Manexo de datos espaciais con R

CENTRO DE INVESTIGACIÓN Y TECNOLOGÍA MATEMÁTICA DE GALICIA

GALICIAN CENTRE FOR MATHEMATICAL RESEARCH AND TECHNOLOGY

Marta Rodríguez Barreiro

marta.rodriguez.barreiro@usc.es

UNIVERSIDADE DA CORUÑA

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Universidade de Vigo

CITMAGA CENTRO DE INVESTIGACIÓN E TECNOLOXÍA MATEMÁTICA DE GALICIA

itmati

Índice

Os datos espaciais

Un dato espacial é aquel que ten unha referencia xeográfica, é dicir, posúe un atributo que nos permite situalo nun mapa.

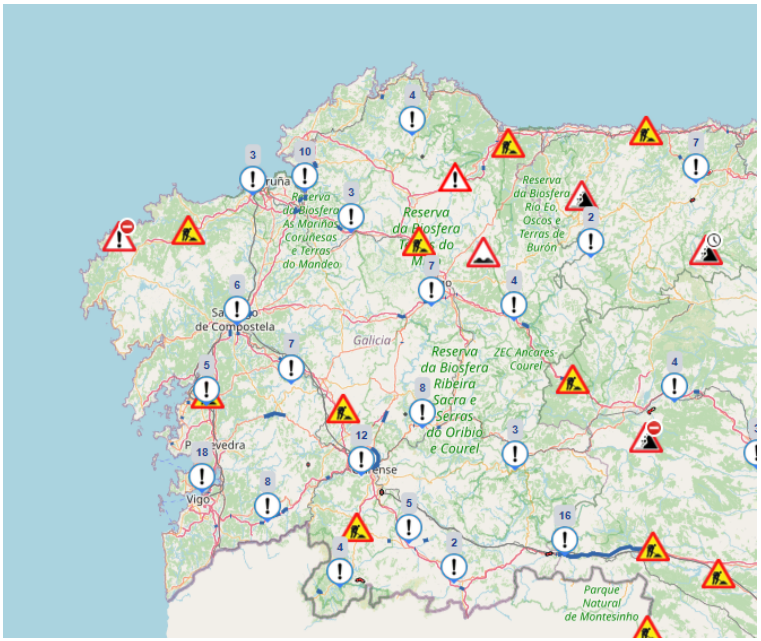

Figura: Captura de pantalla da páxina de incidencias da DGT.

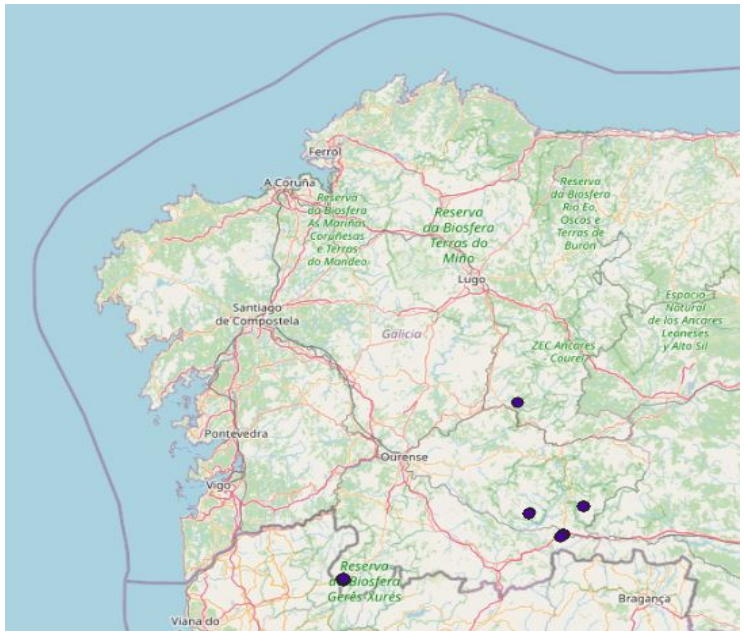

Figura: Descargas de auga realizadas por helicópteros de extinción.

A xeoestatística

Nunha análise estatística, é importante considerar a compoñente espacial asociada ós datos. Por exemplo, analizando a humidade do combustible nunha rexión, é esperable que os datos próximos no espacio sexan máis semellantes que aqueles que se sitúan máis lonxe.

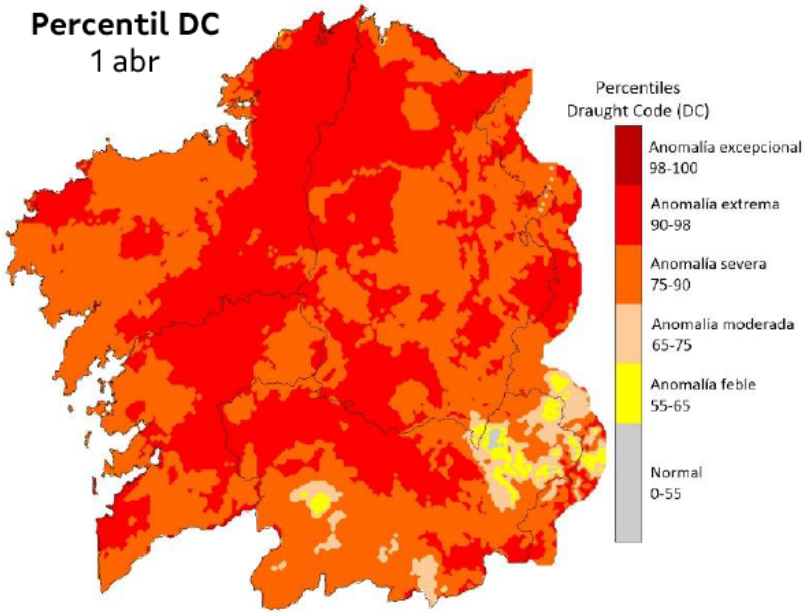

Figura: Imaxe extraída do boletín semanal de risco de incendio forestal da Xunta de Galicia.

A xeoestatística

A xeoestatística é a rama da estatística que se encarga no estudo de procesos espaciais, tendo en conta a presencia dunha posible dependencia espacial ou espacio-temporal nos datos que se analizan.
A incorporación da dimensión espacial dos datos facilita unha comprensión máis completa da realidade observada.
O emprego da xeoestatística para incorporar a dimensión espacial ou espacio-temporal incrementouse notablemente nas últimas décadas, xa que a natureza dos datos incorpora esta dimensión espacial de forma natural, especialmente en campos como:

Xeoloxía, hidroloxía, ecoloxía, ciencias medioambientais, meteoroloxía, epidemioloxía, xeografía, economía, astronomía, procesado de imaxes...

Índice

Tipos de datos espaciais

Existen dous tipos principais de datos espaciais:

- **Formato vectorial**. Representan obxectos **discretos** con límites claros. Empréganse coordenadas para definir as posicións espaciais "exactas" dos datos. Representan obxectos mediante xeometrías que están constituídas por pares de coordenadas, as máis habituais son puntos, liñas e polígonos.
- **Formato ráster**. Representan unha superficie **continua**. Trátase dunha grella regular que determina un conxunto de celas ou píxeles que teñen asociados un ou máis valores. O valor de cada cela do ráster representa o valor da área que representa, xa sexa o valor total da área, o valor medio, o valor puntual do centro da cela...

Datos vectoriais

- Os obxectos máis simples de datos vectoriais son os **puntos**. Confórmanse por un par de coordenadas (x, y) e poden conter ademais atributos asociados. Os puntos poden agruparse nun único obxecto multipunto.
- Da unión secuencial dos puntos xorden os obxectos tipo **liña**. Poden agruparse como un obxecto multiliña e tamén poden conter atributos asociados.
- As xeometrías máis complexas son os **polígonos**. Xorden da unión pechada de obxectos tipo liñas, e, como nos casos anteriores, poden conter atributos asociados e poden unirse dando lugar a obxectos multipolígono.

Datos vectoriais

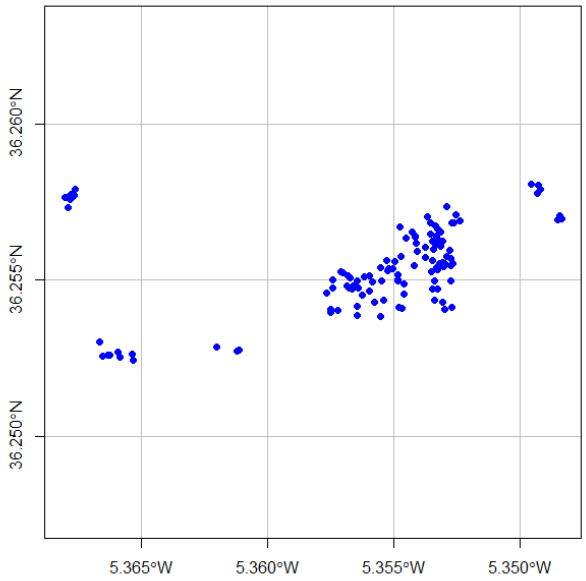

Figura: Descargas de auga de helicópteros.

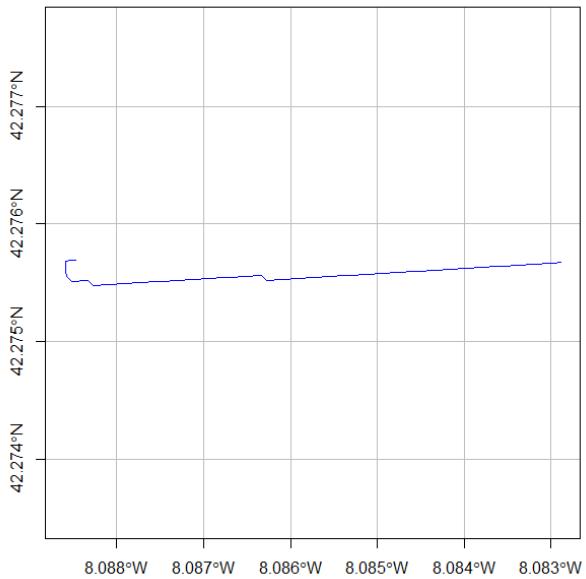

Figura: Anaco de pista forestal.

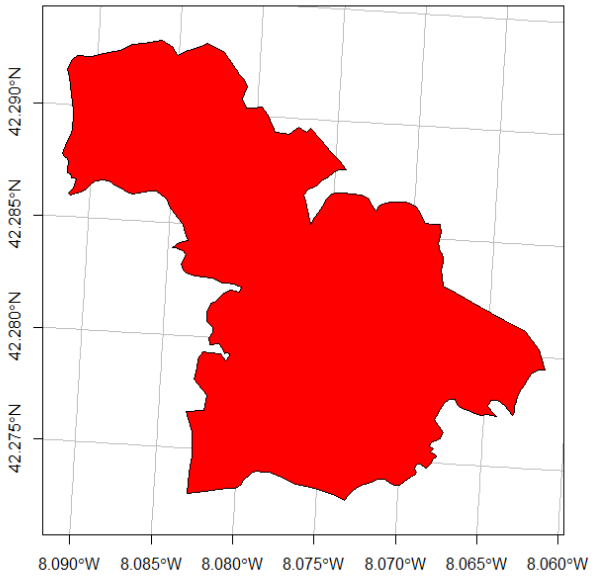

Figura: Perímetro dun incendio forestal.

A libraría *sf*

A libraría *sf* é a principal ferramenta de R para a manipulación de datos vectoriais. Emprega o modelo de xeometrías de **características simples**, que é un estándar para formas xeográficas vectoriais.
Os obxectos gárdanse en formato *data.frame*, cunha columna que contén información sobre as coordenadas.

```
Simple feature collection with 121 features and 6 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -5.36805 ymin: 36.25243 xmax: -5.34835 ymax: 36.25808
Geodetic CRS:  +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
First 10 features:
       id                  timestamp direction   speed liters altitud                  geometry
1  147106 2022-07-27T07:29:45.000+0000       176 33.3333   1200     850 POINT (-5.35327 36.25663)
2  147118 2022-07-27T07:57:07.000+0000        31 24.7222   1200     850 POINT (-5.36791 36.25765)
3  147124 2022-07-27T10:03:44.000+0000       326 23.3333   1200     850 POINT (-5.35325 36.25471)
4  147125 2022-07-27T10:06:27.000+0000       348 19.7222   1200     850 POINT (-5.35647 36.25416)
5  147126 2022-07-27T10:08:32.000+0000       211 20.2778   1200     850 POINT (-5.35252 36.25709)
6  147127 2022-07-27T10:10:45.000+0000       249 10.0000   1200     850  POINT (-5.3526 36.25682)
7  147128 2022-07-27T10:13:06.000+0000       225 10.5556   1200     850 POINT (-5.35268 36.25682)
8  147129 2022-07-27T10:15:33.000+0000       103 19.4444   1200     850 POINT (-5.35354 36.25682)
9  147130 2022-07-27T10:19:06.000+0000       103 36.9444   1200     850 POINT (-5.35721 36.25402)
10 147132 2022-07-27T10:22:59.000+0000       111 34.7222   1200     850 POINT (-5.35767 36.25457)
```

Figura: Obxecto espacial con descargas de auga dos helicópteros de extinción.

A libraría *sf*

Para crear obxectos xeométricos básicos (*simple feature geometry*) existen as funcións **st_point()**, **st_linestring()**, **st_polygon()**.

```
> point1 <- st_point(c(1,1))
> point1
POINT (1 1)
> class(point1)
[1] "XY"    "POINT" "sfg"
```

As xeometrías simples poden combinarse a obxectos múltiples (*simple feature collection*) coa función **st_sfc()**. Estes obxectos incorporan un sistema de coordenadas.

```
> point2 <- st_point(c(2,2))
> multipoint <- st_sfc(point1, point2)
> multipoint
Geometry set for 2 features
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 1 ymin: 1 xmax: 2 ymax: 2
CRS:           NA
POINT (1 1)
POINT (2 2)
> class(multipoint)
[1] "sfc_POINT" "sfc"
```

A libraría *sf*

Ademais, pódenselle engadir atributos á colección creando obxectos completos (*simple feature object*) coa función **st_sf()**.

```
> df <- data.frame(id=c("punto1", "punto2"), cantidade = c(5, 3))
> multipoint_sf <- st_sf(df, multipoint)
> multipoint_sf
Simple feature collection with 2 features and 2 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 1 ymin: 1 xmax: 2 ymax: 2
CRS:           NA
      id cantidade  multipoint
1 punto1         5 POINT (1 1)
2 punto2         3 POINT (2 2)
> class(multipoint_sf)
[1] "sf"         "data.frame"
```

Todas as funcións da libraría que operan sobre datos espaciais comezan por **st_**.
Para ver todas as funcións que están dispoñibles na libraría, pódese executar en R a orden:

```
> methods(class = "sf")
```

A libraría *sf*

Entre as funcións máis destacadas para a manipulación de obxectos espaciais podemos destacar:

- **st_read()**: lee un obxecto sf.
- **st_write()**: escribe un obxecto sf.
- **st_crs()**: permite obter ou establecer o sistema de referencia de coordenadas (CRS) do obxecto.
- **st_transform()**: transforma o obxecto a un novo sistema de referencia de coordenadas (CRS).
- **st_intersection()**: permite intersecar obxectos sf.
- **st_union()**: permite combinar diferentes obxectos sf.
- **st_coordinates()**: permite obter as coordenadas dun obxecto sf.
- **st_as_sf()**: permite converter outro obxecto a un obxecto sf.

A libraría *sf*

- Os obxectos sf poden representarse de xeito directo con *ggplot*.
- É compatible co universo *tidyverse*.
- Permite ler e escribir directamente de bases de datos como PostGIS.
- A función **st_as_sf()** permite converter un *data.frame* que contén coordenadas a un obxecto sf.

```
> d <- data.frame(ciudad = c("Madrid", "Barcelona", "Valencia", "Zaragoza"),
+                 long = c(-3.70256, 2.15899, -0.37966, -0.87734),
+                 lat = c(40.4165, 41.38879, 39.47391, 41.65606),
+                 habitantes = c(3506730, 1732066, 844424, 701299))
> class(d)
[1] "data.frame"
> d_sf <- st_as_sf(d, coords = c("long", "lat"), crs=st_crs(4326))
> class(d_sf)
[1] "sf"         "data.frame"
> d_sf
Simple feature collection with 4 features and 2 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -3.70256 ymin: 39.47391 xmax: 2.15899 ymax: 41.65606
Geodetic CRS:  WGS 84
     ciudad habitantes                    geometry
1    Madrid    3506730  POINT (-3.70256 40.4165)
2 Barcelona    1732066  POINT (2.15899 41.38879)
3  Valencia     844424 POINT (-0.37966 39.47391)
4  Zaragoza     701299 POINT (-0.87734 41.65606)
```

A libraría *sf*

```
> library(mapview)
> mapview(d_sf)
```

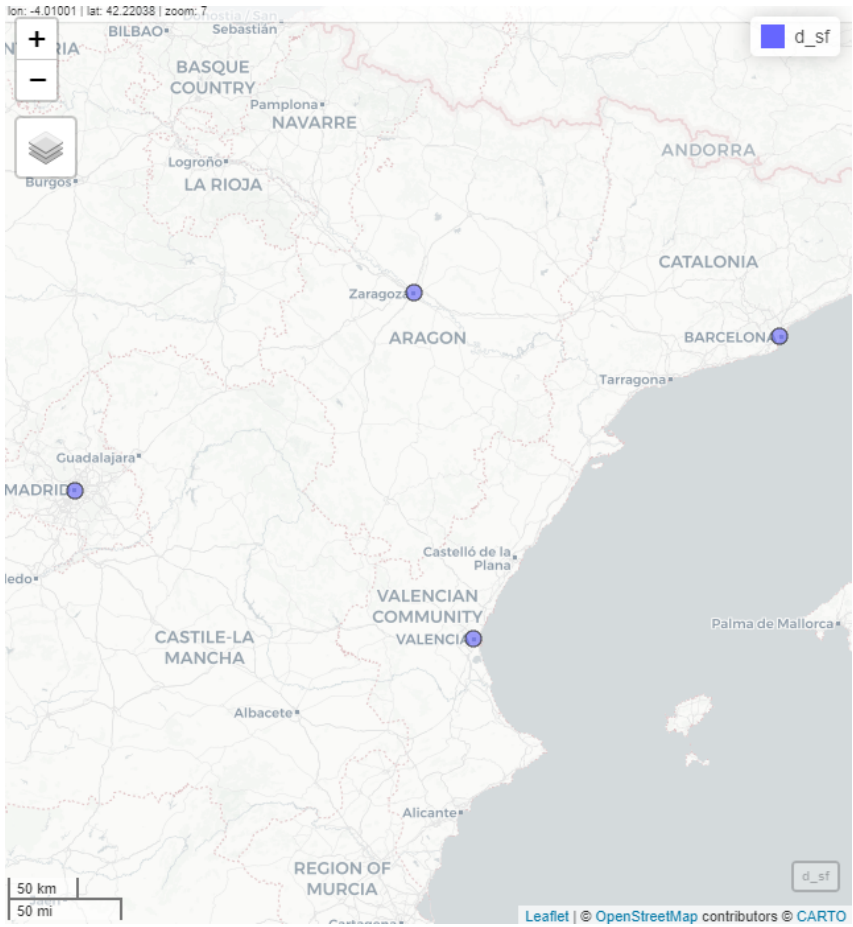

Figura: Exemplo de visualización dun obxecto sf nun mapa coa libraría mapview.

Tipos de datos vectoriais

O *Shapefile* é o formato de datos máis utilizado no software de análise espacial. Os ficheiros shapefile compóñense como mínimo de 3 arquivos:

- **.shp**: contén a xeometría dos datos.
- **.shx**: contén os índices dos obxectos que compoñen a capa.
- **.dbf**: contén a información dos atributos asociados ás entidades.

Pode estar formado doutros arquivos que conteñen información adicional, como a proxección, os metadatos...

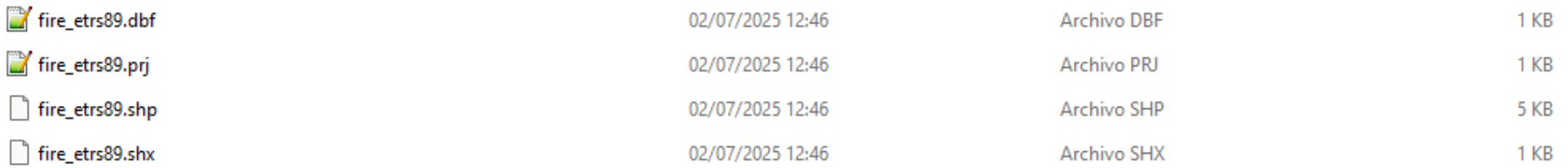

Figura: Exemplo dun ficheiro shapefile.

Pódense ler directamente os ficheiros shapefile coa función **st_read()** pasándolle como argumento o directorio do ficheiro.

Tipos de datos vectoriais

O *Geopackage* é un formato de datos espacial aberto, a súa extensión é *gpkg*. Admite datos vectoriais e ráster. Presenta varios beneficios fronte ó formato pechado shapefile:

- Maior capacidade de almacenamento (pode chegar ós 140 TB fronte ós 2 GB que admite shapefile).
- É un formato de datos aberto, o que o fai compatible con diferentes programas e plataformas.
- Componse dun único arquivo.

Figura: Exemplo dun ficheiro geopackage.

Os ficheiros geopackage pódense ler coa función **st_read()** pasándolle como argumento o directorio do ficheiro.

Tipos de datos vectoriais

O formato *GeoJSON* é o máis común para a representación de obxectos xeográficos na web. Está baseado en JavaScript, polo que é moi popular para publicar información a través de xeoportais ou servidores de mapas. Tamén é un formato aberto, garda as coordenadas como texto en JSON, e pode almacenar información de puntos, liñas ou mesmo polígonos.

Figura: Exemplo dun ficheiro geoJSON.

Os ficheiros geoJSON pódense ler coa función **st_read()** pasándolle como argumento o directorio do ficheiro.

Tipos de datos vectoriais


```
{
"type": "FeatureCollection",
"name": "obstaculostest3",
"crs": {
    "type": "name",
    "properties": {
        "name": "urn:ogc:def:crs:OGC:1.3:CRS84"
    }
},
"features": [{
        "type": "Feature",
        "properties": {
            "id": 1
        },
        "geometry": {
            "type": "MultiLineString",
            "coordinates": [[[-8.083717745712535, 42.275823863901429], [-8.083776763549238, 42.275711933521478]]
        }
    }
]
}
```


Figura: Contido dun ficheiro geopackage.

Datos ráster

Os datos ráster adoitan empregarse para representar unha superficie continua. Teñen unha estrutura máis compacta que os datos vectoriais, xa que non é necesario gardar as coordenadas de cada cela dada a súa forma regular. Un ráster está definido por:

- Un sistema de referencia de coordenadas (CRS).
- As coordenadas da orixe.
- O tamaño das celas en cada dirección (horizontal e vertical).
- O número de celas en cada dirección (horizontal e vertical).
- Un vector cos valores asociados a cada cela.

Datos ráster

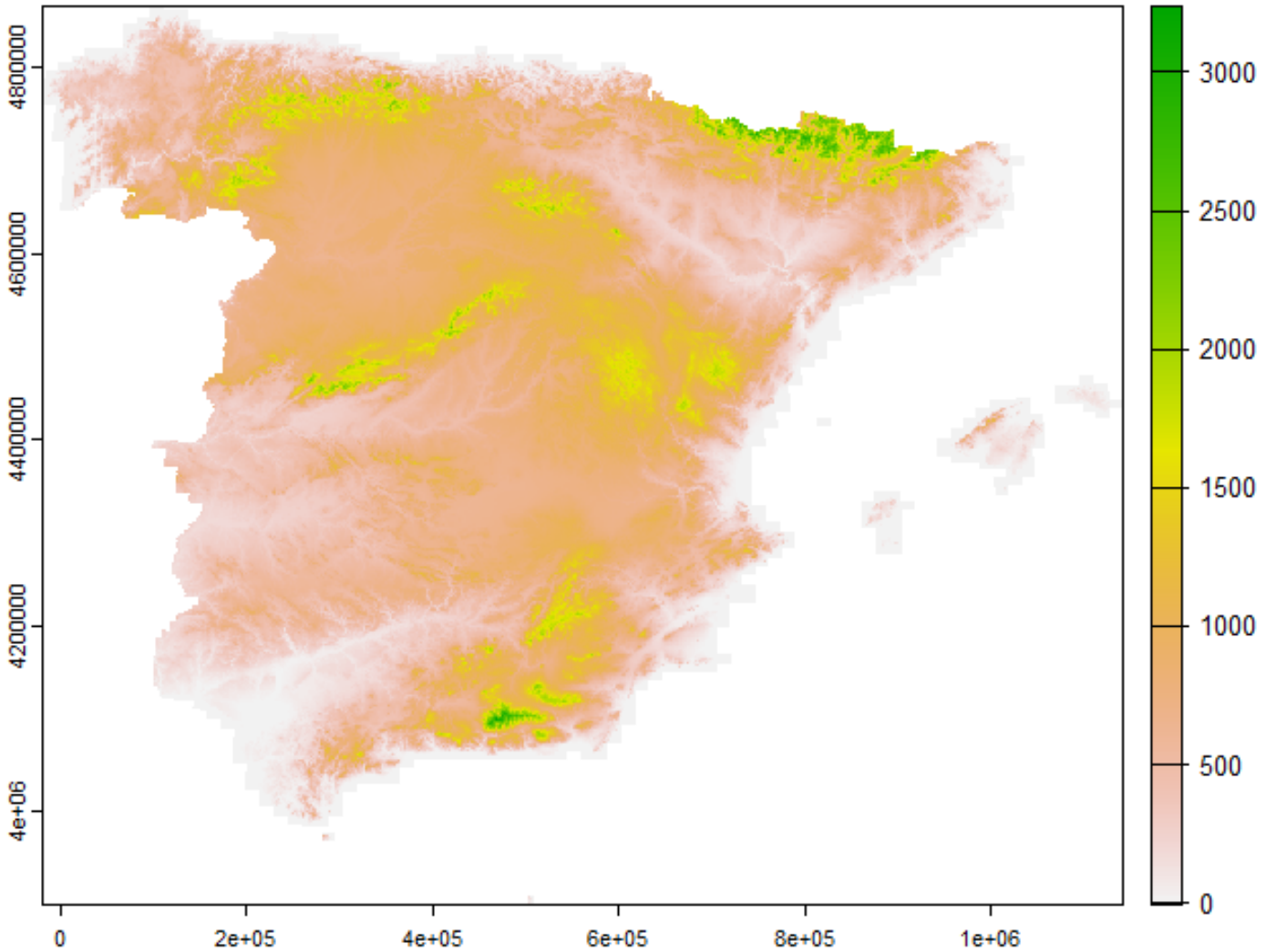

Figura: Ráster que contén o MDT de España.

A libraría terra

A libraría *terra* é a principal ferramenta de R para traballar con datos ráster na actualidade. Os obxectos ráster son representados coa clase *SpatRaster*. Tamén ten unha clase (*SpatVector*) para representar datos vectoriais.

```
class       : SpatRaster
dimensions  : 4824, 5801, 1  (nrow, ncol, nlyr)
resolution  : 200, 200  (x, y)
extent      : -19700, 1140500, 3901100, 4865900  (xmin, xmax, ymin, ymax)
coord. ref. : WGS 84 / UTM zone 30N (EPSG:32630)
source      : MDT_Esp_200.tif
name        : MDT_Esp_200
min value   :    -161.199
max value   :    3441.887
```

Figura: Obxecto SpatRaster co MDT de España.

A libraría terra

A función **rast()** permite crear obxectos *SpatRaster*.

```
> r <- rast(nrows=20, ncols=20, xmin=0, xmax=360)
> r
class       : SpatRaster
dimensions  : 20, 20, 1  (nrow, ncol, nlyr)
resolution  : 18, 9  (x, y)
extent      : 0, 360, -90, 90  (xmin, xmax, ymin, ymax)
coord. ref. : lon/lat WGS 84 (CRS84) (OGC:CRS84)
> class(r)
[1] "SpatRaster"
attr(,"package")
[1] "terra"
```

Se non se define o CRS de forma explícita establece a proxección WGS84 por defecto.

A libraría terra

Existen varias funcións para acceder e manipular o obxecto ráster:

- **nrow()**: indica o número de filas da grella.
- **ncol()**: indica o número de columnas da grella.
- **dim()**: indica as dimensións do obxecto.
- **ncell()**: indica o número de celas da grella.
- **nlyr()**: indica o número de capas do ráster.
- **values()**: permite establecer e acceder ós valores do ráster.
- **crs()**: permite establecer e acceder ó valor do sistema de coordenadas de referencia (CRS).

A libraría terra

Moitas funcións xenéricas como **min()**, **round()**, operacións matemáticas... poden usarse de xeito directo cos obxectos ráster.

```
> r <- rast(ncol = 10, nrow = 10,
+           xmin = -50, xmax = 50, ymin = 20, ymax = 60)
> values(r) <- 1:ncell(r)
> r
class       : SpatRaster
dimensions  : 10, 10, 1  (nrow, ncol, nlyr)
resolution  : 10, 4  (x, y)
extent      : -50, 50, 20, 60  (xmin, xmax, ymin, ymax)
coord. ref. : lon/lat WGS 84 (CRS84) (OGC:CRS84)
source(s)   : memory
name        : lyr.1
min value   :     1
max value   :   100
> r2 <- r * r
> r2
class       : SpatRaster
dimensions  : 10, 10, 1  (nrow, ncol, nlyr)
resolution  : 10, 4  (x, y)
extent      : -50, 50, 20, 60  (xmin, xmax, ymin, ymax)
coord. ref. : lon/lat WGS 84 (CRS84) (OGC:CRS84)
source(s)   : memory
name        : lyr.1
min value   :     1
max value   : 10000
```

A libraría terra

Tamén se poden crear obxectos multibanda, no que cada capa corresponde ó valor dun atributo diferente.

```
> s <- c(r, r2)
> s
class       : SpatRaster
dimensions  : 10, 10, 2  (nrow, ncol, nlyr)
resolution  : 10, 4  (x, y)
extent      : -50, 50, 20, 60  (xmin, xmax, ymin, ymax)
coord. ref. : lon/lat WGS 84 (CRS84) (OGC:CRS84)
source(s)   : memory
names       : lyr.1, lyr.1
min values  :     1,     1
max values  :   100, 10000
> class(s)
[1] "SpatRaster"
attr(,"package")
[1] "terra"
```

A libraría terra

Pódese acceder á cada capa do obxecto multibanda.

```
> min(s[[2]])
class       : SpatRaster
dimensions  : 10, 10, 1  (nrow, ncol, nlyr)
resolution  : 10, 4  (x, y)
extent      : -50, 50, 20, 60  (xmin, xmax, ymin, ymax)
coord. ref. : lon/lat WGS 84 (CRS84) (OGC:CRS84)
source(s)   : memory
name        :   min
min value   :     1
max value   : 10000
> max(s[[1]])
class       : SpatRaster
dimensions  : 10, 10, 1  (nrow, ncol, nlyr)
resolution  : 10, 4  (x, y)
extent      : -50, 50, 20, 60  (xmin, xmax, ymin, ymax)
coord. ref. : lon/lat WGS 84 (CRS84) (OGC:CRS84)
source(s)   : memory
name        : max
min value   :   1
max value   : 100
> plot(s[[1]])
```

A libraría terra

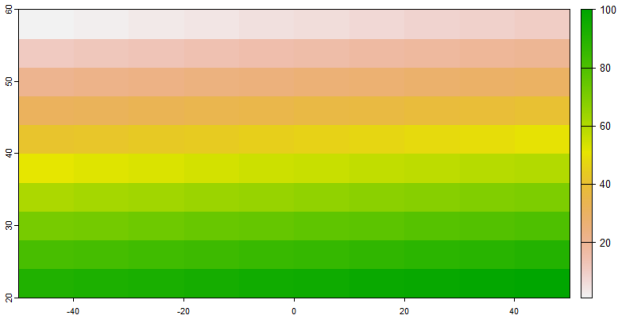

Figura: Representación da primeira banda do ráster (plot(s[[1]])).

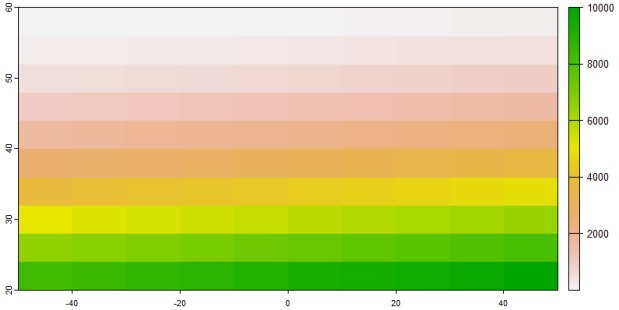

Figura: Representación da primeira banda do ráster (plot(s[[2]])).

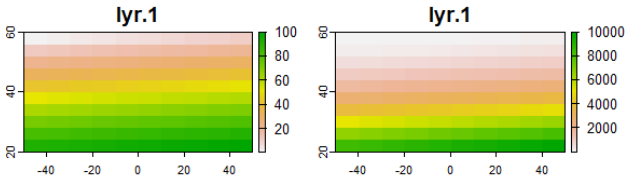

Figura: Representación completa do ráster (plot(s)).

Tipos de datos ráster

O formato ráster máis empregado é o *GeoTiff* ou *Tiff*. A diferenza entre os dous formatos é que o formato geotiff contén información engadida sobre a proxección da imaxe, que non contén o formato tiff.

Figura: Exemplo fun ficheiro tiff.

Os ficheiros ráster pódense ler coa función **rast()** pasándolle como argumento o directorio do ficheiro. En arquivos multibanda, pódese escoller ler soamente unha capa do ficheiro.

Índice

Paquetes de R para a obtención de datos espaciais

Existen diversas librarías de R que permiten o acceso sinxelo a diferentes bases de datos que se poden usar para obter datos espaciais como as fronteiras administrativas de diferentes rexións, datos climatolóxicos ou datos de OpenStreetMap, por exemplo.

- Fronteiras administrativas: **rnaturalearth**, **tidycensus**, **tigris**, **mapSpain**...
- Datos climatolóxicos: **geodata**, **climateR**, **chirps**, **elevatr**, **meteoland**...
- Datos de OpenStreetMap: **osmdata**

Recursos online

- **GeoServer**: é un servidor aberto para compartir data xeoespacial, `https://geoserver.org/`.
- **IDEE**: é o servidor oficial para acceder a datos e servizos xeográficos de España, `https://www.idee.es/`.
- **IDE**: é o servidor oficial para acceder a datos do Ministerio para a Transición Ecolóxica e o Reto Demográfico, `https://www.miteco.gob.es/es/cartografia-y-sig/ide.html`.
- **CNIG**: é unha plataforma do Instituto Xeográfico Nacional e o O. A. Centro Nacional de Información Xeográfica que pon a disposición da cidadanía información xeográfica de España, `https://centrodedescargas.cnig.es/CentroDescargas/catalogo`.

Datos satelitales

Os datos satelitales son información capturada por sensores orbitais que permiten observar e analizar diferentes partes da superficie terrestre.

- **Copernicus Data Space Ecosystem (ESA)**: é o programa de observación terrestre da UE, permite acceder a datos dos satélite Sentinel, `https://dataspace.copernicus.eu/data-availability`.
- **NASA Earthdata Search**: permite acceder a diversos conxuntos de datos e imaxes da Terra da NASA, `https://www.earthdata.nasa.gov/data/catalog`.
- **Meteosat**: satélites meteorolóxicos operados pola Organización Europea para a Explotación de Satélites Meteorolóxicos (EUMETSAT), `https://user.eumetsat.int/data-access/data-centre`.

Google Earth Engine

É unha plataforma baseada na nube que permite ós usuarios visualizar e analizar imaxes de satélite da Terra.
En R, a libraría **rgee** permite descargar imaxes de forma rápida e sinxela con GEE. En realidade, esta libraría fai uso da libraría *reticulate* para conectar con Python, quen ten a conexión nativa coa API de GEE.

A solución que nós adoptamos para descargar e manipular de forma sinxela as imaxes de satélite foi desenvolver un *script* en Python, empregando a API que GEE ten para esta linguaxe (chamada **ee**), e executar desde o código de R ese *script* de Python.

Esta API permite buscar, filtrar (por exemplo, pola cantidade de nubes que se atopan na imaxe) e exportar imaxes nas datas e na área que queiramos.

API de Python para Google Earth Engine

Para empezar, hai que iniciar un proxecto e autenticarse utilizando as credenciais.

```
credentials = ee.ServiceAccountCredentials(service_account, credenciales)
ee.Initialize(credentials)
```

A continuación pódese acceder ó catálogo de imaxes. Pódese escoller entre *Features*, que son obxectos xeométricos con atributes; *Images*, que son como os anteriores pero poden incluír diversas bandas con diferente información; ou *Collections*, que son conxuntos dos elementos anteriores. Por exemplo, co seguinte código accedemos a unha colección de imaxes de Copernicus filtrando polas datas e a área que nos interesa. Ademais, engadimos un filtro para evitar as imaxes que teñen moitas nubes.

```
CollectionSentinel = ee.ImageCollection('COPERNICUS/S2_SR_HARMONIZED').filterDate(start_date, end_date).filterMetadata(
    'CLOUD_COVERAGE_ASSESSMENT', 'less_than', filtnub).filterBounds(area_estudio)
```

Recursos na web para os datos espaciais

- https://rubenfcasal.github.io/estadistica_espacial/
- https://cengel.github.io/R-spatial/
- https://www.paulamoraga.com/book-spatial/index.html
- https://rspatialdata.github.io/
- https://rpubs.com/RICARDOROBAYO95/813711

Manexo de datos espaciais con R

CENTRO DE INVESTIGACIÓN Y TECNOLOGÍA MATEMÁTICA DE GALICIA

GALICIAN CENTRE FOR MATHEMATICAL RESEARCH AND TECHNOLOGY

Marta Rodríguez Barreiro

marta.rodriguez.barreiro@usc.es

UNIVERSIDADE DA CORUÑA

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Universidade de Vigo

CITMAGA CENTRO DE INVESTIGACIÓN E TECNOLOXÍA MATEMÁTICA DE GALICIA