HW2

Haojie Liu

2025-04-16

1. Download the file country__demographics.csv from the Module 2 datasets folder. I want you to tell me something about the linear relationship between GNP (gross national product = crude measure of a country's wealth) and LEXPF (the life expectancy of females). You will encounter some problems. There are two helpful hints below.

```
c_demog <- read.csv("country_demographics.csv", sep=",", header=T)
c_demog2 <-c_demog[complete.cases(c_demog),]
```

```
with(c_demog, cor(GNP, LEXPF))
```

A. Generate the correlation coefficient between GNP and female life expectancy. Interpret the value of the correlation coefficient.

```
## [1] NA
```

```
with(c_demog2, cor(GNP, LEXPF))
```

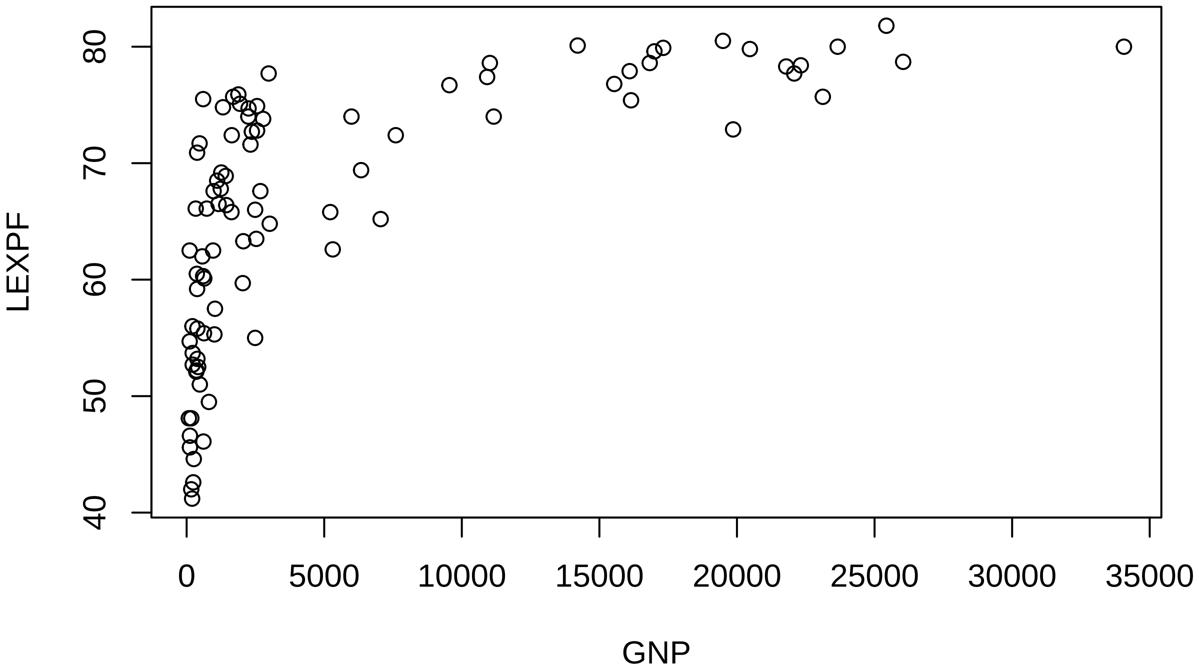
B. (The NA GNP values might cause problems - use Hint 1 and continue.)

```
## [1] 0.6500464
```

GNP and female life expectancy are positively correlated, where they increase and decrese at the same time.

```
with(c_demog2, plot(GNP, LEXPF))
```

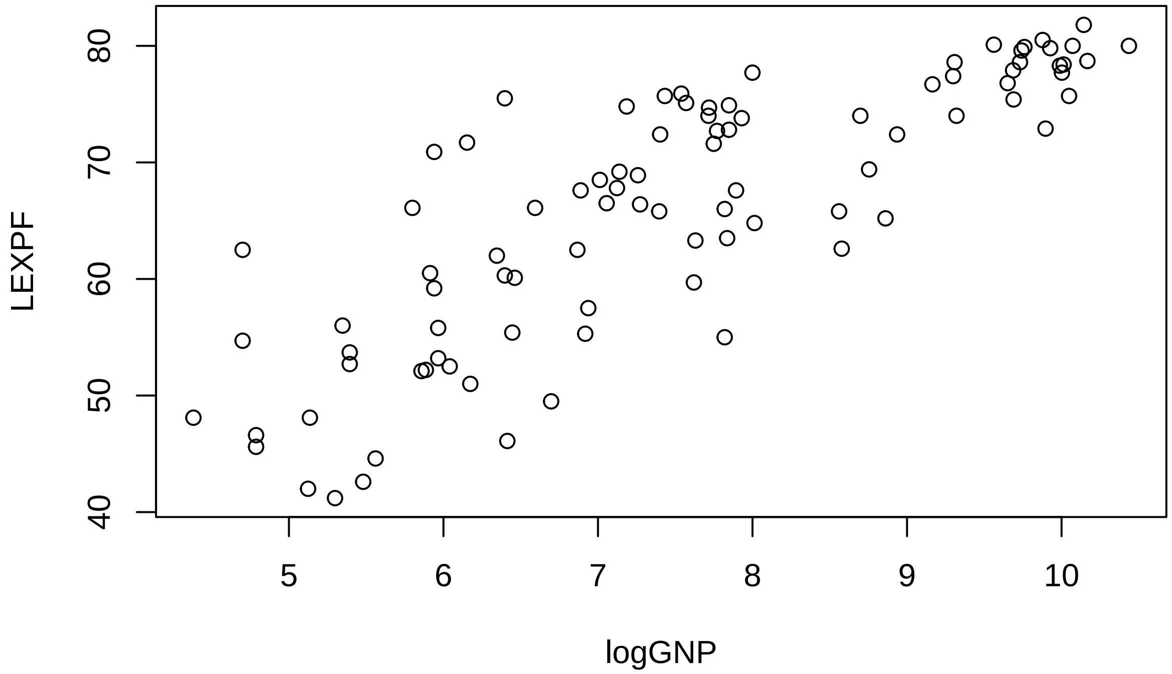
. Plot the relationship between GNP and female life expectancy. What do you see?



Not clear relationship showing on the plot

```
logGNP <- log(c_demog2$GNP)
with(c_demog2, plot(logGNP, LEXPF))
```

D. (The GNP data exhibit considerable skew and this is impacting the relationship between the two variables. Get rid of the skew by logging GNP – use Hint 2 and continue)



```
with(c_demog2, cor(logGNP, LEXPF))
```

E. Take the logarithm of GNP and recalculate the correlation coefficient between logged GNP and female life expectancy. Compare the result with what you found above. Check the data again in another plot. Does the plot look less impacted by skew?

```
## [1] 0.8233949
```

Positive relationship become stronger after transform the data.

Hint 1

> c_demog <- read.csv("country_demographics.csv", sep=",", header=T)
>
> c_demog2 <-c_demog[complete.cases(c_demog),]

Hint 2

> logGNP <- log(c_demog2$GNP)

2. Find two of your own shapefiles, one with at least 100 points and the other containing at least 50 areas (polygons). These could come from the same database. (Don’t use the shapefefiles on the class website.) A shapefile is a collection of data and information used to store locational and geographic information. When downloading a shapefile, be sure to obtain and include all of the necessary components of the shapfile. The ‘st_read’ command will not import a shapefile, unless all of the necessary components are present in the shapefile folder.

```
library(tmap)
library(tmaptools)
library(raster)
```

```
## Loading required package: sp
```

```
library(sf)
```

```
## Linking to GEOS 3.13.0, GDAL 3.8.5, PROJ 9.5.1; sf_use_s2() is TRUE
```

```
library(dplyr)
```

```
##
## Attaching package: ’dplyr’

## The following objects are masked from ’package:raster’:
##
##     intersect, select, union

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

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

```
LAschoolBound <- st_read("/Users/liuhaojie/Desktop/STATSM222/Week3/LAschoolbound")
```

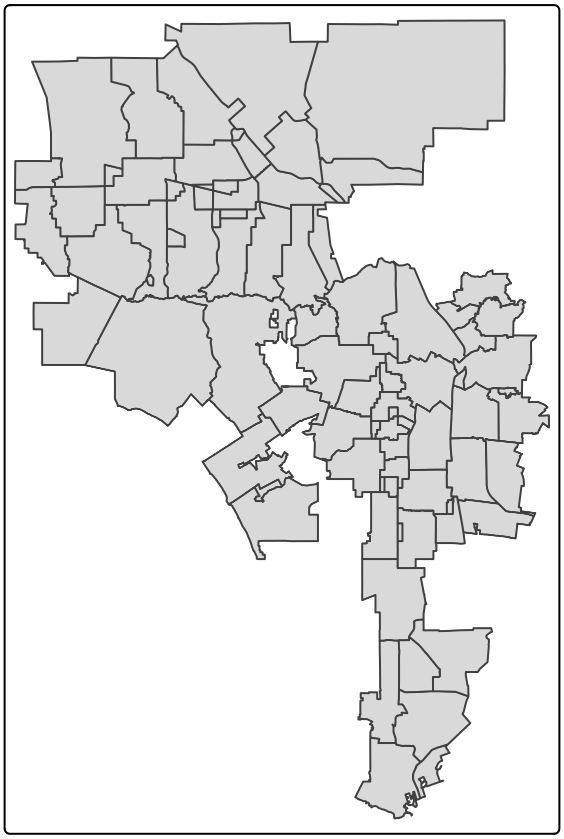
```
## Reading layer ‘LAschoolbound’ from data source
##   ‘/Users/liuhaojie/Desktop/STATSM222/Week3/LAschoolbound’ using driver ‘ESRI Shapefile’
## Simple feature collection with 85 features and 6 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: -13210060 ymin: 3989206 xmax: -13151600 ymax: 4076520
## Projected CRS: WGS 84 / Pseudo-Mercator
```

```
SCUpoint <- st_read("/Users/liuhaojie/Desktop/STATSM222/Week3/SCUpoint")
```

```
## Reading layer ‘SCUpoint’ from data source
##   ‘/Users/liuhaojie/Desktop/STATSM222/Week3/SCUpoint’ using driver ‘ESRI Shapefile’
## Simple feature collection with 3179 features and 19 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 6305226 ymin: 1581825 xmax: 6652590 ymax: 2111965
## Projected CRS: NAD83 / California zone 5 (ftUS)
```

```r
qtm(LAschoolBound)
```

A. Map one of the variables in the polygon shapefile and briefly interpret what it tells you.

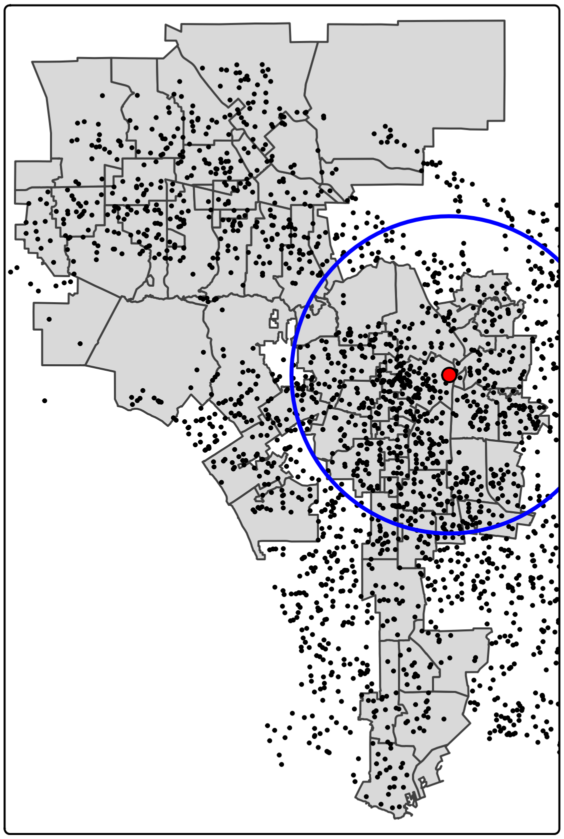


A simple map of LA districts.

```r
SCUpoint_wgs <- st_transform(SCUpoint, 4326)


mc_coords <- st_coordinates(SCUpoint_wgs) |> colMeans()

mc <- st_sfc(st_point(mc_coords), crs = 4326) |> st_transform(st_crs(SCUpoint))
mc <- st_sf(geometry = mc)

coords_proj <- st_coordinates(SCUpoint)
center_proj <- st_coordinates(mc)
dists <- sqrt((coords_proj[,1] - center_proj[1,1])^2 + (coords_proj[,2] - center_proj[1,2])^2)
sd_radius <- sd(dists)
sd_circle <- st_buffer(mc, dist = sd_radius)

tm_shape(LAschoolBound) + tm_polygons() +
  tm_shape(SCUpoint) +
  tm_dots(size = 0.1) +
  tm_shape(mc) +
  tm_symbols(shape = 21, col = "red", size = 0.5, border.col = "black")+
  tm_shape(sd_circle) +
  tm_borders(lwd = 2, col = "blue")
```

B. Map the points in the second shapefile, find the mean center and standard distance and interpret what you see.

```
##

## -- tmap v3 code detected -----------------------------------------------------

## [v3->v4] 'symbols()': use 'fill' for the fill color of polygons/symbols
## (instead of 'col'), and 'col' for the outlines (instead of 'border.col').
## This message is displayed once every 8 hours.
```



We can see most of the college, university and schools are locate around downtown LA.

3. On the Module 2 Datasets page are two files – us_state_popln_history.csv and us_state_centroids.csv. (If the state centroids are not working for you, you might have to find your own coordinates (lon, lat) for the centroid of each state.) The population-history file reports the population of all 50 states and Washington DC over 10 year intervals from 1870 to 2020. You should filter out (delete) the observations for Alaska and Hawaii, focusing only on the contiguous US states. (You will have to deal with two NA "values" early on for Oklahoma. Changing those to 0 is a solution in this case.)

```
statepop <- read.csv("us_state_popln_history.csv")
statepop[is.na(statepop)] <- 0
statepop <- statepop %>%
  filter(!state %in% c("Alaska", "Hawaii"))

centroids <- read.csv("us_state_centroids.csv")
pop_centroids <- centroids %>%
  inner_join(statepop, by = "state") %>%
  select(state, lon, lat, pop1960)
```

A. Read into R the state population data.

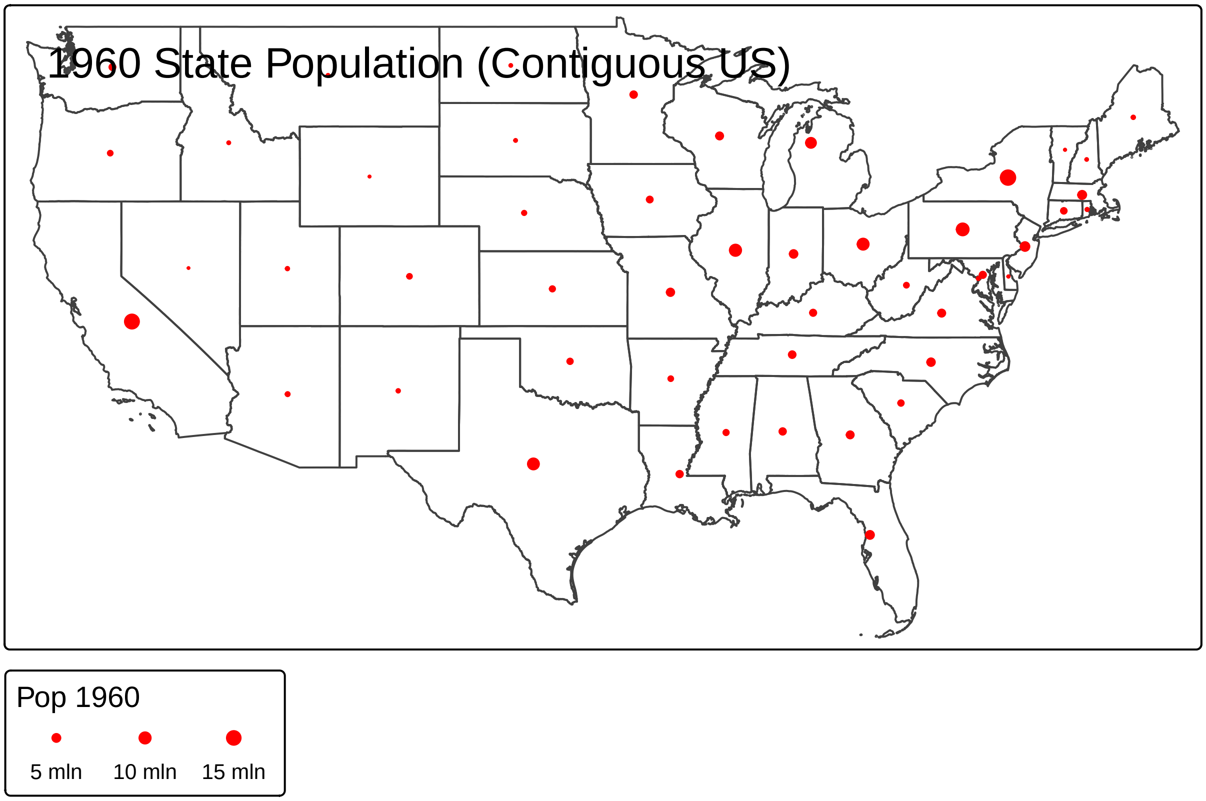
```
state_bound <- st_read("cb_2018_us_state_5m")
```

B. Find a shapefile that you can use to link to the state population totals.

```
## Reading layer `cb_2018_us_state_5m' from data source
##   `/Users/liuhaojie/Desktop/STATSM222/Week3/cb_2018_us_state_5m'
##   using driver `ESRI Shapefile'
## Simple feature collection with 56 features and 9 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -179.1473 ymin: -14.55255 xmax: 179.7785 ymax: 71.35256
## Geodetic CRS:  NAD83
```

```
state_bound <- state_bound %>%
  filter(STUSPS %in% c(
    "AL", "AR", "AZ", "CA", "CO", "CT", "DC", "DE", "FL", "GA",
    "IA", "ID", "IL", "IN", "KS", "KY", "LA", "MA", "MD", "ME",
    "MI", "MN", "MO", "MS", "MT", "NC", "ND", "NE", "NH", "NJ",
    "NM", "NV", "NY", "OH", "OK", "OR", "PA", "RI", "SC", "SD",
    "TN", "TX", "UT", "VA", "VT", "WA", "WI", "WV", "WY"
  ))

pop_sf <- st_as_sf(pop_centroids, coords = c("lon", "lat"), crs = 4326)

# Plot over contiguous US map
tm_shape(state_bound) +
  tm_borders() +
  tm_shape(pop_sf) +
  tm_dots(size = "pop1960", col = "red", scale = 0.5, title.size = "Pop 1960") +
  tm_layout(title = "1960 State Population (Contiguous US)", legend.outside = TRUE)
```

```
##

## -- tmap v3 code detected -------------------------------------------------------

## [v3->v4] `tm_tm_dots()`: migrate the argument(s) related to the scale of the
## visual variable `size` namely 'scale' (rename to 'values.scale') to size.scale
## = tm_scale_continuous(<HERE>).
## i For small multiples, specify a 'tm_scale_' for each multiple, and put them in
##   a list: 'size.scale = list(<scale1>, <scale2>, ...)'
## [v3->v4] `tm_layout()`: use `tm_title()` instead of `tm_layout(title = )`
```

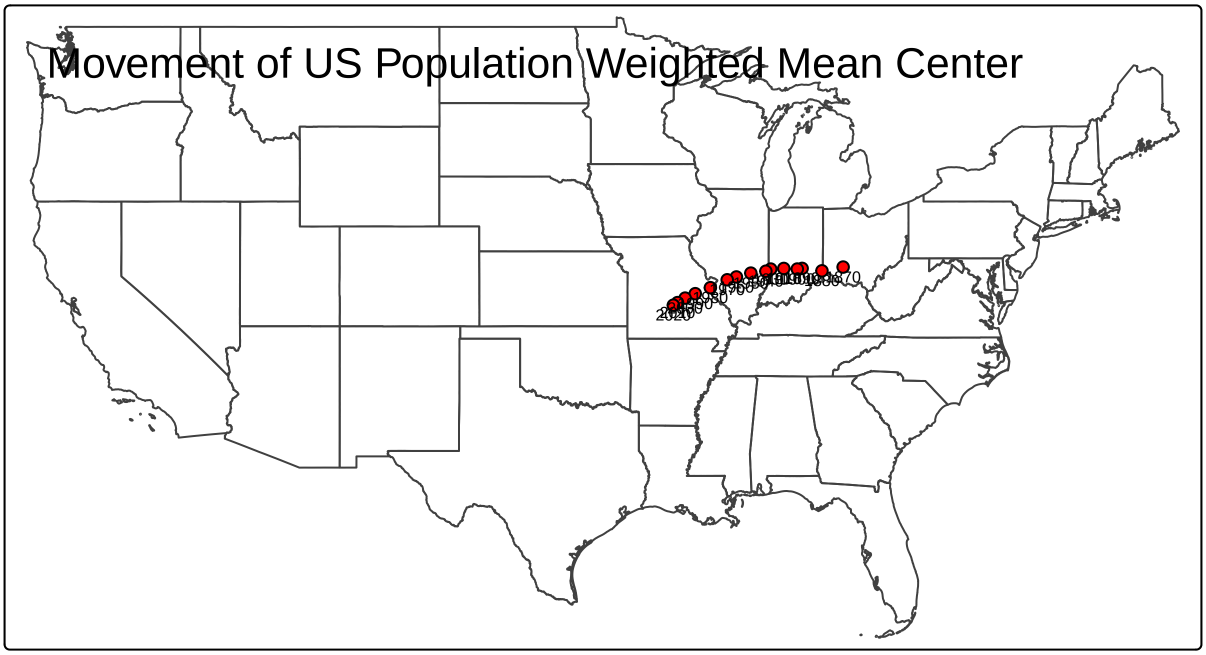


```
data <- inner_join(statepop, centroids, by = "state")

get_weighted_center <- function(df, lon_col, lat_col, weight_col) {
  x <- sum(df[[lon_col]] * df[[weight_col]]) / sum(df[[weight_col]])
  y <- sum(df[[lat_col]] * df[[weight_col]]) / sum(df[[weight_col]])
  return(c(lon = x, lat = y))
}
```

```r
decades <- grep("^pop", names(data), value = TRUE)

centers <- lapply(decades, function(decade) {
  coords <- get_weighted_center(data, "lon", "lat", decade)
  data.frame(decade = gsub("pop", "", decade), lon = coords[1], lat = coords[2])
})

centers_df <- do.call(rbind, centers)

centers_sf <- st_as_sf(centers_df, coords = c("lon", "lat"), crs = 4326)

tm_shape(state_bound) +
  tm_borders() +
  tm_shape(centers_sf) +
  tm_symbols(shape = 21, col = "red", size = 0.4, border.col = "black") +
  tm_text("decade", size = 0.5, ymod = -0.5) +
  tm_layout(title = "Movement of US Population Weighted Mean Center")
```

C. Calculate the weighted mean center of the US population for each decade and show how it has moved on your tmap map. (You can calculate the weighted mean center in R just using the state centroids, or whatever other coordinates that you obtain, along with the population values. Let us not worry about the transformation from lat-lon coordinates to an XY space for the moment.) Briefly report what you see.

```
##

## -- tmap v3 code detected -------------------------------------------------------

## [v3->v4] `tm_layout()`: use `tm_title()` instead of `tm_layout(title = )`
```



The populaton center are moving towards to the west from 1870 to 2020.