GGWR Drug",
    sub="Quantile (Equal-Frequency) Class Intervals")
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```



These are very smooth maps. Let's try it with a smaller bandwidth.

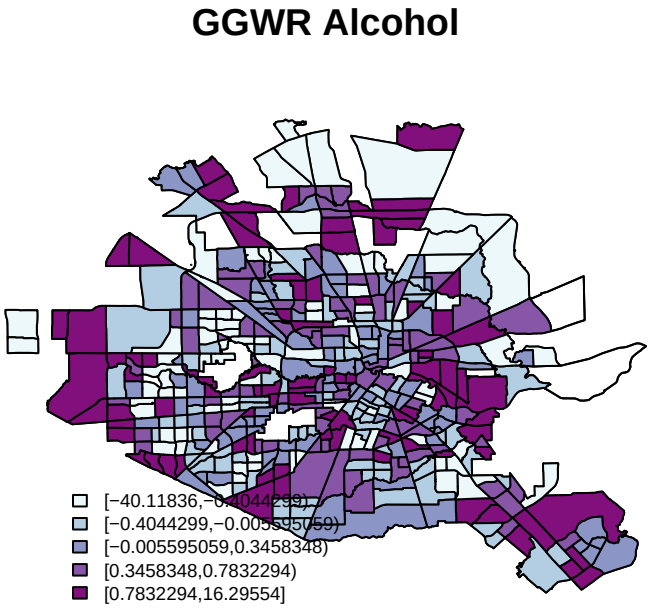
```
# Try with a smaller, fixed bandwidth (1/8 the size of the other)
# This is closer to what is presented in Figure 5 of Waller et al. 2007

houston.ggwr = ggwr(violence ~ Z.log.total + Z.log.drug + offset(log(pop2000)),
                    data = houston,
                    family=poisson,
                    bandwidth = houston.bw/8, longlat=TRUE)

# Now to map the outcomes

# The houston.ggwr object contains an element called SDF which contains
# the spatial data frame (SDF) of the outcomes.
# We will want to map the (spatially varying) intercept and the parameters
# associated with each of our two covariates.
#  The notation is tricky since the spatial data frame is within the
# houston.ggwr object.

intercepts = houston.ggwr$SDF@data$X.Intercept
alcohol.effects = houston.ggwr$SDF@data$Z.log.total
drug.effects = houston.ggwr$SDF@data$Z.log.drug


# Plot the outlines first
plot(houston)
```

```
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(intercepts, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="GGWR Intercepts",
    sub="Quantile (Equal-Frequency) Class Intervals")
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```




```
# Plot the outlines first
plot(houston)
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(alcohol.effects, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="GGWR Alcohol",
    sub="Quantile (Equal-Frequency) Class Intervals")
```

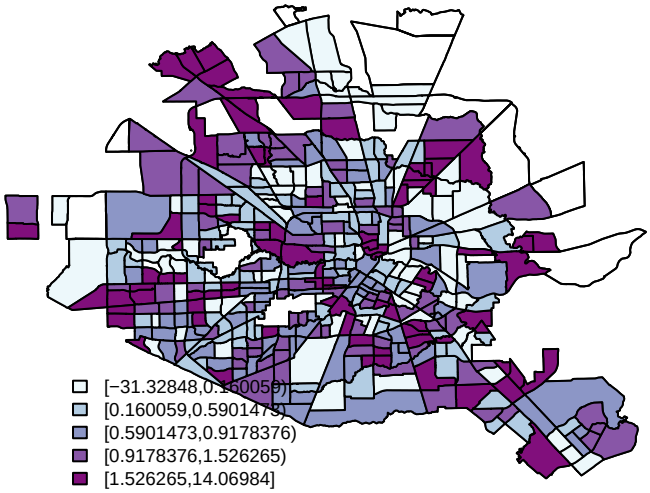
```
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```



Quantile (Equal−Frequency) Class Intervals

```
# Plot the outlines first
plot(houston)
# Define the number of classes
nclr <- 5  # quintiles
# Use RColorBrewer to choose the colors
plotclr <- brewer.pal(nclr,"BuPu")
class <- classIntervals(drug.effects, nclr, style="quantile")
colcode <- findColours(class, plotclr)
#Fill in the tracts with the colors
plot(houston, col=colcode, add=T)
#Add a title
title(main="GGWR Drug",
    sub="Quantile (Equal-Frequency) Class Intervals")
#Add a legend  (Coordinates are in longitude, latitude).
legend(-95.7, 29.65, legend=names(attr(colcode, "table")),
    fill=attr(colcode, "palette"), cex=0.6, bty="n")
```

GGWR Drug



Quantile (Equal−Frequency) Class Intervals