A-3.R

pradyuth

Tue Feb 20 22:40:38 2018

```
#Name : Pradyuth Vangur
#Assignment 3

#Problem 1: Dataframes and ggplot2

#Question a
#install package maps and ggplot2

#Installing and loading the package maps and ggplot2
library(maps)
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

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

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

```
#Question b
#Creating dataframe
da_fr <- map_data(map = "state")

#Dimensions of dataframe
dim(da_fr)
```

```
## [1] 15537     6
```

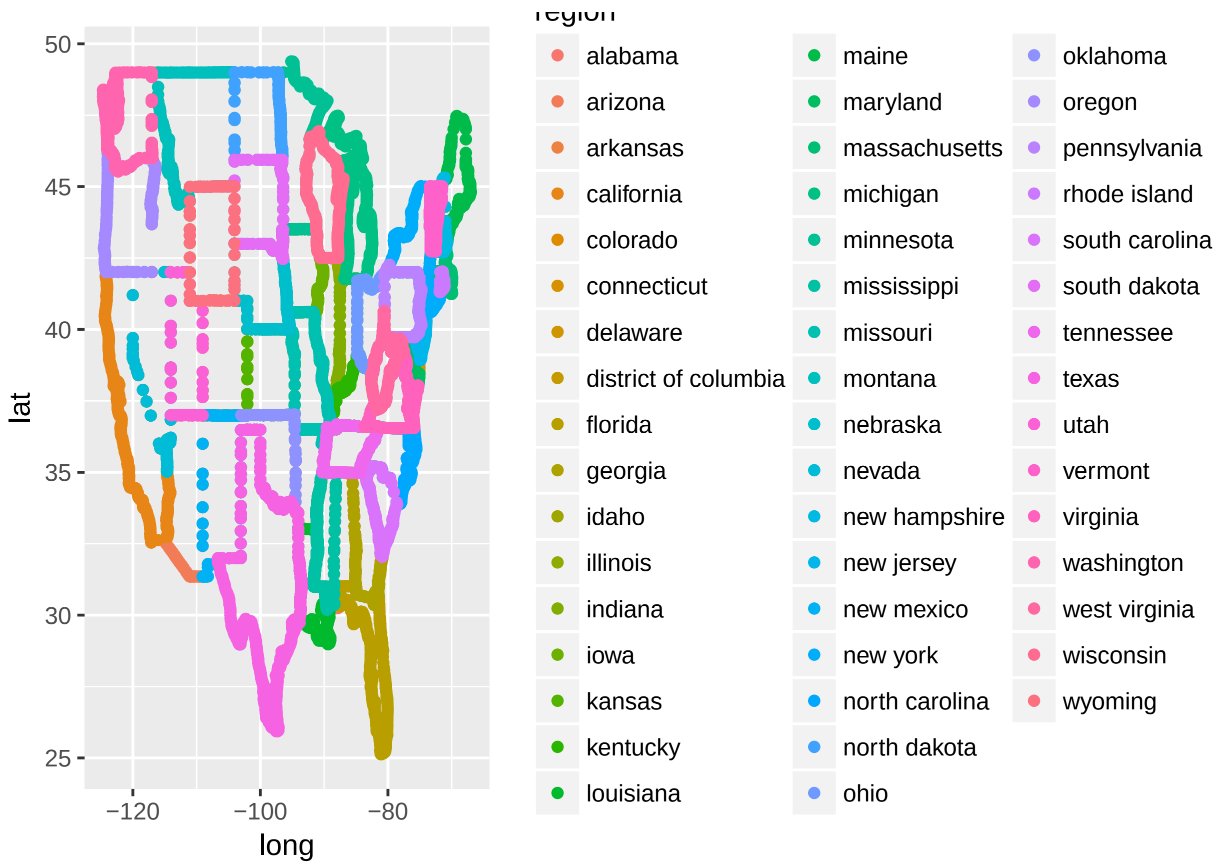
```
#Column names
names(da_fr)
```

```
## [1] "long"      "lat"       "group"     "order"     "region"    "subregion"
```

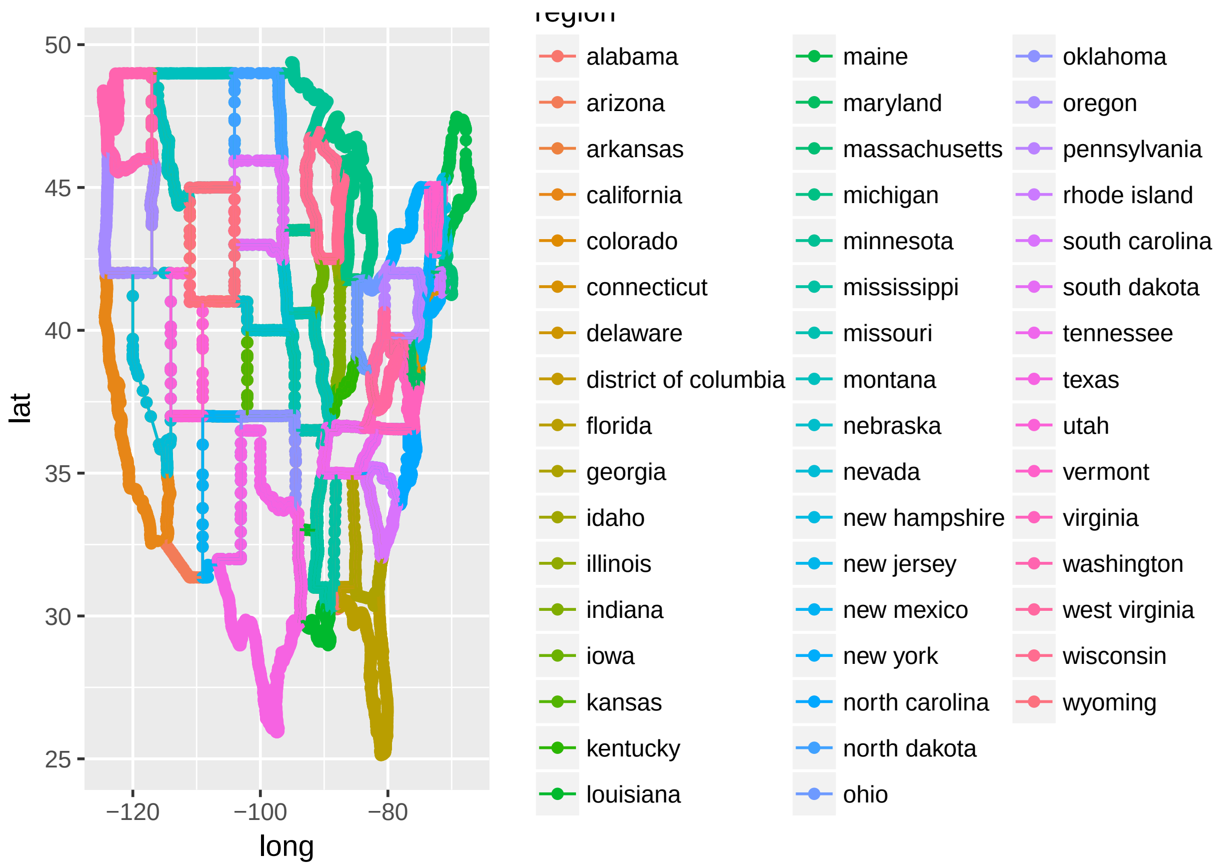
```
#Question c
#Printing the number of unique values
unique(da_fr$region)
```

```
##  [1] "alabama"              "arizona"              "arkansas"
##  [4] "california"           "colorado"             "connecticut"
##  [7] "delaware"             "district of columbia" "florida"
## [10] "georgia"              "idaho"                "illinois"
## [13] "indiana"              "iowa"                 "kansas"
## [16] "kentucky"             "louisiana"            "maine"
## [19] "maryland"             "massachusetts"        "michigan"
## [22] "minnesota"            "mississippi"          "missouri"
## [25] "montana"              "nebraska"             "nevada"
## [28] "new hampshire"        "new jersey"           "new mexico"
## [31] "new york"             "north carolina"       "north dakota"
## [34] "ohio"                 "oklahoma"             "oregon"
## [37] "pennsylvania"         "rhode island"         "south carolina"
## [40] "south dakota"         "tennessee"            "texas"
## [43] "utah"                 "vermont"              "virginia"
## [46] "washington"           "west virginia"        "wisconsin"
## [49] "wyoming"
```

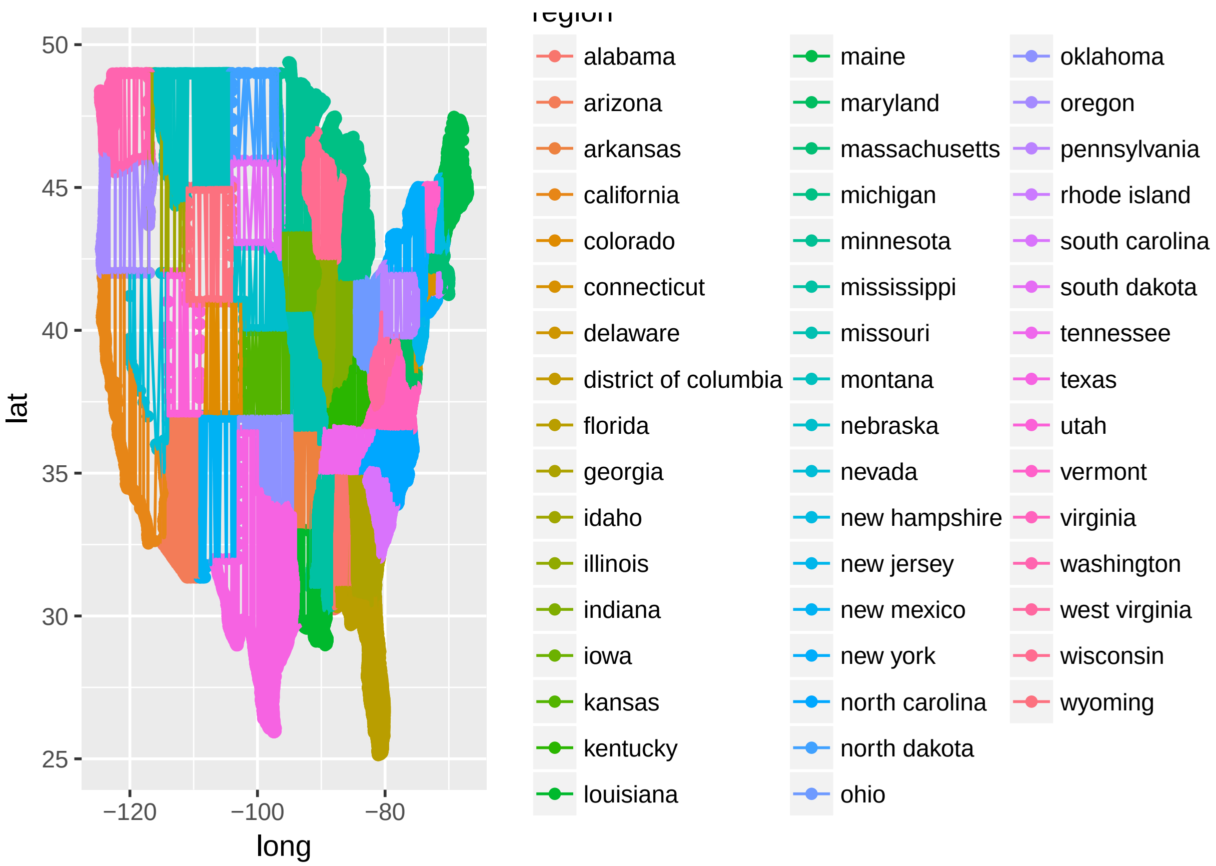
```
#Question d
ggplot(da_fr, aes(x = long, y = lat, col = region)) +
  geom_point()
```



```
#Question e
ggplot(da_fr, aes(x = long, y = lat,group = group, col = region)) +
  geom_point() +
  geom_path()
```



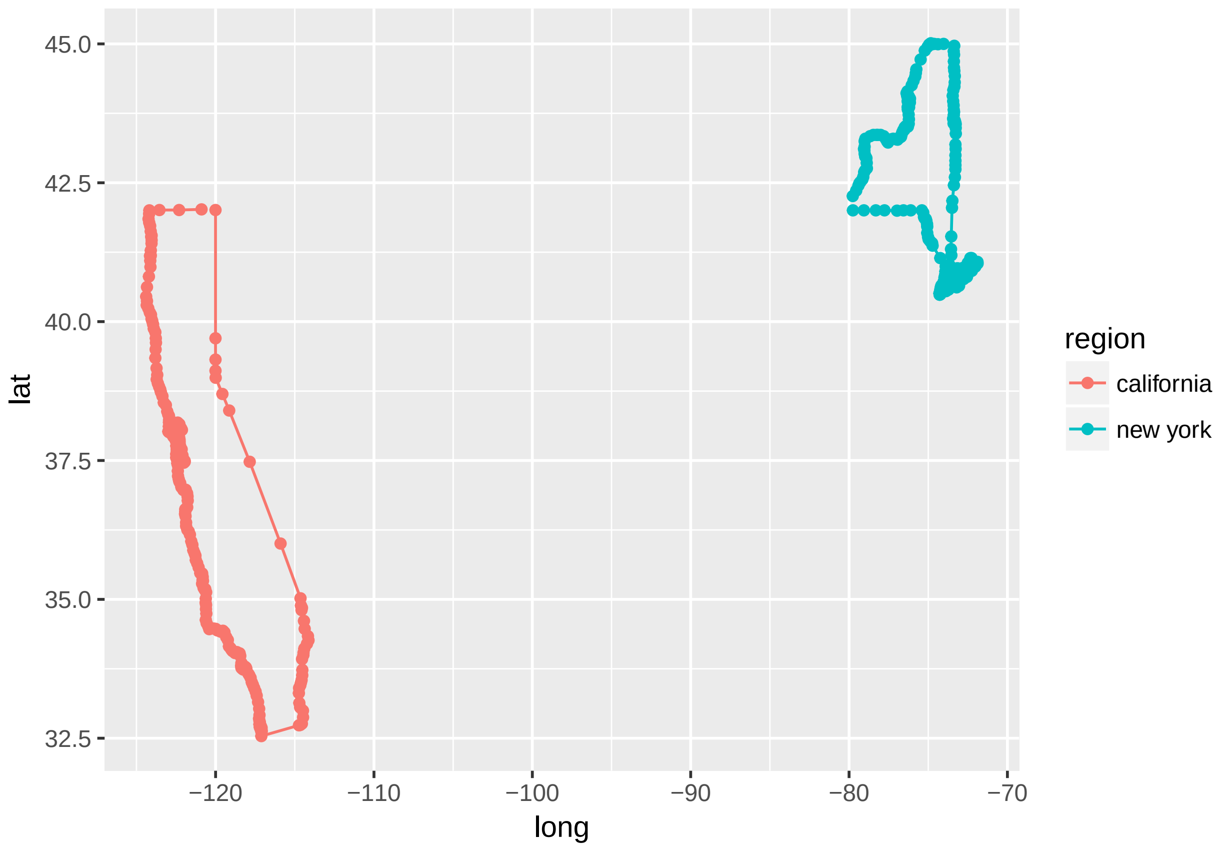
```
#We observe that the path is being laud out in the order of points and gives a
#picture of the map.

ggplot(da_fr, aes(x = long, y = lat,group = group, col = region)) +
  geom_point() +
  geom_line()
```



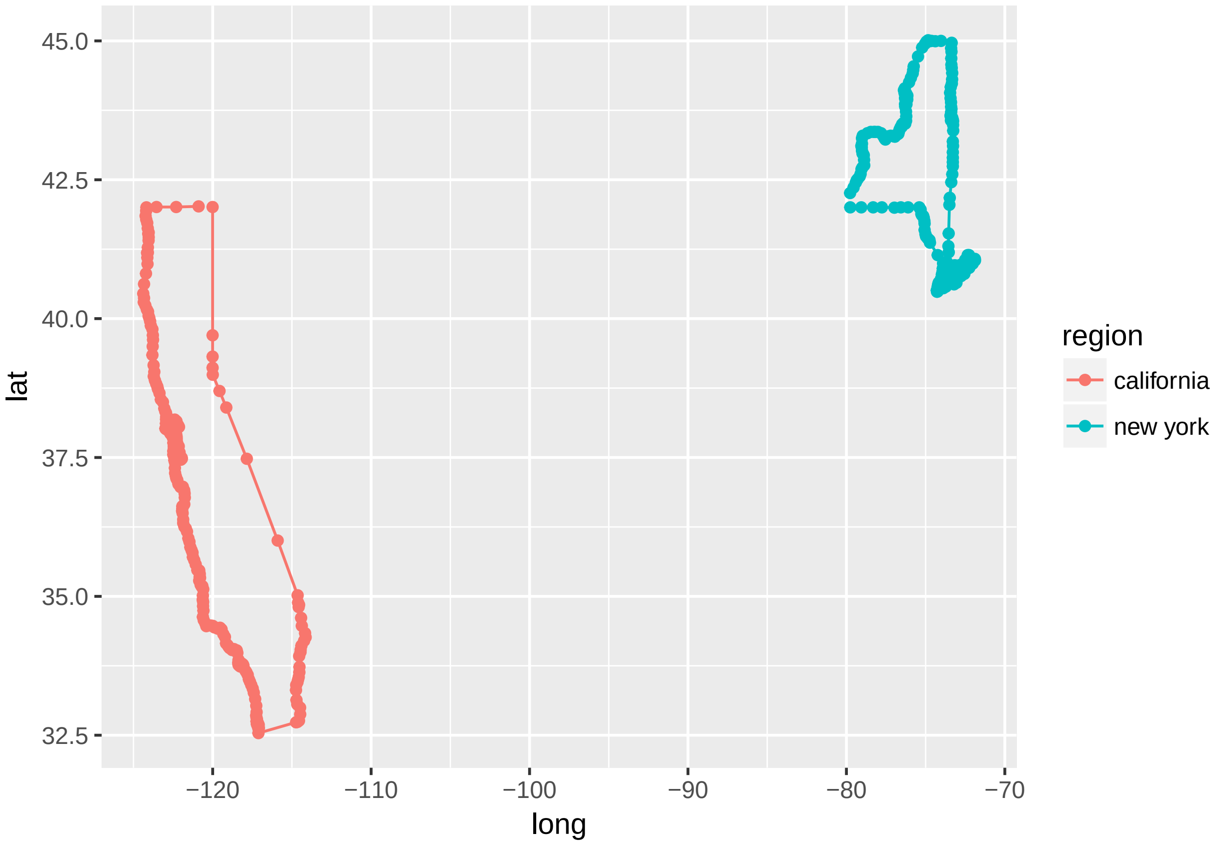
```
#Here, the lines try to connect each points which belong to the same region

#Question f
#Selecting rows of new york and california
da_dr_st <- which(da_fr$region == c("new york" , "california"))
```

```
## Warning in da_fr$region == c("new york", "california"): longer object
## length is not a multiple of shorter object length
```

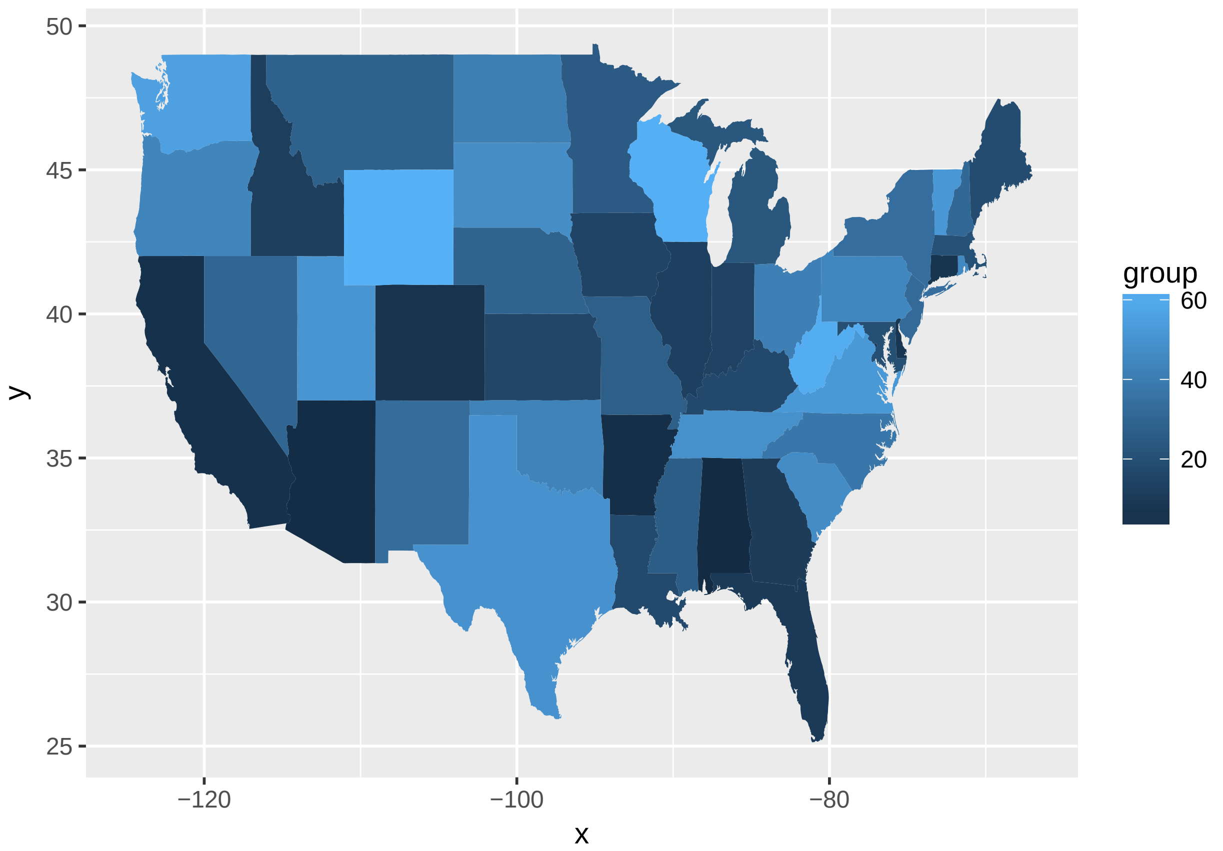
```
#making a dataframe of only those countries
da_fr_st <- da_fr[da_dr_st,]

#Creating plot for two states
ggplot(da_fr_st, aes(x = long, y = lat,group = group, col = region)) +
  geom_point() +
  geom_path()
```



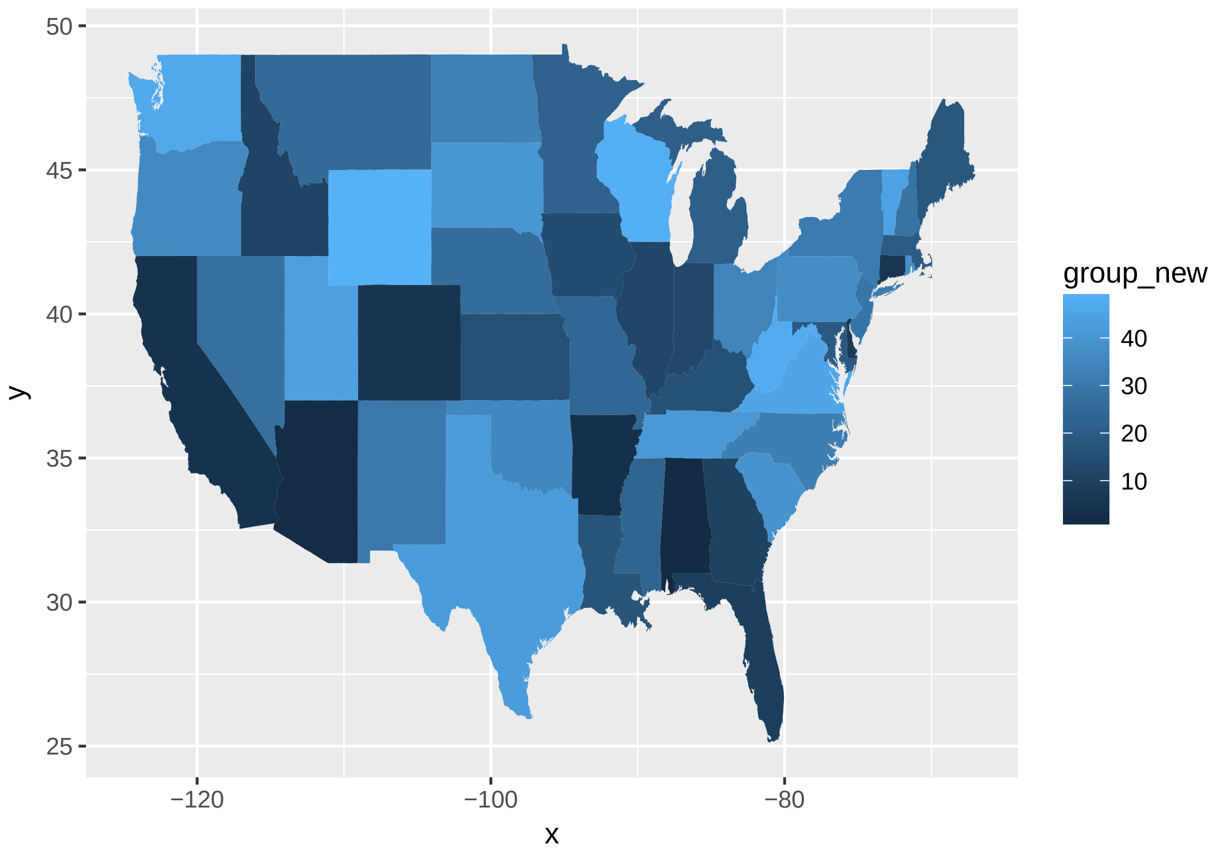
```
#Question g
ggplot(sample(da_fr_st), aes(x = long, y = lat,group = group, col = region)) +
  geom_point() +
  geom_path()
```



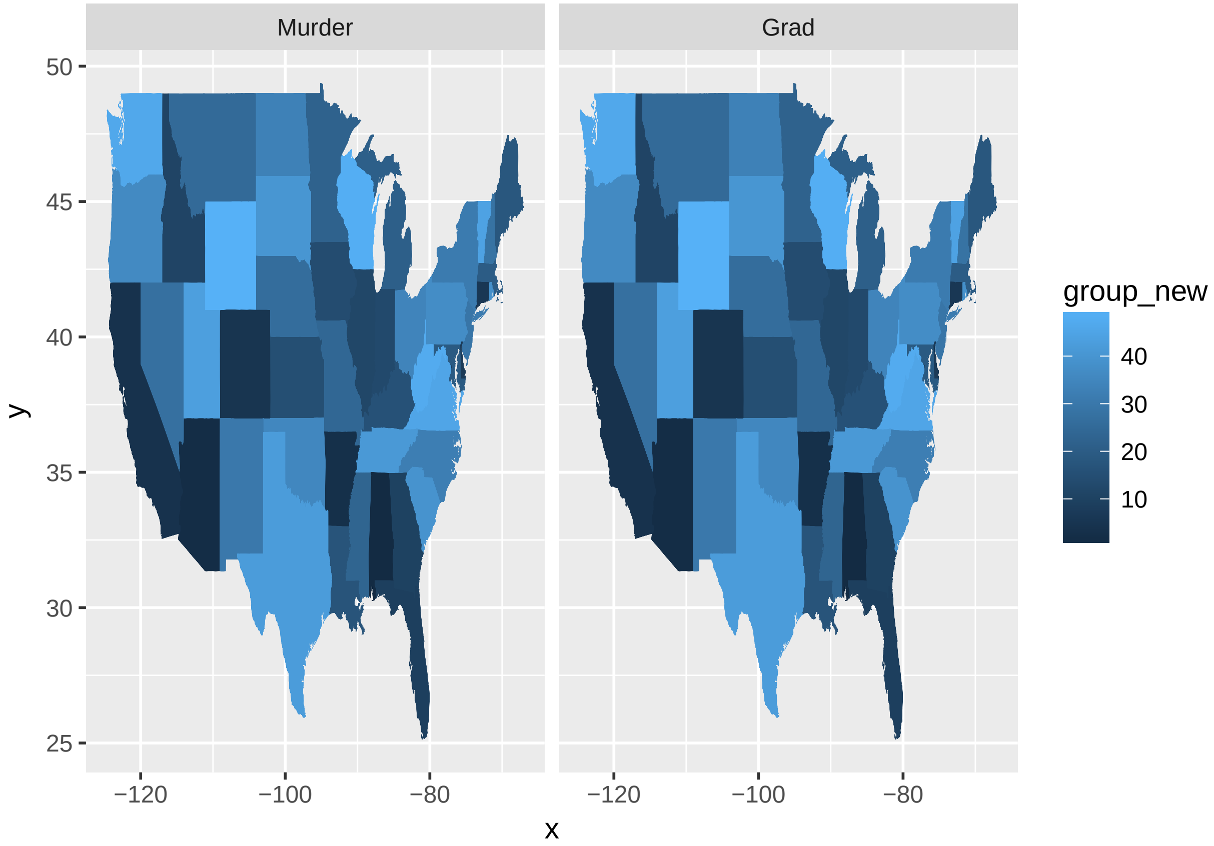
```
#Question h
#states_map is da_fr in this case
ggplot() + geom_map(map = da_fr, map_id=da_fr$region, data = da_fr, aes(fill=group)) +
  expand_limits(x = da_fr$long, y = da_fr$lat)
```



```
#Question i
#Assigning correct values to states
states <- unique(da_fr$region)
group_new <- c(rep(0, nrow(da_fr)))


for( i in 1:nrow(da_fr) ){
  m <- which(da_fr$region[i] == states)
  group_new[i] <- m
}

da_fr_new <- cbind(da_fr, group_new)

ggplot() + geom_map(map = da_fr_new, map_id=da_fr_new$region, data = da_fr_new, aes(fill=group_new)) +
  expand_limits(x = da_fr_new$long, y = da_fr_new$lat)
```

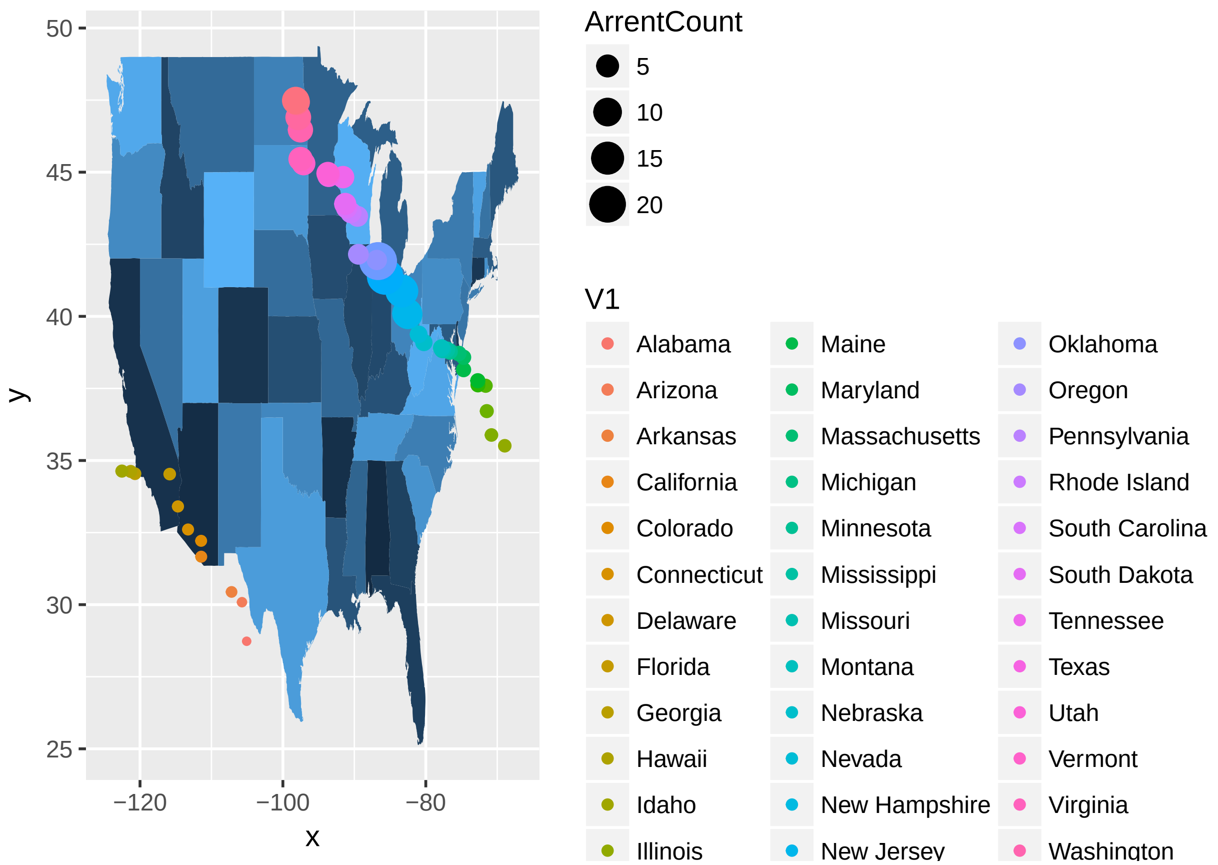


```
#Question j
#Assigning each row it's murder count
data(state)
InfoValue <- c(rep(0, nrow(state.x77)))
InfoValue <- (state.x77[,1] * state.x77[,5]) / 10000
InfoValue_round <- round(InfoValue)

#Adding it onto a new data frame
state_new <- cbind(state.name, state.x77, InfoValue, InfoValue_round)
state_new_df <- as.data.frame(state_new, row.names = F)

#Question k
InfoType_1 <- rep("Murder", nrow(da_fr))
InfoType_2 <- rep("Grad", nrow(da_fr))
InfoType_n <- c(InfoType_1, InfoType_2)
InfoValue_n <- rep(0, nrow(da_fr))
new_value <- rep(0, nrow(da_fr))

state_new_df_2 <- state_new_df[-2,]
state_name_new <- state.name[-2]

#Creating a function to map values into
value <- function(data, col, states_map){
  new_column <- rep(0, nrow(data))
  for(i in 1:nrow(data)){
    new_column[i] <- as.numeric(levels(droplevels(col[i][1])))
  }

  for(i in 1:nrow(states_map)){
    l <- as.numeric(which(states_map$region[i] == tolower(state_name_new)))
    if(sum((states_map$region[i] == tolower(state_name_new))) == 0) {next}
    new_value[i] = new_column[l]
  }
  return(new_value)
}

Murder_values <- value(data = state_new_df_2, col = state_new_df_2$Murder , states_map = da_fr)
Illiteracy_values <- value(data = state_new_df_2, col = state_new_df_2$Illiteracy , states_map = da_fr)

da_fr_M <- cbind(da_fr, InfoType_1, Murder_values, group_new)
names(da_fr_M)[7] <- c("InfoType")
names(da_fr_M)[8] <- c("InfoValue")

da_fr_I <- cbind(da_fr, InfoType_2, Illiteracy_values, group_new)
names(da_fr_I)[7] <- c("InfoType")
names(da_fr_I)[8] <- c("InfoValue")

#Final dataframe
da_fr_MI <- rbind(da_fr_M, da_fr_I)

#Question l
ggplot() + geom_map(map = da_fr_MI, data = da_fr_MI, map_id=da_fr_MI$region, aes(fill=group_new)) +
  expand_limits(data = da_fr, x = da_fr$long, y = da_fr$lat) +
  facet_grid(. ~ InfoType)
```



```
#Question m
states <- unique(da_fr$region)
group_new_1 <- c(rep(0, nrow(da_fr)))
states_df <- as.data.frame(state.x77)
states_df_new <- states_df[-2,]

#Displaying a loop to map the calumns into another
for( i in 1:nrow(da_fr) ){
  m <- which(da_fr$region[i] == states)
  if(sum(da_fr$region[i] == states) == 0){next}
  group_new_1[i] <- states_df_new$Illiteracy[m]
}

da_fr_new_il <- cbind(da_fr_new, group_new_1)


vec <- vector()
MeanLat <- rep(0, length(states))
MeanLong <- rep(0, length(states))

#Finding the mean values of lat and long of each region
for(i in 1:length(states)){
  vec <- which(da_fr_new_il$region == states[i])
  MeanLong[i] <- mean(da_fr_new_il$long[vec])
  MeanLat[i] <- mean(da_fr_new_il$lat[vec])
}
```

```
data_4 <- as.data.frame(cbind(state.name[-2],MeanLat, MeanLong, ArrentCount = as.numeric(levels(droplev

data_4$MeanLat <- as.numeric(levels(droplevels(data_4$MeanLat)))
data_4$MeanLong <- as.numeric(levels(droplevels(data_4$MeanLong)))
data_4$ArrentCount <- as.numeric(levels(droplevels(data_4$ArrentCount)))


q <- ggplot() + geom_map(map = da_fr_new, map_id=da_fr_new$region, data = da_fr_new, aes(fill=group_new
  expand_limits(x = da_fr_new$long, y = da_fr_new$lat)

q + geom_point(data = data_4, aes(x = MeanLong, y = MeanLat, col = V1, size = ArrentCount))
```



```
#Question n
#Function for calculating the mean of latitude and longitude
fun <- function(X){
  vec <- which(X == states)
  MeanLong <- mean(da_fr$long[vec])
  MeanLat <- mean(da_fr$lat[vec])
  return(c(MeanLong, MeanLat))
}

states_n <- states[-2]

#Function to calculate the values for the whole dataset
head(sapply(states, fun))
```

```
##        alabama   arizona  arkansas california  colorado connecticut
## [1,] -87.46201 -87.48493 -87.52503  -87.53076 -87.57087   -87.58806
## [2,]  30.38968  30.37249  30.37249   30.33239  30.32665    30.32665
##       delaware district of columbia   florida   georgia     idaho
## [1,] -87.59379            -87.59379 -87.67400 -87.81152 -87.88026
## [2,]  30.30947             30.28655  30.27509  30.25790  30.24644
##       illinois   indiana      iowa    kansas  kentucky louisiana     maine
## [1,] -87.92037 -87.95475 -88.00632 -88.01778 -88.01205 -87.99486 -87.95475
## [2,]  30.24644  30.24644  30.24071  30.25217  30.26936  30.27509  30.27509
##       maryland massachusetts  michigan minnesota mississippi  missouri
## [1,] -87.90318     -87.82870 -87.80006 -87.80006   -87.81724 -87.84016
## [2,]  30.28082      30.28655  30.28655  30.32665    30.34385  30.38395
##        montana  nebraska    nevada new hampshire new jersey new mexico
## [1,] -87.85162 -87.87453 -87.90318     -87.92610  -87.93183  -87.94329
## [2,]  30.40114  30.41260  30.42406      30.44698   30.49281   30.52719
##       new york north carolina north dakota      ohio  oklahoma    oregon
## [1,] -87.92037      -87.91464    -87.92610 -87.92037 -87.94902 -87.98913
## [2,]  30.56157       30.58449     30.61886  30.67043  30.69908  30.79075
##      pennsylvania rhode island south carolina south dakota tennessee
## [1,]    -88.00632    -88.01778      -88.03497    -88.04642 -88.05215
## [2,]     30.79648     30.80221       30.79075     30.75638  30.72773
##          texas      utah   vermont  virginia washington west virginia
## [1,] -88.05215 -88.06361 -88.06934 -88.08080  -88.08080     -88.09799
## [2,]  30.71054  30.68762  30.68189  30.63033   30.61314      30.60741
##      wisconsin   wyoming
## [1,] -88.10944 -88.11518
## [2,]  30.59595  30.58449
```