

Empleando R como Sistema de Información Geográfica (SIG)

Gustavo Ahumada

Sept 13, 2019

1. Representación simple de datos espaciales

Los objetos espaciales usualmente son representados por datos tipo vector. Tales tipos de datos consisten en la geometría o la forma de los objetos. Por ejemplo, un conjunto de datos vector que contienen el límite de los países del mundo (geometría) y también contienen su respectivo tamaño poblacional para el año 2010. Por otra parte, se tienen los datos espaciales continuos, los cuales son frecuentemente representados con datos tipo raster. A continuación se representante este tipo de datos utilizando R. Parte del código que será utilizado en esta sección está disponible en <https://r-spatial.org/spatial/index.html>.

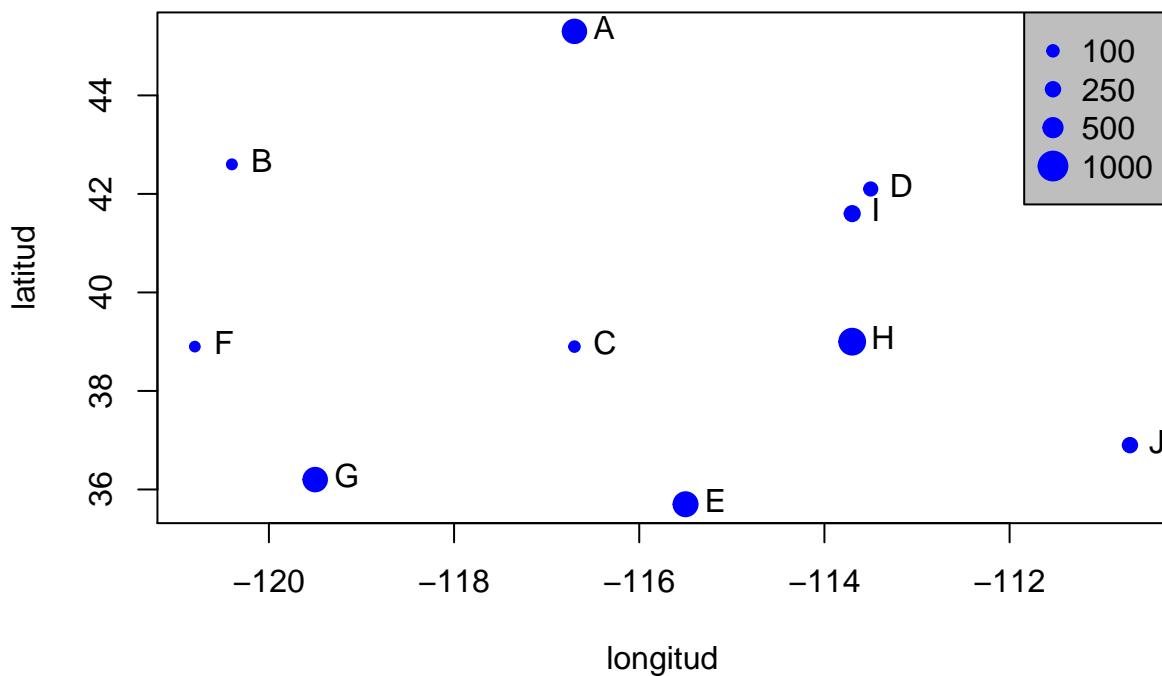
1.1. Datos tipo vector

```
# Cargar librerías
library(pacman)
pacman::p_load(raster, sf, maptools, rgdal, ggplot2, tidyverse, broom)

# Creación de 10 estaciones climáticas (llamadas de A a J)
nombre <- LETTERS[1:10]
longitud <- c(-116.7, -120.4, -116.7, -113.5, -115.5,
              -120.8, -119.5, -113.7, -113.7, -110.7)
latitud <- c(45.3, 42.6, 38.9, 42.1, 35.7, 38.9,
            36.2, 39, 41.6, 36.9)
est_climatic<- cbind(longitud, latitud)
# Simulación de datos de precipitación
set.seed(0)
precip <- round((runif(length(latitud))*10)^3)

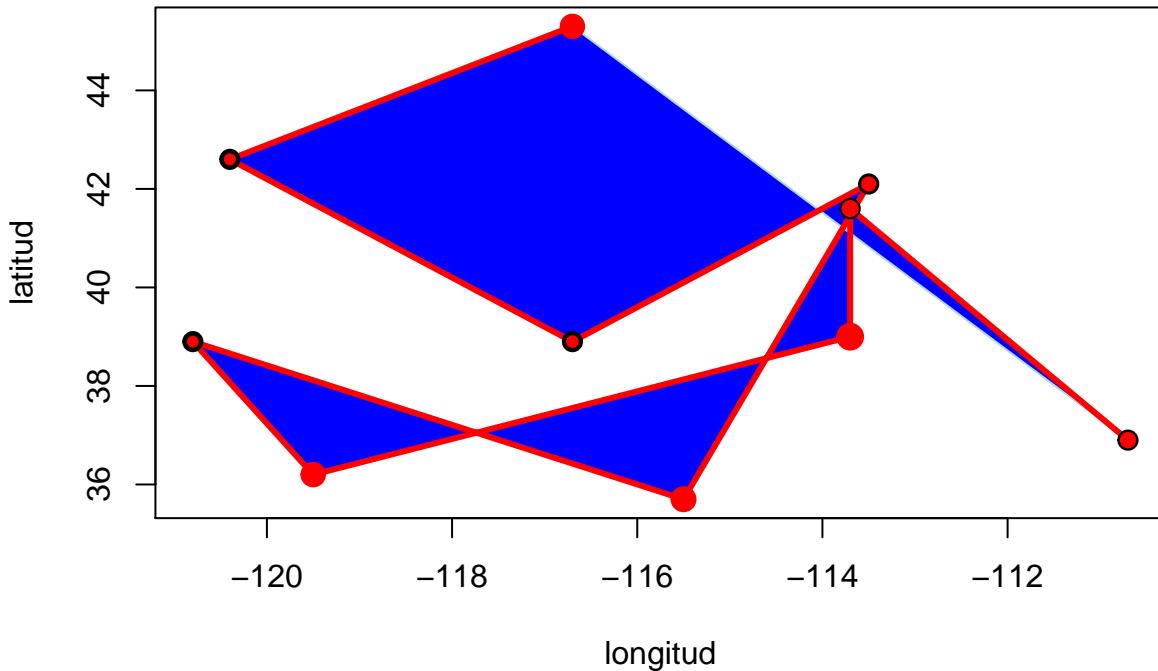
nivel_precip <- 1 + precip/500
plot(est_climatic, cex=nivel_precip, pch=20, col='blue', main='Precipitación por estaciones')
# Adicionar etiquetas
text.default(est_climatic, nombre, pos = 4)
# Adicionar leyenda
breaks <- c(100, 250, 500, 1000)
legend.psize <- 1+breaks/500
legend("topright", legend=breaks, pch=20, pt.cex=legend.psize, col='blue', bg='gray')
```

Precipitación por estaciones



```
# Adicionar puntos, lineas y polígonos al plot.
{longitud <- c(-116.7, -120.4, -116.7, -113.5, -115.5,
              -120.8, -119.5, -113.7, -113.7, -110.7)
latitud <- c(45.3, 42.6, 38.9, 42.1, 35.7, 38.9,
            36.2, 39, 41.6, 36.9)
x <- cbind(longitud, latitud)
plot(est_climatic, main='Precipitación por estaciones')
polygon(x, col='blue', border='light blue')
lines(est_climatic, lwd=3, col='red')
points(x, cex=2, pch=20)
points(est_climatic, cex=nivel_precip, pch=20, col='red', main='Precipitation by station')}
```

Precipitación por estaciones



```
# Tabla de datos
tabla <- data.frame(longitud, latitud, nombre, precip)
tabla
```

```
##   longitud latitud nombre precip
## 1     -116.7    45.3     A    721
## 2     -120.4    42.6     B     19
## 3     -116.7    38.9     C     52
## 4     -113.5    42.1     D    188
## 5     -115.5    35.7     E    749
## 6     -120.8    38.9     F      8
## 7     -119.5    36.2     G    725
## 8     -113.7    39.0     H   843
## 9     -113.7    41.6     I   289
## 10    -110.7    36.9     J   249
```

1.2. Datos tipo raster

```
# Crear esqueleto de una base de datos raster
rast <- raster(ncol=10, nrow=10, xmx=-80, xmn=-150, ymn=20, ymx=60)
rast
```

```
## class       : RasterLayer
```

```

## dimensions : 10, 10, 100 (nrow, ncol, ncell)
## resolution : 7, 4 (x, y)
## extent      : -150, -80, 20, 60 (xmin, xmax, ymin, ymax)
## crs         : +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0

# Asignar valores a objetos tipo raster
values(rast) <- runif(ncell(rast))
rast

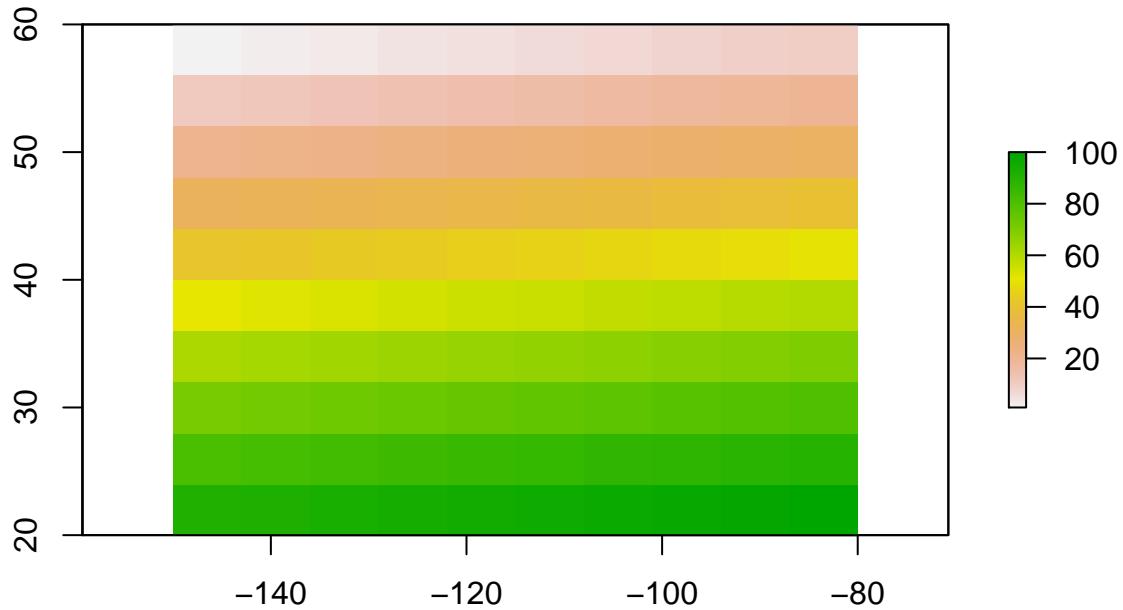
## class       : RasterLayer
## dimensions : 10, 10, 100 (nrow, ncol, ncell)
## resolution : 7, 4 (x, y)
## extent     : -150, -80, 20, 60 (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0
## source     : memory
## names      : layer
## values     : 0.01339033, 0.9926841 (min, max)

# Asignar el número de celdas
values(rast) <- 1:ncell(rast)
rast

## class       : RasterLayer
## dimensions : 10, 10, 100 (nrow, ncol, ncell)
## resolution : 7, 4 (x, y)
## extent     : -150, -80, 20, 60 (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0
## source     : memory
## names      : layer
## values     : 1, 100 (min, max)

plot(rast)

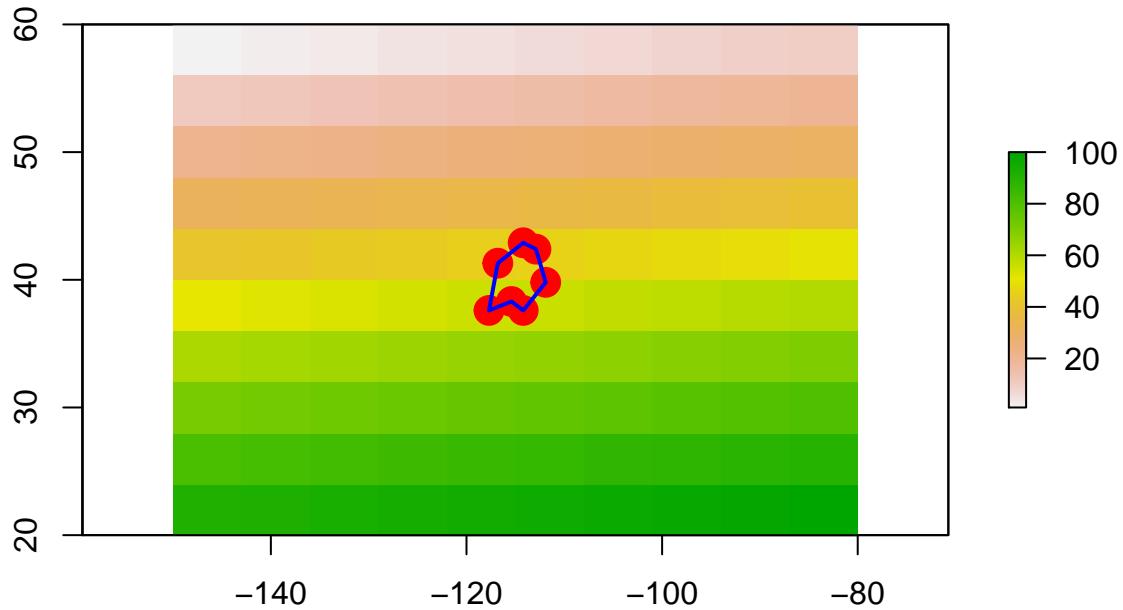
```



```

# Crear un esqueleto de una base de datos raster
rast <- raster(ncol=10, nrow=10, xmx=-80, xmnn=-150, ymn=20, ymx=60)
rast
# Asignar valores a objetos tipo raster
values(rast) <- runif(ncell(rast))
rast
# Asignar el número de celdas
values(rast) <- 1:ncell(rast)
rast
# Plottear objeto tipo raster
plot(rast)
# Adicionar puntos y polígonos
longitud <- c(-116.8, -114.2, -112.9, -111.9, -114.2, -115.4, -117.7)
latitud <- c(41.3, 42.9, 42.4, 39.8, 37.6, 38.3, 37.6)
lonlat <- cbind(longitud, latitud)
polos <- spPolygons(lonlat, crs='+proj=longlat +datum=WGS84')
points(lonlat, col='red', pch=20, cex=3)
plot(pols, border='blue', lwd=2, add=TRUE)}

```



2. Datos de vectores geográficos en R

```

# Definir directorio de trabajo
# en Windows sería algo a esto setwd("C:/Users/gusahu/Google Drive/Workshop_socber/Section_II")
knitr::opts_knit$set(root.dir = '/Users/gusahu/Google Drive/Workshop_GIS_with_R/Section_II')

# Cargar librerías
library(pacman)
pacman::p_load(raster, rgdal, rgeos, tidyverse, stringr, sf)

# Limpiar ambiente de trabajo
g <- gc(reset = TRUE)
rm(list = ls())
options(scipen = 999)

# Importar archivos shape (archivos tipo vector)
mps <- shapefile('./data/mpios_geo_ok2.shp')

# Plottear la geometría o el archivo .shp
plot(mps)

```



```

# Realizar la unión (join) de una tabla con un shapefile
# importar archivos shape (archivos tipo vector)
shp <- st_read('./data/mpios_geo_ok.shp')

## Reading layer `mpios_geo_ok' from data source `/Users/gusahu/Google Drive/Workshop_GIS_with_R/Section
## Simple feature collection with 1123 features and 13 fields
## geometry type: POLYGON
## dimension: XY
## bbox: xmin: -81.73882 ymin: -4.228614 xmax: -66.84722 ymax: 13.38857
## epsg (SRID): 4326
## proj4string: +proj=longlat +datum=WGS84 +no_defs

shp

## Simple feature collection with 1123 features and 13 fields
## geometry type: POLYGON
## dimension: XY
## bbox: xmin: -81.73882 ymin: -4.228614 xmax: -66.84722 ymax: 13.38857
## epsg (SRID): 4326
## proj4string: +proj=longlat +datum=WGS84 +no_defs
## First 10 features:
##   OBJECTID ID_ESPACIA AREA_OFICI           NOM_MUNICI COD_DEPTO FID_1
## 1          1      23189       644    CIÉNAGA DE ORO      23     7
## 2          2      23570       819    PUEBLO NUEVO      23     7
## 3          3      23068      1929      AYAPEL        23     7

```

```

## 4      4    23580      2062 PUERTO LIBERTADOR      23      7
## 5      5    23686      470      SAN PELAYO      23      7
## 6      6    70221      56      COVEÑAS      70     19
## 7      7    70523      174      PALMITO      70     19
## 8      8    70820      282      TOLÚ      70     19
## 9      9    70713      1089      SAN ONOFRE      70     19
## 10     10   70473      168      MORROA      70     19
##   ID_ESPAC_1 AREA_OFI_1 NOMBRE_DPT area_ha nombre Shape_Leng
## 1      23    26506 CÓRDOBA 63748.80 <NA> 1.6826813
## 2      23    26506 CÓRDOBA 79499.60 <NA> 1.5143324
## 3      23    26506 CÓRDOBA 193262.00 <NA> 2.0672053
## 4      23    26506 CÓRDOBA 165073.00 <NA> 2.5372155
## 5      23    26506 CÓRDOBA 47256.90 <NA> 1.5313400
## 6      70    10917 SUCRE  4897.34 <NA> 0.4252619
## 7      70    10917 SUCRE  17625.40 <NA> 0.6416914
## 8      70    10917 SUCRE  31535.50 <NA> 0.8868461
## 9      70    10917 SUCRE 107114.00 <NA> 1.8556544
## 10     70    10917 SUCRE 17855.40 <NA> 0.5933308
##   Shape_Area          geometry
## 1  0.052360437 POLYGON ((-75.7108 8.982489...
## 2  0.065246166 POLYGON ((-75.25133 8.32488...
## 3  0.158564544 POLYGON ((-75.18418 8.39827...
## 4  0.135181866 POLYGON ((-75.51257 7.94786...
## 5  0.038817484 POLYGON ((-75.7108 8.982489...
## 6  0.004028337 POLYGON ((-75.6075 9.468675...
## 7  0.014496629 POLYGON ((-75.54257 9.39386...
## 8  0.025952615 POLYGON ((-75.61853 9.40672...
## 9  0.088225137 POLYGON ((-75.37307 9.62956...
## 10 0.014690851 POLYGON ((-75.28508 9.35258...

```

```

# Importar archivos en formato .csv (.xls)
tbl <- read_csv('./data/produccion_cacao.csv')

```

```

## Parsed with column specification:
## cols(
##   COD_DPTO = col_double(),
##   DPTO = col_character(),
##   MPIO = col_character(),
##   COD_MUNI = col_double(),
##   CULTIVO = col_character(),
##   PERIODO = col_double(),
##   AREA_SEMBRADA_HA = col_double(),
##   AREA_COSECHADA_HA = col_double(),
##   PRODUCCION = col_double(),
##   RDTOS = col_double()
## )

```

```

tbl

```

```

## # A tibble: 5,478 x 10
##   COD_DPTO DPTO  MPIO  COD_MUNI CULTIVO PERIODO AREA_SEMBRADA_HA
##   <dbl> <chr> <chr> <dbl> <chr> <dbl> <dbl>
## 1      5 ANTI~ APAR~    5045 CACAO     2007    1830

```

```

## 2      5 ANTI~ TURBO    5837 CACAO    2007      1002
## 3      5 ANTI~ MACEO    5425 CACAO    2007      950
## 4      5 ANTI~ YALI     5885 CACAO    2007      580
## 5      5 ANTI~ NECO~    5490 CACAO    2007      523
## 6      5 ANTI~ VEGA~    5858 CACAO    2007      520
## 7      5 ANTI~ REME~    5604 CACAO    2007      405
## 8      5 ANTI~ DABE~    5234 CACAO    2007      315
## 9      5 ANTI~ "NAR~    5483 CACAO    2007      311
## 10     5 ANTI~ SAN ~   5659 CACAO    2007      307
## # ... with 5,468 more rows, and 3 more variables: AREA_COSECHADA_HA <dbl>,
## #   PRODUCCION <dbl>, RDTOS <dbl>

# Años disponible en archivo .csv
yrs <- unique(tbl$PERIODO)
yrs

## [1] 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017

tbl <- tbl %>% # pipe (%>%) es un operador que permite encadenar funciones
  filter(PERIODO == 2017) %>%
  mutate(MPIO = iconv(MPIO, to = 'latin1')) # transformación de variables en un data frame

# Ver atributos de objetos
str(tbl)

## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 526 obs. of 10 variables:
## $ COD_DPTO      : num  91 91 5 5 5 5 5 5 5 ...
## $ DPTO          : chr "AMAZONAS" "AMAZONAS" "ANTIOQUIA" "ANTIOQUIA" ...
## $ MPIO           : chr NA "LETICIA" "TURBO" "CACERES" ...
## $ COD_MUNI       : num 91540 91001 5837 5120 5490 ...
## $ CULTIVO        : chr "CACAO" "CACAO" "CACAO" "CACAO" ...
## $ PERIODO        : num 2017 2017 2017 2017 2017 ...
## $ AREA_SEMBRADA_HA: num 29 8 2476 1874 1856 ...
## $ AREA_COSECHADA_HA: num 29 6 1761 870 1265 ...
## $ PRODUCCION     : num 15 2 881 696 632 702 828 506 270 184 ...
## $ RDTOS          : num 0.5 0.4 0.5 0.8 0.5 0.5 0.6 0.6 0.6 0.5 ...

str(shp)

## Classes 'sf' and 'data.frame': 1123 obs. of 14 variables:
## $ OBJECTID : Factor w/ 1123 levels "1","10","100",...: 1 241 352 463 574 685 795 905 1016 2 ...
## $ ID_ESPACIA: Factor w/ 1123 levels "11001","13001",...: 288 297 282 299 307 915 925 934 930 923 ...
## $ AREA_OFICI: num 644 819 1929 2062 470 ...
## $ NOM_MUNICI: Factor w/ 1043 levels "ABEJORRAL","ÁBREGO",...: 205 664 71 684 815 238 613 949 809 553
## $ COD_DEPTO : Factor w/ 34 levels "05","08","11",...: 10 10 10 10 10 22 22 22 22 ...
## $ FID_1     : Factor w/ 34 levels "0","1","10","11",...: 32 32 32 32 32 12 12 12 12 ...
## $ ID_ESPAC_1: Factor w/ 34 levels "05","08","11",...: 10 10 10 10 10 22 22 22 22 ...
## $ AREA_OFI_1: num 26506 26506 26506 26506 26506 ...
## $ NOMBRE_DPT: Factor w/ 34 levels "AMAZONAS","ANTIOQUIA",...: 15 15 15 15 15 30 30 30 30 30 ...
## $ area_ha   : num 63749 79500 193262 165073 47257 ...
## $ nombre    : Factor w/ 0 levels: NA NA NA NA NA NA NA NA ...
## $ Shape_Leng: num 1.68 1.51 2.07 2.54 1.53 ...

```

```

## $ Shape_Area: num  0.0524 0.0652 0.1586 0.1352 0.0388 ...
## $ geometry  :sfc_POLYGON of length 1123; first list element: List of 1
##   ..$ : num [1:1769, 1:2] -75.7 -75.7 -75.7 -75.7 -75.7 ...
##   ..- attr(*, "class")= chr  "XY" "POLYGON" "sfg"
##   - attr(*, "sf_column")= chr "geometry"
##   - attr(*, "agr")= Factor w/ 3 levels "constant","aggregate",...: NA NA NA NA NA NA NA NA NA ...
##   ..- attr(*, "names")= chr  "OBJECTID" "ID_ESPACIA" "AREA_OFICI" "NOM_MUNICI" ...

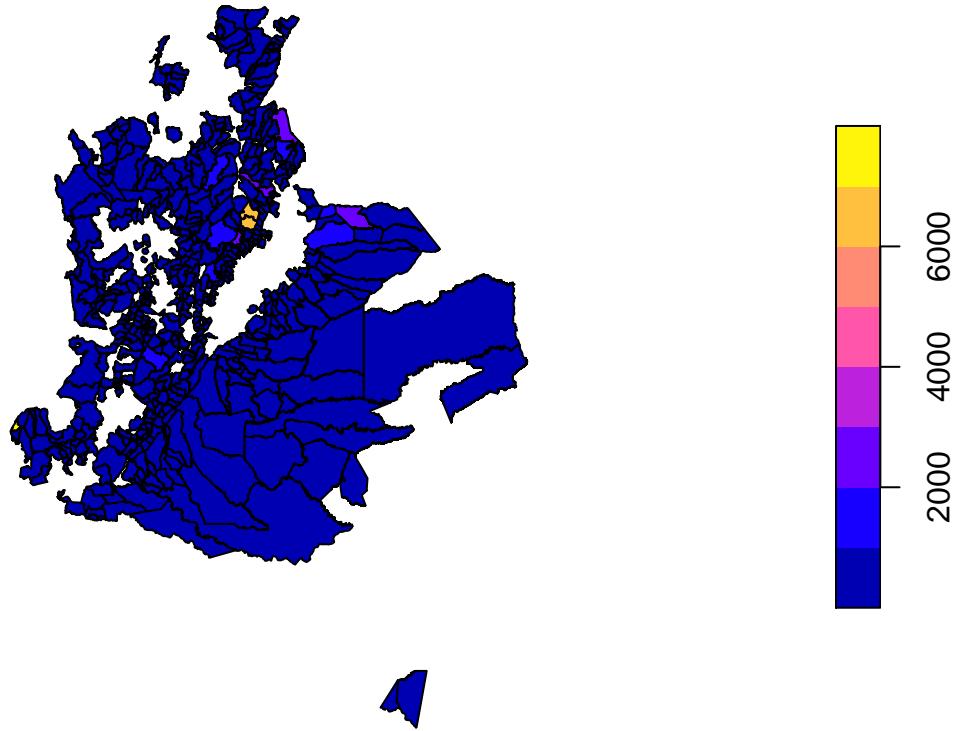
# Transformación ID espacial
shp <- shp %>%
  mutate(ID_ESPACIA = as.numeric(as.character(ID_ESPACIA)))
str(shp)

## Classes 'sf' and 'data.frame':  1123 obs. of  14 variables:
## $ OBJECTID : Factor w/ 1123 levels "1","10","100",...: 1 241 352 463 574 685 795 905 1016 2 ...
## $ ID_ESPACIA: num  23189 23570 23068 23580 23686 ...
## $ AREA_OFICI: num  644 819 1929 2062 470 ...
## $ NOM_MUNICI: Factor w/ 1043 levels "ABEJORRAL","ÁBREGO",...: 205 664 71 684 815 238 613 949 809 553
## $ COD_DEPTO : Factor w/ 34 levels "05","08","11",...: 10 10 10 10 10 22 22 22 22 22 ...
## $ FID_1      : Factor w/ 34 levels "0","1","10","11",...: 32 32 32 32 32 12 12 12 12 12 ...
## $ ID_ESPAC_1: Factor w/ 34 levels "05","08","11",...: 10 10 10 10 10 22 22 22 22 22 ...
## $ AREA_OFI_1: num  26506 26506 26506 26506 26506 ...
## $ NOMBRE_DPT: Factor w/ 34 levels "AMAZONAS","ANTIOQUIA",...: 15 15 15 15 15 30 30 30 30 30 ...
## $ area_ha    : num  63749 79500 193262 165073 47257 ...
## $ nombre     : Factor w/ 0 levels: NA NA NA NA NA NA NA NA ...
## $ Shape_Leng: num  1.68 1.51 2.07 2.54 1.53 ...
## $ Shape_Area: num  0.0524 0.0652 0.1586 0.1352 0.0388 ...
## $ geometry  :sfc_POLYGON of length 1123; first list element: List of 1
##   ..$ : num [1:1769, 1:2] -75.7 -75.7 -75.7 -75.7 -75.7 ...
##   ..- attr(*, "class")= chr  "XY" "POLYGON" "sfg"
##   - attr(*, "sf_column")= chr "geometry"
##   - attr(*, "agr")= Factor w/ 3 levels "constant","aggregate",...: NA NA NA NA NA NA NA NA NA ...
##   ..- attr(*, "names")= chr  "OBJECTID" "ID_ESPACIA" "AREA_OFICI" "NOM_MUNICI" ...

# Realizar unión entre una table y un shapefile
fnl <- inner_join(x = shp, y = tbl, by = c('ID_ESPACIA' = 'COD_MUNI'))
fnl %>%
  dplyr::select(PRODUCCION) %>%
  plot()

```

PRODUCCION



```
# Exportar el nuevo archivo espacial con las nuevas características  
st_write(obj = fnl, dsn = './_shp', layer = 'producción_cacao', driver = 'ESRI shapefile')  
  
## Writing layer `producción_cacao' to data source `./_shp' using driver `ESRI shapefile'  
## features: 526  
## fields: 22  
## geometry type: Polygon
```

3. Datos de ráster geográfico en R

Descripción de los datos empleados

Los datos utilizados a lo largo de la sección 3 provienen de WorldClim-Global Data <http://www.worldclim.org>. Para esta sección empleamos datos climáticos para el año 2050 con una resolución de 10 minutos (tamaño de píxel de 20Km*20Km). El modelo climático mediante el cual fueron construidos los datos es el modelo HadGEM2-CC bajo un escenario de alta probabilidad de ocurrencia. Para más información visitar <https://portal.enes.org/models/earthsystem-models/metoffice-hadley-centre/hadgem2-es>.

3.1 Manipulación de archivos tipo ráster

```

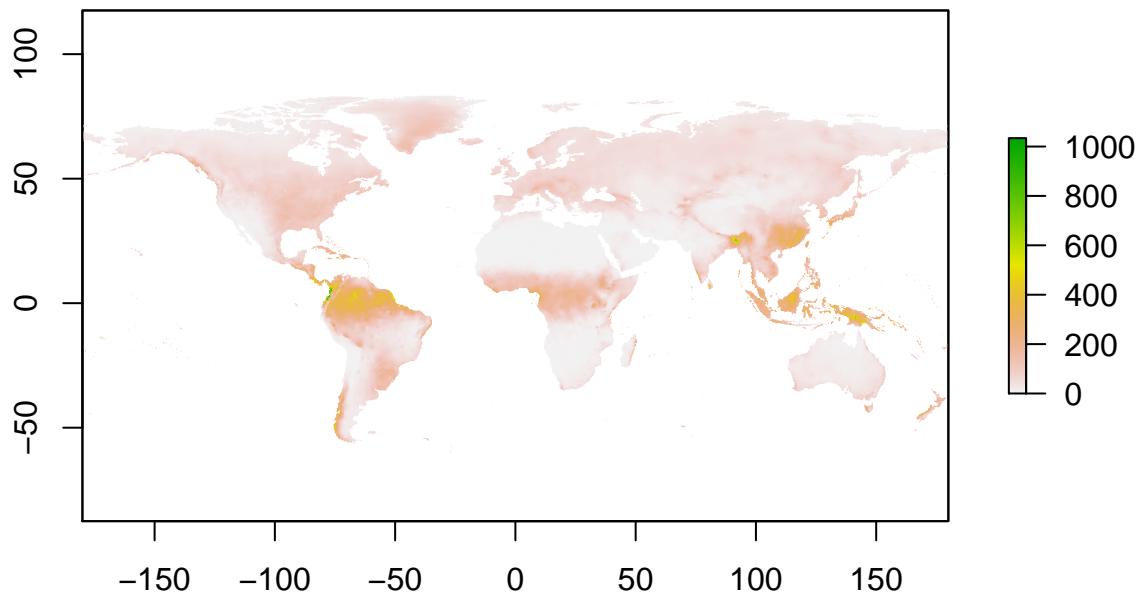
# Definir directorio de trabajo
# en Windows sería algo a esto setwd("C:/Users/gusahu/Google Drive/Workshop_socber/Section_III")
knitr:::opts_knit$set(root.dir = '/Users/gusahu/Google Drive/Workshop_GIS_with_R/Section_III')

# Cargar librerias
library(pacman)
pacman::p_load(raster, rgdal, maptools)

# Limpiar ambiente de trabajo
g <- gc(reset = TRUE)
rm(list = ls())
options(scipen = 999)

# Importar archivo raster de precipitación del mes de mayo
pr_mayo<- raster('./hg85pr50/hg85pr505.tif')
plot(pr_mayo)

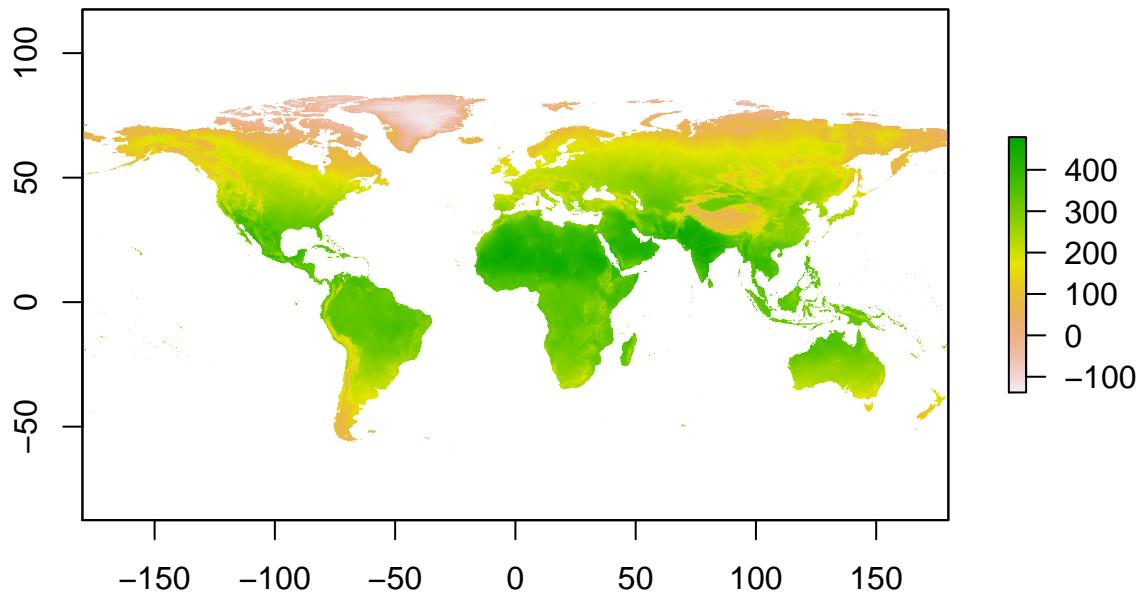
```



```

# Importar archivo raster de temperatura máxima del mes de mayo
tx_mayo<- raster('./hg85tx50/hg85tx505.tif')
plot(tx_mayo)

```



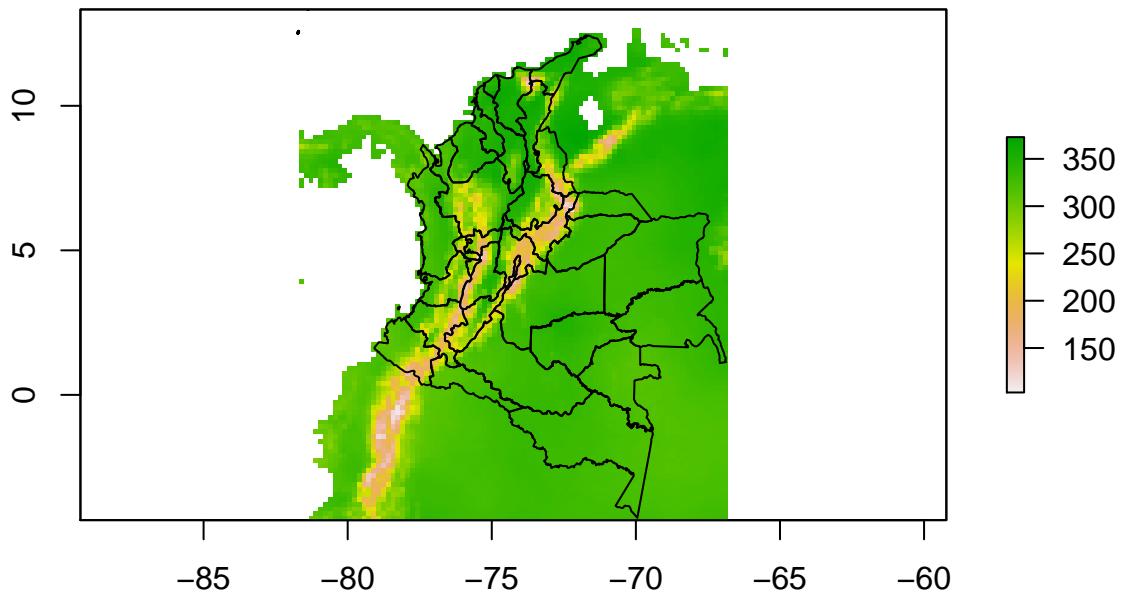
```

# Cargar capa de polígono
{col<- readOGR('./shp_dpto/deptos_wgs84.shp')}
# Cortar raster al área del polígono
tx_mayo_col<- crop(tx_mayo,col)
#Plotear raster
plot(tx_mayo_col, main = "Tmax. 2050 HadGEM CC RCP 8.5-Colombia (c*10)")
plot(col, add=TRUE)}

## OGR data source with driver: ESRI Shapefile
## Source: "/Users/gusahu/Google Drive/Workshop_GIS_with_R/Section_III/shp_dpto/deptos_wgs84.shp", layer
## with 33 features
## It has 5 fields

```

Tmax. 2050 HadGEM CC RCP 8.5–Colombia (c*10)



3.2 Operaciones entre rásteres - cálculo Indice de Aridez de Martonne (1926)

En esta sub-sección vamos a realizar algunas operaciones entre archivos ráster, y a través de estas operaciones realizaremos el cálculo del índice de aridez de Martonne ($I_m = P/(T_m + 10)$). Para tal fin, llevaremos a cabo un conjunto de operaciones puntuales.

Primer paso: cargar capas ráster.

```
# cargar raster por lotes
grids<-list.files("./hg85tn50", pattern = ".tif", full.names = TRUE) # temperatura mínima
grids2<-list.files("./hg85tx50", pattern = ".tif", full.names = TRUE) # temperatura máxima
grids3<-list.files("./hg85pr50", pattern = ".tif", full.names = TRUE) # precipitación

# crear raster stack
tn_stack<- stack(grids)
tx_stack<- stack(grids2)
pr_stack<- stack(grids3)
```

Segundo paso: operar rásteres

```
tn_promedio <- ((sum(tn_stack))/120) # temperatura mínima promedio anual
tx_promedio <- ((sum(tx_stack))/120) # temperatura máxima promedio anual
tm_promedio <- mean(tn_promedio, tx_promedio) # cálculo temperatura media anual
pr_total <- sum(pr_stack) # precipitación acumulada anual
```

Tercer paso: cargar una capa vectorial de polígono

```
col<- readOGR('./shp_dpto/deptos_wgs84.shp')
```

```
## OGR data source with driver: ESRI Shapefile
```

```
## Source: "/Users/gusahu/Google Drive/Workshop_GIS_with_R/Section_III/shp_dpto/deptos_wgs84.shp", layer=
```

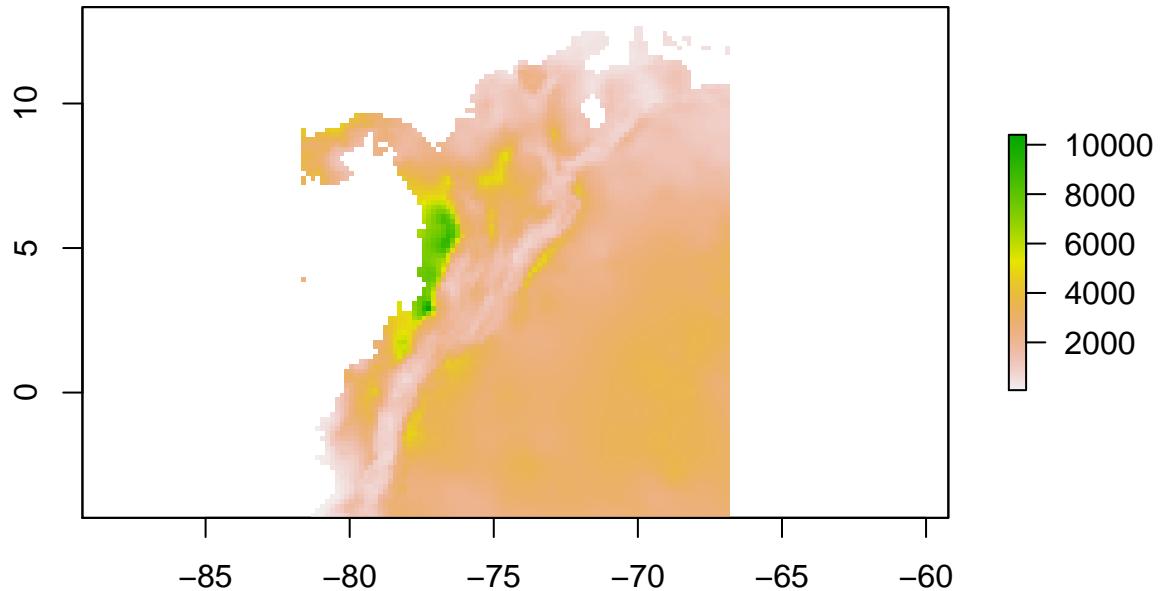
```
## with 33 features
```

```
## It has 5 fields
```

Cuarto paso: cortar raster al área del polígono

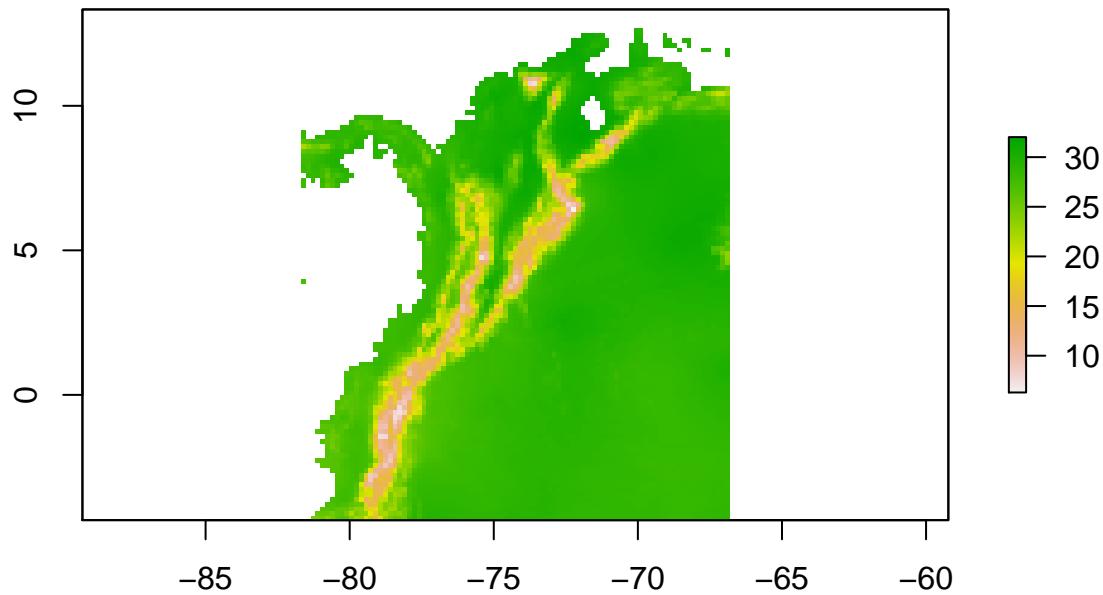
```
pr_total_col<- crop(pr_total,col)
```

```
plot(pr_total_col) # plot precipitación anual acumulada
```



```
tm_promedio_col<- crop(tm_promedio,col)
```

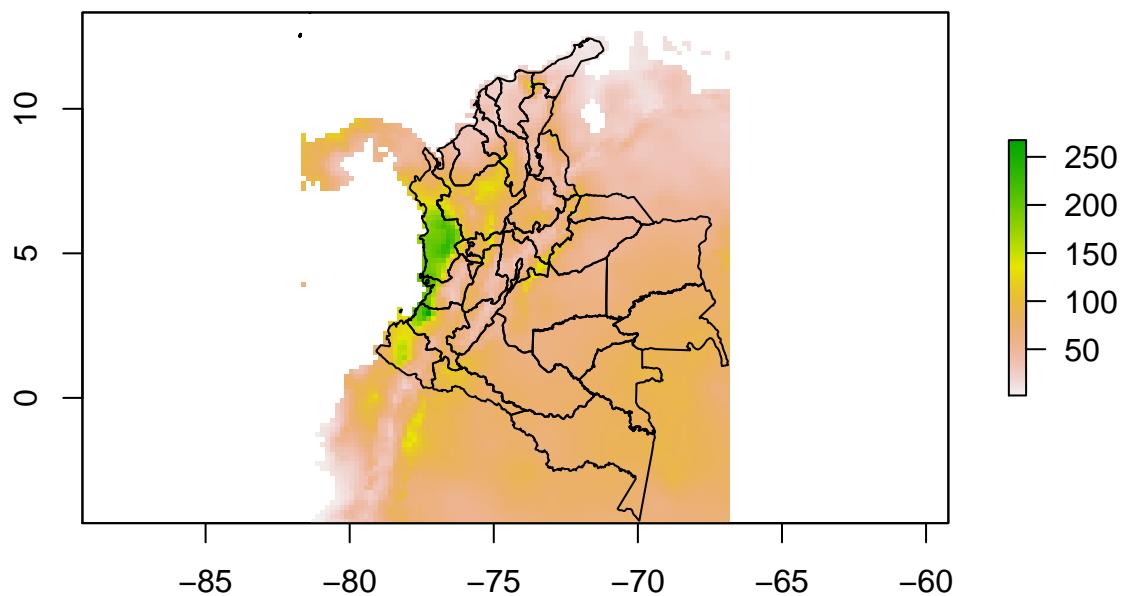
```
plot(tm_promedio_col) # plot temperatura media anual
```



Quinto paso: calcular Índice de Aridez de Martonne para el año 2050

```
ind_a <- pr_total_col / (tm_promedio_col + 10)
plot(ind_a, main = "Índice de Aridez de Martonne año 2050-Colombia")
plot(col, add=TRUE) # adicionar capa vectorial de polígono
```

Índice de Aridez de Martonne año 2050–Colombia



Sexto paso: exportar a *.tif

```
writeRaster(ind_a, filename = "Ia_2050", format="GTiff", overwrite=TRUE)
```