Datos espaciales

intro-R

Febrero 2018

Secciones

1. ¿Por qué R?

2. Raster vs. Vector

3. Raster

4. Vector

5. "sf" y "raster"

Secciones

1. ¿Por qué R?

2. Raster vs. Vector

3. Raster

4. Vector

5. "sf" y "raster"

¿Por qué R?

¿Por qué usar R para manejar datos espaciales?

¿Por qué R?

¿Por qué usar R para manejar datos espaciales?

- **repetibilidad**
- **soluciones "custom"**

¿Por qué R?

¿Por qué usar R para manejar datos espaciales?

- **repetibilidad**
- **soluciones "custom"**
- velocidad

¿Por qué R?

¿Por qué usar R para manejar datos espaciales?

- **repetibilidad**
- **soluciones "custom"**
- velocidad
- integración en nuestro flujo de trabajo

Secciones

1. ¿Por qué R?

2. Raster vs. Vector

3. Raster

4. Vector

5. "sf" y "raster"

Raster vs. Vector

Existen dos formas principales de almacenar información espacial: raster y vector.

- **Raster:** imágenes de una o más bandas. La unidad mínima de información es el *píxel*

Raster vs. Vector

Existen dos formas principales de almacenar información espacial: raster y vector.

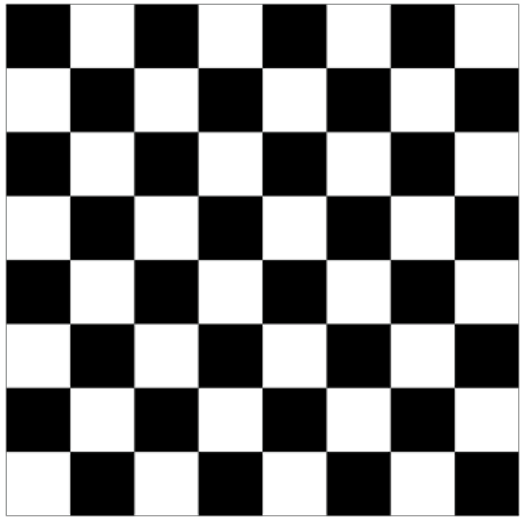
- **Raster:** imágenes de una o más bandas. La unidad mínima de información es el *píxel*
- **Vector:** coordenadas de cada objeto espacial + tabla de atributos. Varios tipos de objetos espaciales (*features*): puntos, líneas, polígonos, multipuntos...

La pregunta del millón

¿Qué formato es mejor?

La comparación

La comparación



La comparación

La comparación

En general

Los rásters servirán para información continua y los formatos vectoriales, para información discreta

Secciones

1. ¿Por qué R?

2. Raster vs. Vector

3. Raster

4. Vector

5. "sf" y "raster"

Estructura

Un archivo ráster es una imagen georreferenciada. Tiene varias propiedades:

- Extensión (coordenadas de las esquinas)
- Sistema de referencia (CRS)
- Resolución (tamaño de píxel)

Estructura

Un archivo ráster es una imagen georreferenciada. Tiene varias propiedades:

- Extensión (coordenadas de las esquinas)
- Sistema de referencia (CRS)
- Resolución (tamaño de píxel)

Formatos

Existen multitud de formatos ráster disponibles y utilizados en la actualidad. El más utilizado en la actualidad es el GeoTIFF (.tif)

Librería "raster"

```
madrid <- raster("ortofoto_madrid.bsq")
writeRaster(madrid, "ortofoto_madrid.tif", format = "GeoTIFF")
```

Librería "raster"

```
madrid <- raster("ortofoto_madrid.bsq")
writeRaster(madrid, "ortofoto_madrid.tif", format = "GeoTIFF")
```

Podemos conocer información de nuestro ráster de varias maneras:

```
crs(madrid)     # Sistema de referencia (CRS)
extent(madrid) # Extensión (coordenadas de las esquinas)
res(madrid)     # Resolución del ráster
nbands(madrid) # Número de bandas
plot(madrid)    # Ver una imagen del ráster
hist(madrid)    # Ver el histograma del ráster
madrid          # Un resumen de toda esta información
```

Bandas

Para importar un ráster de varias bandas usaremos la función "stack".
Podemos seleccionar cada banda como si fuese una columna en una matriz:

```
madrid <- stack("ortofoto_madrid.tif")
madrid[,1]      # Seleccionar solo la primera banda
plot(madrid)    # Veremos una imagen por cada banda
hist(madrid)    # Veremos un histograma por cada banda
```

Aritmética Ráster

Es muy frecuente hacer operaciones con uno o varios rásters:

```
madrid_doble <- madrid * 2
madrid_binario <- madrid > 100 # Binarización
```

Aritmética Ráster

Es muy frecuente hacer operaciones con uno o varios rásters:

```
madrid_doble <- madrid * 2
madrid_binario <- madrid > 100 # Binarización
```

También podemos usar calc, *sobre todo para varias bandas*:

```
mi_funcion <- function(x) {x[,1] + x[,2]^2 - 100}
madrid_modificado <- calc(madrid, fun = mi_funcion)
```

Secciones

1. ¿Por qué R?

2. Raster vs. Vector

3. Raster

4. Vector

5. "sf" y "raster"

Estructura

En el formato vectorial guardamos, fundamentalmente, dos cosas:

- Información espacial (*features*)
- Tabla de atributos

Estructura

En el formato vectorial guardamos, fundamentalmente, dos cosas:

- Información espacial (*features*)
- Tabla de atributos

Podemos tener también información adicional (metadatos):

- Sistema de coordenadas
- Índice de la tabla de atributos
- Índices espaciales

Formatos históricos

Existen muchos formatos vectoriales pero el más utilizado sin duda es el shapefile. Tiene tres ficheros *obligados*:

- .shp: la información espacial
- .dbf: la tabla de atributos en formato dBase
- .shx: los índices de la información espacial, para buscar y juntar rápidamente la información espacial y la tabla de atributos.

Formatos históricos

Existen muchos formatos vectoriales pero el más utilizado sin duda es el shapefile. Tiene tres ficheros *obligados*:

- .shp: la información espacial
- .dbf: la tabla de atributos en formato dBase
- .shx: los índices de la información espacial, para buscar y juntar rápidamente la información espacial y la tabla de atributos.

Además, puede tener multitud de ficheros opcionales, de los cuales los más frecuentes son:

- .prj: el sistema de coordenadas (CRS) y proyección
- .sbn y .sbx: índices espaciales.

Simple Features

En el año 2004 el Open Geospacial Consortium propone un estándar (norma ISO) para ordenar la información vectorial conocido como Simple Features.
La clave: toda la información *en un solo fichero*

Formatos modernos

El formato GeoPackage (.gpkg) implementa este estándar. Además, es FOSS

Librería "sf"

"sf" permite importar y manejar información espacial en R siguiendo el estándar Simple Features. Un objeto importado por sf será siempre un data frame.

```
setwd("../directorio_de_trabajo/datos/")
municipios <- st_read("municipios.shp")
st_write(municipios, "municipios.gpkg")
```

Librería "sf"

"sf" permite importar y manejar información espacial en R siguiendo el estándar Simple Features. Un objeto importado por sf será siempre un data frame.

```
setwd("../directorio_de_trabajo/datos/")
municipios <- st_read("municipios.shp")
st_write(municipios, "municipios.gpkg")
```

También podemos usar calc, *sobre todo para varias bandas*:

```
arboles <- st_as_sf(arboles,
                    coords = c("longitude", "latitude"),
                    crs = 4326)
```

"sf" + "tidyverse" = Gloria Bendita

Al ser todos los objetos importados por sf un data frame, podemos manipularlos con las herramientas del tidyverse:

```
municipios %>%
  select(habitantes, area) %>%
    mutate(densidad = habitantes/area) %>%
      ggplot() + geom_sf(aes(fill = densidad))
```

Operaciones espaciales

Podemos llevar a cabo multitud de operaciones espaciales con funciones de sf. Aquí van solo unas pocas:

```
st_geometry(municipios)
st_geometry(municipios) <- NULL
st_crs(municipios)
st_transform(municipios, 4326)
st_buffer(municipios, 100)
st_centroid(municipios)
st_intersects(municipios, zepas)
```

Secciones

1. ¿Por qué R?

2. Raster vs. Vector

3. Raster

4. Vector

5. "sf" y "raster"

A veces tenemos que combinar información vectorial y ráster. Por desgracia, las librerías sf y raster son, a día de hoy, mayoritariamente incompatibles.

A veces tenemos que combinar información vectorial y ráster. Por desgracia, las librerías sf y raster son, a día de hoy, mayoritariamente incompatibles.

Sí se puede

Es posible utilizar conjuntamente información vectorial y ráster en R utilizando el predecesor de sf, sp. No vamos a hacerlo en este curso.