

# mapSpain: Mapas de España en R

Grupo de Usuarios de R de Madrid

Diego Hernangómez

2021-11-25

# Bienvenidos a mapSpain

Introducción

Diccionarios

Límites políticos

Imágenes

Otros recursos

# Motivación

**mapSpain** facilita la creación de mapas de los diferentes niveles administrativos de España.

Además, proporciona también la posibilidad de usar imágenes de servicios WMS/WMTS de manera estática (como imagen georeferenciada) o dinámica (en mapas leaflet).

Adicionalmente, **mapSpain** dispone de funciones que permiten normalizar nombres de las CCAA y provincias, lo que facilita el proceso de manipulación y transformación de datos (no necesariamente espaciales).

Las **fuentes de información** empleadas en **mapSpain** son:

- **GISCO** (Eurostat) via el paquete **giscoR**.
- **Instituto Geográfico Nacional** (IGN)
- Distintos organismos públicos de España que proporcionan servicios de teselas WMTS/WMS (<https://www.idee.es/web/idee/segun-tipo-de-servicio>).

Los objetos resultantes se proporcionan en formato **sf** o **SpatRaster(terra)**.

Página web: <https://ropenspain.github.io/mapSpain/>



# Instalación

## CRAN

```
install.packages("mapSpain", dependencies = TRUE)
```

## Dev version

Usando el r-universe:

```
# Enable this universe
options(repos = c(
  ropenspain = "https://ropenspain.r-universe.dev",
  CRAN = "https://cloud.r-project.org"
))

install.packages("mapSpain", dependencies = TRUE)
```

## Remotes

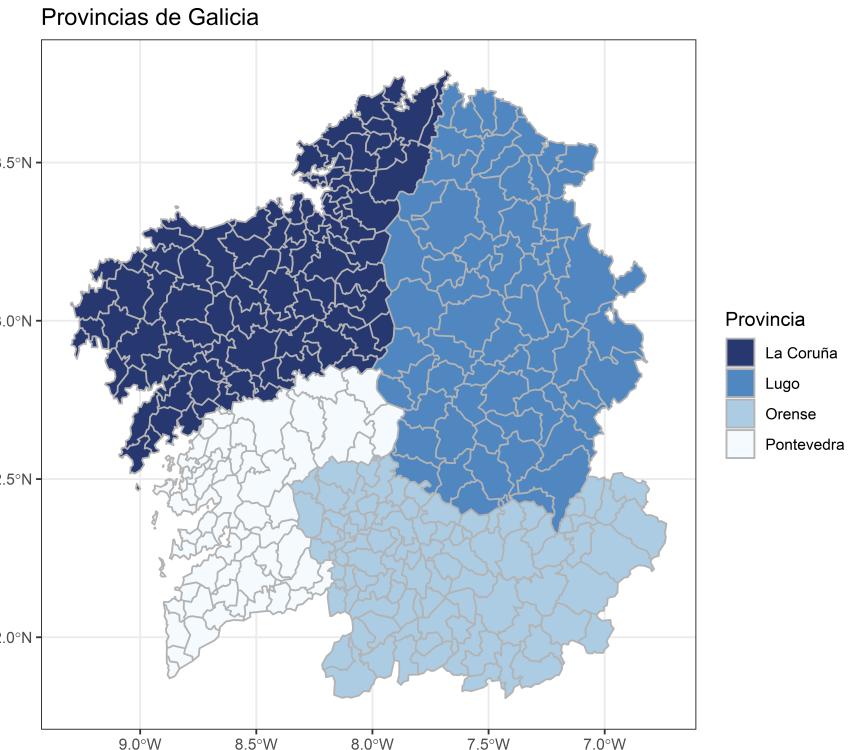
```
remotes::install_github("rOpenSpain/mapSpain", dependencies = TRUE)
```

# Un ejemplo rápido

```
library(mapSpain)
library(tidyverse)

galicia <- esp_get_munic_siane(region = "Galicia") %>%
  # Homogenize labels
  mutate(
    Provincia = esp_dict_translate(ine.prov.name, "es")
  )

ggplot(galicia) +
  geom_sf(aes(fill = Provincia),
          color = "grey70"
  ) +
  labs(title = "Provincias de Galicia") +
  scale_fill_discrete(
    type =
      hcl.colors(4, "Blues")
  ) +
  theme_bw()
```



# Un ejemplo rápido

Si exploramos el dataset:

```
library(reactable)

reactable(galicia,
  searchable = TRUE, striped = TRUE,
  filterable = TRUE, height = 350
)
```

codauto ame	ine.ccaa.n ame	cpro	ine.prov.na me	cmun	name	LAU_CODE	geometry	Provincia
12	Galicia	15	Coruña, A	001	Abegondo	15001	[object Object]	La Coruña
12	Galicia	15	Coruña, A	002	Ames	15002	[object Object]	La Coruña
12	Galicia	15	Coruña, A	002	Ames	15002	[object Object]	La Coruña
1–10 of 313 rows						Previous		1 2 3 4 5 ... 32 Next

# Comparando mapSpain con otras alternativas

Comparamos ahora **mapSpain** con otros dos paquetes que proporcionan objetos `sf` de distintos países:

```
# rnatural earth
library(rnaturalearth)

esp_rnat <- ne_countries("large", country = "Spain", returnclass = "sf")

# gadm
library(GADMTools)
esp_gadm <- gadm_sf_loadCountries("ESP")

# MapSpain

esp_mapSpain <- esp_get_country(epsg = 4326)

# Prepara el plot

esp_all <- bind_rows(
  esp_rnat,
  esp_gadm$sf,
  esp_mapSpain
)
esp_all$source <- c("rnaturalearth", "gadm", "mapSpain")
```

# Comparando mapSpain con otras alternativas

## rnaturalearth

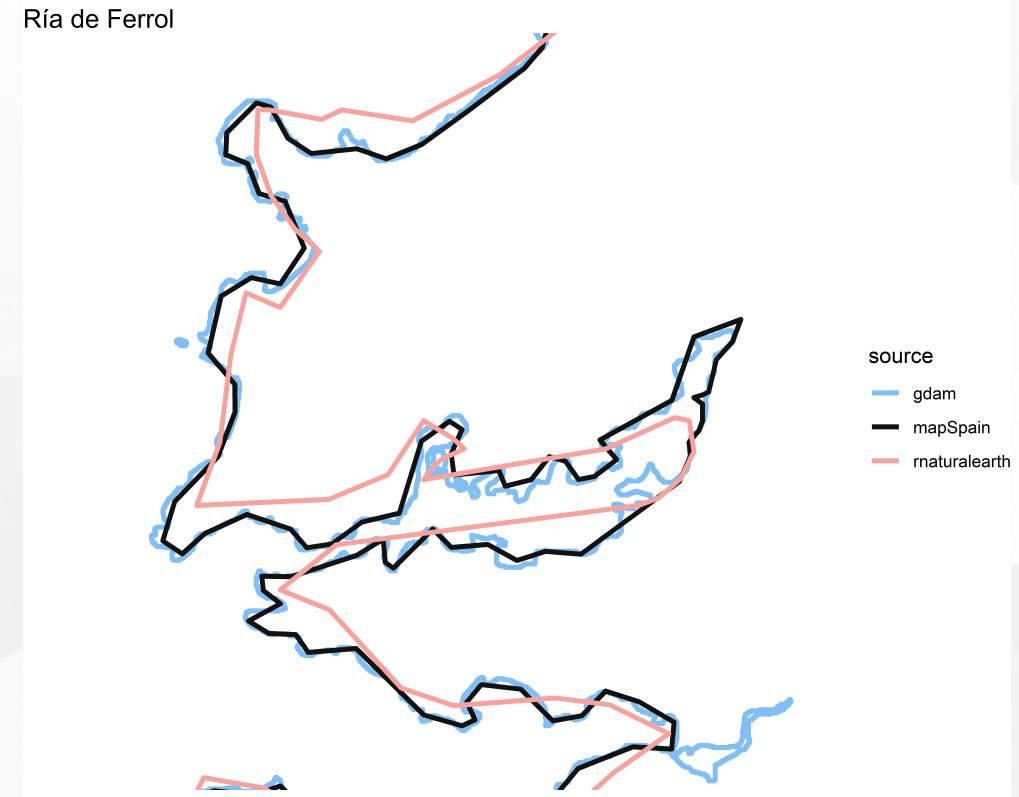
No capta bien el contorno.

## GADM

Proporciona datos muy detallados.

## mapSpain

Resultados satisfactorios.



# Almacenamiento

**mapSpain** es un paquete API que usa recursos web. El comportamiento por defecto consiste en descargar archivos al directorio temporal `tempdir()` para su uso posterior durante la sesión.

La función `esp_set_cache_dir()` permite modificar este comportamiento, estableciendo un directorio de descarga específico para el usuario. Para hacer esta configuración persistente se puede emplear el parámetro `install = TRUE`

```
esp_set_cache_dir("~/R/mapslib/mapSpain", install = TRUE, verbose = TRUE)

#> mapSpain cache dir is: C:/Users/xxxxx/Documents/R/mapslib/mapSpain

munic <- esp_get_munic_siane(verbose = TRUE)

#> Cache dir is C:/Users/xxxxx/Documents/R/mapslib/mapSpain
#> Downloading file from https://github.com/rOpenSpain/mapSpain/raw/sianedata/dist/se89_3_admin_muni_a_x.gpkg

#> See https://github.com/rOpenSpain/mapSpain/tree/sianedata/ for more info
#> trying URL 'https://github.com/rOpenSpain/mapSpain/raw/sianedata/dist/se89_3_admin_muni_a_x.gpkg'
#> Content type 'application/octet-stream' length 5570560 bytes (5.3 MB)
#> downloaded 5.3 MB

#> Download succesful
#> Reading from local file #> C:/Users/xxxxx/Documents/R/mapslib/mapSpain/se89_3_admin_muni_a_x.gpkg
#> 5.3 Mb
```

# Diccionario

# Funciones para trabajar con strings

**mapSpain** proporciona dos funciones relacionadas para trabajar con textos y códigos:

- `esp_dict_region_code()` convierte textos en códigos de CCAA y provincias. Esquemas de codificación soportados:
  - ISO2
  - NUTS
  - INE (codauto y cpro)
- `esp_dict_translate()` traduce textos a diferentes idiomas:
  - Castellano
  - Inglés
  - Catalán
  - Gallego
  - Vasco

Estas funciones pueden ser de utilidad en ámbitos más amplios que necesiten homogeneizar códigos de CCAA y Provincias (Datos COVID ISCI, etc).

# Funciones para trabajar con strings

## esp\_dict\_region\_code()

```
vals <- c("Errioxa", "Coruna", "Gerona", "Madrid")
esp_dict_region_code(vals, destination = "nuts")

#> [1] "ES23"   "ES111"  "ES512" "ES30"

esp_dict_region_code(vals, destination = "cpro")

#> [1] "26"    "15"    "17"    "28"

esp_dict_region_code(vals, destination = "iso2")

#> [1] "ES-RI"  "ES-C"   "ES-GI"  "ES-MD"

# Desde ISO a otros códigos

iso2vals <- c("ES-M", "ES-S", "ES-SG")
esp_dict_region_code(iso2vals, origin = "iso2")

#> [1] "Madrid"    "Cantabria" "Segovia"
```

# Funciones para trabajar con strings

## esp\_dict\_region\_code()

```
iso2vals <- c("ES-GA", "ES-CT", "ES-PV")

esp_dict_region_code(iso2vals,
  origin = "iso2",
  destination = "nuts"
)

#> [1] "ES11" "ES51" "ES21"

# Soporta diferentes niveles
valsmix <- c("Centro", "Andalucía", "Seville", "Menorca")
esp_dict_region_code(valsmix, destination = "nuts")

#> [1] "ES4"     "ES61"    "ES618"   "ES533"

esp_dict_region_code(c("Murcia", "Las Palmas", "Aragón"),
  destination = "iso2"
)

#> [1] "ES-MC"  "ES-GC"  "ES-AR"
```

# Funciones para trabajar con strings

## esp\_dict\_translate()

```
vals <- c(  
  "La Rioja", "Sevilla", "Madrid",  
  "Jaen", "Orense", "Baleares"  
)  
  
esp_dict_translate(vals, lang = "en")  
  
#> [1] "La Rioja"          "Seville"         "Madrid"         "Jaén"  
#> [5] "Ourense"          "Balearic Islands"  
  
esp_dict_translate(vals, lang = "es")  
  
#> [1] "La Rioja" "Sevilla"   "Madrid"    "Jaén"      "Orense"    "Baleares"  
  
esp_dict_translate(vals, lang = "ca")  
  
#> [1] "La Rioja"      "Sevilla"       "Madrid"        "Jaén"  
#> [5] "Ourense"       "Illes Balears"
```

# Funciones para trabajar con strings

## esp\_dict\_translate()

```
vals <- c(  
  "La Rioja", "Sevilla", "Madrid",  
  "Jaen", "Orense", "Baleares"  
)  
  
esp_dict_translate(vals, lang = "eu")  
  
#> [1] "Errioxa"          "Sevilla"         "Madril"        "Jaén"  
#> [5] "Ourense"         "Balear Uharteak"  
  
esp_dict_translate(vals, lang = "ga")  
  
#> [1] "A Rioxa"          "Sevilla"         "Madrid"        "Xaén"  
#> [5] "Ourense"         "Illas Baleares"
```

# Límites políticos

# Límites políticos

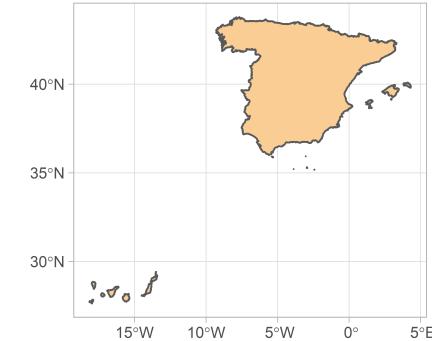
`mapSpain` contiene un set de funciones que permiten obtener límites políticos a diferentes niveles:

- Todo el país
- **NUTS** (Eurostat): Clasificación estadística de Eurostat. Niveles 0 (país), 1, 2 (CCAA) y 3.
- CCAA
- Provincias
- Municipios

Para CCAA, Provincias y Municipios hay dos versiones: `esp_get_xxxx()` (fuente: GISCO) y `esp_get_xxxx_siane()` (fuente: IGN).

La información se proporciona en diferentes proyecciones y niveles de resolución.

```
esp <- esp_get_country(moveCAN = FALSE)  
  
ggplot(esp) +  
  geom_sf(fill = "#f9cd94") +  
  theme_light()
```



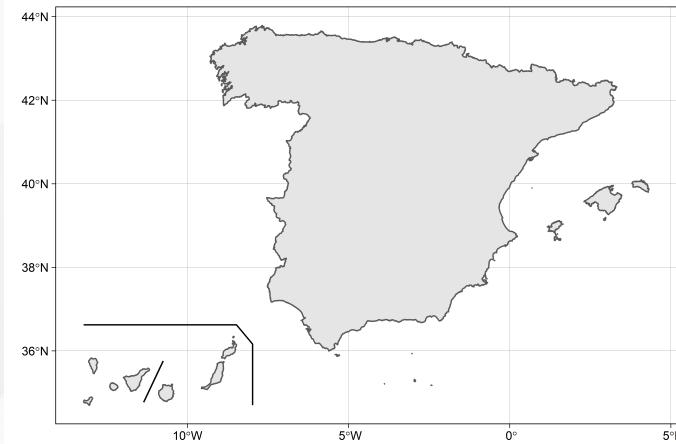
# Límites políticos

## El caso Canarias

Por defecto, **mapSpain** "desplaza" Canarias para una mejor visualización en la mayoría de sus funciones. Este comportamiento se puede desactivar usando `moveCAN = FALSE`(ver anterior ejemplo).

Proporcionamos funciones adicionales que permiten representar líneas alrededor de la inserción del mapa ([ejemplos](#)).

```
esp_can <- esp_get_country()  
can_prov <- esp_get_can_provinces()  
can_box <- esp_get_can_box()  
  
ggplot(esp_can) +  
  geom_sf() +  
  geom_sf(data = can_prov) +  
  geom_sf(data = can_box) +  
  theme_linedraw()
```



Cuando se trabaja con imágenes, mapas interactivos o se desean realizar análisis espaciales, se debe usar `moveCAN = FALSE`

# Límites políticos

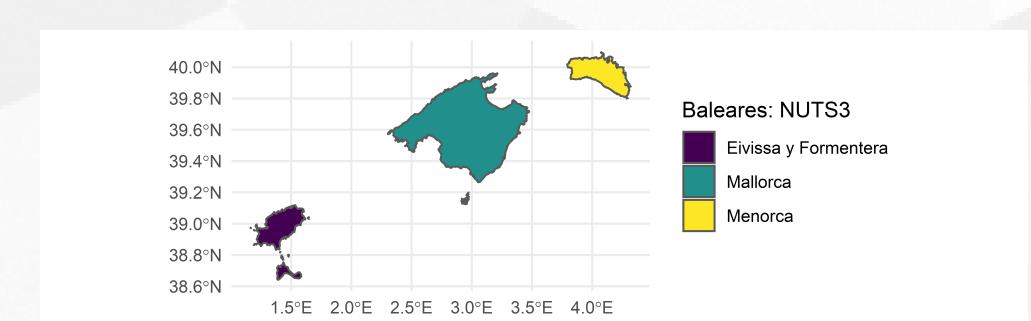
## NUTS

```
nuts1 <- esp_get_nuts(resolution = 20, epsg = 3035, nuts_l  
ggplot(nuts1) +  
  geom_sf() +  
  theme_linedraw() +  
  labs(title = "NUTS1: Baja Resolución")
```



```
# Baleares NUTS3  
nuts3_baleares <- c("ES531", "ES532", "ES533")  
paste(esp_dict_region_code(nuts3_baleares, "nuts"), collapse = "")  
#> [1] "Eivissa y Formentera, Mallorca, Menorca"
```

```
nuts3_sf <- esp_get_nuts(region = nuts3_baleares)  
ggplot(nuts3_sf) +  
  geom_sf(aes(fill = NAME_LATN)) +  
  labs(fill = "Baleares: NUTS3") +  
  scale_fill_viridis_d() +  
  theme_minimal()
```

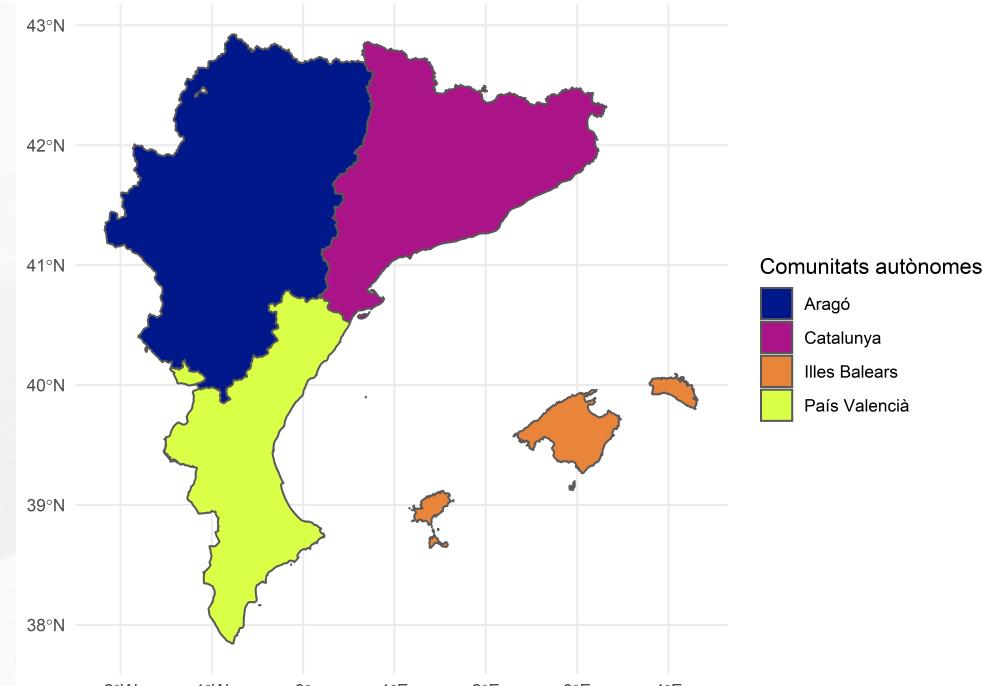


# Límites políticos CCAA

```
ccaa <- esp_get_ccaa(ccaa = c(
  "Catalunya",
  "Comunidad Valenciana",
  "Aragón",
  "Baleares"
))

ccaa <- ccaa %>% mutate(
  ccaa_cat = esp_dict_translate(ccaa$ine.ccaa.name, "ca")
)

ggplot(ccaa) +
  geom_sf(aes(fill = ccaa_cat)) +
  labs(fill = "Comunitats autònomes") +
  theme_minimal() +
  scale_fill_discrete(type = hcl.colors(4, "Plasma"))
```



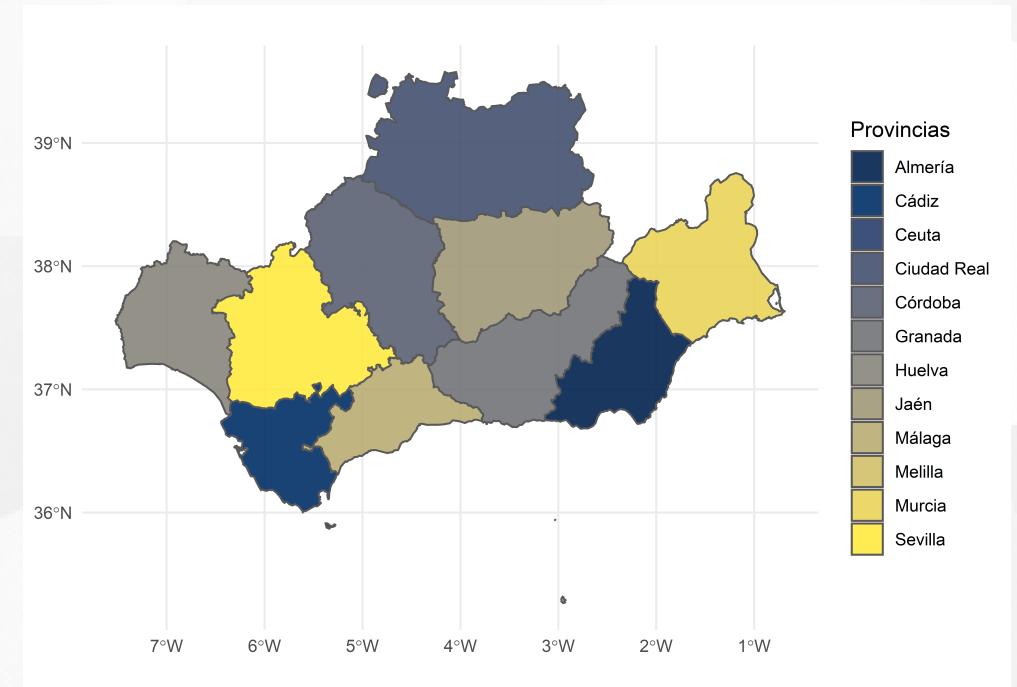
# Límites políticos

## Provincias (usando versión \*\_siane)

Si pasamos una entidad de orden superior (e.g. Andalucía) obtenemos todas las provincias de esa entidad.

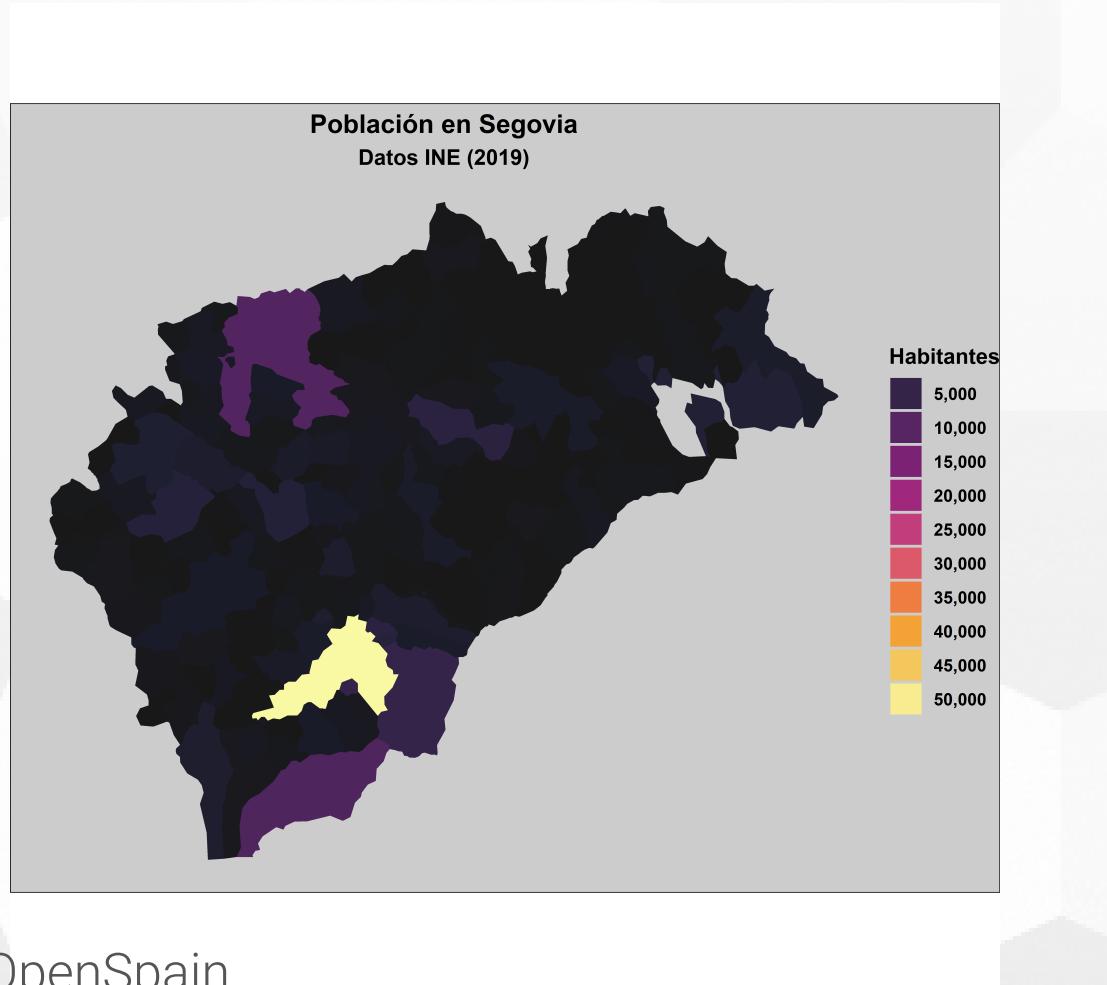
```
provs <- esp_get_prov_siane(c
  "Andalucía", "Ciudad Real",
  "Murcia", "Ceuta", "Melilla"
))

ggplot(provs) +
  geom_sf(aes(fill = prov.shortname.es),
  alpha = 0.9
) +
  scale_fill_discrete(type = hcl.colors(12, "Cividis")) +
  theme_minimal() +
  labs(fill = "Provincias")
```



# Límites políticos

## Municipios



```
munic <- esp_get_munic(region = "Segovia") %>%
  # Datos de ejemplo: Población INE
  left_join(mapSpain::pobmun19, by = c("cpro", "cmun"))

ggplot(munic) +
  geom_sf(aes(fill = pob19), alpha = 0.9, color = NA) +
  scale_fill_gradientn(
    colors = hcl.colors(100, "Inferno"),
    n.breaks = 10,
    labels = scales::label_comma(),
    guide = guide_legend()
  ) +
  labs(
    fill = "Habitantes",
    title = "Población en Segovia",
    subtitle = "Datos INE (2019)"
  ) +
  theme_void() +
  theme(
    plot.background = element_rect("grey80"),
    text = element_text(face = "bold"),
    plot.title = element_text(hjust = .5),
    plot.subtitle = element_text(hjust = .5)
  )
```

# Límites políticos

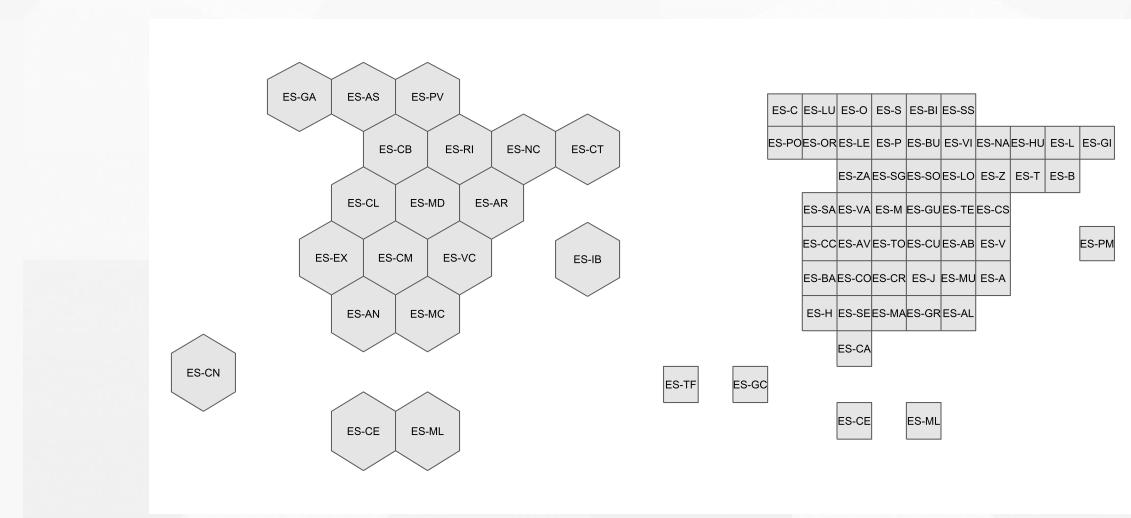
## Hexbin maps

Disponibles como cuadrados y hexágonos, para provincias y CCAA.

```
cuad <- esp_get_hex_ccaa()
hex <- esp_get_grid_prov()

ggplot(cuad) +
  geom_sf() +
  geom_sf_text(aes(label = iso2.ccaa.code)) +
  theme_void()

ggplot(hex) +
  geom_sf() +
  geom_sf_text(aes(label = iso2.prov.code)) +
  theme_void()
```



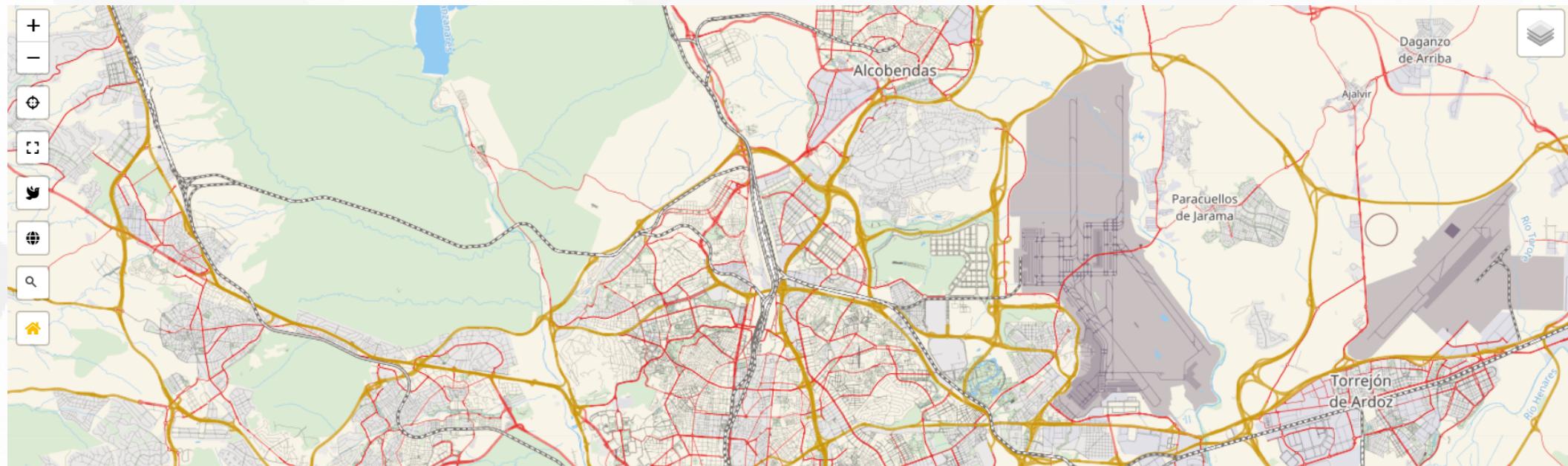
# Imágenes

# Imágenes

**mapSpain** permite usar también imágenes de mapas (satélite, mapas base, carreteras, etc.) proporcionados por diferentes organismos públicos (<https://www.idee.es/web/idee/segun-tipo-de-servicio>).

Las imágenes se pueden emplear para la creación de mapas estáticos (imágenes obtenidas como capas ráster de 3 o 4 bandas) o como fondo de mapas dinámicos, a través del paquete **leaflet**.

Los proveedores se han extraído del plugin para **leaflet** **leaflet-providerESP**.



# Imágenes

## Creación de mapas estáticos

Tenemos varias opciones que podemos emplear para componer mapas base:

```
madrid_munis <- esp_get_munic_siane(region = "Madrid")
base_pnoa <- esp_getTiles(madrid_munis, "PNOA", bbox_expand = 0.1, zoommin = 1)

ggplot() +
  layer_spatraster(base_pnoa) +
  geom_sf(
    data = madrid_munis, color = "blue", fill = "blue",
    alpha = 0.25, lwd = 0.5
  ) +
  theme_minimal() +
  labs(title = "Municipios en Madrid")

# Usando la opción mask
madrid <- esp_get_munic_siane(munic = "^Madrid$")

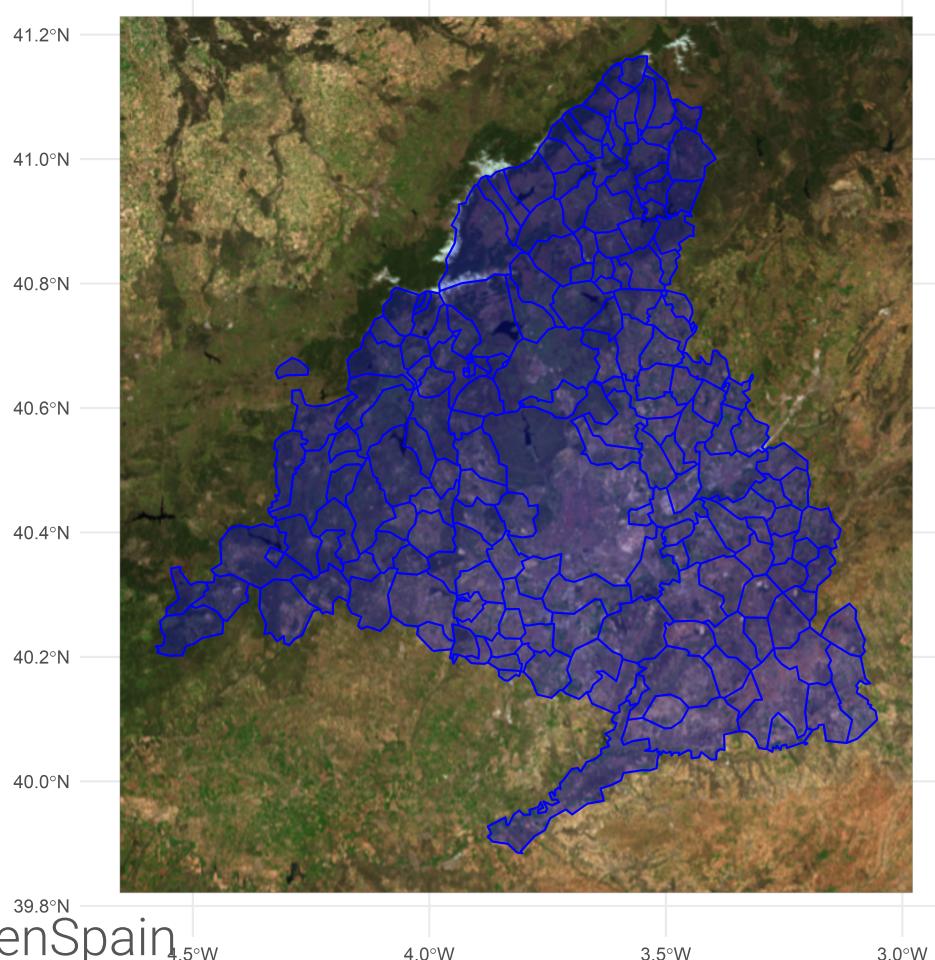
madrid_mask <- esp_getTiles(madrid, mask = TRUE, crop = TRUE, zoommin = 2)

ggplot() +
  layer_spatraster(madrid_mask) +
  theme_void() +
  labs(title = "Mapa Base de la Comunidad de Madrid",
       caption = "CC BY 4.0 www.iderioja.org"
  )
```

# Imágenes

## Creación de mapas estáticos

Municipios en Madrid



Mapa Base de la Comunidad de Madrid



# Imágenes

## Mapas dinámicos usando mapSpain

Estas capas se pueden usar también como fondo en mapas estáticos

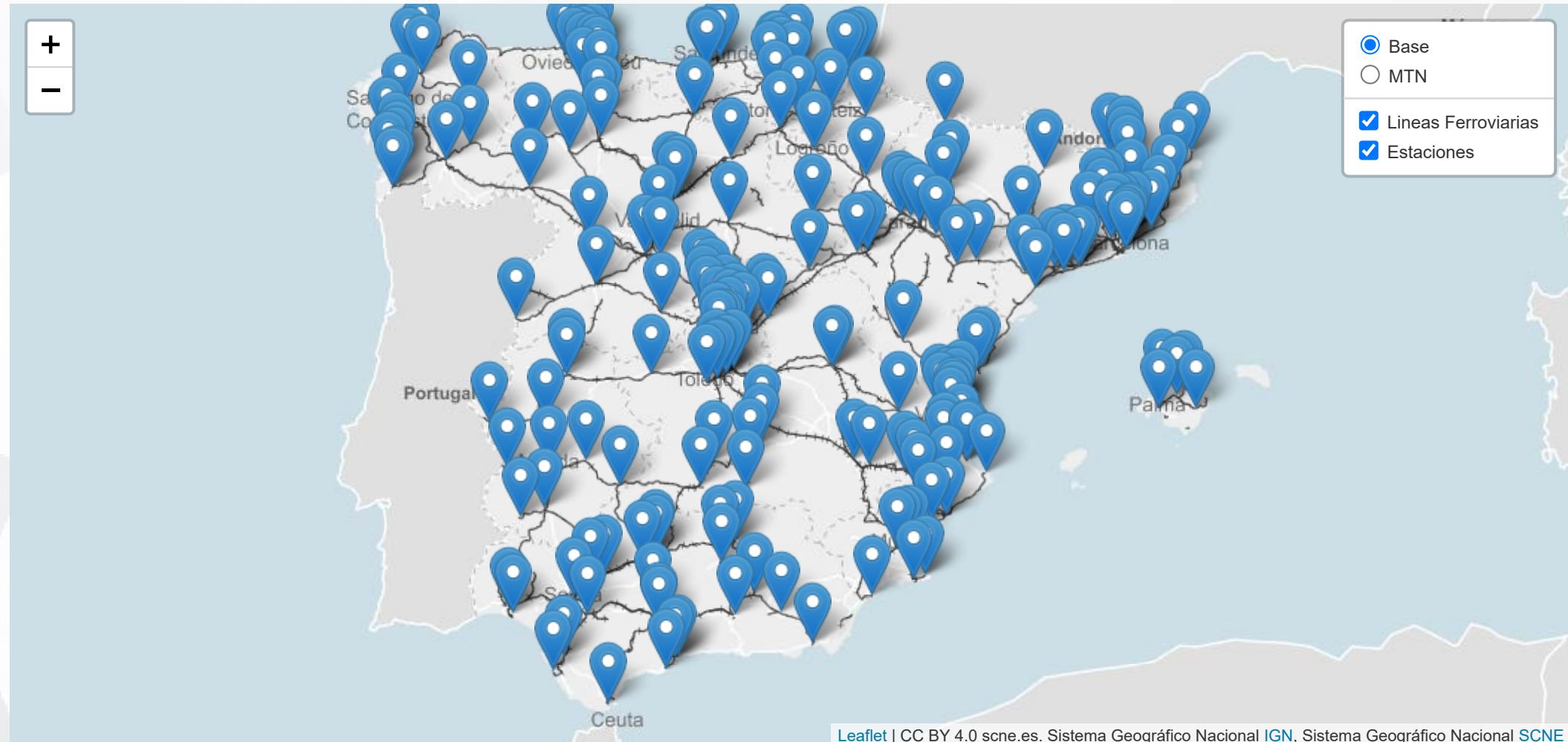
```
stations <- esp_get_railway(spatialtype = "point", epsg = 4326)

library(leaflet)

leaflet(stations) %>%
  addProviderEspTiles("IGNBase.Gris", group = "Base") %>%
  addProviderEspTiles("MTN", group = "MTN") %>%
  addProviderEspTiles("RedTransporte.Ferroviario", group = "Lineas Ferroviarias") %>%
  addMarkers(group = "Estaciones",
    popup = sprintf(
      "<strong>%s</strong>",
      stations$rotulo) %>%
    lapply(htmltools::HTML)
  ) %>%
  addLayersControl(
    baseGroups = c("Base", "MTN"),
    overlayGroups = c("Lineas Ferroviarias", "Estaciones"),
    options = layersControlOptions(collapsed = FALSE)
  )
```

# Imágenes

## Mapas dinámicos usando mapSpain

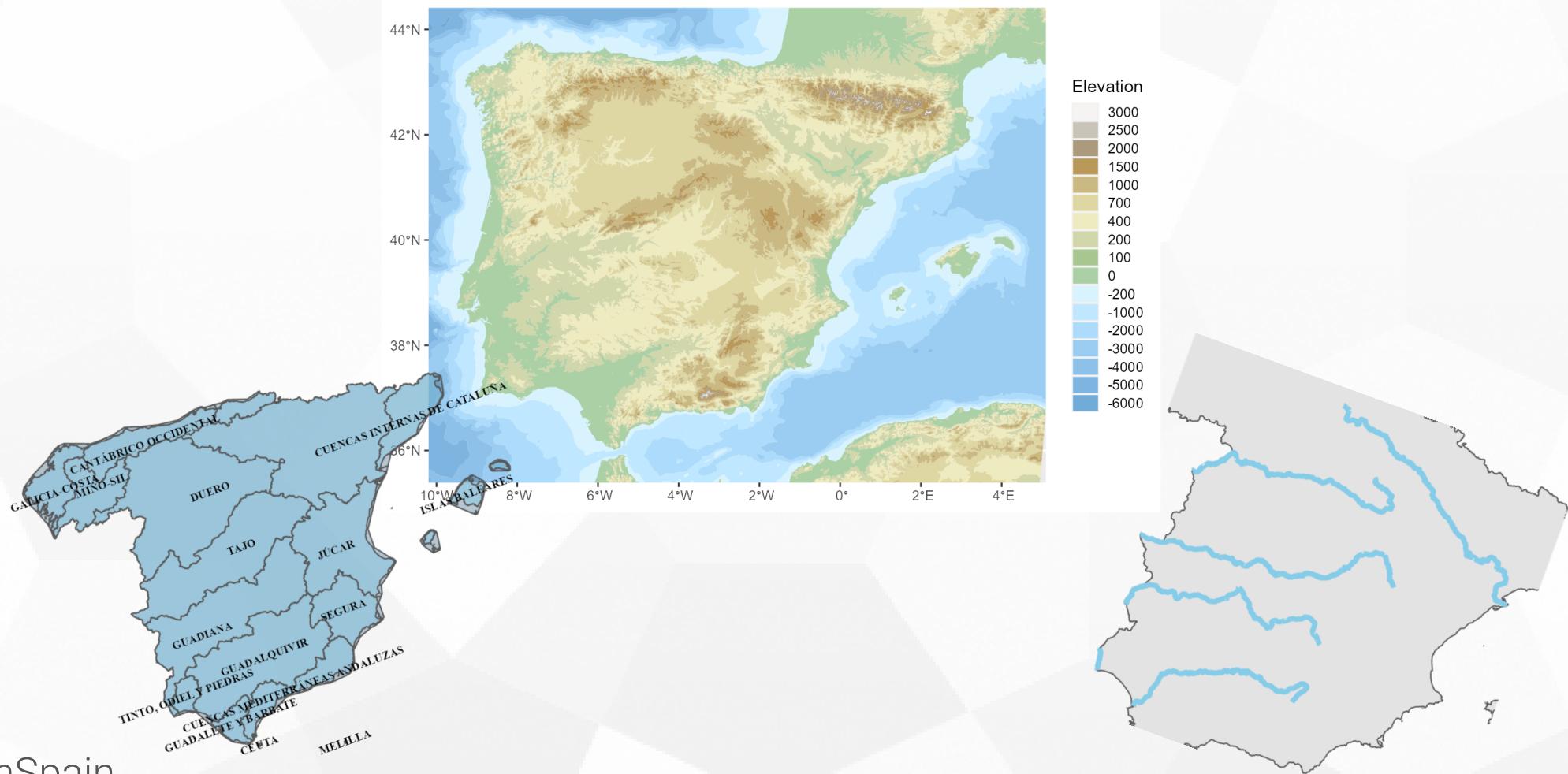


Leaflet | CC BY 4.0 scne.es. Sistema Geográfico Nacional IGN, Sistema Geográfico Nacional SCNE

# Otros recursos

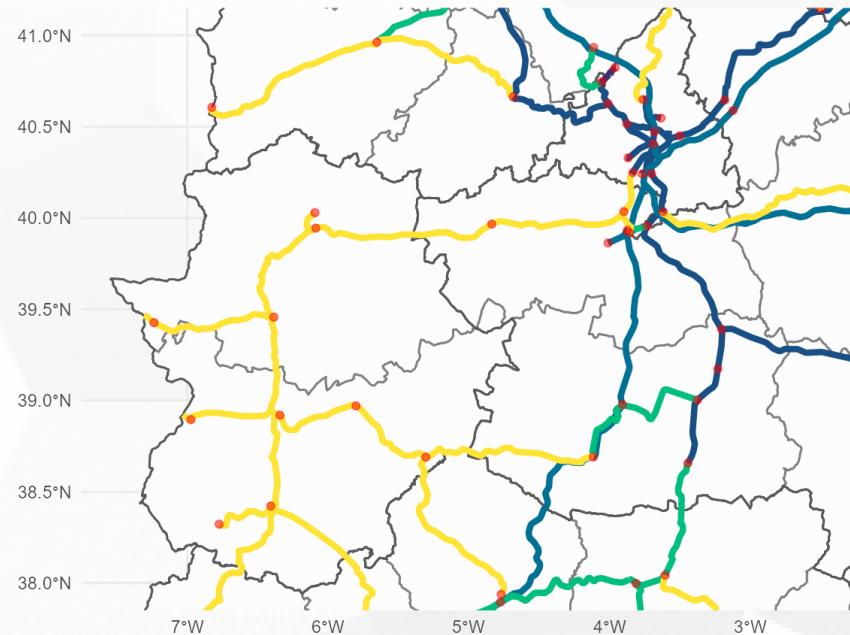
# Otros recursos

mapSpain incluye otras **funciones adicionales** que permiten extraer información sobre altitud, ríos y cuencas hidrográficas de España.



# Otros recursos

Con mapSpain podemos obtener líneas y puntos de **infraestructuras** de España, como carreteras y líneas ferroviarias.



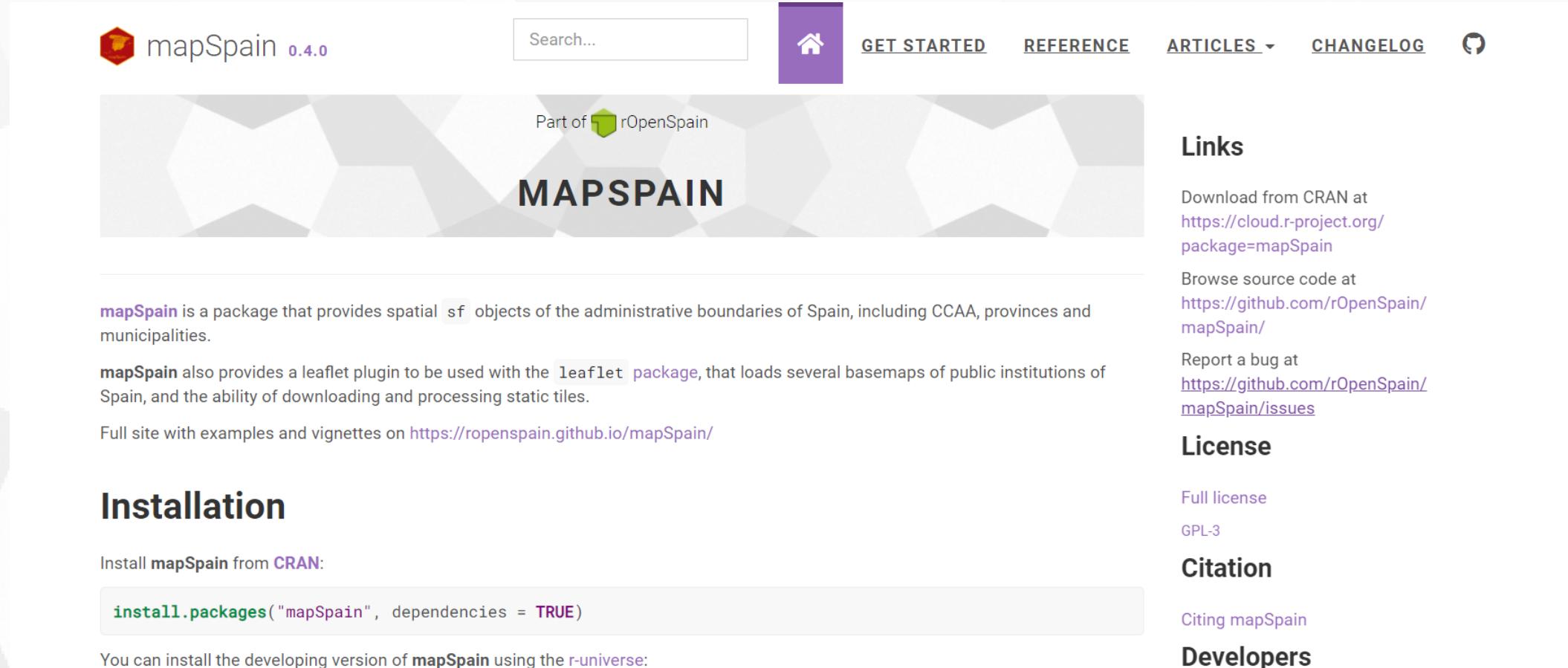
## Road type

- Autopistas de peaje
- Autopistas, autovías y doble calzada
- Otras carreteras
- Red de carreteras del Estado

# Otros recursos

**MUY RECOMENDABLE:** visitar la página de la documentación <https://ropenspain.github.io/mapSpain/>.

Presenta gran cantidad de ejemplos con códigos reproducibles:



The screenshot shows the homepage of the mapSpain package documentation. At the top, there's a navigation bar with a search bar, a purple home icon, and links for "GET STARTED", "REFERENCE", "ARTICLES", and "CHANGELOG". Below the header, it says "Part of rOpenSpain" and features a large title "MAPSPAIN" with a background graphic of grey hexagons. The main content area describes the package as providing spatial `sf` objects of Spain's administrative boundaries and a Leaflet plugin. It includes a link to the full site at <https://ropenspain.github.io/mapSpain/>. On the right side, there are sections for "Links", "License", "Citation", and "Developers", each with a corresponding link.

**Links**

Download from CRAN at  
[https://cloud.r-project.org/  
package=mapSpain](https://cloud.r-project.org/package=mapSpain)

Browse source code at  
[https://github.com/rOpenSpain/  
mapSpain/](https://github.com/rOpenSpain/mapSpain/)

Report a bug at  
[https://github.com/rOpenSpain/  
mapSpain/issues](https://github.com/rOpenSpain/mapSpain/issues)

**License**

[Full license](#)  
[GPL-3](#)

**Citation**

[Citing mapSpain](#)

**Developers**

Install mapSpain from CRAN:

```
install.packages("mapSpain", dependencies = TRUE)
```

You can install the developing version of mapSpain using the r-universe:

# Gracias

Presentación creada con **xaringan**

GitHub repo: <https://github.com/rOpenSpain/mapSpain>

Docs: <https://ropenspain.github.io/mapSpain/>