MA615-Map assignment

Group2

10/27/2020

1.Reading data

First, we need to import the data form data ‘hurr_tracks’ and ‘rain’

```
#read the data from package `hurricaneexposuredata`
data("hurr_tracks")
data("rain")

#filter the 'Floyd-1999' and 'Allison-2001' from 'hurr_tracks' dataset
#we will use this rounte in mapping
Floyd_route <- filter(hurr_tracks, storm_id=="Floyd-1999")
Allison_route <- filter(hurr_tracks, storm_id=="Allison-2001")

#filter the 'Floyd-1999' and 'Allison-2001' from 'rain' dataset
original_Floyd_rain <- filter(rain, storm_id=="Floyd-1999")
original_Allison_rain <- filter(rain, storm_id=="Allison-2001")
```

2.Data preparing

Now, we need prepare two data fram for mapping.

```
#pick the 'fips' and sum of precip in both hurricanes.
original_Floyd_rain <- original_Floyd_rain %>%
  group_by(fips,storm_id) %>%
  summarise(precip_sum = sum(precip), .groups = "drop")

original_Allison_rain <- original_Allison_rain %>%
  group_by(fips,storm_id) %>%
  summarise(precip_sum = sum(precip),.groups = "drop")
```

‘county.fips’ is a database matching FIPS codes to maps package county and state names. And we will use it to merge the map we want.

```
#some fips in 'county.fips' only have 4 numbers
#And we need change it to 5 number, such as '1025' to '01025'
county_fips <- county.fips
county_fips$fips <- str_pad(county_fips$fips,5,side = "left",pad = "0")
```

First, we need create a new data frame which contains divided region information.

```
Floyd_rain_region <- merge(original_Floyd_rain,county_fips, by="fips") %>%
  separate(polyname, sep = ",", into = c("region","subregion"))

Allison_rain_region <- merge(original_Allison_rain,county_fips, by="fips") %>%
  separate(polyname, sep = ",", into = c("region","subregion"))
```

Then, we list the states required and turn it suitable for mapping.

```
states <- c("texas","oklahoma","kansas","louisiana","arkansas","missouri","iowa",
"wisconsin","michigan","illinois","indiana","ohio","kentucky","tennessee",
"alabama","mississippi","florida","georgia","south carolina","north carolina",
"virginia","west virginia","maryland","delaware","pennsylvania","new jersey",
"new york","connecticut","rhode island","massachusetts","vermont",
"new hampshire","maine")

#turn data from the maps package in to a data frame suitable for plotting with ggplot2
map_states <- map_data("county", states)
```

Next, we merge two data frames prepared which both contain region information. And now, we get the data which contains important information for mapping such as longitude and latitude.

```
#use 'merge' to combine two data frames.
Floyd_rain <- merge(Floyd_rain_region,map_states, by=c("region","subregion"))
Allison_rain <- merge(Allison_rain_region, map_states,by=c("region","subregion"))
```

At last, break the sum of precip into different range, and we get the data frame for mapping.

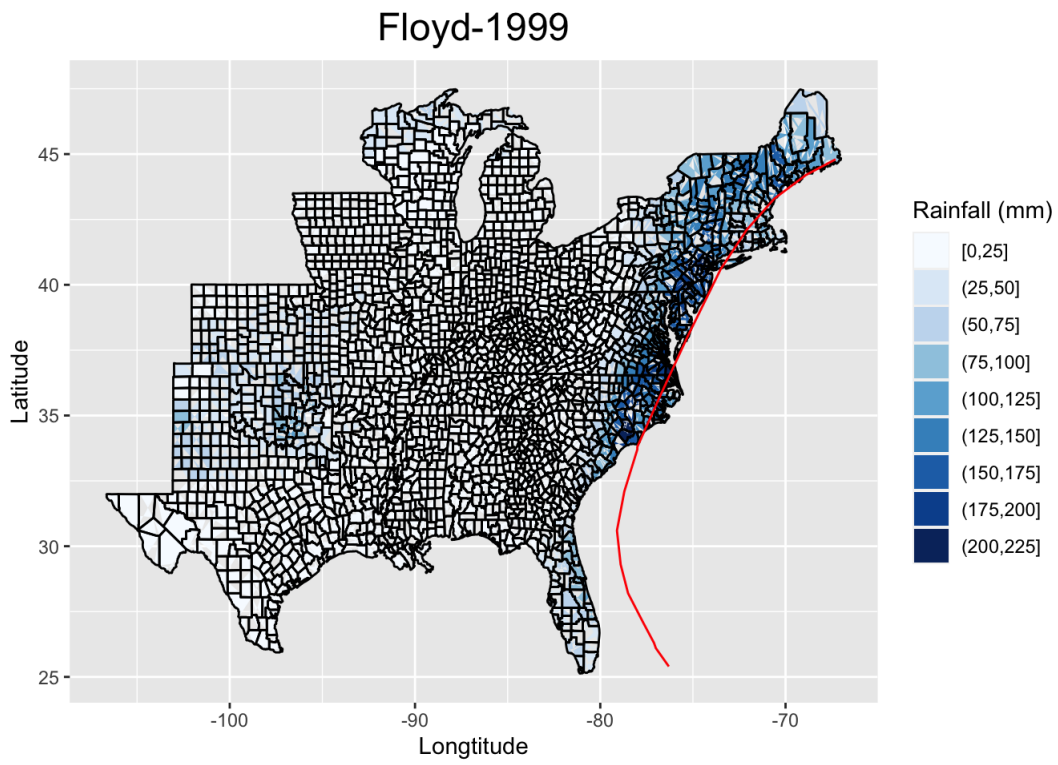
```
#divide sum of precip into 7 parts which is
#[0,25] (25,50] (50,75] (75,100] (100,125]  (125,150]
#(150,175] (175,200] (200,225]
Floyd_rain$cut <- cut(Floyd_rain$precip_sum,
                      breaks=c(0,25,50,75,100,125,150,175,200,225),
                      include.lowest = T)
#divide sum of precip into 2 parts
Allison_rain$cut <- ifelse(Allison_rain$precip_sum<175, "Unexposed", "Exposed")
```

3.mapping with package ‘ggplot’

Now, we can make maps now, use package ‘ggplot’ first.

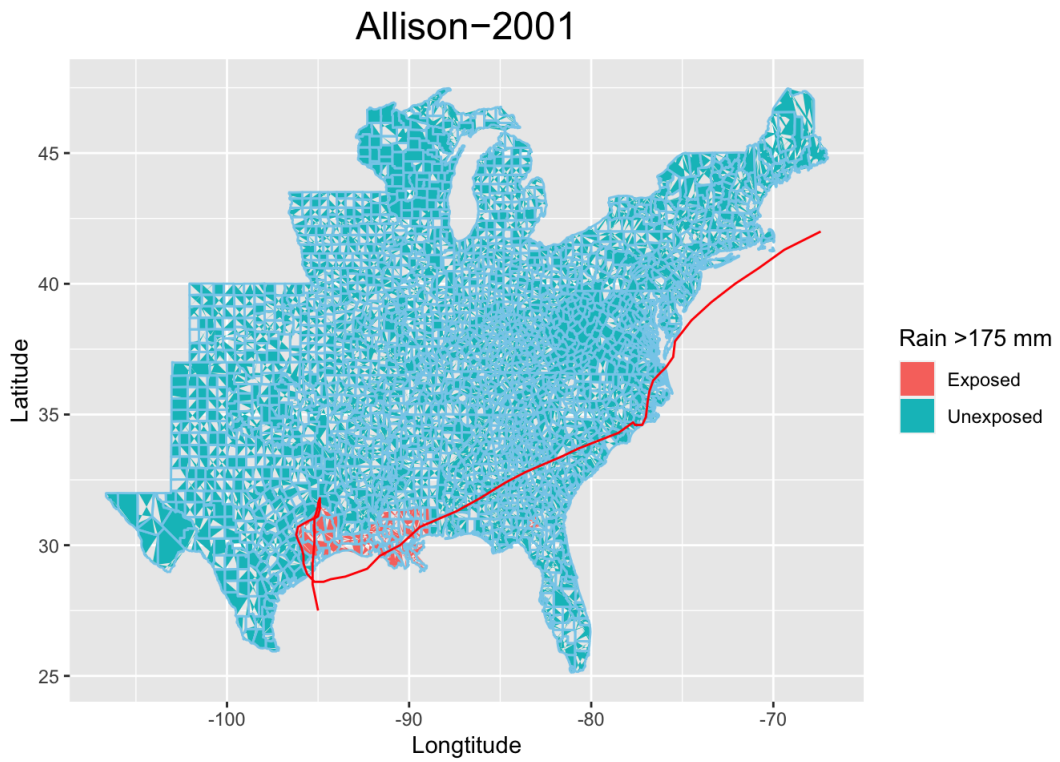
3.1 ‘Floyd-1999’

```
Floyd_rain_plot <- ggplot()+
  geom_polygon(Floyd_rain, mapping=aes(x=long, y=lat, group=group, fill=cut))+
  #connects the observations in the order in which they appear in the 'map_states'
  geom_path(map_states, mapping=aes(x=long, y=lat, group=group),color="black")+
  #draw the route
  geom_path(Floyd_route, mapping = aes(x=longitude, y=latitude),color="red")+
  #display discrete values on a map
  scale_fill_brewer(palette="Blues")+
  #change the range of x and y
  xlim(min(map_states$long),max(map_states$long))+
  ylim(min(map_states$lat),max(map_states$lat))+
  #change the name of x, y, and title
  xlab("Longtitude")+ylab("Latitude")+ggtitle("Floyd-1999")+
  #add marks
  labs(fill="Rainfall (mm)")+
  theme(plot.title = element_text(hjust = 0.5, size = 18))

Floyd_rain_plot
```



3.2 ‘Allison–2001’

```
Allison_rain_plot <- ggplot()+
  geom_polygon(Allison_rain, mapping=aes(x=long, y=lat, group=group, fill=cut))+
  #connects the observations in the order in which they appear in the 'map_states'
  geom_path(map_states, mapping=aes(x=long, y=lat, group=group),color="sky blue")+
  #draw the route
  geom_path(Allison_route, mapping = aes(x=longitude, y=latitude),color="red")+
  #change the range of x and y
  xlim(min(map_states$long),max(map_states$long))+
  ylim(min(map_states$lat),max(map_states$lat))+
  #change the name of x, y, and title
  xlab("Longtitude")+ylab("Latitude")+ggtitle("Allison-2001")+
  #add marks
  labs(fill="Rain >175 mm")+
  theme(plot.title = element_text(hjust = 0.5, size = 18))

Allison_rain_plot
```



4.mapping with package ‘tmap’

Next, we try use package ‘tmap’ for mapping.

4.1 Data preparing

First, we need to transform the data ‘map_states’ into spatial version.

```
tmap_states <- st_as_sf(map("county", states,plot = FALSE, fill = TRUE))
```

Then, we creat a new data frame which is in spatial format from ‘Floyd_rain’ and ‘Allison_rain’

```
#for Floyd-1999
t_Floyd_rain <- Floyd_rain %>%
  select(region, subregion, cut) %>%
  mutate(ID=str_c(region,subregion,sep = ",")) %>%
  select(ID,cut) %>%
  rename(`Rainfall(mm)`=cut)

#combine the data
t_Floyd_rain <- left_join(t_Floyd_rain,tmap_states, by="ID")
#change the class again, from data.frame to sf
t_Floyd_rain <- st_as_sf(t_Floyd_rain)

#for Allison-2001
t_Allison_rain <- Allison_rain %>%
  select(region, subregion, cut) %>%
  mutate(ID=str_c(region,subregion,sep = ",")) %>%
  select(ID,cut) %>%
  rename(`Rainfall > 175mm`=cut)

#combine the data
t_Allison_rain <- left_join(t_Allison_rain, tmap_states, by="ID")
#change the class again, from data.frame to sf
t_Allison_rain <- st_as_sf(t_Allison_rain)
```

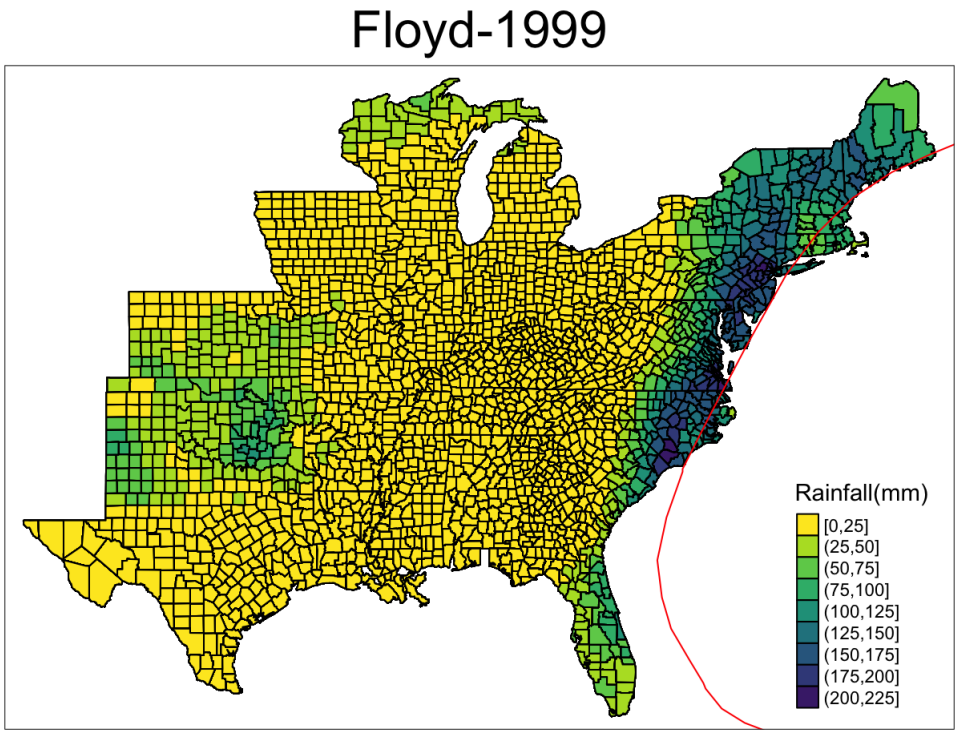
Lastly, we still need to transform data ‘Floyd’ and ‘Allison’ into spatial ones.

```
#for Floyd-1999
t_Floyd_route <- cbind(Floyd_route$longitude,Floyd_route$latitude)%>%
  Line() %>% Lines(ID='Floyd-1999') %>%
  list() %>% SpatialLines()

#for Allison-2001
t_Allison_route <- cbind(Allison_route$longitude,Allison_route$latitude)%>%
  Line() %>% Lines(ID='Allison-2001') %>%
  list() %>% SpatialLines()
```

4.2 ‘Floyd-1999’

Now, we get the spatial form of data we want, and we try to draw.

```
t_Floyd_rain_plot <- tm_shape(t_Floyd_rain)+

  tm_polygons(border.col="black",lwd=1,
              col='Rainfall(mm)',
#each unique value will correspond to a distinct category when using 'cat'
              style="cat",
              title="Rainfall(mm)",
#use package 'viridis' to maintain gradual change visual effect
              palette=viridis(n=10,direction=-1))+
  #add the route
  tm_shape(t_Floyd_route)+
  tm_lines(col = "red")+
  tm_layout(main.title='Floyd-1999',main.title.position="center",main.title.size = 2)
t_Floyd_rain_plot
```



4.3 Allison-2001

```
t_Allison_rain_plot <- tm_shape(t_Allison_rain)+
  tm_polygons(border.col="black",lwd=1,
              col='Rainfall > 175mm',
#each unique value will correspond to a distinct category when using 'cat'
              style="cat",
              title="Rainfall > 175mm",
#use package 'viridis' to maintain gradual change visual effect
              palette=magma(n=7,direction=-1))+
  #add the route
  tm_shape(t_Allison_route)+
  tm_lines(col = "red")+
  tm_layout(main.title='Allison-2001',main.title.position="center",main.title.size = 2)
t_Allison_rain_plot
```