Travelling_sales_person

Pankaj Shah

6/15/2019

Library

```
library(tidyverse)  # dplyr, ggplot
```

```
## Warning: package 'tibble' was built under R version 3.5.2
```

```
## Warning: package 'stringr' was built under R version 3.5.2
```

```
library(ggmap)      # for geom_leg() call
library(RCurl)      # for reading csv from github
```

```
## Warning: package 'RCurl' was built under R version 3.5.2
```

```
library(dbscan)     # clustering
library(geosphere)  # calculates distances
library(TSP)        # solves travelling salesperson problem
library(gridExtra)  # combines ggplot charts
```

Read Medical School Data:

```
Locations <-
    getURL("https://raw.githubusercontent.com/shahnp/data/master/Medschools.txt") %>%
    read.csv(text=., stringsAsFactors = F) %>%  # the dot means "this data" in the pipe
    as.data.frame()

#identify the following variables
    Locations$ID    <- Locations$UnitID # the unique identifier for each site
    Locations$lon   <- Locations$lon        # Y coord
    Locations$lat   <- Locations$lat        # X coord
    Locations$value <- Locations$Enrollment  # size of points, weight for clustering

#What does this data describe
data_title <- "Medical Schools by Enrollment (2015)"

#Want to use groups?
    #If yes, write "TRUE" and assign a group.
    #If no, write "FALSE" and then select clustering method

  use_group <- TRUE
  Locations$group <- Locations$Region #column to group by geographically

#clsutering method
  clustMethod <- c("ward.D", "ward.D2", "single", "complete",
                   "average", "mcquitty", "median", "centroid")
  which_clustMethod <- 1 #defaults to 1: hclust(method = "ward.D")
```

```
head(Locations)
```

```
##   UnitID                                Institution.Name        lon      lat
## 1 100663         University of Alabama at Birmingham  -86.80917 33.50223
## 2 100858                               Auburn University  -85.48278 32.60469
## 3 102094                  University of South Alabama  -88.18189 30.69508
## 4 102377                          Tuskegee University  -85.71031 32.43102
## 5 104179                        University of Arizona -110.95077 32.23207
## 6 106263 University of Arkansas for Medical Sciences  -92.32094 34.74972
##   Enrollment    Region HBCU Hospital                Control     ID value
## 1      12334 Southeast   No      Yes                 Public 100663 12334
## 2      22035 Southeast   No       No                 Public 100858 22035
## 3      12922 Southeast   No      Yes                 Public 102094 12922
## 4       2962 Southeast  Yes       No Private not-for-profit 102377  2962
## 5      36580 Southwest   No       No                 Public 104179 36580
## 6       2069 Southeast   No      Yes                 Public 106263  2069
##       group
## 1 Southeast
## 2 Southeast
## 3 Southeast
## 4 Southeast
## 5 Southwest
## 6 Southeast
```

What if user_group == F

```
if(use_group == F){

    clusters <- hclust(dist(select(Locations, lat, lon)),
                       method = clustMethod[which_clustMethod])

    #summary(clusters$height)
    Locations$group <- paste0("cluster ",
                              cutree(clusters, h = mean(clusters$height)))

    cluster_bar <-
        Locations %>%
        group_by(group) %>%
        summarise(nPoints = n()) %>%
        arrange(-nPoints) %>%
        ungroup() %>%
        mutate(Ord = row_number())

    ggplot(cluster_bar) +
      geom_boxplot(aes(y=nPoints, x = 1)) +
      coord_flip() +
      ggtitle(paste0("Distribution of points among the ",
                     nrow(cluster_bar), " clusters")) +
      theme(axis.text.y = element_blank(),
            axis.title.y = element_blank(),
            aspect.ratio = .25)
```

```
} else {
    Locations$group <- Locations$group
}
```

Airport data

```
Airports <- getURL("https://raw.githubusercontent.com/shahnp/data/master/airports.txt") %>%
                  read.csv(text=., stringsAsFactors = F) %>%
                    as.data.frame()
```

```
head(Airports)
```

```
##    ID      lat        lon
## 1 BHM 33.56389  -86.75231
## 2 HSV 34.63719  -86.77506
## 3 LIT 34.72944  -92.22478
## 4 PHX 33.43428 -112.01158
## 5 BUR 34.20067 -118.35867
## 6 FAT 36.77619 -119.71814
```

Create centers by finding the centroids of each group, then find the closest site to the centroid, then calculate distances to that central location

```
Regions <-
    Locations %>%
    select(group) %>%
    distinct() %>%
    arrange(group)

Centroid <-
    Locations %>%
    select(ID, group, lon, lat) %>%
    group_by(group) %>%
        mutate(lon.C = median(lon),
               lat.C = median(lat)) %>%
        ungroup() %>%

    #get distances to centroid of region
    rowwise() %>%
        mutate(dist.mi = distGeo(p1 = c(lon, lat),
                                 p2 = c(lon.C, lat.C))/1609.34) %>%
        ungroup() %>%
    #identify central facility
    group_by(group) %>%
        arrange(dist.mi) %>%
        mutate(Ord = row_number()) %>%
        ungroup() %>%
    mutate(Central = ifelse(Ord == 1, "Central", "Other")) %>%
    #assign central facility as navigation point (lon.C, lat.C)
    group_by(group) %>%
        arrange(Ord) %>%
        mutate(lon.C = ifelse(Ord == 1, lon, 0),
```

```
               lat.C = ifelse(Ord == 1, lat, 0)) %>%
        mutate(lon.C = min(lon.C),
               lat.C = max(lat.C)) %>%
        ungroup() %>%
    #get distances for each site to central facility
    rowwise() %>%
        mutate(dist.mi = distGeo(p1 = c(lon, lat),
                                 p2 = c(lon.C, lat.C))/1609.34) %>%
                                 #convert from meters to miles
        ungroup()
```

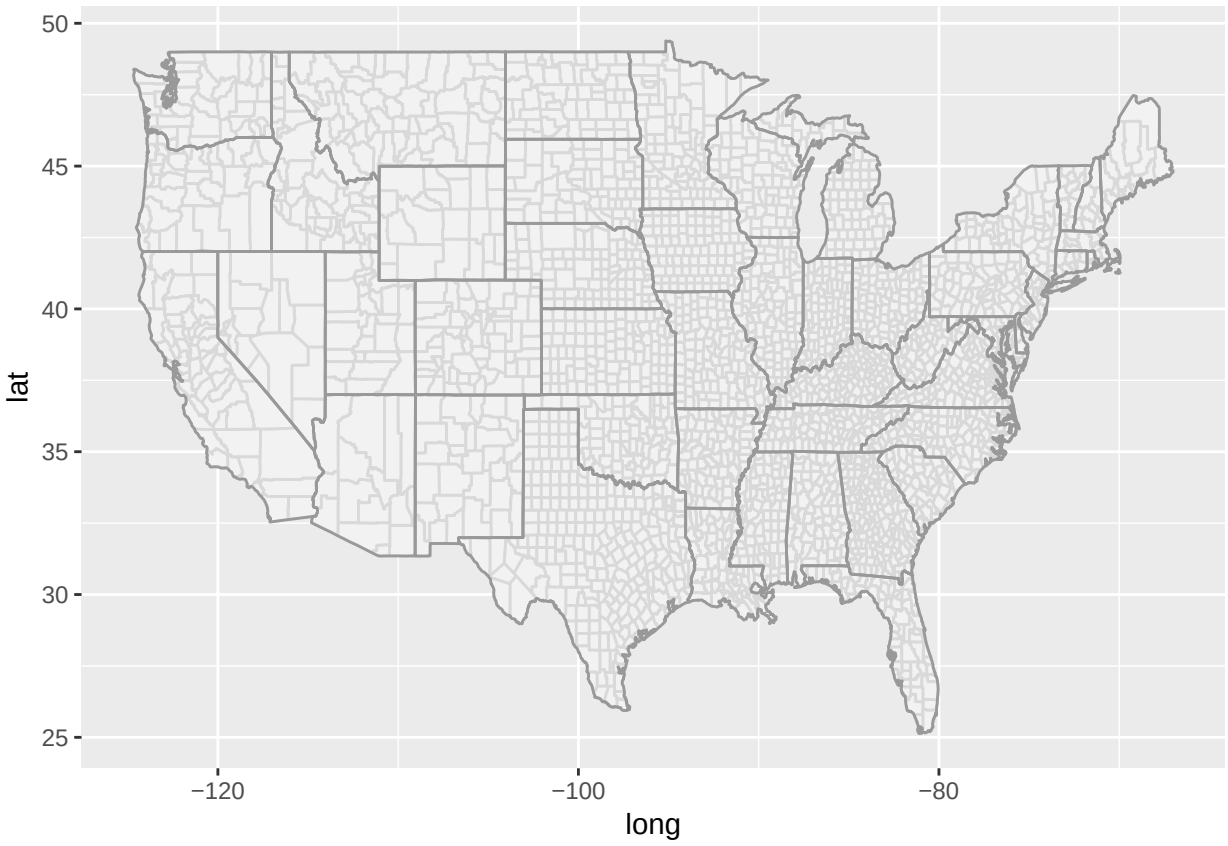
Generate map elements

```
#map elements that repeat across maps

map_airports <- geom_point(data = Airports, aes(x = lon, y = lat),
                           size = 3, alpha = .5, color = "black",
                           fill = "yellow", shape = 24)

map_legend <- theme(legend.position = "bottom",
                    axis.text = element_blank(),
                    axis.ticks = element_blank(),
                    axis.title = element_blank(),
                    panel.background = element_rect(fill = "white"))

all_states <- map_data("state")
```

```
##
## Attaching package: 'maps'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

```
all_counties <- map_data("county")

basemap <-
      ggplot() +
      geom_polygon( data=all_counties, aes(x=long, y=lat, group = group),
                    colour="grey85", fill="GREY95") +
      geom_polygon( data=all_states, aes(x=long, y=lat, group = group),
                    colour="grey60", fill = NA, size = .5)
basemap
```



Generate polygon outlines

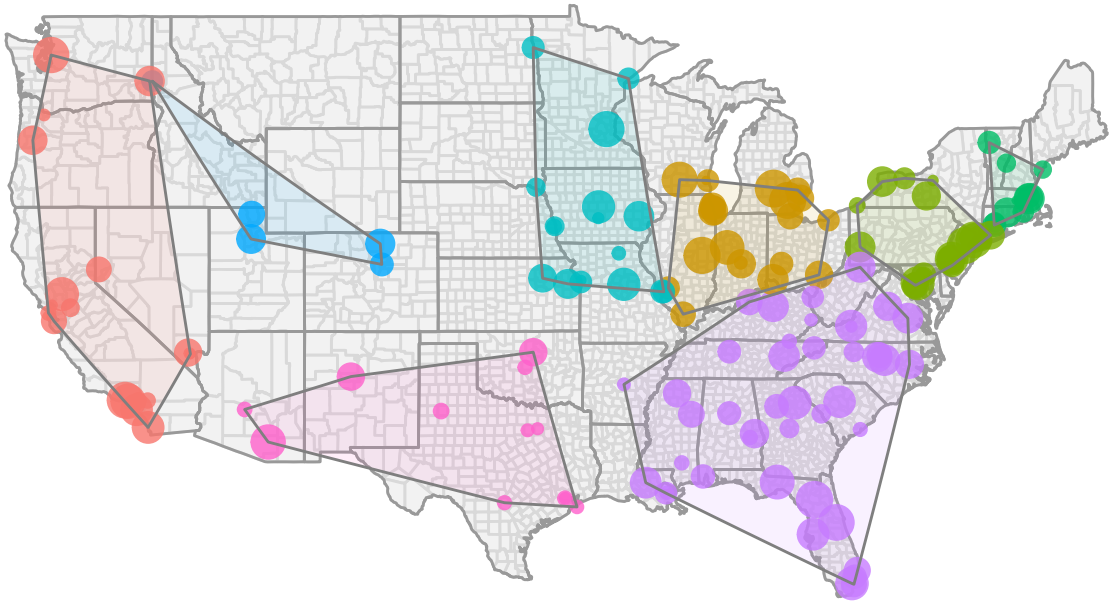
```
group_outline <- data.frame(lon = vector("numeric", 0),
                            lat = vector("numeric", 0),
                            group = vector(class(Regions$group), length = 0))

for(i in 1:nrow(Regions)){
  group_points <-  Locations %>%
                      filter(group == Regions$group[i]) %>%
                      select(lon, lat)
  select_hull <- chull(group_points)
  select_hull <- c(select_hull, select_hull[1])

  group_outline <- rbind(group_outline,
                         group_points[select_hull, ] %>%
                          mutate(group = Regions$group[i]))

}
```

Show basemap

```
# Make map

basemap +
geom_point(data = Locations,
           aes(lon, lat, color = factor(group), size = value), alpha = .8)+
geom_polygon(data = group_outline,
             aes(lon, lat, group = group,
```

```
                  fill = factor(group)), color = "grey50", alpha = .1) +
coord_map() +
theme(legend.position = "none",
      axis.text = element_blank(),
      axis.ticks = element_blank(),
      axis.title = element_blank(),
      panel.background = element_rect(fill = "white")) +
ggtitle(paste0(data_title, ": ",
               nrow(Regions), " groups, n=", sum(Locations$value), " (uses ",
               ifelse(use_group == T, "original groups", "clustering algorithm"), ")"))
```

Medical Schools by Enrollment (2015): 8 groups, n=2516476 (uses original grou



Create maps

Create three maps within each group:

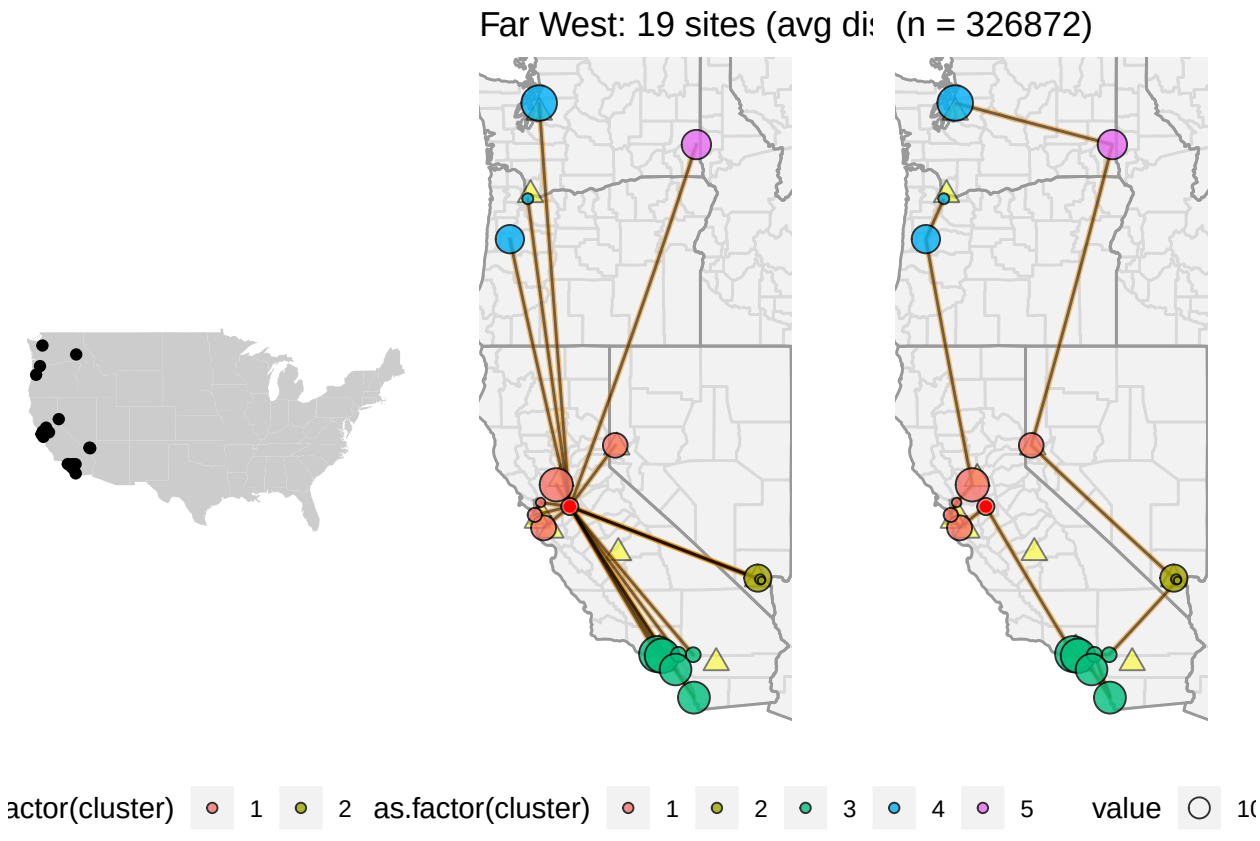
- map showing region
- straight lines to central location
- shortest path through all points

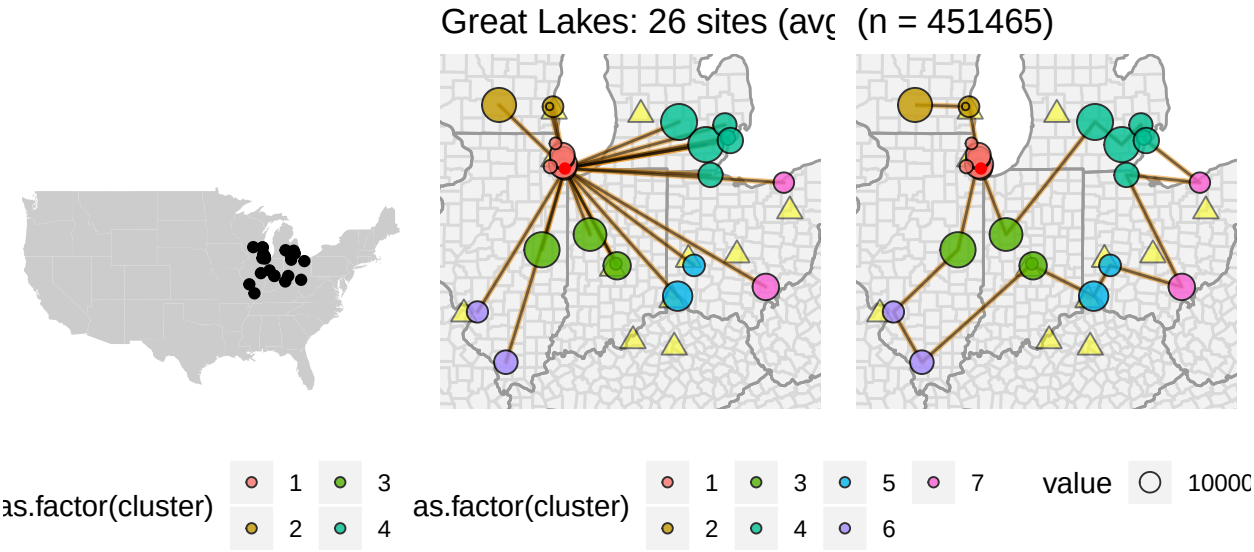
Points are clustered within each region as a guide for the decision making process.

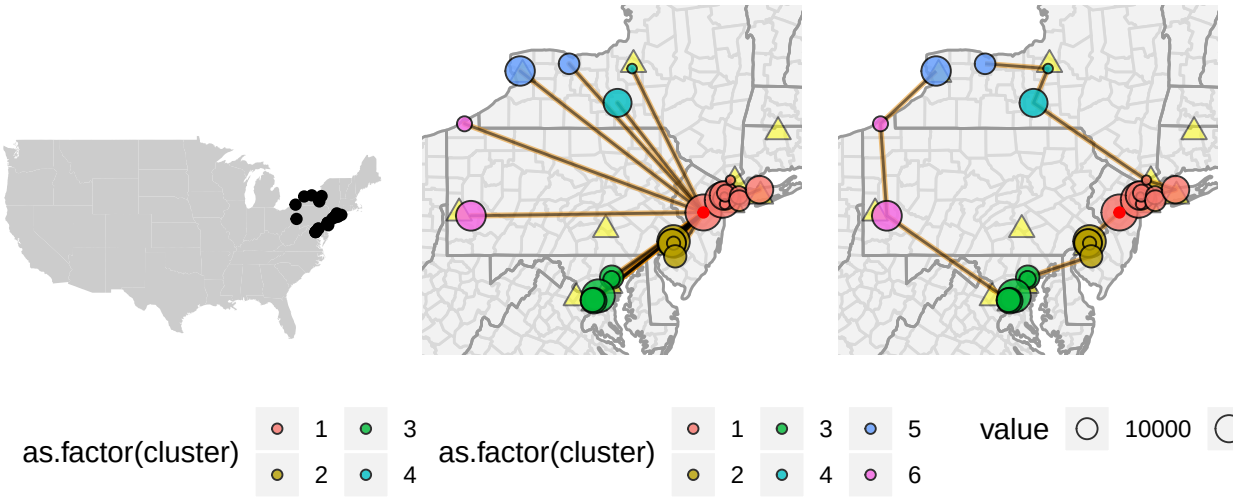
```
#for loop to generate maps
for(i in 1:nrow(Regions)){

    RegionLoop <-
        filter(Centroid, group == Regions$group[i]) %>%
        left_join(select(Locations, ID, value), by = "ID")

    #clusters
      clustersRL <- hclust(dist(select(RegionLoop, lat, lon)),
                           method = clustMethod[which_clustMethod])
```

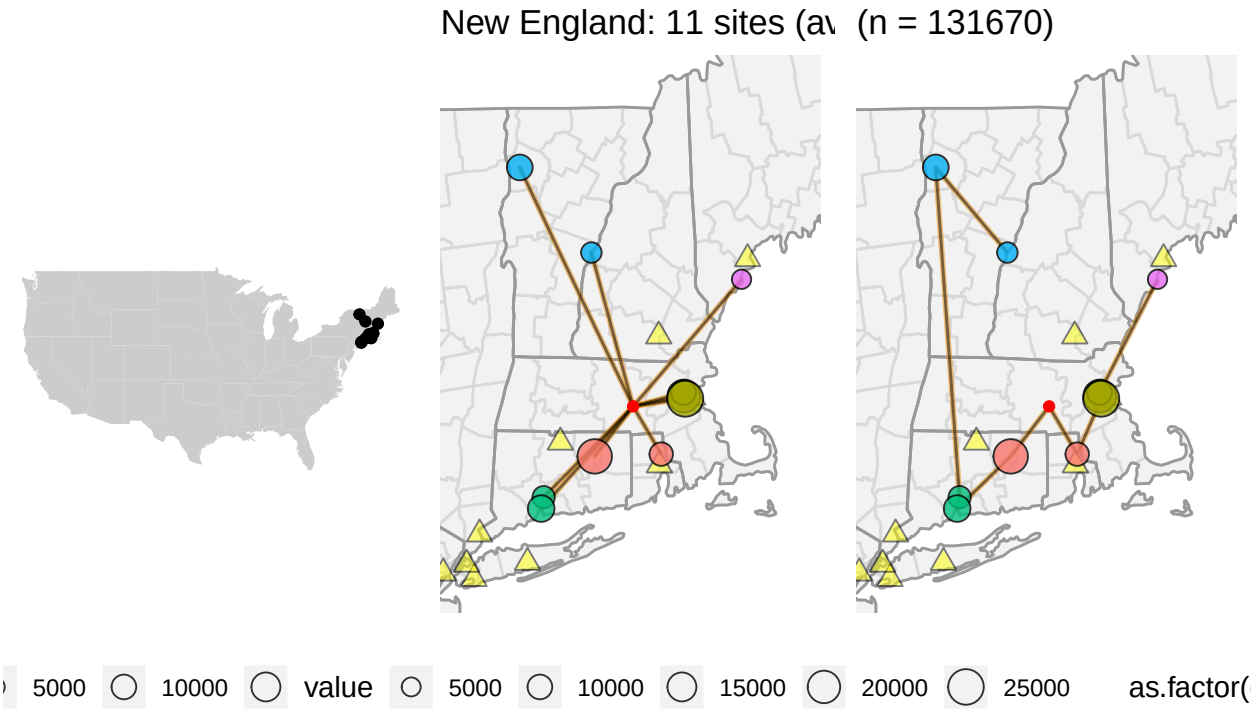
```
  RegionLoop$cluster <- cutree(clustersRL, h = mean(clustersRL$height))

  #table(RegionLoop$cluster)

#creates shortest routes
  df <- select(RegionLoop, lon,lat)
  tsp <- TSP(dist(df))
  tour <- solve_TSP(tsp, method="nearest_insertion", control=list(rep=10))
  path.tsp <- unname(tour)
  RegionLoop$Path <- as.integer(tour)

#plot(lat~lon, df[path.tsp,], type="b",asp=1)

#map elements that repeat across maps
  map_locations <- geom_point(data = RegionLoop,
                              aes(x = lon, y= lat, size = value,
                                  fill = as.factor(cluster)),
                              alpha = .8, color = "black", shape = 21)

  map_location_central <-
    geom_point(data = filter(RegionLoop, Central == "Central"),
               aes(lon, lat),
               color = "red")

  map_projection <-
    coord_map(projection="mercator",
              xlim=c(min(RegionLoop$lon)-1, max(RegionLoop$lon)+1),
              ylim=c(min(RegionLoop$lat)-1, max(RegionLoop$lat)+1))

#show location of group
baseplot <-
  ggplot() + coord_map() +
  geom_polygon(data=all_states, aes(x=long, y=lat, group = group),
               color = NA, fill = "grey80", size = .5) +
   geom_point(data = RegionLoop, aes(x = lon, y= lat)) +
  #map_location_central +
  theme_nothing()

#plots 1 & 2
plot1 <- #map of distance to most central location
   basemap + map_projection + map_airports + #ggplot() +
   geom_leg(data = RegionLoop,
           aes(x = lon, xend = lon.C,
               y = lat, yend = lat.C),
            color = "#E58700", size = 1, alpha = .5) +
    geom_leg(data = RegionLoop,
           aes(x = lon, xend = lon.C,
               y = lat, yend = lat.C),
            color = "black", size = .5, alpha = .5) +
    map_locations + map_location_central + map_legend +
    ggtitle(paste0(RegionLoop$group[1],
                   ": ", nrow(RegionLoop), " sites (avg dist ",
                   round(median(RegionLoop$dist.mi), 0), " mi.)"))
```

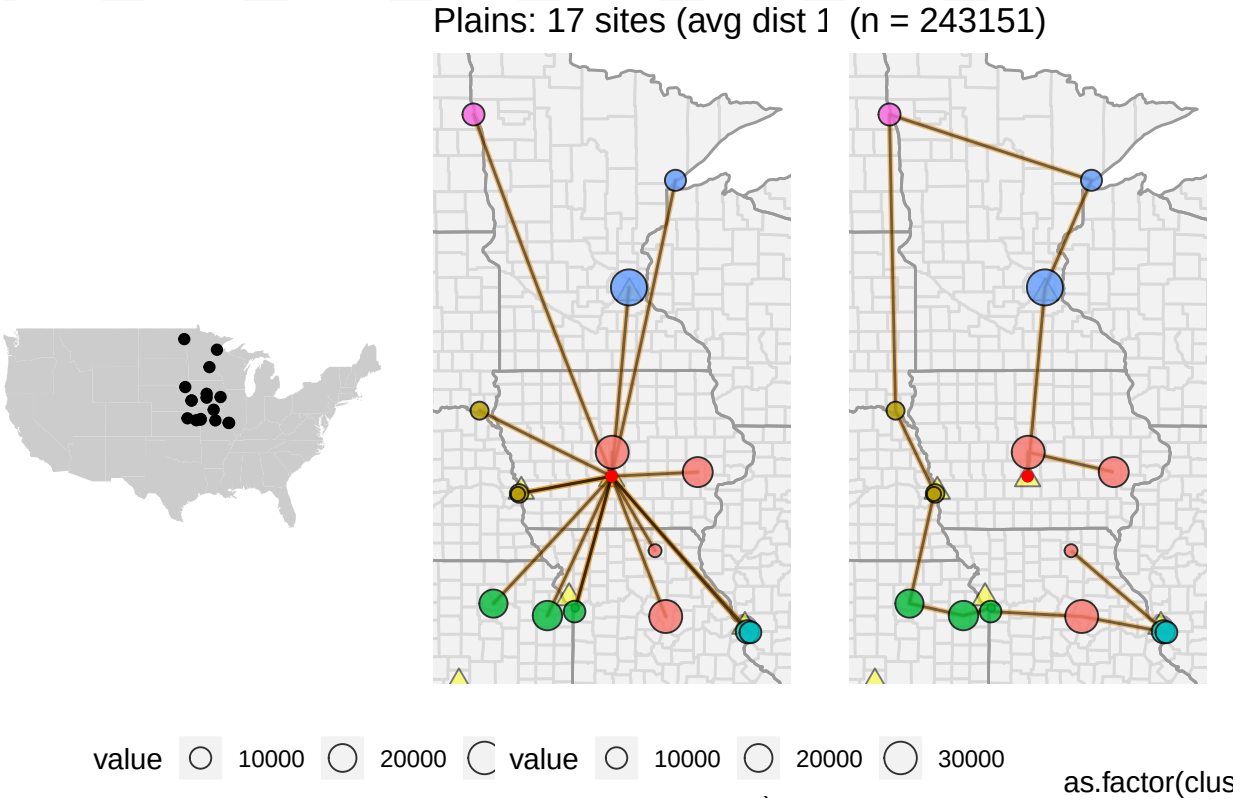
```
    plot2 <- #map of shortest routes
        basemap + map_projection + map_airports +
        geom_path(data = RegionLoop[path.tsp,],
                  aes(x=lon, y=lat), color = "#E58700", size = 1, alpha = .5) +
        geom_path(data = RegionLoop[path.tsp,],
                  aes(x=lon, y=lat), color = "black", size = .5, alpha = .5) +
        map_locations + map_location_central + map_legend +
        ggtitle(paste0("(n = ", sum(RegionLoop$value), ")"))

  #put all plots together
    grid.arrange(baseplot, plot1, plot2,ncol=3)

} #end of for loop
```

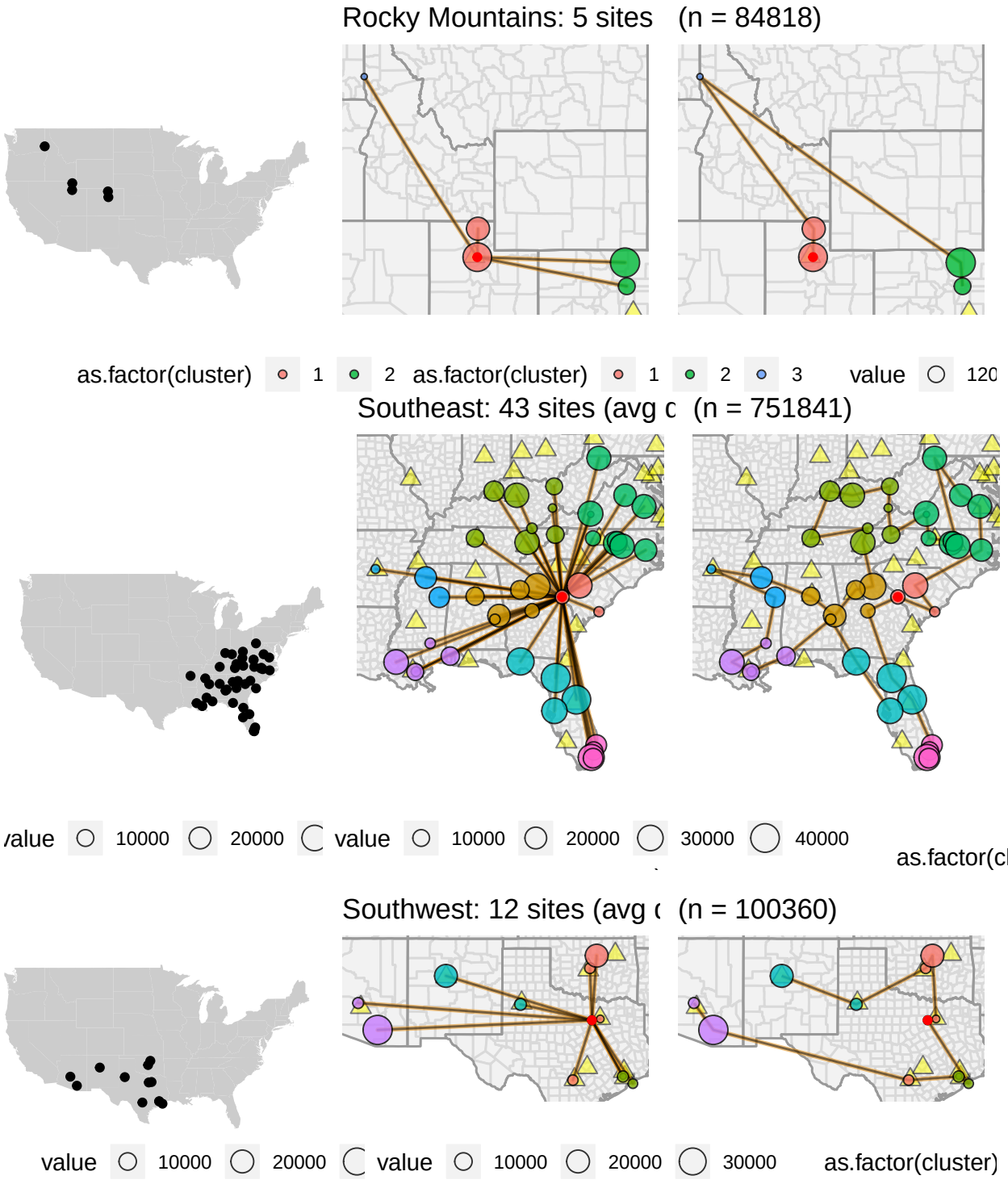



Great Lakes: 26 sites (avg
(n = 451465)
as.factor(cluster)
1
2
3
4
as.factor(cluster)
1
2
3
4
5
6
7
value
10000


Mid East: 30 sites (avg dis
(n = 426299)
as.factor(cluster)
1
2
3
4
as.factor(cluster)
1
2
3
4
5
6
value
10000


New England: 11 sites (av (n = 131670)
5000
10000
value
5000
10000
15000
20000
25000
as.factor(


Plains: 17 sites (avg dist 1 (n = 243151)
value
10000
20000
value
10000
20000
30000
as.factor(clus


Rocky Mountains: 5 sites (n = 84818)
as.factor(cluster) 1 2
as.factor(cluster) 1 2 3
value 120
Southeast: 43 sites (avg c (n = 751841)
value 10000 20000
value 10000 20000 30000 40000
as.factor(cl
Southwest: 12 sites (avg c (n = 100360)
value 10000 20000
value 10000 20000 30000
as.factor(cluster)