

NIDO DEL BÚHO PRESENTA:

# Estadística y Econometría Espacial con R, Módulo I

## Clase 1: Introducción al análisis espacial con R



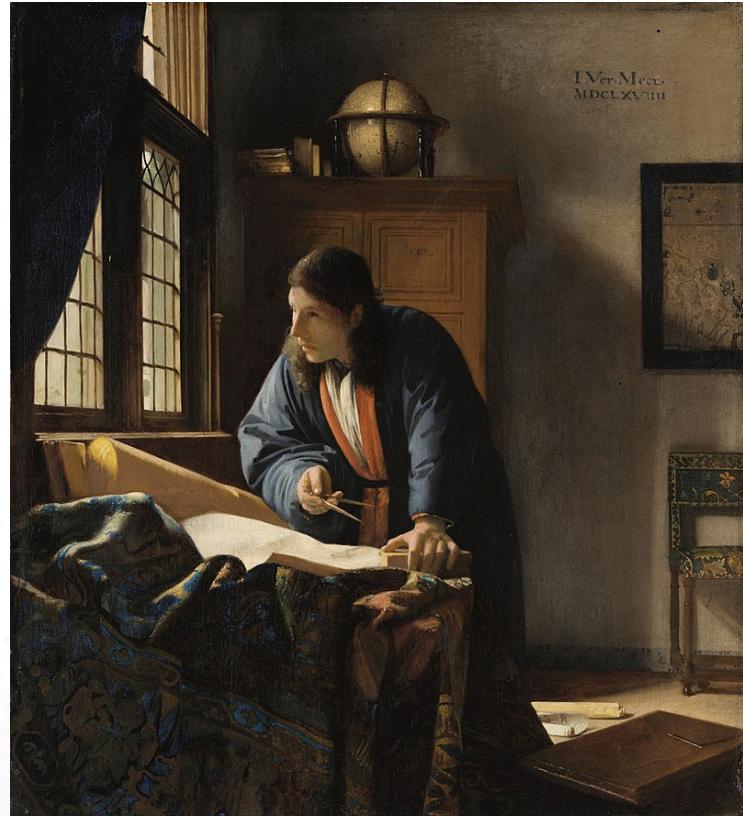
**PRESENTADO POR**  
Alex Bajaña | Pablo Sarango

---

# La revolución de los datos espaciales

En el **siglo pasado**:

- Existía una aguda escasez de datos y herramientas para el análisis geográfico.
- Los primeros geógrafos utilizaron diversas herramientas, como barómetros, brújulas y sextantes, para avanzar en el conocimiento del mundo



---

# La revolución de los datos espaciales

Por ejemplo:

- Solo con la invención del cronómetro marino en 1761 fue posible calcular la longitud en el mar, lo que permitió a los barcos tomar rutas más directas.



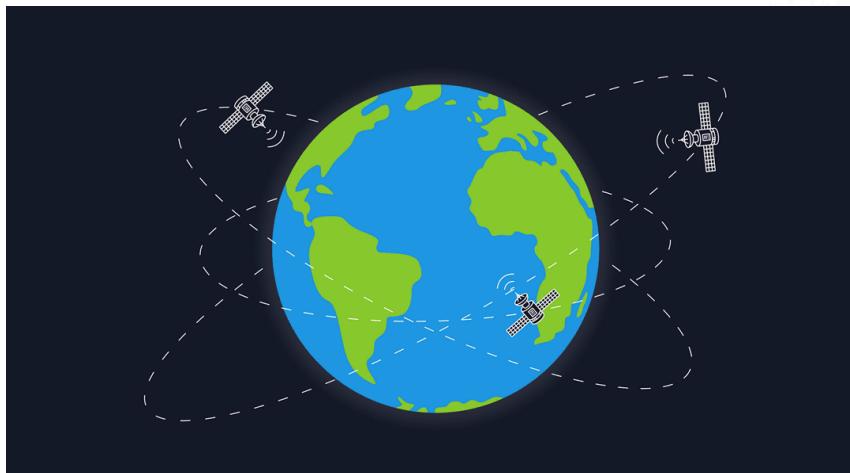
---

# La revolución de los datos espaciales

En la **actualidad**:

**Existe el problema contrario:  
demasiados datos; demasiadas  
herramientas**

- La mayoría de los teléfonos disponen ya de un receptor de posicionamiento global (GPS).
- Datos de sensores satelitales.
- Científicos ciudadanos miden incesantemente cada rincón del mundo.



---

# La revolución de los datos espaciales

Esta revolución de los datos espaciales impulsa la demanda de hardware informático de alto rendimiento y software eficiente y escalable para manejar y extraer la señal del ruido.



# Software Libre

El software libre se presenta como una alternativa particularmente atractiva porque ofrece a los usuarios un conjunto de libertades esenciales: ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software.

Y las ventajas que esto ofrece son:

- Ritmo de desarrollo y longevidad
- Interoperabilidad
- Reproductibilidad
- Comunidad

# Software Libre

## Ritmo de desarrollo y longevidad

Cientos de personas envían cada día informes de errores y sugieren nuevas funciones y mejoras de la documentación de los proyectos de código abierto.

## Interoperabilidad

Mientras que los productos patentados tienden a ser monopolios difíciles de mantener, el software de código abierto es más parecido a una federación de herramientas modulares que pueden combinarse de diferentes maneras.

# Software Libre

## Reproducibilidad

El software de código abierto elimina una importante barrera a la reproducibilidad al permitir que otros comprueben sus hallazgos o apliquen sus métodos en nuevos contextos utilizando las mismas herramientas.

## Comunidad

La comunidad le permite obtener soporte mucho más rápido y, a menudo, de mayor calidad que con un equipo de soporte centralizado.

# Progreso de software geoespacial

- En las últimas décadas gracias a organizaciones como **OSGeo**, las técnicas geográficas avanzadas ya no son un privilegio exclusivo de aquellos con acceso a hardware y software costosos.
- Ahora, cualquier persona puede descargar y ejecutar software de alto rendimiento para geocomputación.



# Enfoque GUI vs CLI

- Uno de estos productos es **QGIS**, que es un Sistema de Información Geográfica de software libre y de código abierto que permite crear, editar, visualizar, analizar y publicar información geoespacial.
- Si bien estos productos de software ofrecen potentes funcionalidades, su énfasis en la interfaz gráfica de usuario (GUI) puede limitar la capacidad de los usuarios para documentar y compartir sus flujos de trabajo de manera precisa y replicable.
- Por lo que un enfoque de línea de comandos (CLI), proporciona un entorno flexible para registrar y compartir los pasos de análisis, permitiendo a otros usuarios replicar los resultados y verificar su validez.



# Introducción a R

- R es un lenguaje y entorno multiplataforma de código abierto para el análisis estadístico, la visualización de datos y el aprendizaje automático.
- Con una amplia gama de paquetes, R también soporta la estadística geoespacial avanzada y modelización.
- Además, su naturaleza extensible facilita la integración con otros lenguajes mediante paquetes como [Rcpp](#) y [reticulate](#), posibilitando el acceso a código C++ y Python.



# Instalación de R

## Windows

Para instalar R en Windows:

1. Haga clic en el enlace [Descargar R para Windows.](#)
2. A continuación, haga clic en el enlace [base.](#)
3. Finalmente, haga clic en el primer enlace de la parte superior de la nueva página. Este enlace debería decir algo parecido a [Descargar R 4.4.0 para Windows.](#)

## Mac

Para instalar R en un Mac:

1. Haga clic en el enlace [Descargar R para Mac.](#)
2. A continuación, haga clic en el enlace del [paquete R-4.4.0](#) (o en el enlace del paquete de la versión más reciente de R).

# Instalación de RStudio

- En este curso accederemos a R a través de la interfaz que ofrece RStudio.
- RStudio es un IDE (Integrated Development Environment, o Entorno de Desarrollo Integrado) de código abierto para R, que permite interactuar con R de manera muy simple.

Para descargar Rstudio, bastará con acudir a su web <http://www.rstudio.com> y elegir la opción **Download Rstudio**



Grow your data science skills at posit::conf(2024) | August 12th-14th in Seattle

LEARN MORE X

posit PRODUCTS SOLUTIONS LEARN & SUPPORT EXPLORE MORE PRICING

SEARCH DOWNLOAD RSTUDIO

Turn data  
science into  
business results





# Interfaz de RStudio

RStudio File Edit Code View Plots Session Build Debug Profile Tools Window Help Flights - RStudio

flights-example.R Addins

Source on Save Go to file/function Run Source

```
library(nycflights13) ## package containing flights dataset
library(lubridate)
library(dplyr)
library(ggplot2)

head(flights, n = 3)
daily <- flights %>%
  mutate(date = make_date(year, month, day)) %>%
  count(date) %>%
  mutate(wday = wday(date, label = TRUE))
head(daily, n = 3)
ggplot(daily, aes(wday, n)) +
  geom_boxplot(outlier.colour = "hotpink") +
  labs(x = "Weekday", y = "Flights",
       subtitle = "Number of 2013 New York Flights Each Weekday")

```

(Top Level) R Script

Console Terminal Jobs

~/Documents/Flights/

```
# A tibble: 3 x 19
  year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
  <int> <int> <int>    <dbl>        <dbl>    <dbl>    <int>        <dbl>    <dbl> <chr>
1  2013     1     1      517        515      2       830        819      11  UA
2  2013     1     1      533        529      4       850        830      20  UA
3  2013     1     1      542        540      2       923        850      33  AA
# ... with 9 more variables: flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
# distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
> daily <- flights %>%
+   mutate(date = make_date(year, month, day)) %>%
+   count(date) %>%
+   mutate(wday = wday(date, label = TRUE))
> head(daily, n = 3)
# A tibble: 3 x 3
  date           n wday
  <date>       <int> <ord>
1 2013-01-01     842  Tue
```

Environment History Connections Tutorial

Import Dataset 346 MiB List

Global Environment

daily 365 obs. of 3 variables

\$ date: Date[1:365], format: "2013-01-01" "2013-01-02" ...  
\$ n : int [1:365] 842 943 914 915 720 832 933 899 902...  
\$ wday: Ord.factor w/ 7 levels "Sun"<"Mon"<"Tue"<...: 3 ...

Files Plots Packages Help Viewer

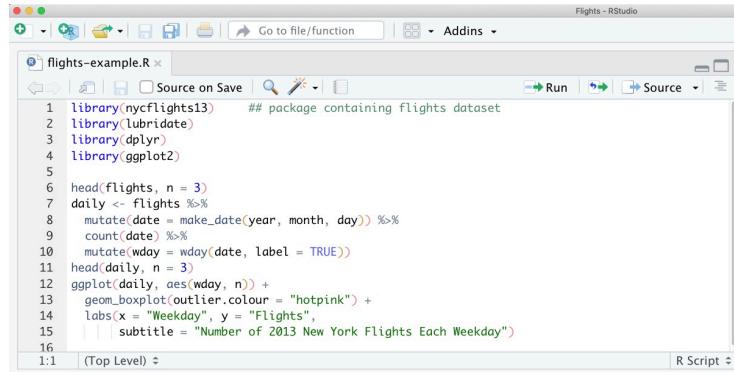
Zoom Export

Number of 2013 New York Flights Each Weekday

Flights

# Interfaz de RStudio

## Editor



```
flights-example.R x
library(nycflights13)    ## package containing flights dataset
library(lubridate)
library(dplyr)
library(ggplot2)

head(flights, n = 3)
daily <- flights %>%
  mutate(date = make_date(year, month, day)) %>%
  count(date) %>%
  mutate(wday = wday(date, label = TRUE))
head(daily, n = 3)
ggplot(daily, aes(wday, n)) +
  geom_boxplot(outlier.colour = "hotpink") +
  labs(x = "Weekday", y = "Flights",
       subtitle = "Number of 2013 New York Flights Each Weekday")
1:1 (Top Level) ▾ R Script ▾
```

Se encuentra en la parte superior izquierda. Aquí es donde se crean y modifican scripts.

# Interfaz de RStudio

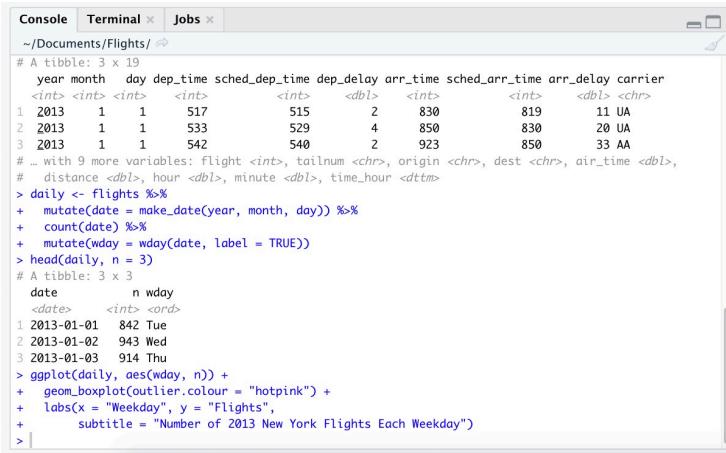
## Entorno de variables (environment)



Se encuentra en la parte superior derecha.  
Aquí se muestran los objetos disponibles  
que hemos creado.

# Interfaz de RStudio

## Consola

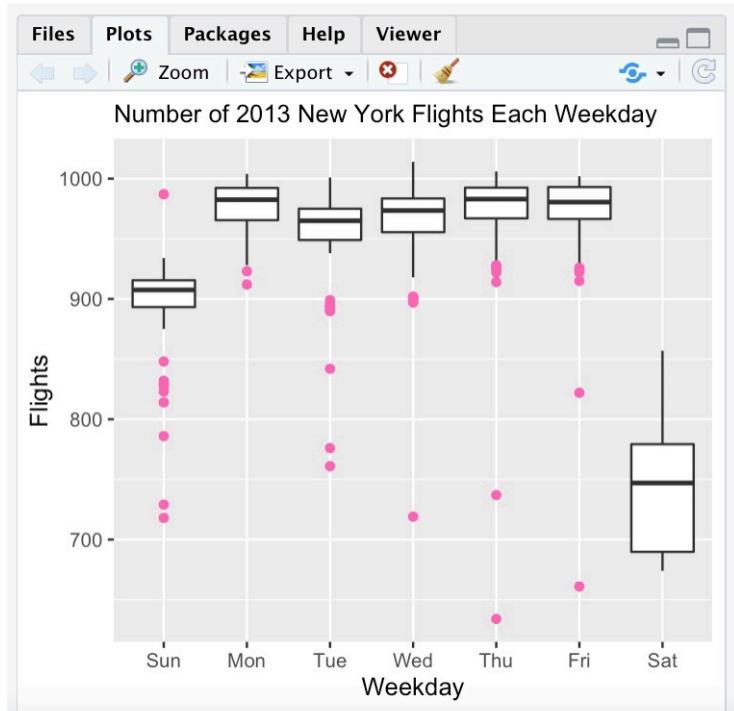


```
Console Terminal × Jobs ×
~/Documents/Flights/ ↵
# A tibble: 3 × 19
  year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
<int> <int> <int> <int> <dbl> <int> <dbl> <int> <dbl> <chr>
1 2013     1     1    517      515     2    830      819     11 UA
2 2013     1     1    533      529     4    850      830     20 UA
3 2013     1     1    542      540     2    923      850     33 AA
# ... with 9 more variables: flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
# distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
> daily <- flights %>%
+   mutate(date = make_date(year, month, day)) %>%
+   count(date) %>%
+   mutate(wday = wday(date, label = TRUE))
> head(daily, n = 3)
# A tibble: 3 × 3
  date          n wday
  <date>     <int> <ord>
1 2013-01-01  842 Tue
2 2013-01-02  943 Wed
3 2013-01-03  914 Thu
> ggplot(daily, aes(wday, n)) +
+   geom_boxplot(outlier.colour = "hotpink") +
+   labs(x = "Weekday", y = "Flights",
+        subtitle = "Number of 2013 New York Flights Each Weekday")
>
```

Se encuentra en la parte inferior izquierda, y es donde se encontrarán la salidas de la ejecución código y cálculos. También es posible escribir código directamente en la consola.

# Interfaz de RStudio

## Utilidades



Se encuentra en la parte inferior derecha.  
Aquí encontramos:

- El panel *Files* que muestra todos los archivos y carpetas del directorio raíz.
- El panel *Plots* que muestra los gráficos creados con R.
- El panel *Packages* que muestra los paquetes instalados y opciones de instalación de paquetes.
- El panel *Help* que muestra la información de una consulta que hagamos.
- El panel *Viewer* donde se puede observar gráficos dinámicos hechos

# Instalar y cargar paquetes

Son colecciones de funciones, documentación tablas de datos, y en ocasiones utilidades como Addins para Rstudio, que permiten llegar a un objetivo establecido.

- **Instalar un paquete**

```
install.packages("sf")
```

- **Instalar varios paquetes**

```
install.packages(c("sf", "terra"))
```

- **Instalar la versión en desarrollo de un paquete**

```
install.packages("devtools")
devtools::install_github("https://github.c
```

- **Cargar la librería**

```
library(sf)
```

# Buscar ayuda

## Dentro de RStudio

- Abrir una ventana de ayuda

```
help.start()
```

- Buscar información acerca de un tema específico

```
help.search("linear regression")
```

- Buscar ayuda sobre una función

```
? ...
```

## Web

- Stackoverflow
  - Rpubs
- Inteligencias Artificiales

# Comentarios

Cuando escribimos código recordemos que alguien más lo va a leer.

- Para comentar una línea se pone el símbolo numeral `#` antes de lo que se quiera comentar.
- Para comentar un párrafo o una sección completa primero se debe seleccionar el contenido a comentar y a continuación presionar las teclas: `ctrl + shift + C`

```
# Esto es un comentario  
library(terra)
```

- Añadir una nueva sección: `ctrl + shift + R`

```
# Este es el título de mi sección -----
```

# Mensajes

Cuando un mensaje aparece, R está indicando el correcto funcionamiento de esa función o sentencia.

- Ejemplo: al cargar la librería tidyverse el mensaje indica todos los conflictos y detalles que implica usar esta librería en el environment.

```
> library(tidyverse)
— Attaching core tidyverse packages ━━━━━━━━━━━━━━━━ tidyverse 2.0.0 ━━━
✓ dplyr    1.1.4   ✓ readr    2.1.5
✓forcats  1.0.0   ✓ stringr  1.5.1
✓ ggplot2  3.5.1   ✓ tibble   3.2.1
✓ lubridate 1.9.3  ✓ tidyr    1.3.1
✓ purrr   1.0.2
— Conflicts ━━━━━━━━━━━━━━━━ tidyverse_conflicts() ━━━
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package to force all conflicts to become errors
```

# Errores

En presencia de errores, R tratará de ser lo más informativo en cuanto al error y su posible ubicación en nuestro código.

**Un error, no devuelve resultados a menos que se escriba una excepción.**

```
> "a"+"b"
```

```
Error in "a" + "b" : argumento no-numérico para operador binario
```

# Advertencias

R devolverá un resultado pero te dirá qué consideraciones debes tomar acerca de tus resultados.

```
> as.numeric("a")
[1] NA
Warning message:
NAs introducidos por coerción
```

**¡Muchas gracias por tu  
atención!**