R Notebook

This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

rm(list = ls())  
library(spdep)

## Loading required package: sp

## Loading required package: spData

## To access larger datasets in this package, install the spDataLarge  
## package with: `install.packages('spDataLarge',  
## repos='https://nowosad.github.io/drat/', type='source')`

## Loading required package: sf

## Linking to GEOS 3.9.1, GDAL 3.2.1, PROJ 7.2.1

library(rflexscan)  
library(sf)  
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5 v purrr 0.3.4  
## v tibble 3.1.5 v dplyr 1.0.7  
## v tidyr 1.1.4 v stringr 1.4.0  
## v readr 2.0.2 v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(readxl)  
library(fields)

## Loading required package: spam

## Loading required package: dotCall64

## Loading required package: grid

## Spam version 2.7-0 (2021-06-25) is loaded.  
## Type 'help( Spam)' or 'demo( spam)' for a short introduction   
## and overview of this package.  
## Help for individual functions is also obtained by adding the  
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.

##   
## Attaching package: 'spam'

## The following objects are masked from 'package:base':  
##   
## backsolve, forwardsolve

## Loading required package: viridis

## Loading required package: viridisLite

##   
## Try help(fields) to get started.

data <- st\_read('C://Users//19795//OneDrive//Desktop//STAT 647//STAT 647 Project//Shapefile//2019\_Cty.shp')

## Reading layer `2019\_Cty' from data source   
## `C:\Users\19795\OneDrive\Desktop\STAT 647\STAT 647 Project\Shapefile\2019\_Cty.shp'   
## using driver `ESRI Shapefile'  
## Simple feature collection with 254 features and 7 fields  
## Geometry type: POLYGON  
## Dimension: XY  
## Bounding box: xmin: -106.6456 ymin: 25.83716 xmax: -93.50804 ymax: 36.5007  
## Geodetic CRS: NAD83

data$num\_FF = data$COUNT\_COUN  
data = subset(data, select = -c(COUNT\_COUN))  
data$FF\_Area = data$num\_FF/(data$SUM\_ALAND1+data$SUM\_AWATER)\*100000000 # USing a factor of 100000000

# Neighborhood based on Adjacency   
A <- st\_touches(data)  
A <- as.matrix(A)  
adj <- apply(A==1,1,which)  
  
  
for (i in 1:length(adj)) {  
 if(lengths(adj[i]) == 0){  
 print(i)  
 }  
}

## [1] 38  
## [1] 71  
## [1] 151  
## [1] 233

expected <- sum(data$FF\_Area) / nrow(data)  
  
fls <- rflexscan(x = as.numeric(data$Long), y = as.numeric(data$Lat),  
 observed = as.numeric(data$FF\_Area),  
 expected = as.numeric(expected),  
 name = data$COUNTYFP10,  
 clustersize = 20,  
 nb = adj)

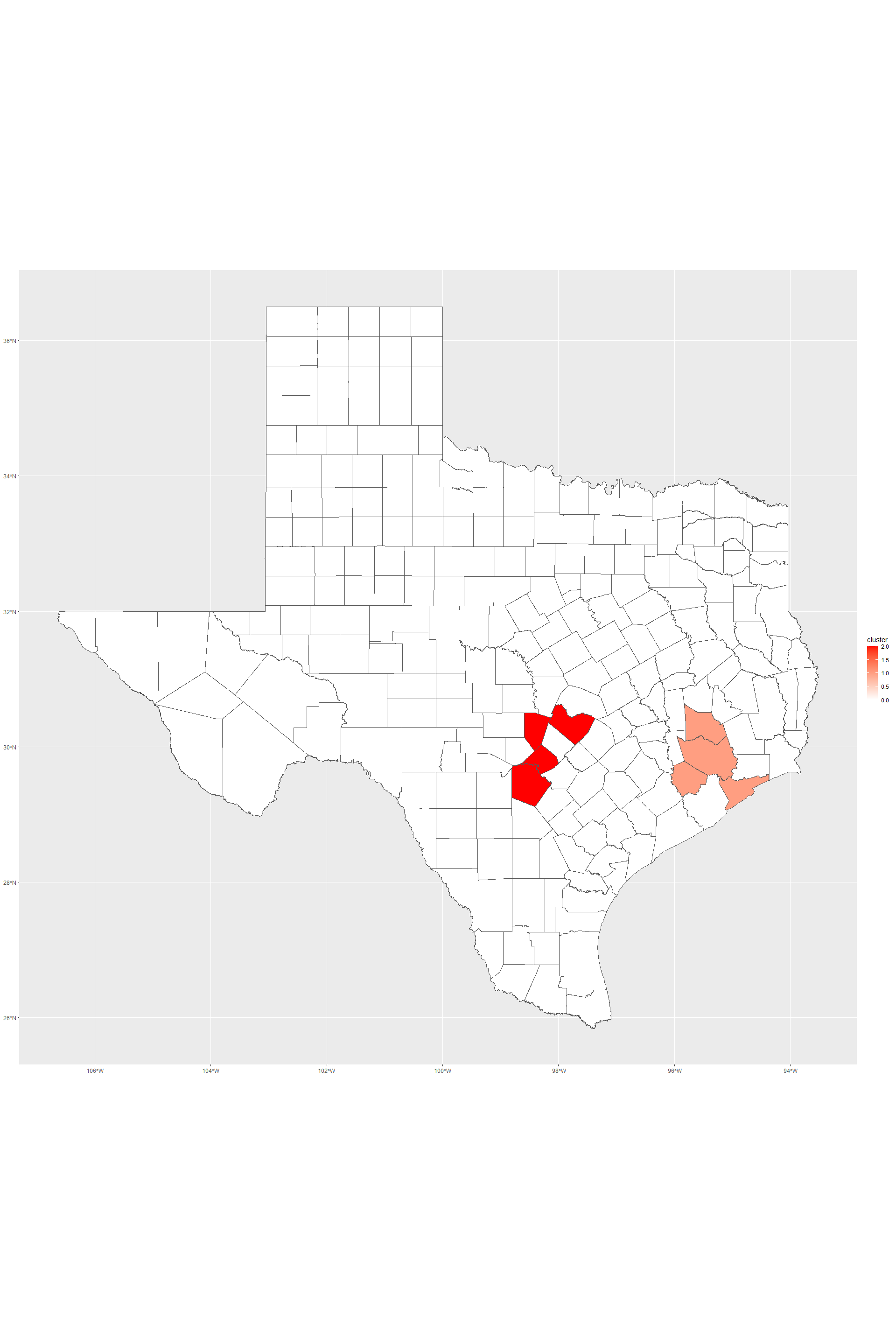
sum <- summary(fls)  
sum$cluster

## NumArea MaxDist Case Expected RR Stats P  
## 1 4 1.1529388 44 1.1496063 38.273973 134.058467 0.001  
## 2 4 1.1211008 16 1.1496063 13.917808 28.933222 0.001  
## 3 2 0.4643589 4 0.5748031 6.958904 4.417193 0.994  
## 4 1 0.0000000 3 0.2874016 10.438356 4.375102 0.994  
## 5 4 0.9188121 4 1.1496063 3.479452 2.194408 1.000

# Saving only significant clusters  
significant.clusters <- as.data.frame(list())  
  
all.clusters.data.frame <- as.data.frame(sum$cluster)  
  
for (row in 1:nrow(all.clusters.data.frame)){  
 if(all.clusters.data.frame[row, ]$P < 0.05){  
 significant.clusters <- rbind(significant.clusters, all.clusters.data.frame[row, ])  
 }  
}

data$cluster = 0 # To store the cluster number for the rows  
# Printing the differnt clusters   
num.clusters <- nrow(significant.clusters)  
  
for (row in 1:nrow(significant.clusters)){  
 for (ct in fls$cluster[[row]]$name){  
 ind <- which(data$COUNTYFP10 == ct)  
 data[ind, ]$cluster <- num.clusters - row + 1 # most significant cluster has higher value  
 }  
}

# Just adjust the plot now.  
  
ggplot(data=data)+geom\_sf(size = 0.05, aes(fill=cluster)) + scale\_fill\_gradient(high = "red", low = "white", na.value=NA)



ggsave("2019\_CT\_cluster\_size\_5.png")

## Saving 20 x 30 in image

Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.