Laboratorio Nacional de Políticas Públicas

CIDE CENTRO DE INVESTIGACIÓN Y DOCENCIA ECONÓMICAS A.C.

Mapas interactivos en leaflet

Periodismo de Datos

Febrero, 2021

M.C. JORGE JUVENAL CAMPOS FERREIRA.
Investigador Asociado.
Laboratorio Nacional de Políticas Públicas
CIDE

Hoja de Ruta.

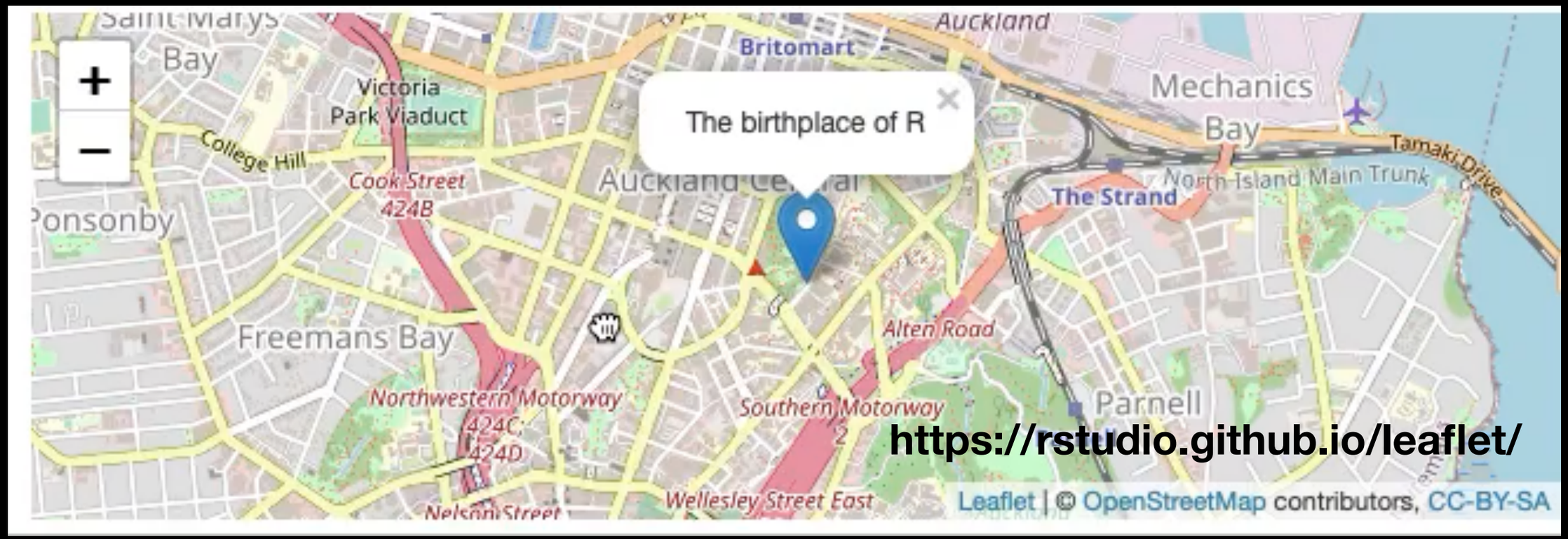
1. ¿Qué necesitamos saber? - Revisión de conceptos básicos.
2. Introducción a la librería {*leaflet*} para visualización de mapas interactivos.
3. Funciones de {leaflet} para plasmar geometrías, crear paletas de colores o elementos de interactividad.
4. Práctica. Elaborando mapas en {*leaflet*}.

Conceptos Básicos

Jorge Juvenal Campos Ferreira

Mapa Interactivo

Un mapa que añade las funciones de interactividad.

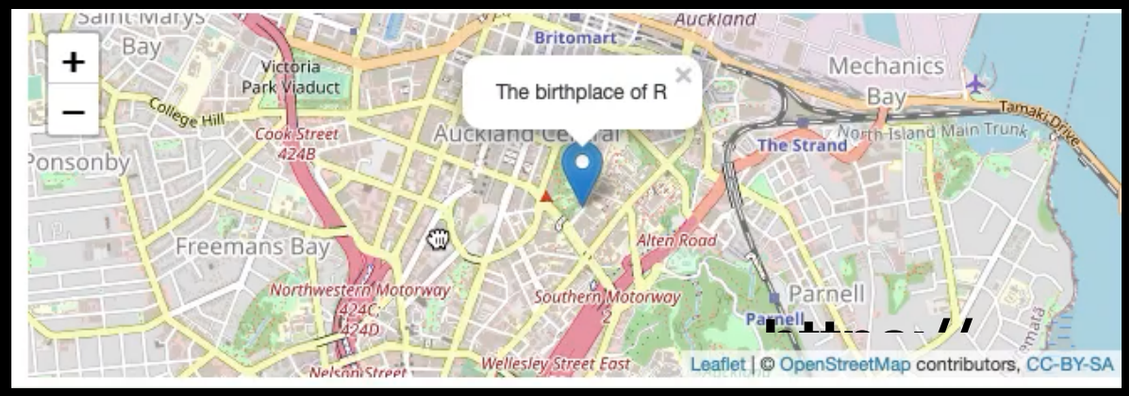


Ejemplo: https://rpubs.com/Juve_Campos/jitomate2019red

¿En qué casos quiero un mapa interactivo?

Si voy a publicar el mapa en internet y:

- Quiero que el usuario pueda explorar los datos.
- Quiero plasmar muchas variables en un mapa.
- La información es muy densa y no se ve bien en presentación estática o en papel
- Voy a hacer una aplicación web

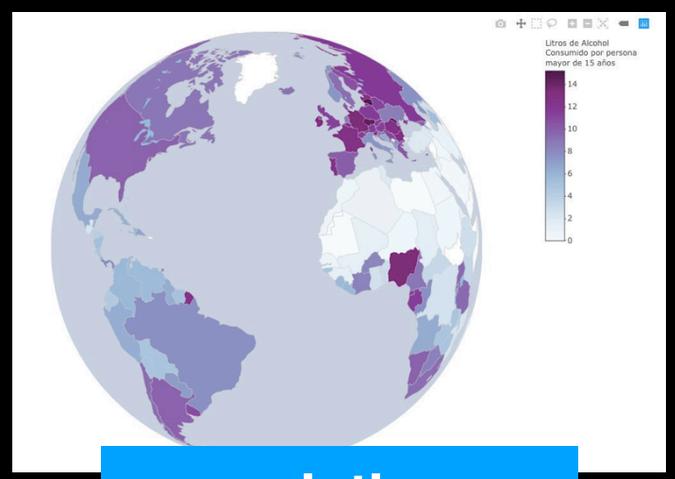


Librerías

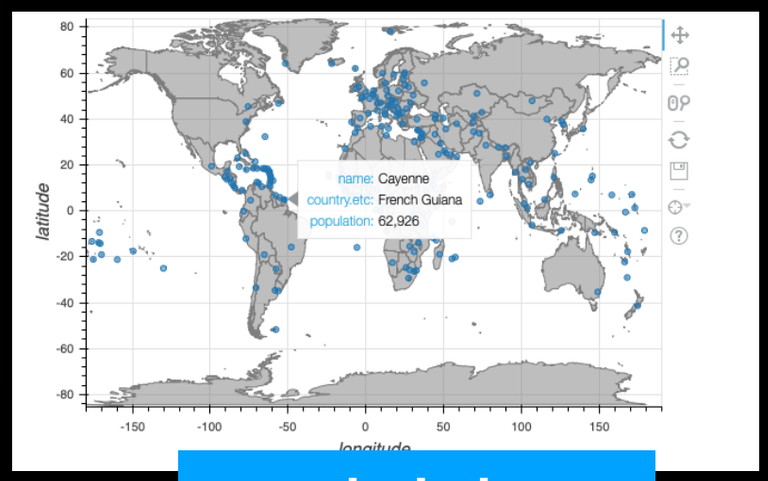
Leaflet, plotly, tmap, mapbox, highcharter, rbokeh, etc.

En esta session utilizaremos leaflet, por ser una de las más comunes para elaborar mapas interactivos.

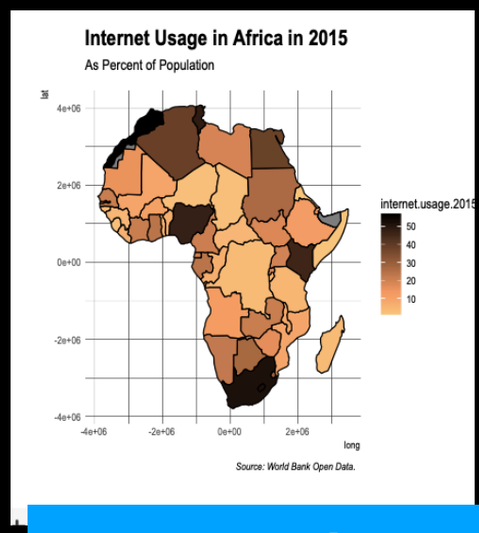
Más referencia en https://bhaskarvk.github.io/user2017.geodataviz/notebooks/03-Interactive-Maps.nb.html



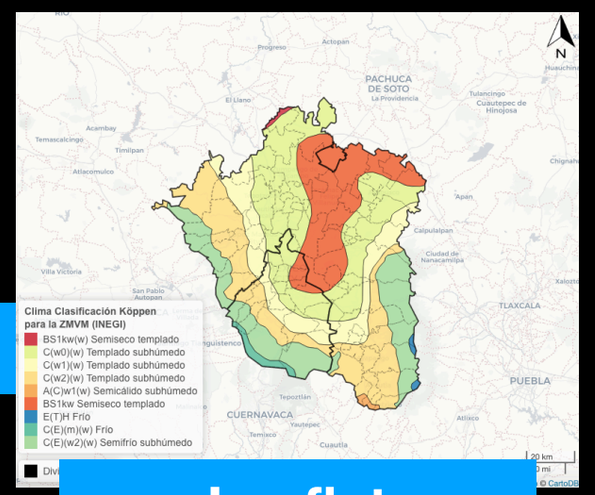
plotly



rbokeh



ggraph

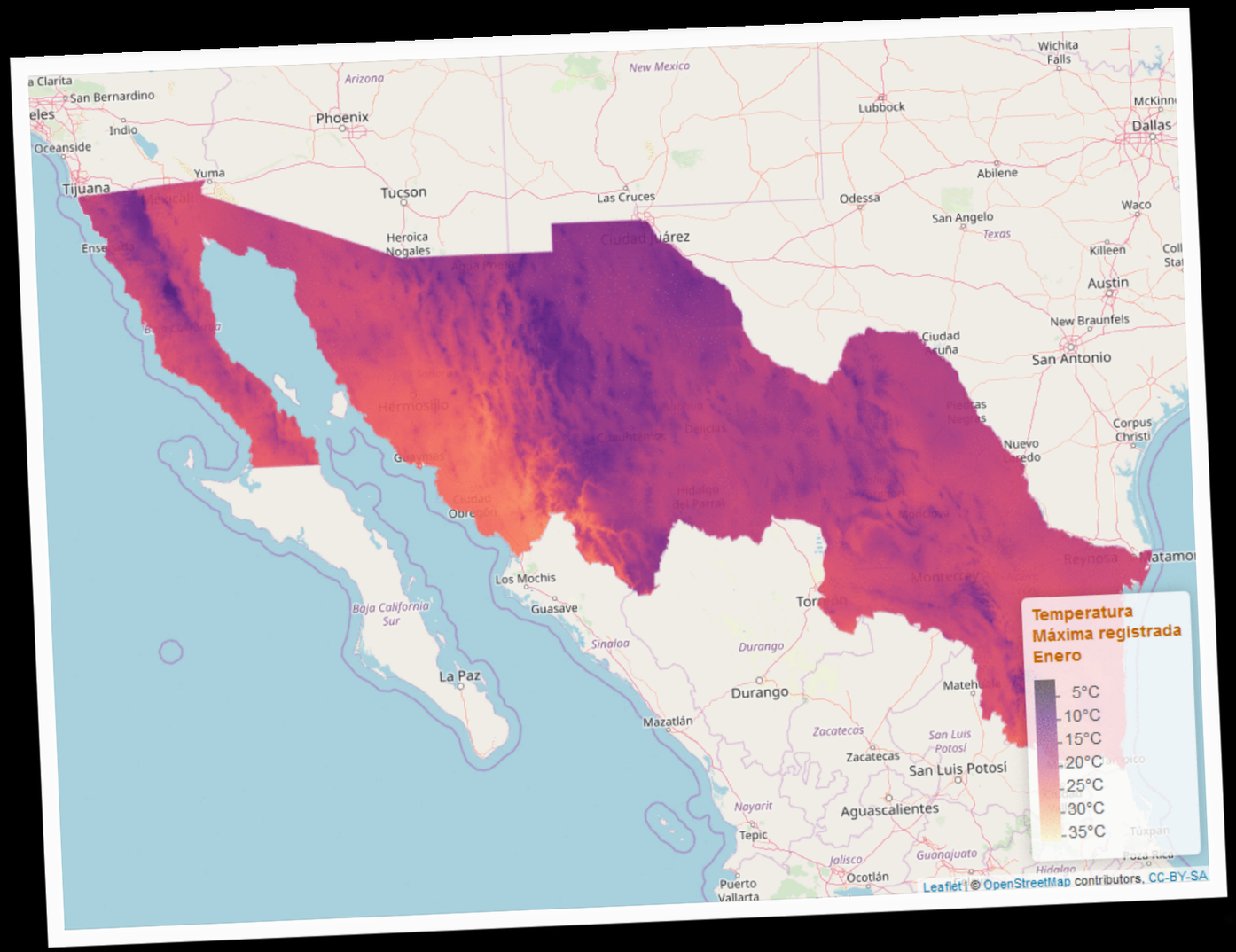


Leaflet

Librería {leaflet}

La librería Leaflet.js es una librería open-source de J**avascript que nos permite crear mapas interactivos.** La librería de R, **{leaflet}, nos permite crear fácilmente mapas con esta librería utilizando la sintaxis de R, sin tener que aprender javascript.**

Referencia principal:
https://rstudio.github.io/leaflet/



Elementos de un mapa interactivo de {*leaflet*}

Jorge Juvenal Campos Ferreira

Labels o tooltips

Los tooltips (en {leaflet} también conocidos como *labels*) son elementos de la interfaz de usuario que se despliegan como cajas de texto informativas cuando hacemos hover (pasamos el cursor) sobre algún elemento de interés.

Esto es un tooltip sobre el punto rojo.

Popups

Los popups, en el contexto de leaflet, son **elementos de la interfaz de usuario que despliegan información cuando el usuario lleva a cabo una acción particular** (típicamente un click sobre algún elemento).

*A estos elementos se les da formato con código HTML.

Paletas de colores

Las paletas de colores son **conjuntos de colores que dan color a los distintos elementos de un mapa** (áreas, líneas, puntos, pixeles). En R, las paletas toman la forma de **funciones**, las cuales **relacionan atributos (datos, valores) con un color correspondiente.**

Las paletas varían en función del tipo de datos que queremos plasmar en un mapa.

Tipo de dato	Función generadora de paleta
Numérico continuo, escala contínua.	*colorNumeric()*
Numerico contínuo, escala discreta	*colorBin() (rangos) o colorQuantile() (cuantiles)*
Categórico	*colorFactor()*

Más info: https://rpubs.com/Juve_Campos/LeafletTutorial

Librerías

Los argumentos básicos que reciben las funciones de paletas son:

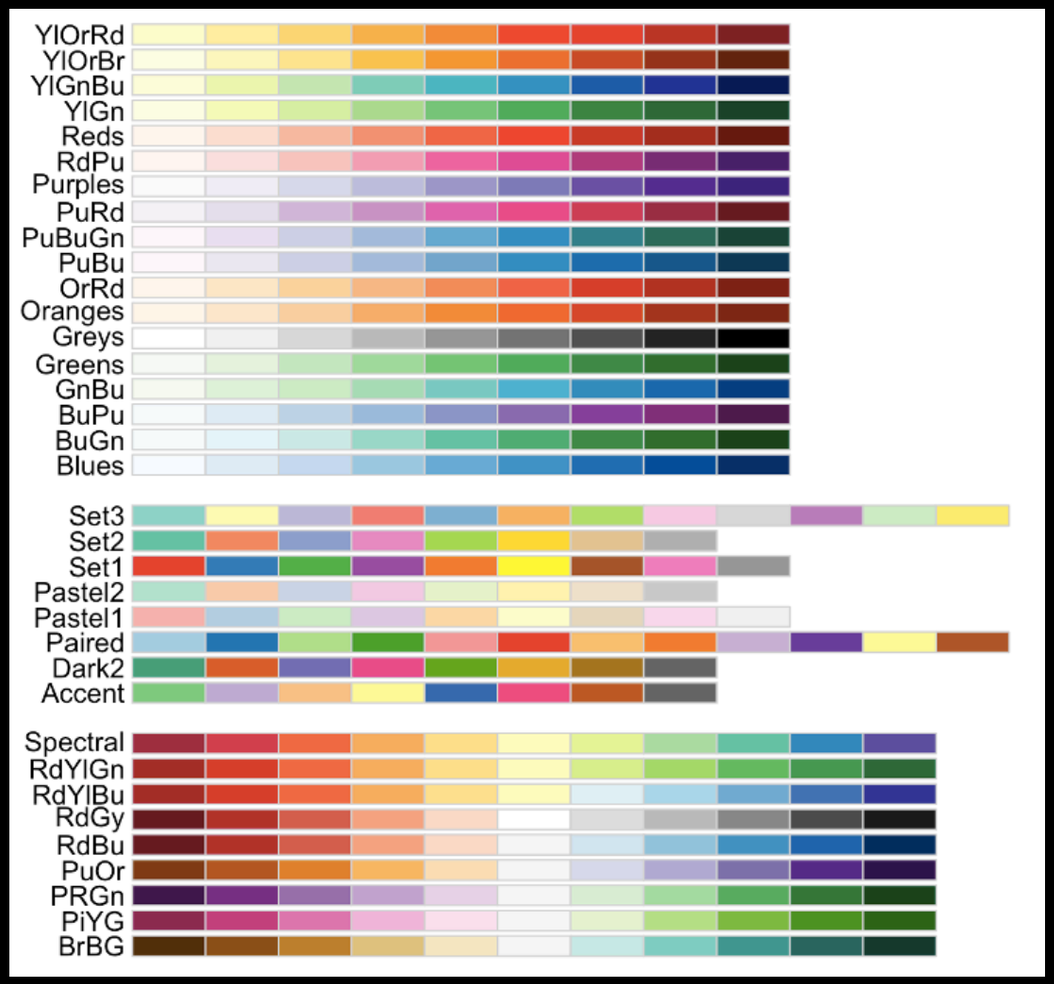
Palette (colores): Este argumento recibe los colores que queremos que conformen a la paleta de colores. Estos pueden ser:

1. Un vector personalizado de colores (en texto o en formato hexadecimal).
2. Un color de la librería “viridis” cargado como cadena de texto: “viridis”, “inferno”, “magma”, “plasma”.
3. Una paleta pre-construida de la librería RColorBrewer, como por ejemplo: “RdYlBu”, “Accent” o “Greens”.
4. Alguna paleta de colores de algúna otra librería (como las famosas paletas de colores de “wespalettes”).

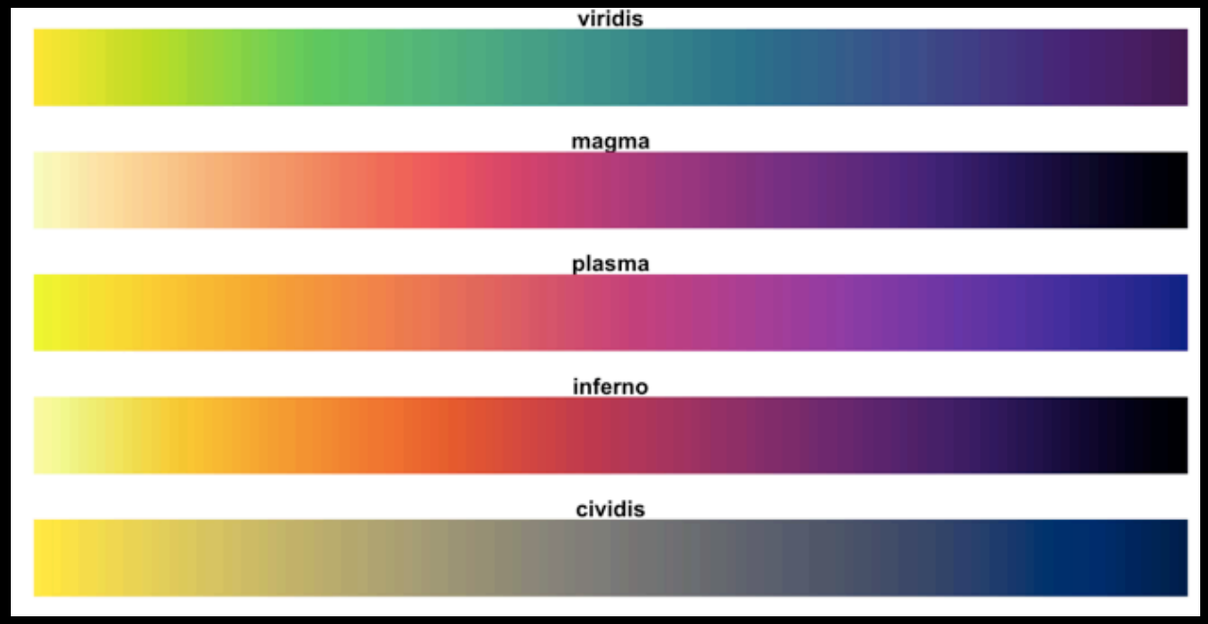
Domain (valores). Este argumento nos solicita el vector de valores a los cuales se les va a asociar un color.

Paletas de colores

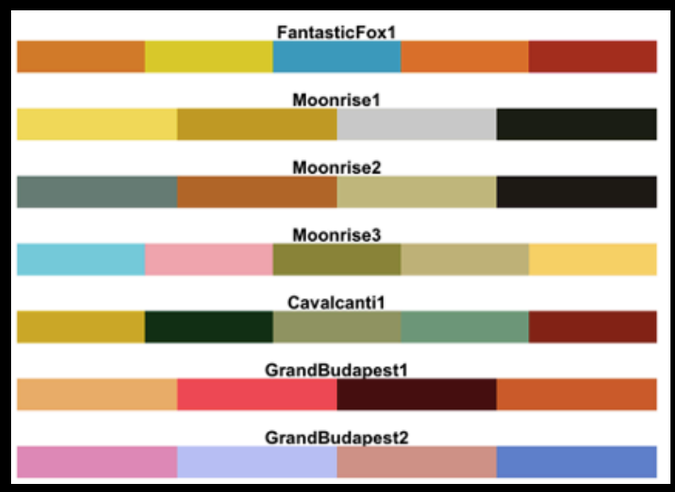
RColorBrewer

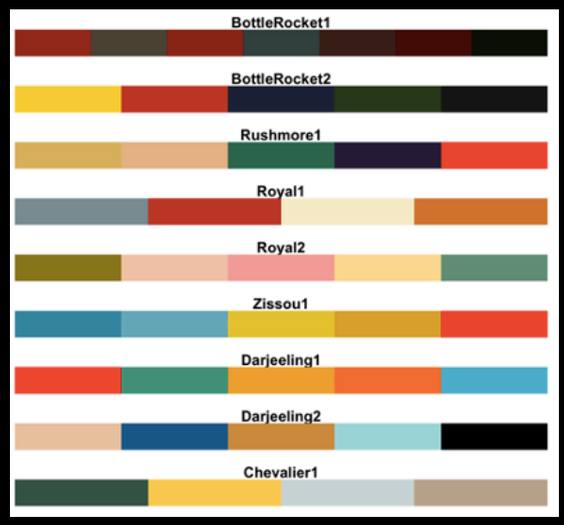


Viridis



wespalettes

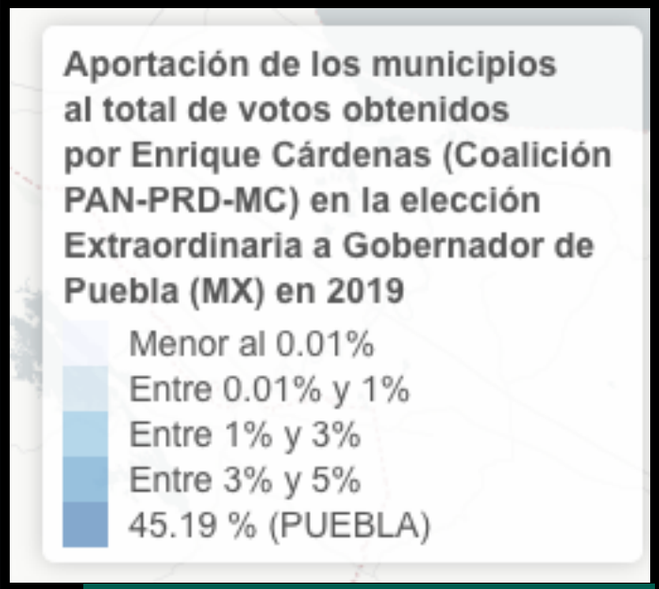




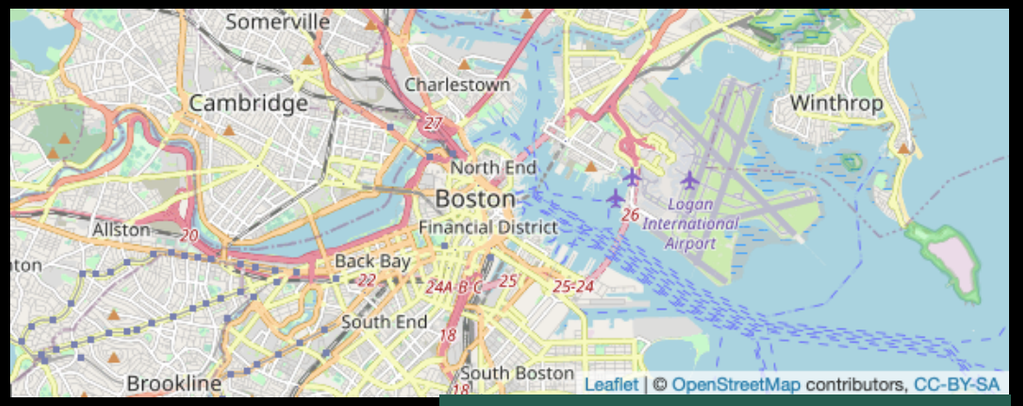
Jorge Juvenal Campos Ferreira

Capas

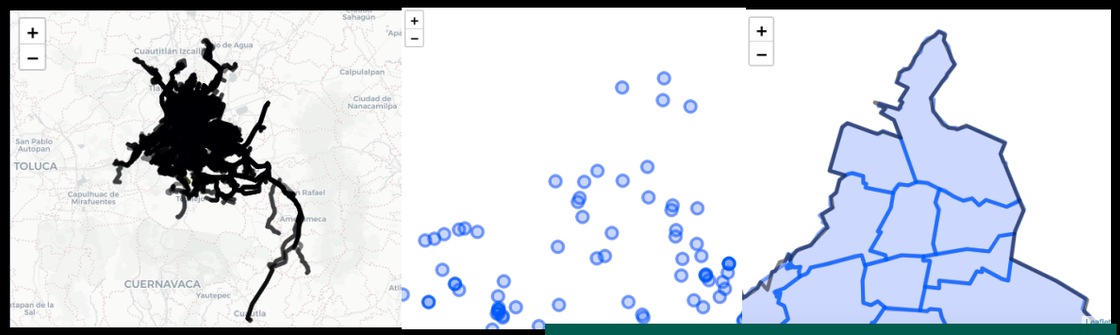
Leaflet maneja distintos tipos de capas:



Leyendas

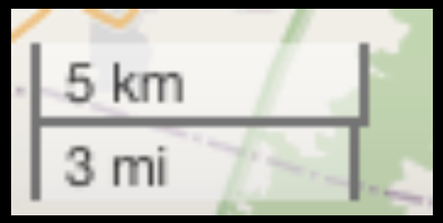


Mapas base

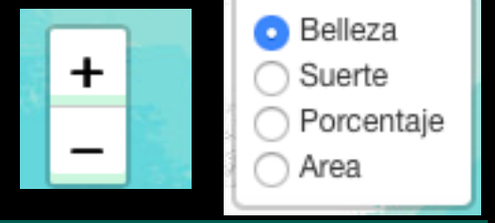


Geometrías

Badges



Información Adicional



Controles

Capas

Leaflet maneja distintos tipos de capas:

-**Mapas base** (tiles). Son imágenes georreferenciadas que dan un contexto sobre el lugar donde viven nuestras geometrías.
-**Capas de geometrías.** Son las geometrías almacenadas en los objetos sf a partir de los cuales vamos a construir nuestros mapas.
-**Controles.** Los controles son elementos de la interfaz de usuario que controlan la interacción del usuario con la información vertida en el mapa. Entre estos controles destacan los botones de zoom y los botones de control de capas.
-**Leyendas.** Elementos del mapa que informan acerca de la paleta de colores que estamos usando.
-**Información adicional.** Leaflet nos permite añadir elementos adicionales a nuestros mapas. Un ejemplo muy usado es la barra de escalas de distancias en el mapa.
-**Badges.** Leaflet nos permite añadir información adicional personalizada a través de imágenes y texto.

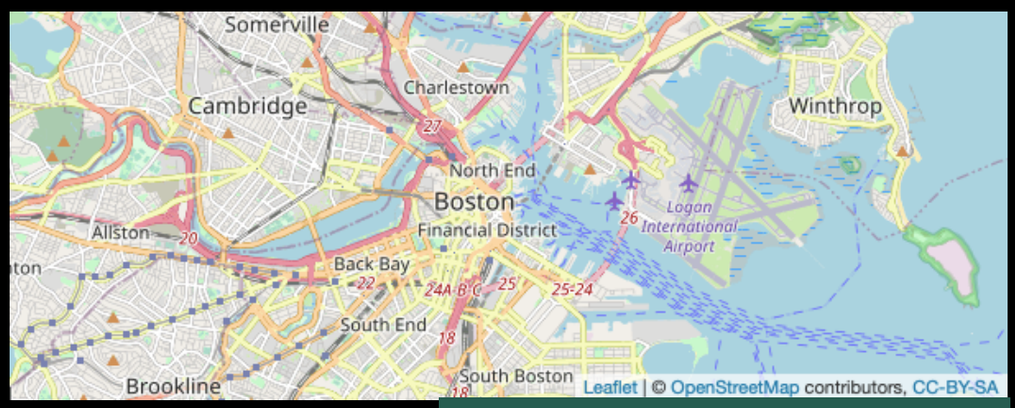
Funciones de {*leaflet*}

Para añadir mapas base:

-addTiles(), agrega un mapa base default al mapa que estamos elaborando. **(Nota: Recordar que leaflet trabaja con el crs = 4326, por lo que si no estamos trabajando con este *crs* van a salir cosas muy extrañas).**

-addProviderTiles("nombre_del_mapa-base"), agrega un mapa base distinto al default. Para ver que mapas base hay disponibles hay que acceder al catálogo a través del objeto **leaflet::providernames** o visitar https://leaflet-extras.github.io/leaflet-providers/preview/

-



Mapas base

Funciones de {*leaflet*}

Para añadir Capas de geometrías:

-**addPolygon()**, para añadir polígonos.

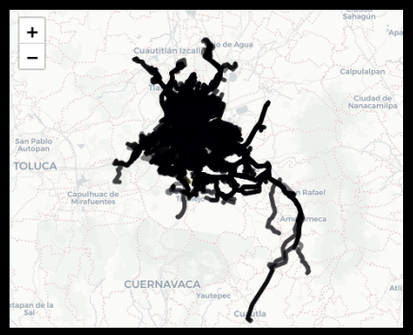
-**addPolylines()**, para añadir líneas (polilíneas).

-**addCircleMarker()**, para añadir círculos en los puntos.

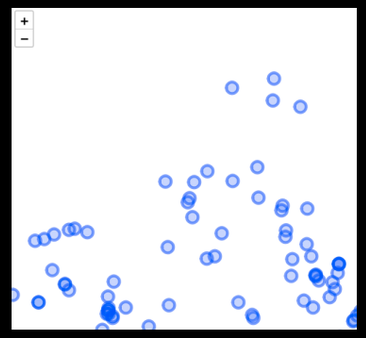
-**addMarker()**, para añadir marcadores sobre los puntos.

-**addRasterImage()**, para añadir imágenes raster.

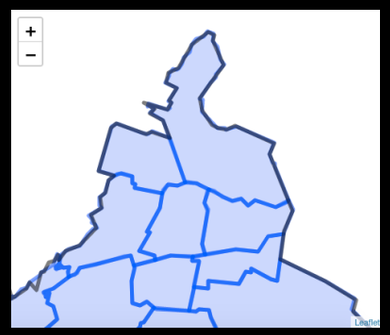
Geometrías



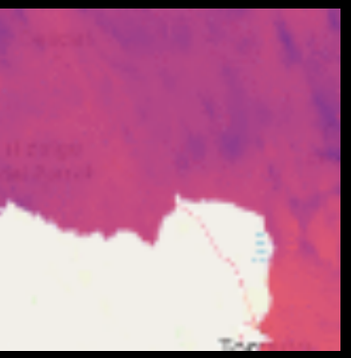
-Líneas



-Puntos



-Polígonos



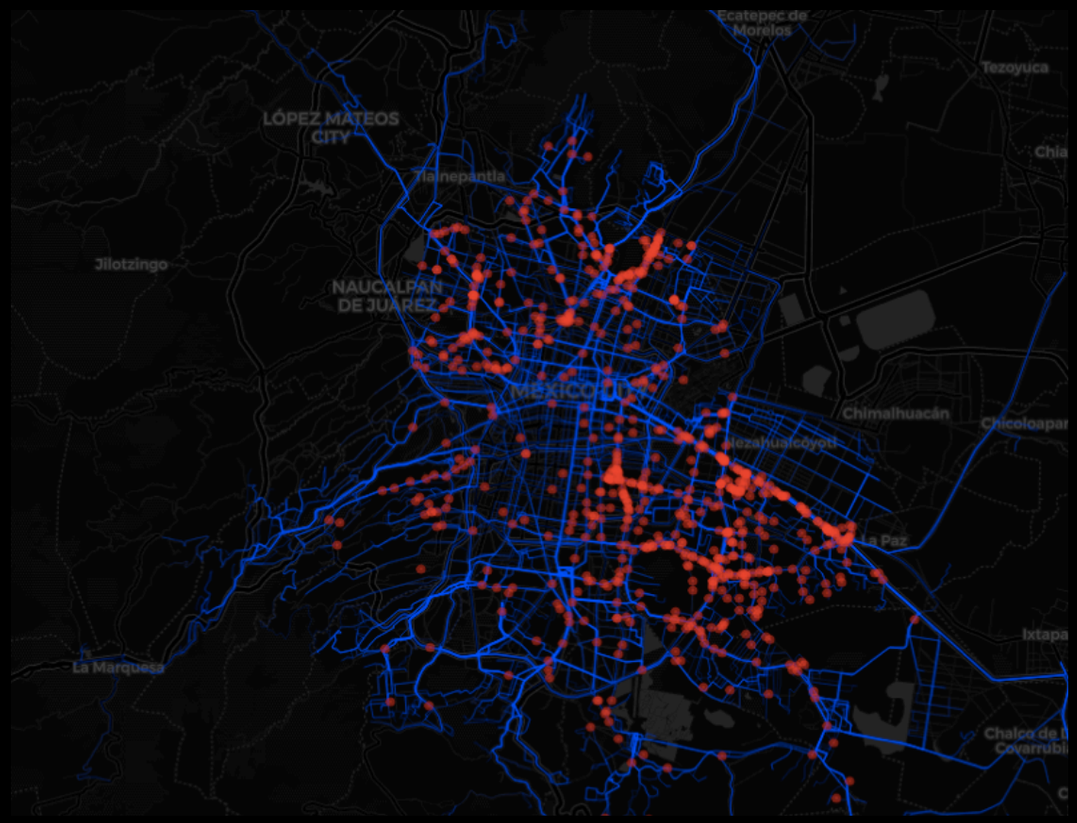
-Rasters

Funciones de {*leaflet*}

Así como ggplot nos permite agregar múltiples geometrías en un solo mapa, igual podemos ir pegando geometrías en un solo mapa. Al igual que *ggplot*(), **podemos agregar estas geometrías nuevas especificando la base de geometrías a través del argumento "data".**

Líneas + puntos =

```
leaflet(datos, options = leafletOptions(zoomControl = FALSE)) %>%
  addProviderTiles("CartoDB.DarkMatter") %>%
  addPolylines(data = rutas,
               weight = 0.3,
               label = rutas$ruta_corre
  ) %>%
  addCircleMarkers(fillColor = "red",
                   color = "red",
                   fillOpacity = 0.3,
                   radius = 0.1
                   )
```

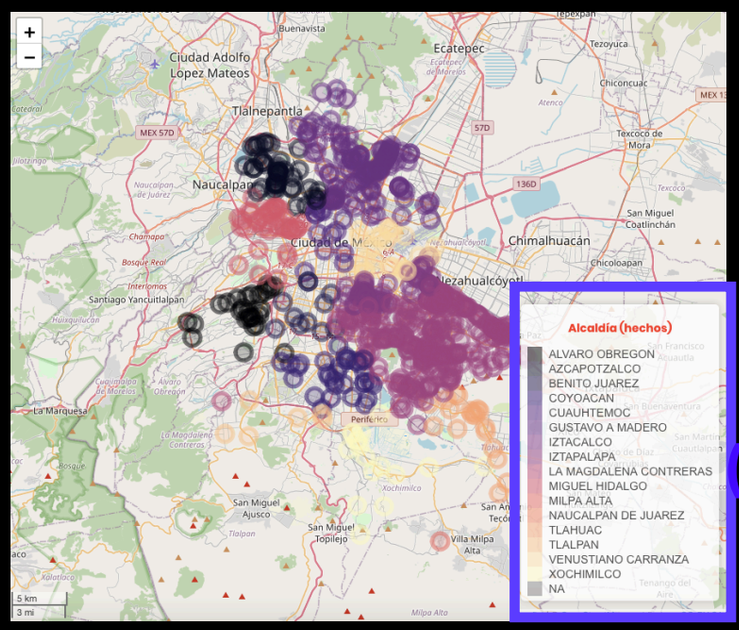


Funciones de {*leaflet*}

Para añadir leyendas.

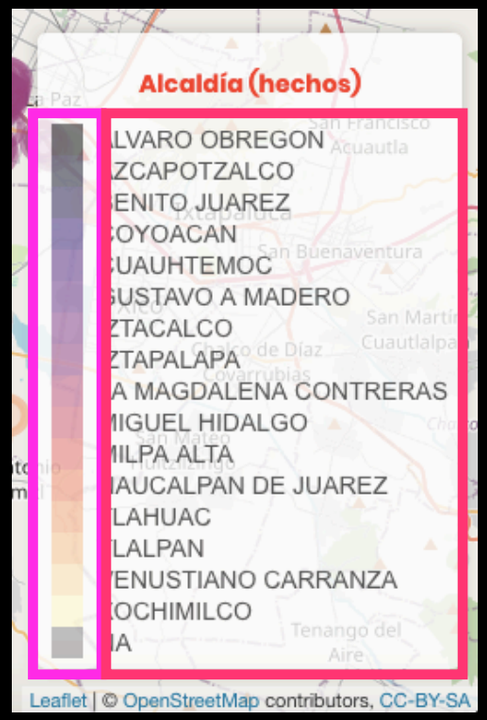
–**addLegend(),** nos permite construir una leyenda en el mapa. Recibe como argumentos (entre otros):

- la paleta de colores,
- el dominio de los datos,
- la posición de la leyenda dentro del mapa,
- el título de la leyenda (formateable con HTML).



Paleta (viridis::magma)

Posición (bottomright)



Título (formateado con HTML y CSS)

Dominio (alcaldías CDMX)

Actividad práctica.

- Abriremos y exploraremos cómo hacer mapas interactivos con R y *leaflet*.