Advanced Statistics
Assignment 3
Claude M. Schrader

Research Question:

This analysis will utilize two methods of interpolation to contrast each model's strengths and weaknesses using carbon emissions in the midatlantic region from 2015.

Methods:

Using the Nearest Neighbor and Inverse Distance Weighting interpolation techniques, this analysis will create a visualization of carbon emissions surrounding the midatlantic region of the United States. Both of these tools are methods of averaging data from point observations for the purpose of getting a more general estimate of the study area.

The choropleth map in Figure 4 was created using the Nearest Neighbor technique. This technique uses the value of the nearest data point to interpolate Carbon Emissions. First it creates a set of polygons representing the sections of the study area affected by a particular point. These polygons are then used as the basis of a choropleth map to visualize the level of emissions on a broader scale.

This method's primary advantage is its simplicity and speed. The code required to create the analysis is very short, and does not require anything in the way of starting values, or calibration data. It would make a good first-pass visualization tool to give an understanding of the spatial distribution of a dataset, which could then be followed up with a more in depth analysis. This general visualization of the emissions distribution could be useful to understand which areas are more in need of health and education outreach, and where to target efforts to reduce emissions.

Nearest Neighbor has a few major disadvantages. It creates a busy image that masks the local geography of the study area. This image tends to be visually dominated by large polygons at the edges of the study area, but these polygons represent areas with a low density of survey points. Counterintuitively, the harder to read area of the map with many small polygons is the section with better underlying data. This method also has a major disadvantage in that areas of the map are interpolated based on only the closest data point. This means that the created visual is sensitive to data collection issues and statistical outliers.

The two isopleth maps in Figure 7 were created using the Inverse Distance Weighting interpolation technique. This method also creates a visualization based on interpolated data from limited data points. Unlike Nearest Neighbor, however, this technique creates a weighted mean based on nearby observations, instead of just one. All observations are considered, although closer data points will be weighted more heavily than those further away. This weight can be altered by setting the alpha value, usually set to 1 or 2.

This technique has a few advantages over Nearest Neighbor. One primary advantage is that since it isn’t dependent on one data point, the visualization will be less sensitive to data collection errors or statistical outliers. Since this technique creates smoother classes than Nearest Neighbor, it better emphasizes the continuous data represented than Nearest Neighbor’s sharp polygons do. Inverse Distance Weighting also tends to create a simpler and more balanced graphic, which should be easier to understand and follow.

One main limitation with this technique is that it is primarily a visualization technique. Because it does not create a test statistic or give you hard statistics, its applications are limited to exploratory uses, or as a first stage analysis before proceeding to more in depth techniques.

Conclusion:

Both Nearest Neighbor and Inverse Distance Weighting are interpolation tools that can provide useful visualizations by summarizing and averaging a limited dataset across a study area. Because they do not provide test statistics, they cannot be used for hypothesis testing. However, as a part of a suite of analyses, they can play an important role in understanding the spatial distribution of data. This information can be invaluable for things like allocating resources and understanding where effort should be best spent.

Figure 1: Midatlantic Carbon Equivalent of Emissions

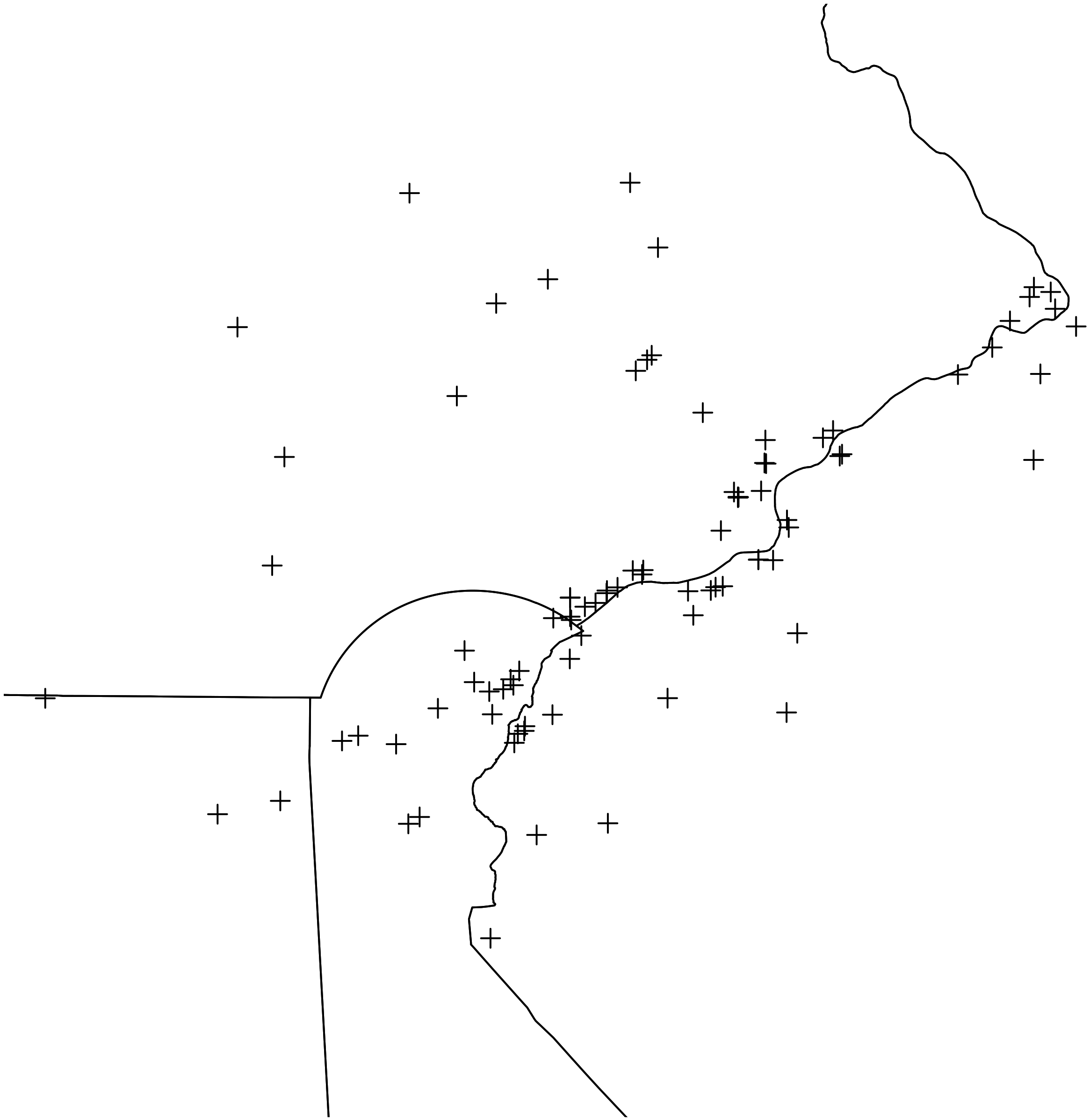


Figure 2: Distribution of Carbon Equivalent of Emissions, 2015

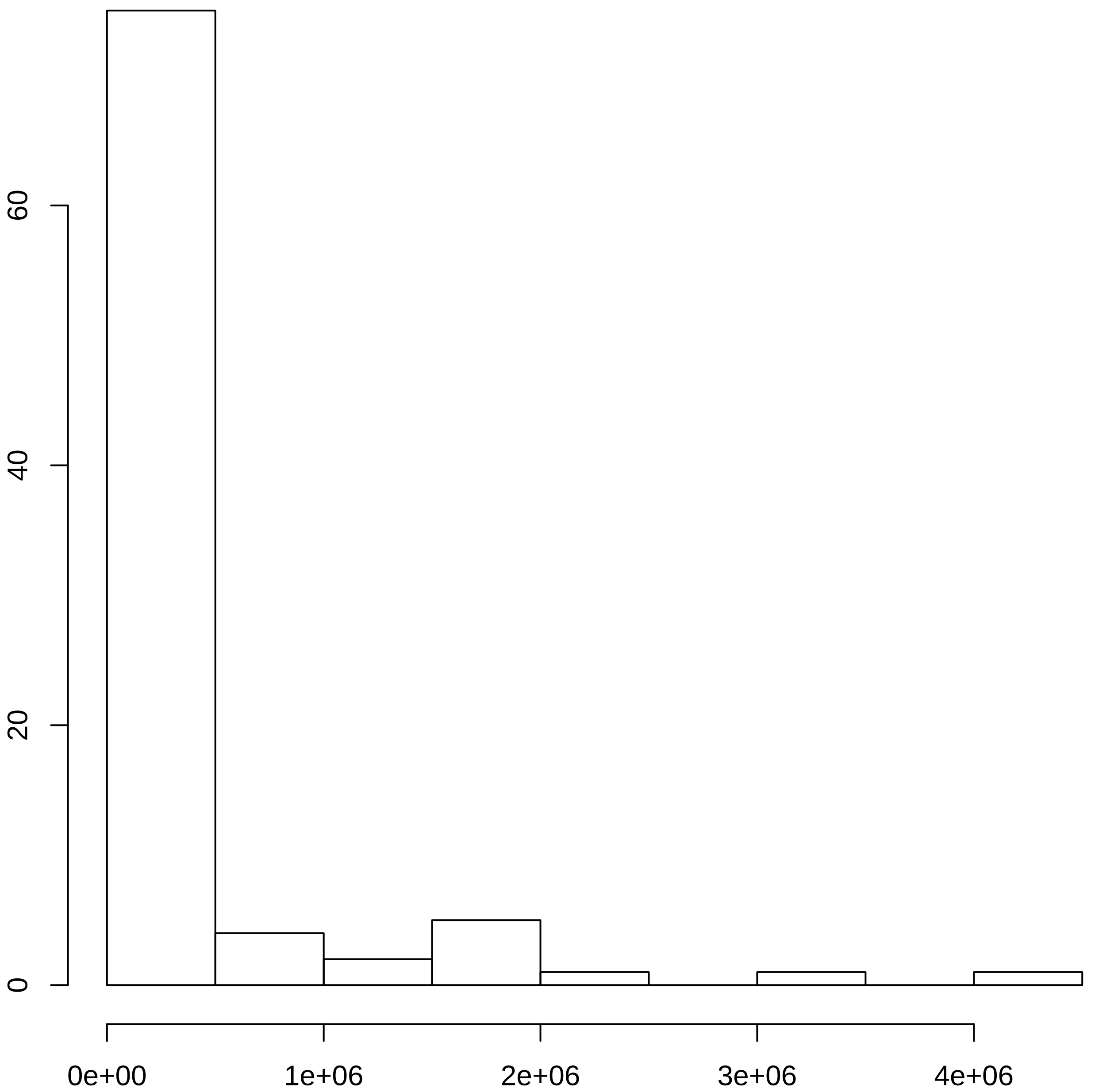


Figure 3: Nearest Neighbor Diagnostic

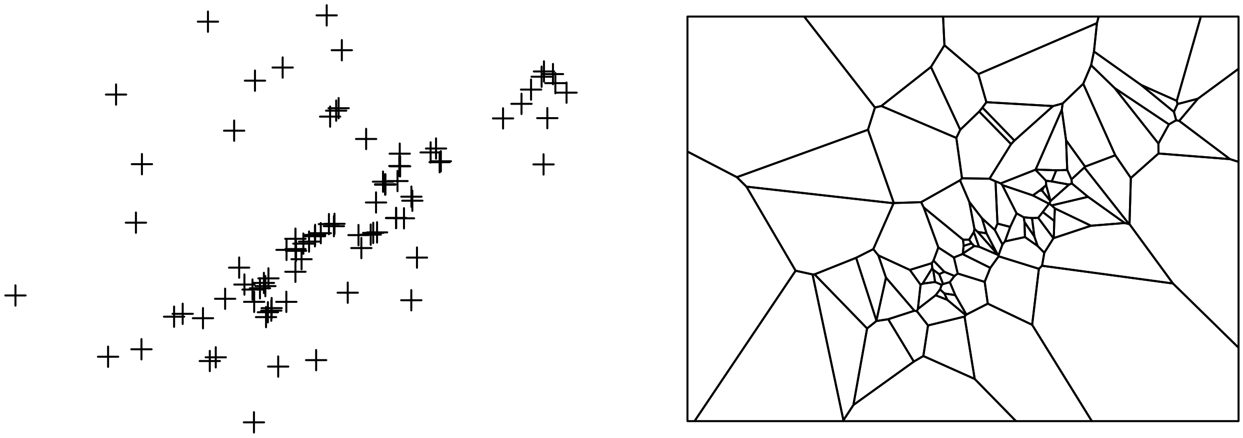


Figure 4: Carbon Equivalent of Emissions, 2015

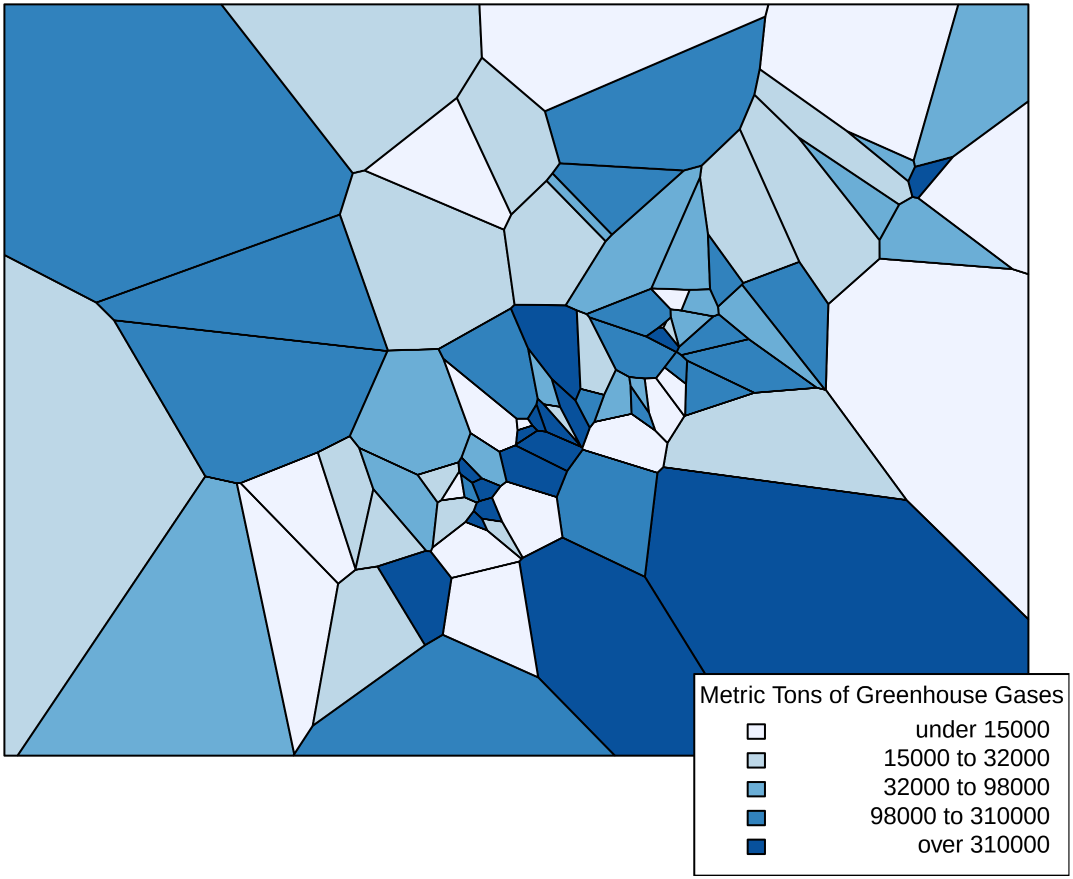


Figure 5: Density Estimate of Carbon Emissions

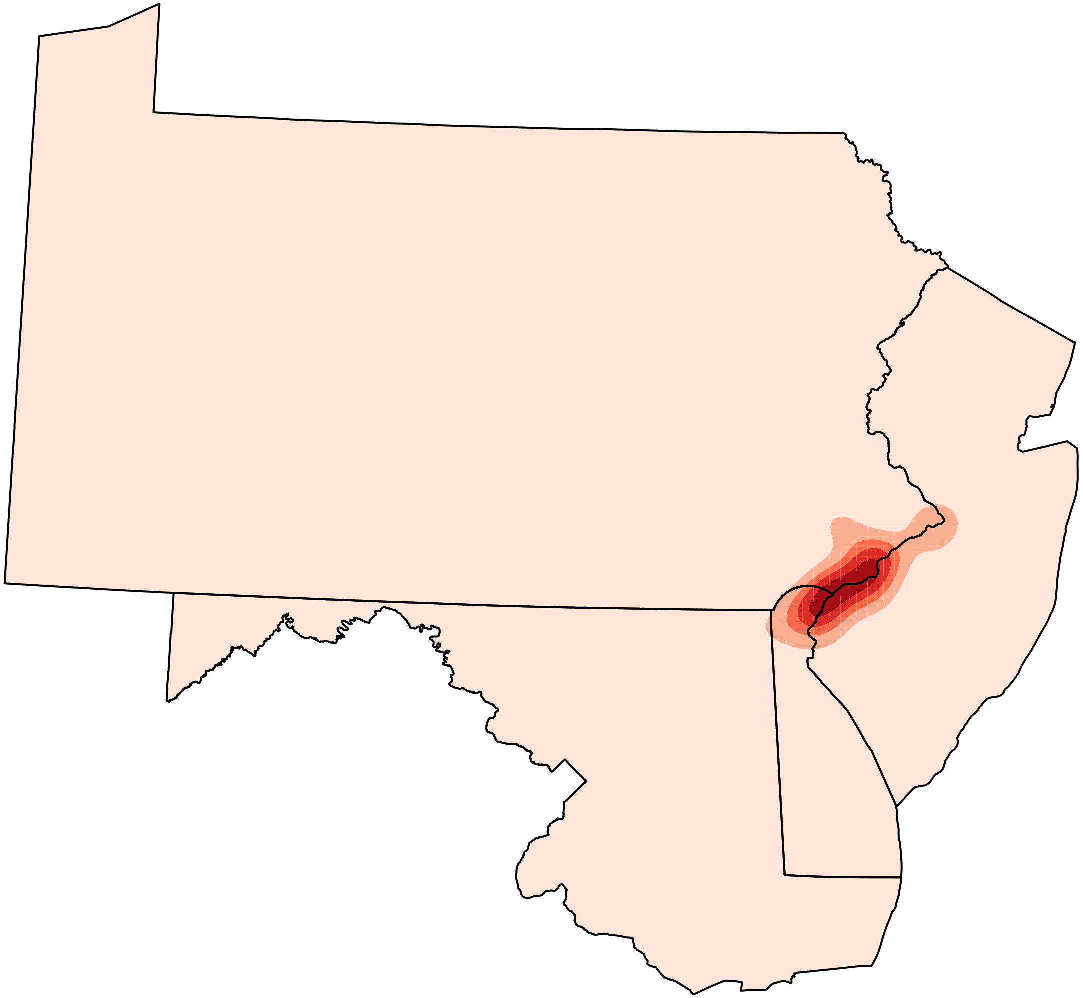


Figure 6: Density Estimate of Carbon Emissions, increased bandwidth

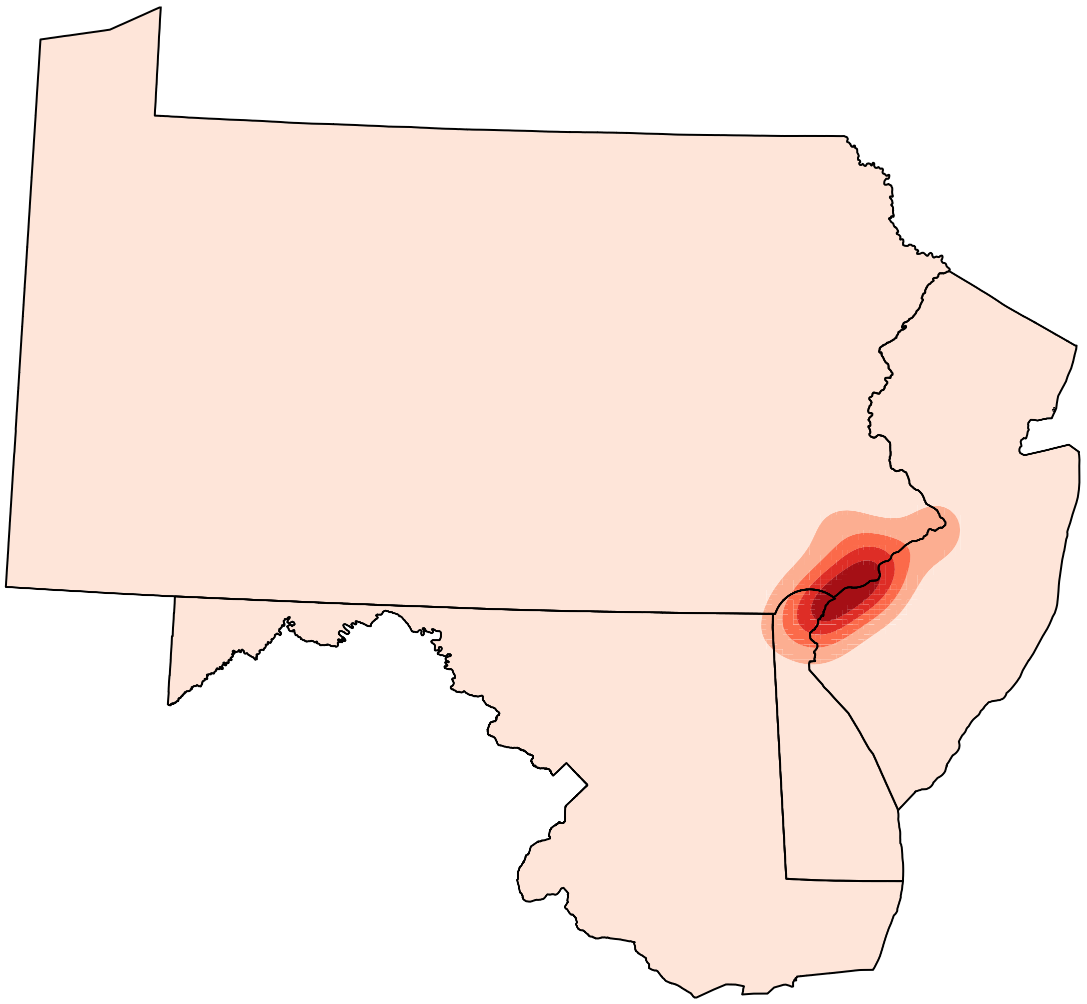
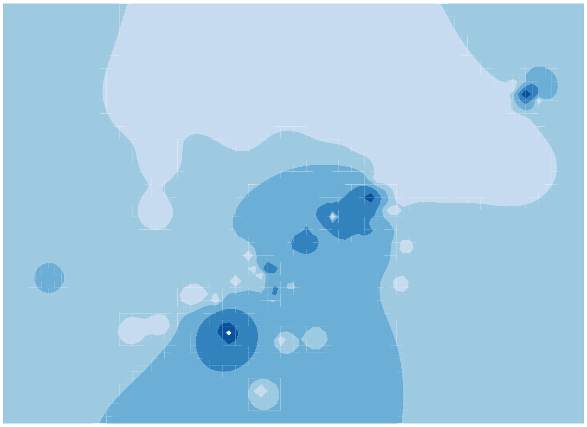
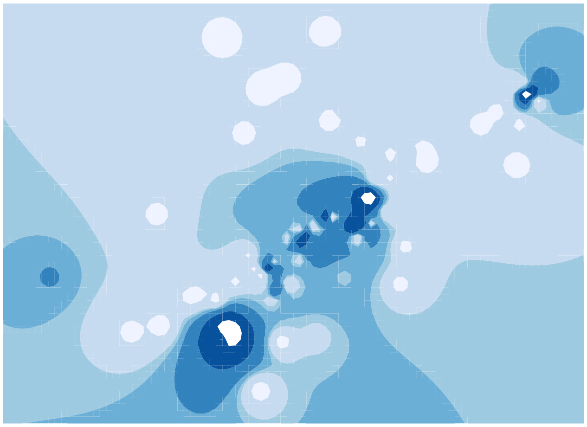


Figure 7: IDW estimate, alpha=1



alpha=2



Appendix: R code

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

setwd("C:/Users/tuj53509/Dropbox/docs/Temple/Advanced Statistics for Urban
Applications/Assignment3")


library(sp)

library(rgdal)

library(maptools)

library(RColorBrewer)

library(GISTools)

require(SpatialEpi)

require(deldir)

library(gstat)

library(classInt)


#load data

midatl_states <- readShapePoly("MidAtlStates_WGS/MidAtlStates_WGS.shp",
proj4string = CRS("+init=EPSG:4326"))

names(midatl_states)

plot(midatl_states)


greenhouse <- read.csv("2015EPA_FLIGHT_GreenhouseGas.csv", header = TRUE)

greenhouse <- as.data.frame(greenhouse)

head(greenhouse)


ghg <- greenhouse[,c("x","y")]

head(ghg)

ghg.spdf <- SpatialPointsDataFrame(greenhouse[,c("x","y")], data =
greenhouse, proj4string = CRS("+init=EPSG:4326"))

head(ghg.spdf)


par(mfrow=c(1,1))

par(mar=c(0,0,0,0))

plot(ghg.spdf)
```

```
ghg.utm <- spTransform(ghg.spdf, CRS("+init=EPSG:32618"))

sts.utm <- spTransform(midatl_states, CRS("+init=EPSG:32618"))

pdf(file="f1.ghgandstates.pdf")

par(mar=c(2,2,2,2))

plot(ghg.utm, main="Figure 1: Midatlantic Carbon Equivalent of Emissions")

plot(sts.utm, add = TRUE)

dev.off()


pdf(file="f2.ghghist.pdf")

par(mfrow=c(1,1), mar=c(2,2,2,2))

hist(ghg.utm$CarbonMetTon, breaks = 10, main="Figure 2: Distribution of
Carbon Equivalent of Emissions, 2015")

dev.off()


#Step 3: Nearest Neighbor Estimation

#set up functions

voronoipolygons = function(layer) {

  crds <- layer@coords

  z <- deldir(crds[,1], crds[,2])

  w <- tile.list(z)

  polys <- vector(mode='list', length=length(w))

  for (i in seq(along=polys)) {

    pcrds <- cbind(w[[i]]$x, w[[i]]$y)

    pcrds <- rbind(pcrds, pcrds[1,])

    polys[[i]] <- Polygons(list(Polygon(pcrds)),

                           ID=as.character(i))

  }

  SP <- SpatialPolygons(polys)

  voronoi <- SpatialPolygonsDataFrame(SP,

                                      data=data.frame(dummy=sapply(slot(SP,
'polygons'),


function(x) slot(x, 'ID'))))

  return(voronoi)

}
```

```
ghg.voro <- voronoipolygons(ghg.utm)

pdf(file="f3.nearestneighbor.pdf")

par(mfrow=c(1,2),mar=c(1,1,4,1))

plot(ghg.spdf)

title("Figure 3: Nearest Neighbor Diagnostic")

plot(ghg.voro)

dev.off()


par(mfrow=c(1,1), mar=c(2,0,2,0))

pdf(file="f4.nearestneighborchoro.pdf")

nneigh.shades <- auto.shading(ghg.utm$CarbonMetTon,
cols=brewer.pal(5,"Blues"))

choropleth(ghg.voro, ghg.utm$CarbonMetTon, shading=nneigh.shades,
main="Figure 4: Carbon Equivalent of Emissions, 2015")

choro.legend(px='bottomright',bg="white",sh=nneigh.shades, cex=.75,
title="Metric Tons of Greenhouse Gases")

dev.off()


#create two KDEs

par(mfrow=c(1,1))

#KDE1 - use the default bandwidth

ghg1.dens <- kde.points(ghg.utm, lims = sts.utm)

masker1 <- poly.outer(ghg1.dens, sts.utm, extend = 100)

pdf(file="f5.kde01.pdf")

level.plot(ghg1.dens)

add.masking(masker1)

plot(sts.utm, add = TRUE)

title("Figure 5: Density Estimate of Carbon Emissions")

dev.off()

#KDE2 - use manually set a bandwidth

ghg2.dens <- kde.points(ghg.utm, h=53000, lims = sts.utm)

masker2 <- poly.outer(ghg2.dens, sts.utm, extend = 100)

pdf(file="f6.kde02.pdf")

level.plot(ghg2.dens)
```

```
add.masking(masker2)

plot(sts.utm, add = TRUE)

title("Figure 6: Density Estimate of Carbon Emissions, increased bandwidth",
cex = .75)

dev.off()


#Step 4: Inverse Distance Weighting

proj4string(ghg.voro) <- CRS("+init=EPSG:32618")

ghg.grid1 <- spsample(ghg.voro,type='regular',n=6000)

#model 1 with alpha = 1.0

ghg.idw.est1 <- gstat::idw(CarbonMetTon~1, ghg.utm,newdata=ghg.grid1,idp=1.0)

#Model 2 with alpha = 2.0

ghg.idw.est2 <- gstat::idw(CarbonMetTon~1, ghg.utm,newdata=ghg.grid1,idp=2.0)


ux <- unique(coordinates(ghg.idw.est1)[,1])

uy <- unique(coordinates(ghg.idw.est1)[,2])

#prediction matrix for idw estimate 1

predmat <- matrix(ghg.idw.est1$var1.pred,length(ux),length(uy))

#prediction matrix for idw estimate 2

predmat2 <- matrix(ghg.idw.est2$var1.pred,length(ux),length(uy))

#calculate breaks

#classbreaks <- classIntervals(ghg.idw.est1$var1.pred,n=5,style="jenks")

#plot idw estimates

pdf(file="f7.idwests.pdf")

par(mar=c(1,1,2,0.1),mfrow=c(1,2))

plot(ghg.voro,border=NA,col=NA)

.filled.contour(ux,uy,predmat,col=brewer.pal(6,'Blues'),
                levels=c(0,125672,308036,352758,434444,682066,1385206))

title("Figure 7: IDW estimate, alpha=1", cex=0.75)

#levels=c(125672,308036,352758,434444,682066,1385206))

#plot IDW estimate 2

plot(ghg.voro,border=NA,col=NA)

.filled.contour(ux,uy,predmat2,col=brewer.pal(6,'Blues'),
                levels=c(0,125672,308036,352758,434444,682066,1385206))
```

```
title("alpha=2", cex=0.75)
dev.off()

#Step 5: Kriging
#,boundaries=seq(0,500000,l=100)
#ghg.vari.fit <- fit.variogram(ghg.vari.est,vgm(1,"Mat",1200000,l=51))
#ghg.vari.est <- variogram(CarbonMetTon~1,ghg.utm)
ghg.vari.est <- variogram(CarbonMetTon~1,ghg.utm)
plot(ghg.vari.est)
#ghg.vari.fit <- fit.variogram(ghg.vari.est,vgm(1, "Mat", 1290000, 1))
ghg.vari.fit <- fit.variogram(ghg.vari.est,vgm("Mat"))
plot(ghg.vari.est)

#try using eyefit package to automatically get a starting fit
install.packages("geoR")
library(geoR)
eyefit(ghg.vari.est)

plot(ghg.vari.est,model=ghg.vari.fit)
```