

Tarea 2

Sara Luz e Iván

Carga de Bibliotecas

```
library(foreign)
library(tidyverse)
library(lubridate)
library(readr)
library(sp)
library(rgdal)
library(pracma)
library(R.utils)
library(geosphere)
library(kableExtra)
library(readr)
```

Solo puedes usar for, while:

1. Obten los centroides de todas las manzanas CDMX con el Censo 2020 del INEGI: liga; Nombre del archivo 09m.

Descarga de la base de datos

Se utilizaron las siguientes ligas:

- [Censo 2020 CDMX](#)
- [DENUE CDMX 2020/11](#)

Carga de Datos

```
mzn_censo = readOGR("./Datos/09_ciudaddemexico/conjunto_de_datos/09m.shp")
```

OGR data source with driver: ESRI Shapefile

Source: "/home/saraluz/Documents/2Sem/DatosEspaciales/Tareas/Datos/09_ciudaddemexico/conjunto_de_datos/09m.shp"

with 66789 features

It has 8 fields

- Son 66,789 manzanas
- con 8 variables

Análisis de los datos

```
mzn_censo@data %>% head() %>% kable() %>% kable_paper()
```

	CVEGEO	CVE_ENT	CVE_MUN	CVE_LOC	CVE_AGEBC	CVE_MZA	AMBITO
0	0901000010898031	09	010	0001	0898	031	Urbana
1	0901000012269024	09	010	0001	2269	024	Urbana
2	0901000011472068	09	010	0001	1472	068	Urbana
3	0901000011824024	09	010	0001	1824	024	Urbana
4	0901000012377004	09	010	0001	2377	004	Urbana
5	0901000012358031	09	010	0001	2358	031	Urbana

```
mzn_censo@proj4string
```

Coordinate Reference System:

Deprecated Proj.4 representation:

```
+proj=lcc +lat_0=12 +lon_0=-102 +lat_1=17.5 +lat_2=29.5 +x_0=2500000  
+y_0=0 +ellps=GRS80 +units=m +no_defs
```

WKT2 2019 representation:

```
PROJCRS["MEXICO_ITRF_2008_LCC",  
    BASEGEOGCRS["ITRF2008",  
        DATUM["International Terrestrial Reference Frame 2008",  
            ELLIPSOID["GRS 1980",6378137,298.257222101,  
                LENGTHUNIT["metre",1]],  
            ID["EPSG",1061]],  
        PRIMEM["Greenwich",0,
```

```

        ANGLEUNIT["Degree",0.0174532925199433]]],
CONVERSION["unnamed",
METHOD["Lambert Conic Conformal (2SP)",
ID["EPSG",9802]],
PARAMETER["Latitude of false origin",12,
ANGLEUNIT["Degree",0.0174532925199433],
ID["EPSG",8821]],
PARAMETER["Longitude of false origin",-102,
ANGLEUNIT["Degree",0.0174532925199433],
ID["EPSG",8822]],
PARAMETER["Latitude of 1st standard parallel",17.5,
ANGLEUNIT["Degree",0.0174532925199433],
ID["EPSG",8823]],
PARAMETER["Latitude of 2nd standard parallel",29.5,
ANGLEUNIT["Degree",0.0174532925199433],
ID["EPSG",8824]],
PARAMETER["Easting at false origin",2500000,
LENGTHUNIT["metre",1],
ID["EPSG",8826]],
PARAMETER["Northing at false origin",0,
LENGTHUNIT["metre",1],
ID["EPSG",8827]]],
CS[Cartesian,2],
AXIS["(E)",east,
ORDER[1],
LENGTHUNIT["metre",1,
ID["EPSG",9001]]],
AXIS["(N)",north,
ORDER[2],
LENGTHUNIT["metre",1,
ID["EPSG",9001]]]]

```

```
mzn_censo@polygons[[2]]@Polygons[[1]]@coords %>% head()
```

	[,1]	[,2]
[1,]	2791388	821465.2
[2,]	2791376	821459.9
[3,]	2791365	821457.5
[4,]	2791361	821457.4
[5,]	2791360	821467.2
[6,]	2791369	821490.3

Los datos se encuentran en coordenadas un sistema de coordenadas proyectadas

```
plot(mzn_censo, main="Colonias CDMX")
```

Colonias CDMX



- Lo cambiamos a Latitud y Longitud

```
m = seq(1,length(mzn_censo))
manzanas <- tibble(mzn= numeric(),lon= numeric(),lat= numeric())

for (i in m) {

  # Agrupamos las coordenadas de la i-ésima manzana a convertir en un data.frame
  x<-mzn_censo@polygons[[i]]@Polygons[[1]]@coords[,1]#longitud
  y<-mzn_censo@polygons[[i]]@Polygons[[1]]@coords[,2]#latitud
  d <- data.frame(lon=x, lat=y)
  coordinates(d) <- c("lon", "lat")

#Escribimos la proyección actual de nuestro sistema de coordenadas geográficas
proj4string(d) <- CRS("+proj=lcc +lat_1=17.5 +lat_2=29.5 +lat_0=12 +lon_0=-102 +x_0=2500
#En CRS.new escribimos la proyección a la que queremos convertir nuestros datos
CRS.new <- CRS("+proj=longlat +datum=WGS84 +no_defs")
```

```

#Transformamos el sistema
d_proyectado <- spTransform(d, CRS.new)

#Convertirmos los resultados es un nuevo data.frame
D<-data.frame(mzn=i, d_proyectado)#conviértelo a data frame para que los veas como tabla
manzanas <- manzanas %>% add_row(D)
}

#head(D) %>% kable() %>% kable_paper()

```

Cálculo del centroide

```

centroide <- manzanas %>%
  group_by(mzn) %>%
  summarise(c_lon=mean(lon), c_lat=mean(lat))

```

-Añadimos clave de la manzana

```

clave_mzn <- mzn_censo@data %>% select(CVE_MZA)
centroide <- centroide %>% add_column(clave_mzn)
head(centroide) %>% kable() %>% kable_paper()

```

mzn	c_lon	c_lat	CVE_MZA
1	-99.23021	19.35925	031
2	-99.21853	19.36545	024
3	-99.24615	19.37789	068
4	-99.22448	19.34755	024
5	-99.21162	19.36945	004
6	-99.25178	19.36845	031

2. Encuentra la ubicación de todos los OXXO de la CDMX, asegúrate de filtrar correctamente los datos para evitar falsos positivos.

Carga de Datos

```

datos_denue <- read.csv('./Datos/denue_09_1120_csv/denue_09_csv/conjunto_de_datos/denue_in'

```

Limpieza de datos

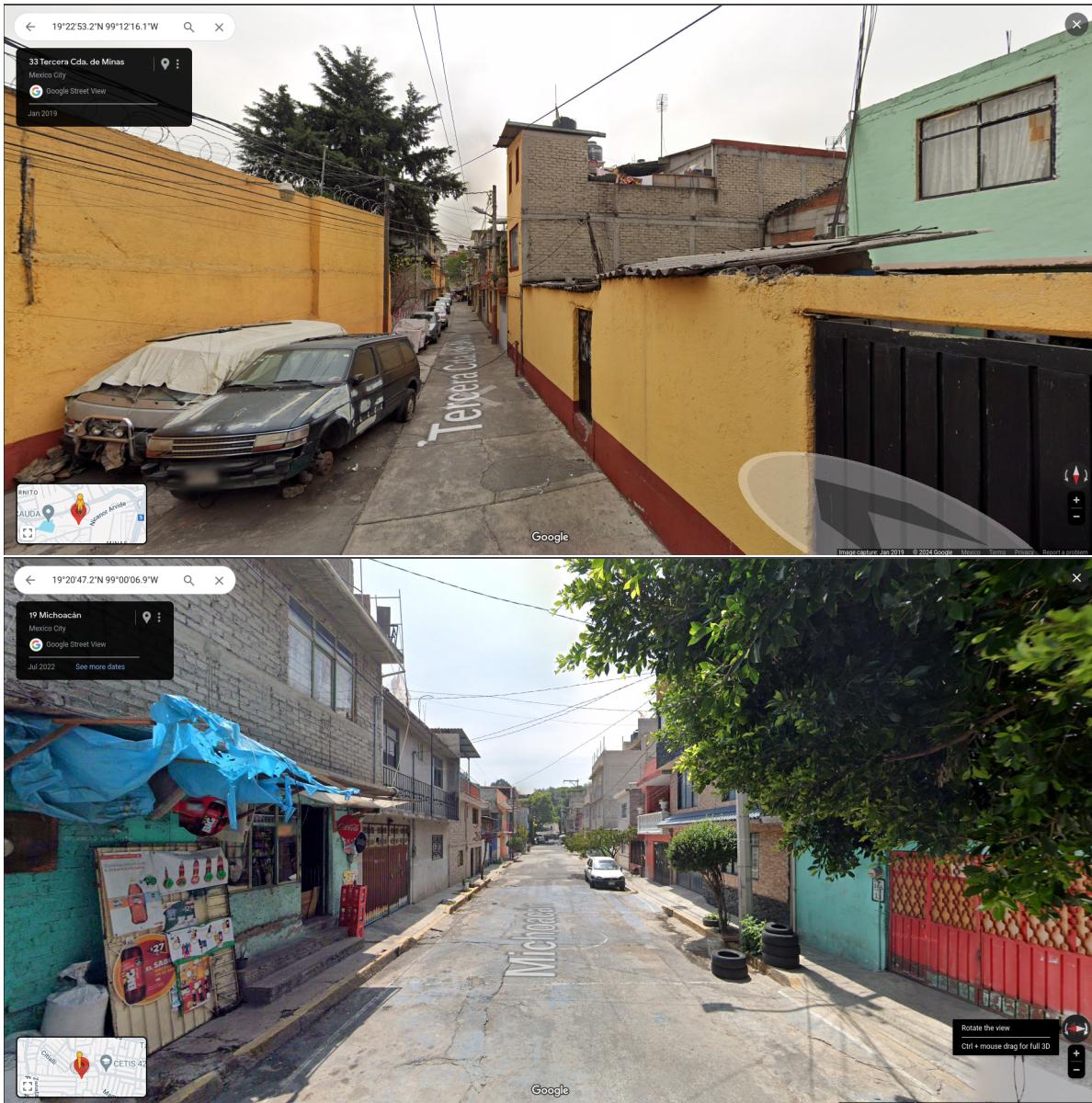
- Primero filtramos todos aquellos que puedan ser un OXXO pero tengan otra razón social

```
datos_denue %>% select(nom_estab, raz_social, latitud, longitud) %>%
  filter(str_detect(nom_estab, regex("oxxo", ignore_case = TRUE)) | 
    str_detect(nom_estab, regex("oxo", ignore_case = TRUE))) ) %>%
  filter(raz_social!="CADENA COMERCIAL OXXO SA DE CV") %>% head() %>% kable() %>% kable_p
```

nom_estab	raz_social	latitud	longitud
ABARROTES LOXO		19.30498	-99.03738
ABARROTES LOXXO		19.37335	-99.01004
ABARROTES OCOMOXOTLA		19.21709	-99.21397
CADENA COMERCIAL OXXO		19.49076	-99.07192
MINISUPER XOXO		19.27646	-99.00568
MISCELANEA OXXO		19.38145	-99.20446

- Usamos Google Maps para revisar las más probables de ser un OXXO, y únicamente una resultó serlo:





- Con lo cual los datos quedarían como:

```
datos_denue <- datos_denue %>% select(id, nom_estab, raz_social, latitud, longitud, manzana)
  filter(nom_estab == "CADENA COMERCIAL OXXO" | raz_social=="CADENA COMERCIAL OXXO SA DE C
  tibble() %>%
  select(id, latitud, longitud, manzana)
datos_denue %>%
  head() %>% kable() %>% kable_paper()
```

	id	latitud	longitud	manzana
6322823	19.44847	-99.13852	2	
6322871	19.41737	-99.16489	11	
6305767	19.49862	-99.11795	4	
6308525	19.38593	-99.11723	3	
6305777	19.50325	-99.15666	10	
6305903	19.44240	-99.17432	7	

3. Determina cuál es OXXO más cercano al Centroide de cada manzana de la CDMX.

- Primero hacemos una prueba

```
distHaversine(c(datos_denue[[1,3]],datos_denue[[1,2]]),c(centroide[[1,2]],centroide[[1,3]])
```

[1] 13832.86

- Posteriormente hacemos un ciclo FOR

OJO hubo que detener este código después de tres días de estarlo corriendo

```
# n = seq(1,nrow(datos_denue))
#
# distancias <- tibble(mzn= numeric(),c_lon= numeric(), c_lat= numeric(), CVE_MZA=character(),
#                       oxxo=integer(), id=integer(), latitud= numeric(), longitud= numeric(),
#                       distancia=numeric())
#
# for (i in m) {
#   for (j in n) {
#     d <- distHaversine(c(datos_denue[[j,3]],datos_denue[[j,2]]),c(centroide[[i,2]],centroide[[i,3]]))
#     D <- data.frame(slice(centroide, i), j, slice(datos_denue,j), d)
#     distancias <- distancias %>% rbind(tibble(D))
#   }
# }
# }

distancias = read_csv("./Datos/distancias_miau.csv",show_col_types = FALSE)
```

4. Determina cuál es el OXXO que es más cercano a más manzanas de la CDMX.

```
distancias %>% group_by(j) %>%
  summarise(dist_tot = sum(d)) %>%
  arrange(dist_tot) %>%
  head()

# A tibble: 6 x 2
      j   dist_tot
  <dbl>     <dbl>
1 352 4379875.
2 1184 4491465.
3 814 4580638.
4 647 4610756.
5 627 4646237.
6 176 4679610.

distancias %>% filter(j==352)

# A tibble: 1,491 x 10
   mzn c_lon c_lat CVE_MZA     j      id latitud longitud manzana      d
   <dbl> <dbl> <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1     1 -99.2 19.4 031     352 6323906  19.4 -99.2    3 1311.
2     2 -99.2 19.4 024     352 6323906  19.4 -99.2    3  584.
3     3 -99.2 19.4 068     352 6323906  19.4 -99.2    3 2650.
4     4 -99.2 19.3 024     352 6323906  19.4 -99.2    3 2368.
5     5 -99.2 19.4 004     352 6323906  19.4 -99.2    3 1182.
6     6 -99.3 19.4 031     352 6323906  19.4 -99.2    3 3039.
7     7 -99.3 19.4 023     352 6323906  19.4 -99.2    3 3190.
8     8 -99.3 19.4 022     352 6323906  19.4 -99.2    3 3148.
9     9 -99.2 19.4 007     352 6323906  19.4 -99.2    3 1634.
10    10 -99.3 19.4 018     352 6323906  19.4 -99.2    3 3700.
# i 1,481 more rows
```

5. ¿Cuál es la distancia euclíadiana entre el OXXO del punto 4 y este en la Paz?

- Primero pasamos las coordenadas a UTM

```

# convertimos a coordenadas espaciales
mydf <- structure(list(longitude = c(-99.22285, -110.3136462), latitude = c(19.36876, 24.1),
                           .Names = c("longitude", "latitude"), class = "data.frame", row.names = c(NA, -1L))

xy <- mydf[,c(1,2)]

spdf <- SpatialPointsDataFrame(coords = xy, data = mydf,
                                 proj4string = CRS("+proj=longlat +datum=WGS84 +ellps=WGS84"))

# Utilizamos las coordenadas geograficas y las proyectamos
d <- data.frame(lon=spdf$longitude, lat=spdf$latitude)
coordinates(d) <- c("lon", "lat")

# Estan en lat,lon entonces declaramos la proyeccion usual
sputm <- SpatialPoints(d, proj4string=CRS("+proj=longlat +datum=WGS84"))

# Ya sabemos que todos los puntos estan en la CDMX y por lo tanto, pertenecen a la UTM Zone
proyeccion<-CRS("+proj=utm +zone=14 +datum=WGS84 +units=m +no_defs ")

# Transformamos los datos
UTM <- spTransform(sputm, proyeccion) %>%
  data.frame()
colnames(UTM) <- c("lon_UTM", "lat_UTM")

pto1 <- c(UTM$lon_UTM[1], UTM$lat_UTM[1])
pto2 <- c(UTM$lon_UTM[2], UTM$lat_UTM[2])

```

- Calculamos distancia euclidiana

```

#Formula
euclidean <- function(a, b) sqrt(sum((a - b)^2))

euclidean(pto1, pto2)

```

[1] 1268998

6. ¿Cuál es la distancia de Haversine entre el OXXO del punto 4 y este en la Paz?

```
distHaversine(c(spdf$longitude[1],spdf$latitude[1]), c(spdf$longitude[2],spdf$latitude[2]))
```

```
[1] 1263152
```

Escribe una rutina de R para cada pregunta. Mantén el orden en tu código y comenta todo.

Colócalo en el Folder de equipo a más tardar, **miércoles 31 de febrero 11 am.**

Puntuación Extra:

Si tuvieras las coordenadas de los centroides de las manzanas y las de los OXXO en UTM Mercator, ¿Cómo podrías ahorrar operaciones para responder las preguntas anteriores? Diseña un planteamiento teórico - geométrico para la pregunta anterior, considera el espacio geométrico en el que están los puntos en UTM y sus propiedades.