

# ANÁLISIS Y TRATAMIENTO DE DATOS CON R

*Con ejemplos e ilustraciones*

**Primera Edición**

Diego Paul Huaraca S.  
MS-PLUS, INC.

*Un aporte de Source Stat Lab Ecuador a la sociedad.*

---

## Índice general

|  |           |
|--|-----------|
| <b>1. Instalación y actualización</b>        | <b>3</b>  |
| 1.1. Descarga de R . . . . .                 | 3         |
| 1.2. Instalación de R . . . . .              | 5         |
| 1.3. Entorno de trabajo . . . . .            | 9         |
| 1.3.1. Archivo . . . . .                     | 9         |
| 1.3.2. Editar . . . . .                      | 10        |
| 1.3.3. Visualizar . . . . .                  | 11        |
| 1.3.4. Misc . . . . .                        | 12        |
| 1.3.5. Paquetes . . . . .                    | 13        |
| 1.3.6. Ventanas . . . . .                    | 14        |
| 1.3.7. Ayuda . . . . .                       | 14        |
| 1.4. Instalación de paquetes . . . . .       | 15        |
| 1.4.1. Repositorio CRAN . . . . .            | 16        |
| 1.4.2. Repositorios externos . . . . .       | 19        |
| 1.4.3. Github . . . . .                      | 19        |
| 1.5. Información de paquetes . . . . .       | 21        |
| 1.6. Actualización de paquetes . . . . .     | 23        |
| 1.7. Actualización de R . . . . .            | 23        |
| 1.8. Obteniendo ayuda . . . . .              | 24        |
| 1.9. Mostrando ejemplos . . . . .            | 26        |
| <b>2. Entornos de desarrollo</b>             | <b>28</b> |
| 2.1. RStudio . . . . .                       | 28        |
| 2.1.1. Instalación y actualización . . . . . | 29        |
| 2.1.2. Funcionamiento . . . . .              | 29        |
| 2.1.3. Ancho de impresión . . . . .          | 30        |
| 2.1.4. Prompt . . . . .                      | 31        |
| 2.1.5. Decimales . . . . .                   | 31        |
| 2.1.6. Respalando información . . . . .      | 32        |
| 2.2. R Analytic Flow . . . . .               | 32        |
| 2.2.1. Ventajas . . . . .                    | 33        |
| 2.2.2. Desventajas . . . . .                 | 33        |

# 1

---

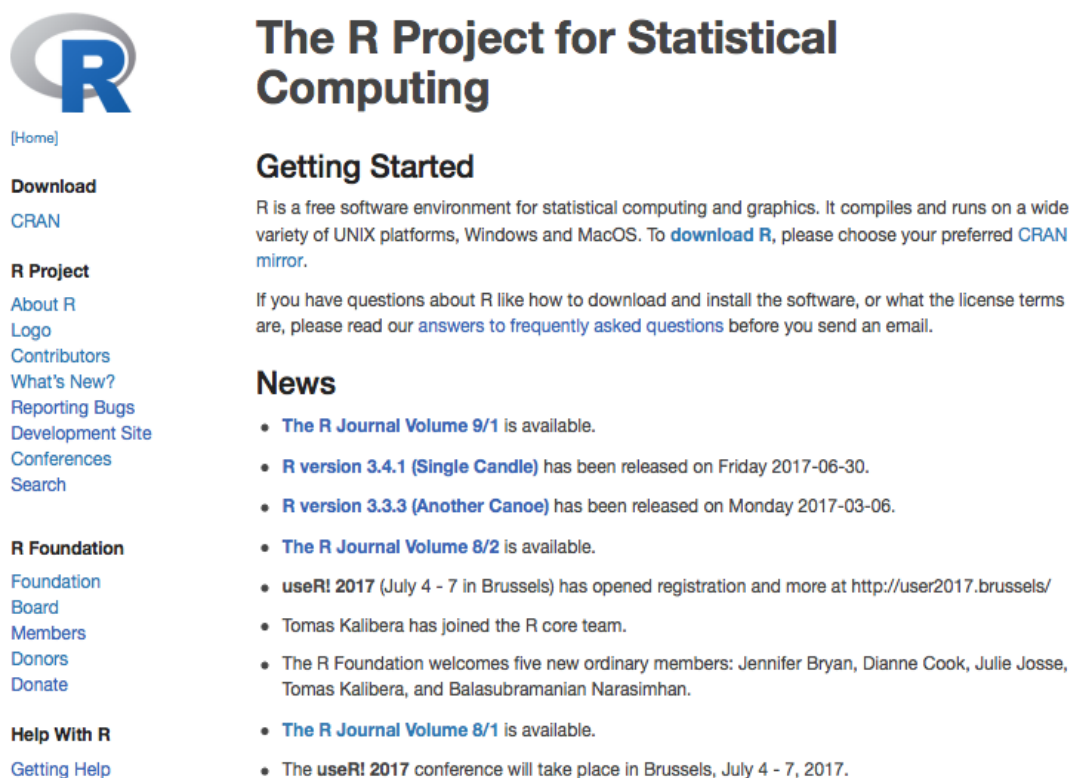
## Instalación y actualización

En esta sección mostraremos los pasos a seguir para la instalación y actualización del programa R, adicionalmente, se intentará dar una explicación amplia sobre las librerías o paquetes que complementan el funcionamiento del programa. Asumiremos que el usuario se encuentra trabajando sobre un computador con sistema Windows o Mac, sin embargo, no existe mayor diferencia en los pasos a seguir en el caso que el usuario utilice un sistema operativo o plataforma diferente.

### 1.1. Descarga de R

La descarga del programa R se describe en los siguientes pasos:

1. Accedemos al sitio web del R Project a través de un navegador tecleando la dirección: <http://www.r-project.org>.



The R Project for Statistical Computing

[\[Home\]](#)

**Download**  
[CRAN](#)

**R Project**  
[About R](#)  
[Logo](#)  
[Contributors](#)  
[What's New?](#)  
[Reporting Bugs](#)  
[Development Site](#)  
[Conferences](#)  
[Search](#)

**R Foundation**  
[Foundation](#)  
[Board](#)  
[Members](#)  
[Donors](#)  
[Donate](#)

**Help With R**  
[Getting Help](#)

**Getting Started**

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To [download R](#), please choose your preferred [CRAN mirror](#).

If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

**News**

- [The R Journal Volume 9/1](#) is available.
- [R version 3.4.1 \(Single Candle\)](#) has been released on Friday 2017-06-30.
- [R version 3.3.3 \(Another Canoe\)](#) has been released on Monday 2017-03-06.
- [The R Journal Volume 8/2](#) is available.
- [useR! 2017](#) (July 4 - 7 in Brussels) has opened registration and more at <http://user2017.brussels/>
- Tomas Kalibera has joined the R core team.
- The R Foundation welcomes five new ordinary members: Jennifer Bryan, Dianne Cook, Julie Josse, Tomas Kalibera, and Balasubramanian Narasimhan.
- [The R Journal Volume 8/1](#) is available.
- The [useR! 2017](#) conference will take place in Brussels, July 4 - 7, 2017.

Figura 1.1: Página web R-Project

- Nos dirigimos al enlace de la CRAN ubicado en la parte superior izquierda y elegimos un repositorio cercano a nuestra localidad. En nuestro caso, seleccionamos el repositorio de la Escuela Politécnica de Litoral.

Ecuador

<http://cran.espol.edu.ec/>

Figura 1.2: Repositorio Espol

- En la parte superior de la página nos dirigimos a la sección **Download and install R**, seguido elegimos el sistema operativo del equipo en el cual deseamos instalar el programa.

**The Comprehensive R Archive Network**

---

**Download and Install R**

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

---

**Source Code for all Platforms**

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (Friday 2017-06-30, Single Candle) [R-3.4.1.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

---

**Questions About R**

- If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

Figura 1.3: Selección del Sistema Operativo

- Elegimos el subdirectorio **base**.

**R for Windows**

Subdirectories:

|  |  |
|--|--|
| <a href="#">base</a><br><br><a href="#">contrib</a><br><br><a href="#">old contrib</a><br><br><a href="#">Rtools</a> | <p>Binaries for base distribution (managed by Duncan Murdoch). This is what you want to <a href="#">install R for the first time</a>.</p> <p>Binaries of contributed CRAN packages (for R &gt;= 2.11.x; managed by Uwe Ligges). There is also information on <a href="#">third party software</a> available for CRAN Windows services and corresponding environment and make variables.</p> <p>Binaries of contributed CRAN packages for outdated versions of R (for R &lt; 2.11.x; managed by Uwe Ligges).</p> <p>Tools to build R and R packages (managed by Duncan Murdoch). This is what you want to build your own packages on Windows, or to build R itself.</p> |
|--|--|

Please do not submit binaries to CRAN. Package developers might want to contact Duncan Murdoch or Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the [R FAQ](#) and [R for Windows FAQ](#).

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

Figura 1.4: Subdirectorios

5. Finalmente, damos click sobre **Download R X.Y.Z for Windows**, con lo cual descargamos la versión de R más reciente.

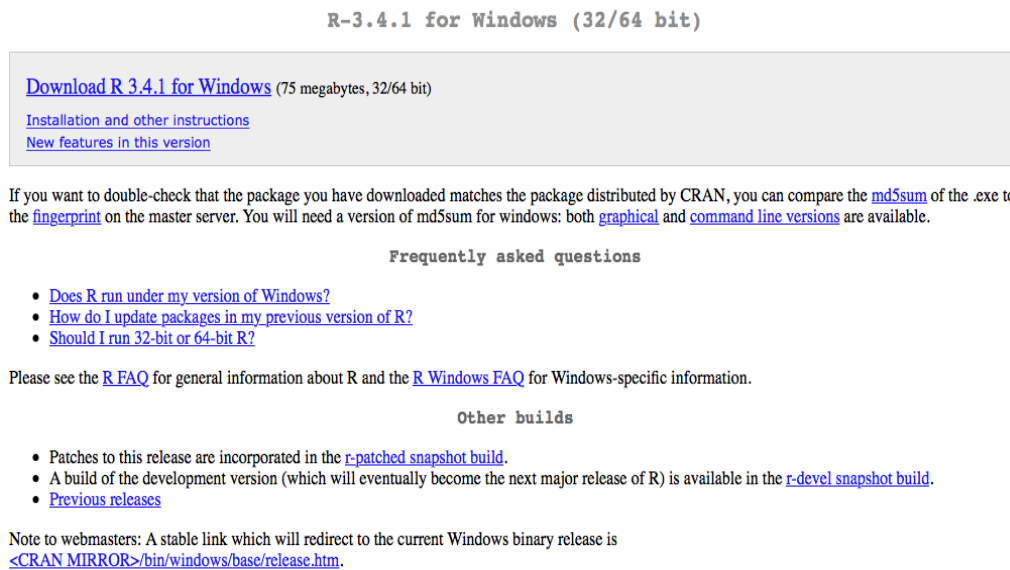


Figura 1.5: Descarga programa

El ejecutable obtenido corre sin inconvenientes sobre la plataforma de 32 o 64 bits, razón por la cual el usuario no debe preocuparse en este detalle.

## 1.2. Instalación de R

Una vez obtenido el ejecutable del programa procedemos con la instalación, para las plataformas Windows y Mac el procedimiento es similar, en el caso de la plataforma GNU Linux al final de esta sección se detalla su procedimiento:

1. Damos doble click sobre el ejecutable y seleccionamos el idioma que nos guiará en el proceso de instalación.

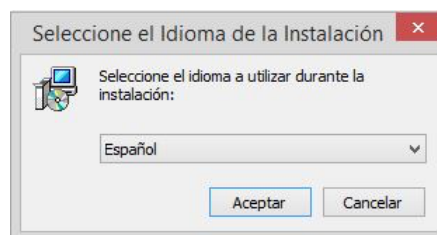


Figura 1.6: Seleccionamos Español como idioma

Cabe aclarar que la selección del idioma aplica únicamente a la etapa de instalación y no en el posterior funcionamiento del programa.

Por *default* el idioma del programa y sus complementos es el *inglés*.

2. En las siguientes dos ventanas damos click sobre el botón *Siguiente*. En caso de requerirlo ponemos revisar los términos establecidos para el uso del software.

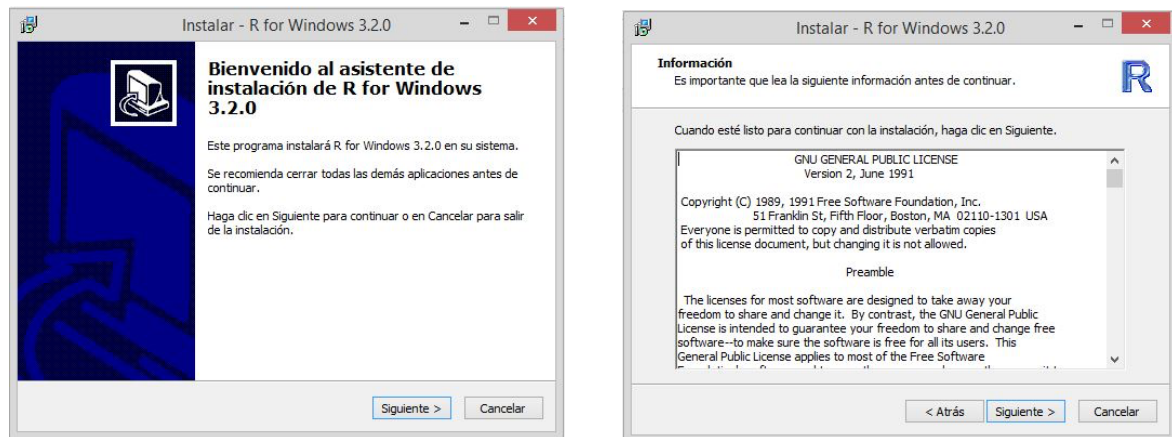


Figura 1.7: Click en Siguiente

3. Seleccionamos el lugar en disco para albergar los archivos del programa o simplemente aceptamos la dirección por defecto (recomendado).

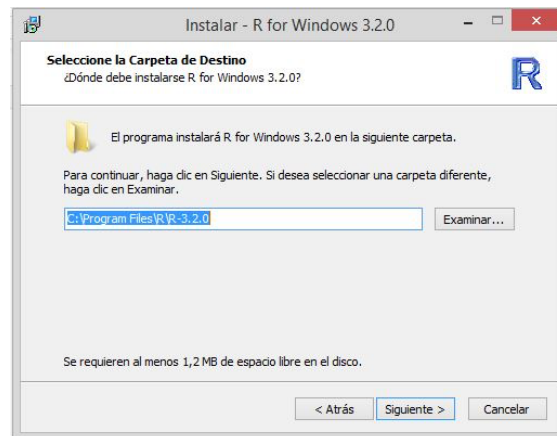


Figura 1.8: Seleccionamos el directorio de instalación

4. Seleccionamos los componentes del programa ha ser instalados, por defecto se instalan todos los componentes, sin embargo no es necesario (se podría instalar únicamente los dos primeros componentes).

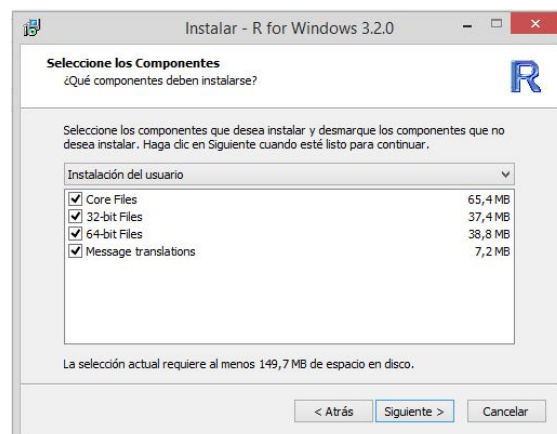


Figura 1.9: Componentes del programa

5. Para las siguientes ventanas simplemente damos click en el botón *Siguiente* de acuerdo a la configuración mostrada a continuación:

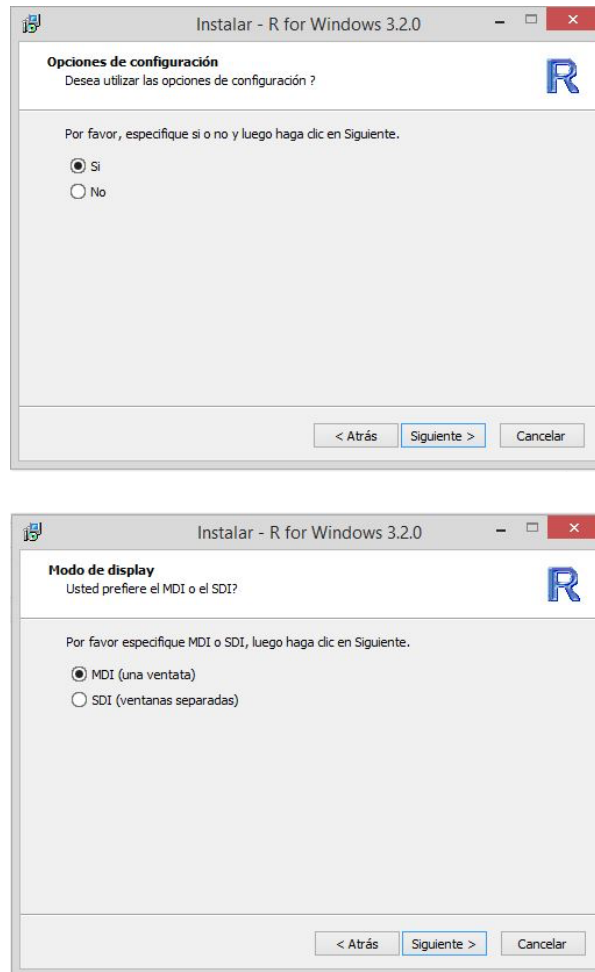


Figura 1.10: Click en *Siguiente*

6. Para la configuración del *proxy* debemos tomar en consideración si la red de internet con la cual trabajamos tiene algún tipo de seguridad o no. En caso afirmativo, seleccionaremos *Internet2* (recomendado), caso contrario *Standard*.

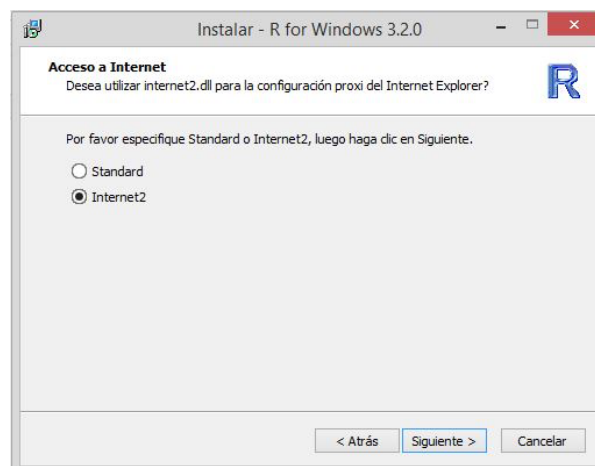


Figura 1.11: Configuración Proxy



7. En las últimas dos ventanas damos click sobre el botón *Siguiente* de acuerdo a la configuración mostrada a continuación:

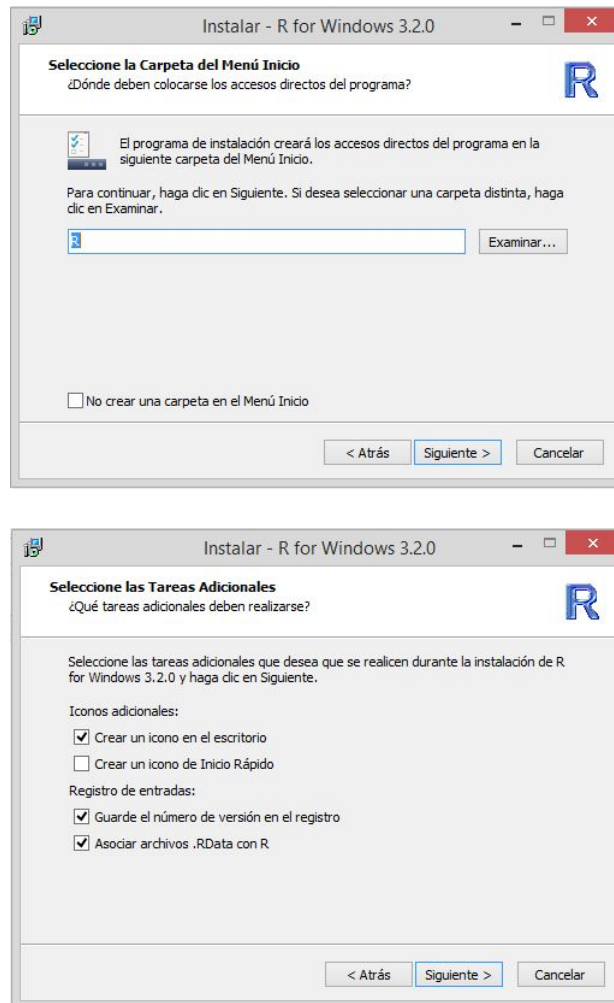


Figura 1.12: Click en **Siguiente**

Con los pasos antes descritos lograremos tener el programa 100 % funcional y listo para que el usuario pueda utilizarlo sin problemas en su computador.

La instalación de R en sistemas operativos GNU Linux se realiza tecleando el siguiente comando en la terminal:

```
~$ sudo apt-get install r-base
```

para añadir el repositorio de CRAN tecleamos:

```
~$ sudo gedit/etc/apt/sources.list
```

El auge tecnológico de los últimos años ha permitido ejecutar varios de los programas clásicos de escritorio sobre teléfonos inteligentes, a lo cual R no es la excepción pues lo podemos correr sobre la plataforma Android<sup>1</sup> logrando de esta manera llevar el programa en todo momento y lugar.

Para los usuarios de iPhone existe una versión de R en iTunes denominada **R Programming Language** la cual es mantenida por Megakey Trans.

<sup>1</sup>[https://play.google.com/store/apps/details?id=com.appsopensource.R&hl=es\\_419](https://play.google.com/store/apps/details?id=com.appsopensource.R&hl=es_419)

## 1.3. Entorno de trabajo

Es de gran importancia para el usuario de R conocer el entorno de trabajo, y es precisamente lo que trataremos en esta sección. Iniciaremos explicando la barra de herramientas, la cual consta de las secciones: Archivo, Editar, Visualizar, Misc, Paquetes, Ventanas y Ayuda.

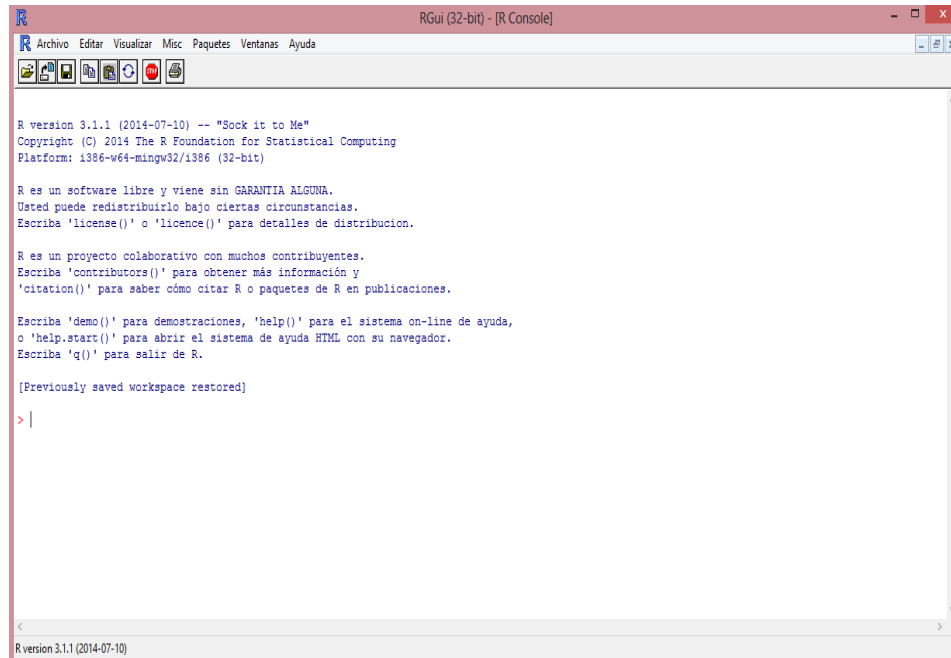


Figura 1.13: Entorno de trabajo de R

### 1.3.1. Archivo

En esta sección se podrá tratar todo lo relacionado con el área de trabajo, manejo de archivos y salida del programa. Los elementos que lo componen son:

- **Interpretar código fuente R:** Hay que recordar que R es un lenguaje de programación derivado del S, es por ello que podemos realizar programas en un editor externo, con esta opción podemos ejecutar dicho programa en la consola siempre y cuando, la extensión de guardado sea del tipo .R.
- **Nuevo script:** Si lo ejecutamos, se nos abrirá un editor de lenguaje R para crear script que luego podemos llamar desde la consola, el tipo de archivo que podemos generar son R o S.
- **Abrir script:** Los script que hemos generados o que tengamos del tipo R o S podemos abrirlos en un editor para poder editarlos.
- **Mostrar archivo(s):** Siver para abrir y poder visualizar o editar cualquier archivo, es interesante para los que tengan relación con el lenguaje R o S. Se diferencia con la opción Abrir script de que éste no puede editarlos directamente, simplemente visualizarlos. Se pueden abrir varios archivos a la vez.
- **Cargar área de trabajo:** Pues como su nombre indica, sirve para cargar un área de trabajo que hayamos configurado y guardado previamente, es útil por ejemplo, cuando hemos definido el tipo de letra, el espacio de trabajo, los colores, etc, y queremos usarlo. La extensión común es .RData aunque también puede cargarse entornos de trabajo con formato antiguo tipo .rda.

- **Guardar área de trabajo:** Cuando se haya configurado el entorno de consola de R, colores, tipo de letra, tamaño, etc, podemos guardarlo para cargarlo posteriormente en futuras aplicaciones ya que el propio programa R no guarda, de momento, dicha configuración. El formato del entorno de trabajo es `.RData`.
- **Cargar Histórico:** Podemos cargar el archivo de comandos que se hayan ejecutado en una sesión previamente guardada. El formato de salida es `.history`.
- **Guardar Histórico:** Con él, podemos guardar los comandos que hayamos introducido por consola en una sesión. El formato de guardado es `.history`.
- **Cambiar dir:** Podemos configurar el directorio de trabajo que está definido por defecto cuando se instaló el programa.
- **Imprimir:** Podemos configurar e imprimir el entorno de trabajo de R, la consola.
- **Guardar en archivo:** Se guardará todo lo que se haya escrito por la consola en formato `.txt` que después podremos recuperar.
- **Salir:** Sirve para salir del programa R, antes nos preguntará si queremos guardar el área de trabajo.

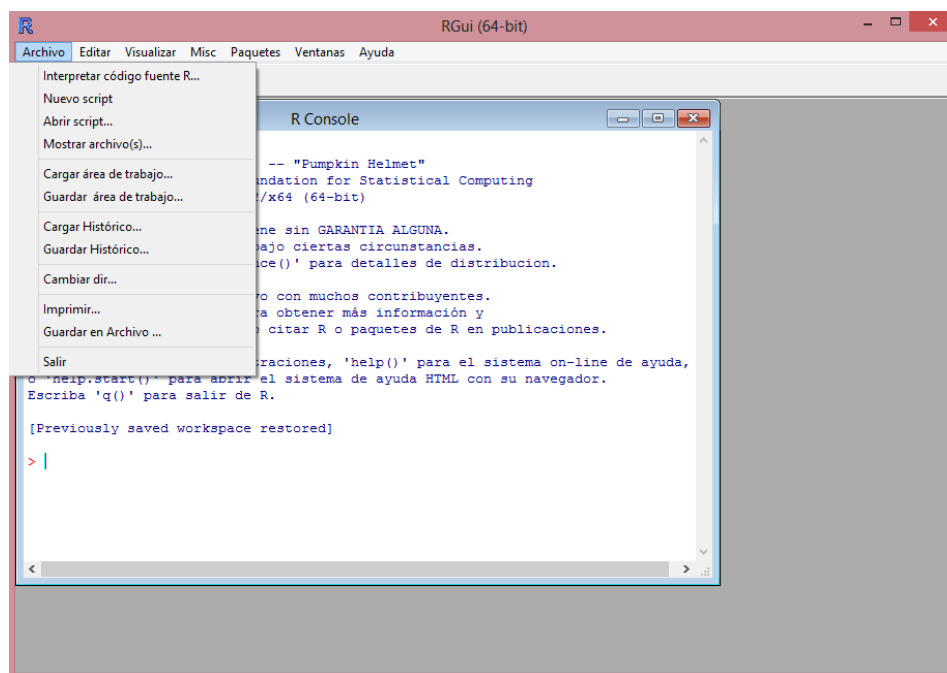


Figura 1.14: Sección Archivo

### 1.3.2. Editar

Es la sección encargada de la edición de datos y de la configuración del entorno de trabajo.

- **Copiar:** Podemos copiar al portapapeles el comando o sentencia o lo que queramos de la consola. Para acceder de forma rápida por teclado se debe pulsar: `Ctrl + C`.
- **Pegar:** Podemos pegar en la consola aquello que tengamos en el portapapeles. Para acceder de forma rápida por teclado se debe pulsar: `Ctrl + V`.

- **Pegar solo comandos:** La diferencia básica con la opción Pegar es que con esta opción, sólo se pegarán en la consola aquello que sea comandos para ejecutarse.
- **Copiar y Pegar:** Todo aquello que copiamos, directamente se pegará, de forma inmediata, en la consola. Para acceder de forma rápida por teclado se debe pulsar: Ctrl + X.
- **Seleccionar todo:** Como su nombre indica, se selecciona todo lo que esté presente en la consola.
- **Limpiar consola:** Se borrarán todo lo que esté presente en la consola, pero ojo, no se borrarán las variables y estructuras definidas. Es útil cuando tenemos en la consola mucha información que ya no es útil. Para acceder de forma rápida por teclado se debe pulsar: Ctrl + L.
- **Editor de datos:** Podemos definir datos (vectores, estructuras, funciones, etc) y guardarlos para posteriormente llamarlos en la consola. Estos datos estarán definidos en la consola cuando se guarde.
- **Preferencias de la interfaz gráfica:** En esta opción es donde podremos configurar todo lo relacionado a la visualización de texto reflejado en la consola: tamaño de letra, colores, tipo de letra, etc. También, se puede configurar el entorno de trabajo, para multiventana o una única ventana.

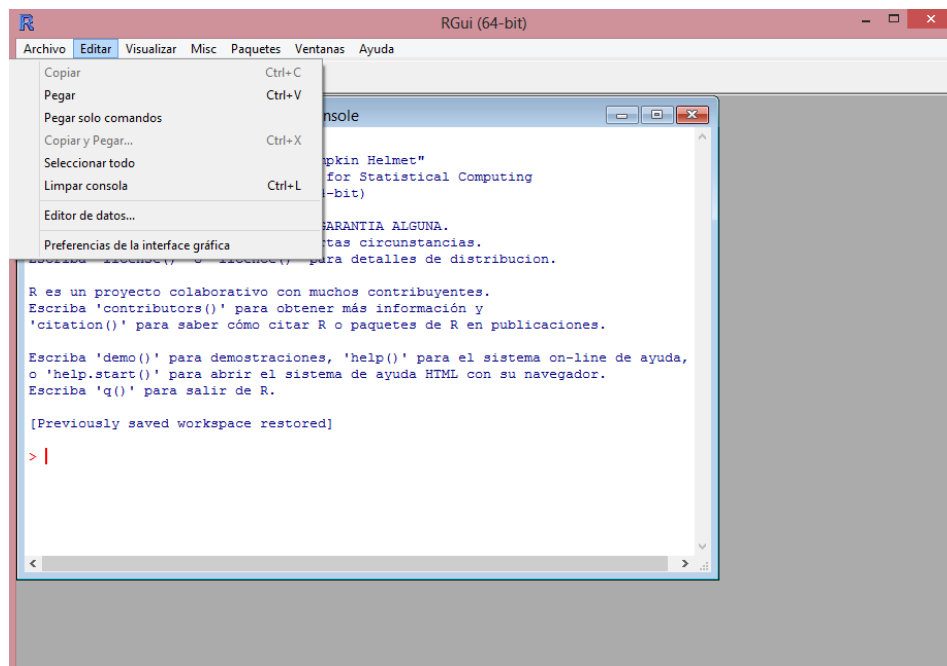


Figura 1.15: Sección Editar

### 1.3.3. Visualizar

Activa la visualización de algunos iconos en la parte superior tales como: Abrir script, Cargar área de trabajo, etc., mismas que ya se detallaron anteriormente.

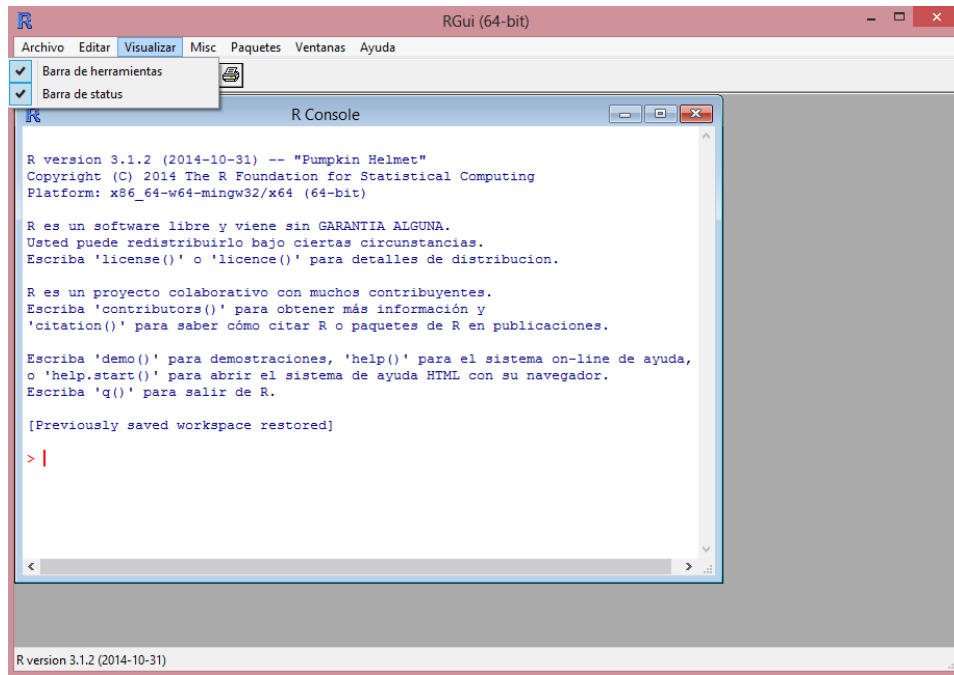


Figura 1.16: Sección Visualizar

#### 1.3.4. Misc

Esta es la sección denominada misceláneas, donde hay más de un control interesante.

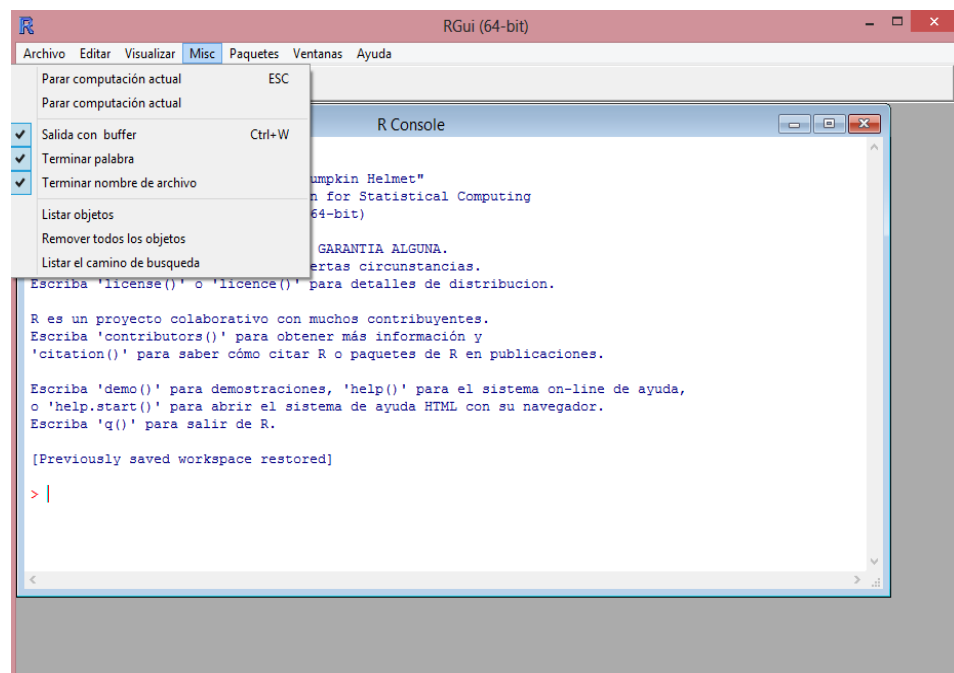


Figura 1.17: Sección Misc

- **Parar computación actual:** Este control es muy interesante y útil, con él, podremos parar cualquier archivo, sentencia o código que esté ejecutando la consola R. Para acceder de forma rápida por teclado se debe pulsar: ESC.
- **Salida con buffer:** Para acceder de forma rápida por teclado se debe pulsar: Ctrl + W.

- **Terminar palabra:** Es una ayuda interactiva, que nos ayuda a completar las palabras mientras estamos escribiendo en caso que la consola las reconozca.
- **Terminar nombre de archivo:** Realiza la misma función que la opción Terminar palabra pero en archivos.
- **Listar objetos:** Se nos mostrará por consola los objetos que hasta en este momento hemos definido en la consola.
- **Remover todos los objetos:** Como su nombre indica, elimina de memoria todos los objetos que hayamos definido (datos, variables, matrices, vectores, etc) en la consola de R. Cuando pulsemos sobre dicha opción, el programa, nos preguntará si realmente queremos borrarlos.
- **Listar el camino de búsqueda:** Nos ofrece por consola las librerías y complementos que tenemos instalados en R.

### 1.3.5. Paquetes

Dedicado al manejo de las librerías que posee el programa R.

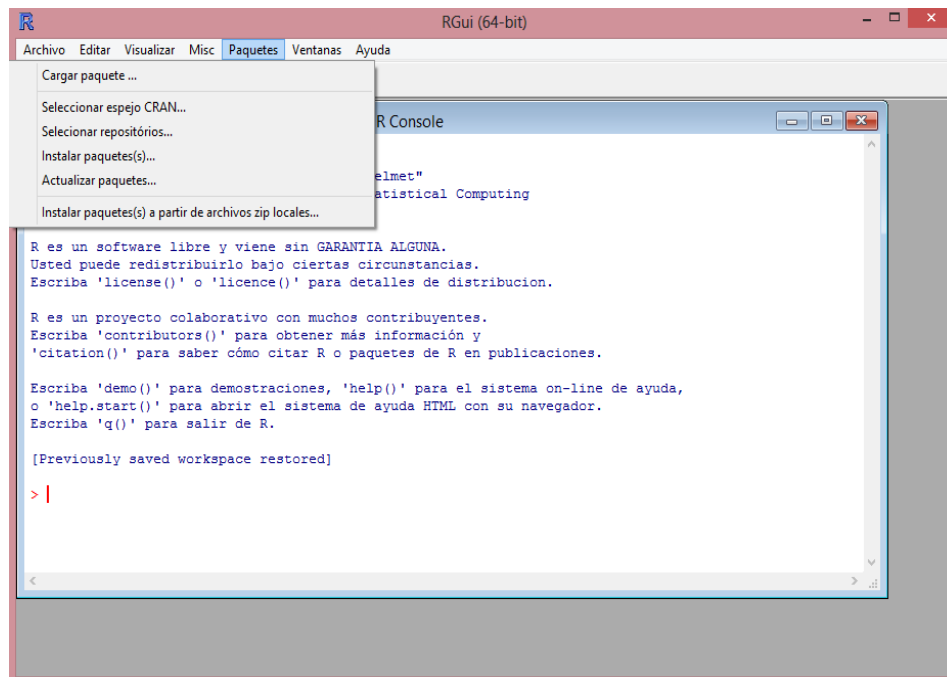


Figura 1.18: Sección Paquetes

- **Cargar paquete:** Con él, podemos cargar en la consola cualquier librería que tengamos instalada.
- **Seleccionar espejo CRAN:** Sirve para configurar el servidor de librerías.
- **Seleccionar repositorios:** En consola nos mostrará los repositorios que tenemos instalados y nos pedirá cual usar para la sesión activa.
- **Instalar paquetes(s):** Podremos actualizar o instalar librerías nuevas en red, para ello, debemos elegir un servidor.
- **Actualizar paquetes(s):** Podemos actualizar las librerías que tengamos instaladas en caso de haber una actualización reciente por red, para ello, debemos seleccionar un servidor

- **Instalar paquetes(s) a partir de archivos zip locales:** En caso de haber descargado una librería y tenerla en nuestro ordenador, podemos instalarlo mediante esta opción.

### 1.3.6. Ventanas

Permite configurar la visualización de las ventanas: consola, área de gráficos y editor de script en modo vertical, horizontal, etc.

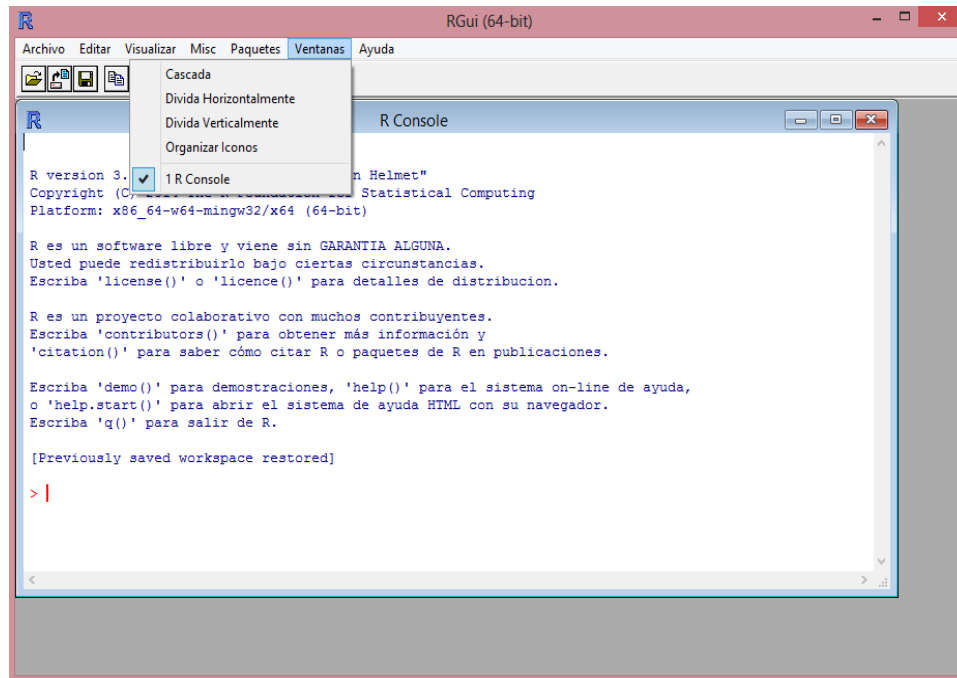


Figura 1.19: Sección Ventanas

### 1.3.7. Ayuda

Es un apartado que debemos tenerlo siempre presente, ya que en él se dispone al usuario, los manuales de utilización y específicos de R.

- **Console:** Nos mostrará en una ventana los atajos por teclado que posee la consola.
- **FAQ en R:** Ayuda en html que se carga desde nuestro ordenador, donde nos ofrece las preguntas frecuentes que se suelen hacer los usuarios al utilizar R. No hace falta estar conectado a Internet.
- **FAQ en R para Windows:** Preguntas frecuentes para los usuarios de Windows.
- **Manuales en PDF:** Una recopilación de los manuales para el manejo del programa, esta opción es bastante interesante.
- **Funciones R(texto):** Es una opción donde podremos introducir sentencias de comando para obtener ayuda sobre ellas, es bastante útil.
- **Ayuda Html:** Se nos abrirá en el navegador una ayuda interactiva, no es necesario estar conectado a Internet.
- **Html search page:** Consiste en una página html donde podremos buscar las instrucciones que queramos consultar.

- **Página principal R:** Es un enlace directo a la web del proyecto R.
- **Página principal de CRAN:** Enlace directo al directorio de librerías de R.
- **Sobre:** Indica información de la compilación que se tenga instalada de R.

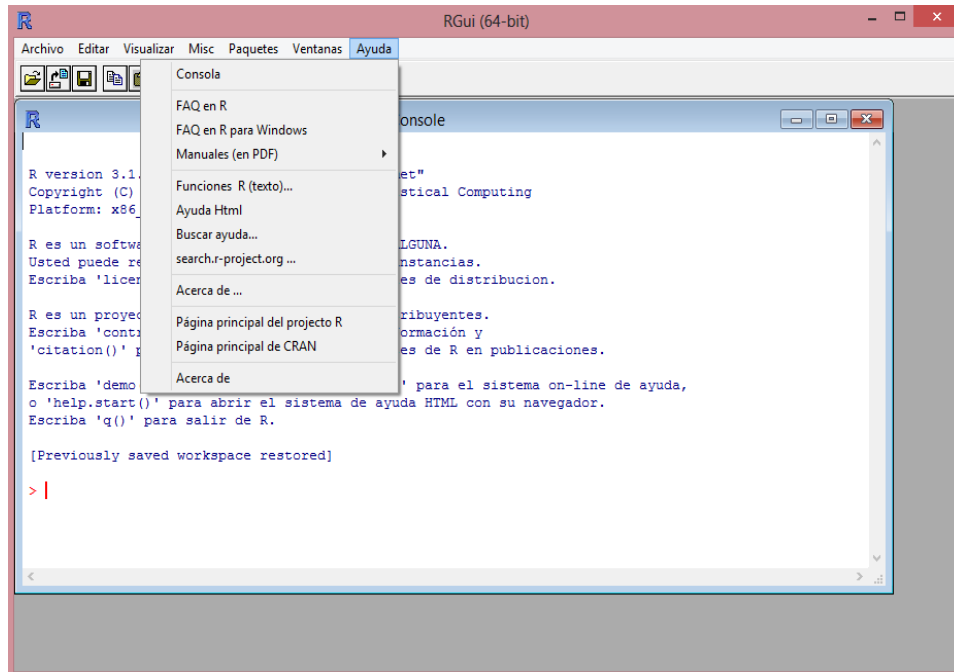


Figura 1.20: Sección Ayuda

## 1.4. Instalación de paquetes

Como habíamos señalado en la sección anterior, una de las principales ventajas de R es su capacidad para incrementar su funcionalidad mediante la incorporación de nuevas bibliotecas o paquetes.

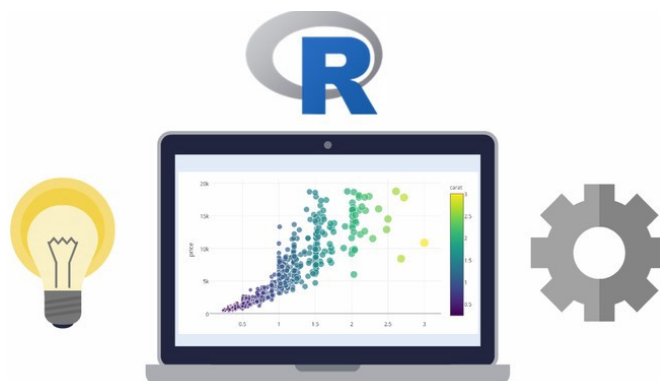


Figura 1.21: Aumento de funcionalidades mediante paquetes

Para entender de mejor manera la necesidad de emplear paquetes usaremos la siguiente metáfora:

*Imaginemos que tener instalado el programa en el computador es como haber adquirido un auto nuevo el cual cumple con ciertas funciones principales y no trae contratiempos, una vez que*



lo empezamos a rodar nos vamos dando cuenta de la necesidad de mejorar el rendimiento de ciertos mecanismos ya existentes del auto debido que lo vamos exponiendo a diferentes ambientes y condiciones. Lo anterior pasa exactamente con R, cuando descargamos e instalamos el software contamos con ciertas características y paquetes básicos pero existen miles de paquetes adicionales que pueden ser agregados para lograr mejorar su funcionalidad y a la vez realizar cosas estupendas.

### 1.4.1. Repositorio CRAN

La instalación de paquetes en R desde el repositorio CRAN es sencillo, una vez conocido el nombre del paquete a ser instalado basta con introducir el siguiente comando en la consola:

```
install.packages("nombre_paquete")
```

Ciertos paquetes de R requieren la instalación de otros paquetes adicionales debido que entre estos comparten algunas funciones (*paquetes sugeridos*) para evitar tener inconvenientes con el funcionamiento total de un paquete se aconseja adicionar el parámetro siguiente:

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

En el caso que se desee instalar una lista de  $n$  paquetes planteamos la siguiente solución:

```
# Listado de paquetes a instalar
paquetes <- c("pckg_1", "pckg_2", ... , "pckg_n")

# Sentencia de instalación
lapply(seq_along(paquetes), function(i){
  install.packages(paquetes[[i]], dependencies=TRUE)
})
```

Una segunda alternativa al momento de instalar los paquetes de R consiste en obtener el archivo zipeado (.zip) desde el repositorio CRAN e instalarlo manualmente a través del menú del programa.

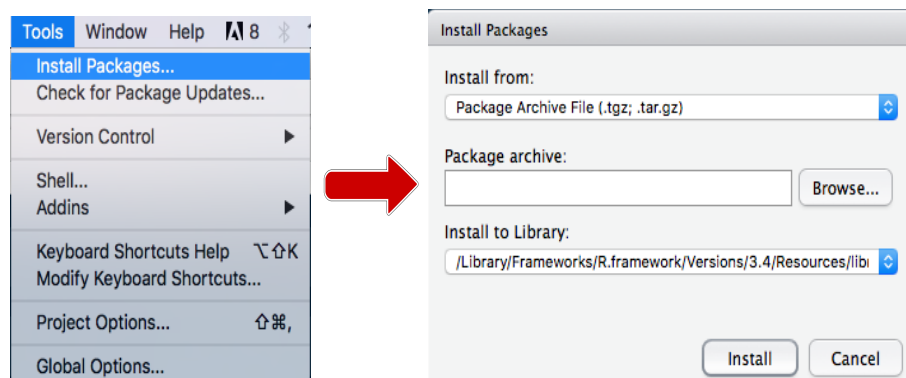


Figura 1.22: Instalación de paquetes mediante archivo .zip

Una vez que se ha instalado un paquete el paso siguiente consiste en *cargar* las funciones y datos del mismo dentro del área de trabajo, para esto disponemos dos comandos: `library` y `require`, sin embargo, lo más recomendable es utilizar el primero de ellos como se explica por medio del siguiente ejemplo:

Para mostrar la diferencia entre de los comandos anteriores tratamos de cargar el paquete `translate2R`, mismo que no ha sido descargado e instalado previamente.

```
library('translate2R')

## Error in library("translate2R") :
## there is no package called translate2R

require('translate2R')

## Loading required package: translate2R
## Warning message:
## In library(package, lib.loc=lib.loc, character.only=TRUE,
##           logical.return=TRUE,
##           : there is no package called translate2R
```

Al emplear `library` observamos que nos arroja un *error*, mientras que al emplear `require` únicamente tenemos una *advertencia*. En el caso puntual que nos encontremos trabajando en un código complejo la segunda opción nos causaría serios problemas dado que permite continuar con la ejecución del proceso sin que el usuario se percate, mientras que la primera opción alerta al usuario y detiene la ejecución del código. Por todo lo antes mencionado, la gran mayoría de usuarios R prefieren usar el comando `library` para cargar los paquetes.

En el caso que se desee cargar  $n$  paquetes se tiene la siguiente solución:

```
# Listado de paquetes a cargar
paquetes <- c("pkg_1", "pkg_2", ... , "pkg_n")

# Sentencia de carga
lapply(paquetes, FUN=library, character.only=TRUE)
```

Si deseamos visualizar el listado completo de los paquetes instalados podemos recurrir al comando:

```
library()
```

Además, si deseamos conocer el lugar en el cual se instalan los paquetes tecleamos:

```
.libPaths()

## [1] "/Library/Frameworks/R.framework/Versions/3.4/Resources/library"
```

En algunas ocasiones los usuarios desean descargar los paquetes dentro de una carpeta específica (`C:/pkg_down`) e instalar los paquetes en otra carpeta diferente (`D:/pkg_inst`), a continuación mostramos una posible solución al problema:

```
# Descarga en repositorio 'destdir' e instalación en 'lib'
install.packages("pkgs", destdir="C:/pkg_down", lib="D:/pkg_inst")

# Instalación en repositorio 'lib.loc'
library("pkgs", lib.loc="D:/pkg_inst")
```

Para conocer la totalidad de paquetes que se encuentran albergados en la CRAN, y a su vez verificar cuáles de ellos se encuentran válidos y disponibles usamos los comandos:

```
installed.packages()  
available.packages()
```

En el caso que se requiera conocer los paquetes que se encuentra cargados en el área de trabajo empleamos el comando:

```
sessionInfo()  
  
## R version 3.4.0 (2017-04-21)  
## Platform: x86_64-apple-darwin15.6.0 (64-bit)  
## Running under: macOS Sierra 10.12.6  
##  
## Matrix products: default  
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib  
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib  
##  
## locale:  
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8  
##  
## attached base packages:  
## [1] stats      graphics  grDevices  utils      datasets  methods    base  
##  
## other attached packages:  
## [1] knitr_1.15.1  
##  
## loaded via a namespace (and not attached):  
## [1] compiler_3.4.0 magrittr_1.5   tools_3.4.0   stringi_1.1.5  
## [5] highr_0.6      stringr_1.2.0 evaluate_0.10
```

El comando anterior imprime información acerca de la versión y plataforma del ejecutable de R que se encuentra utilizando, adicionalmente muestra información de la localidad y de los paquetes adjuntos o cargados.

Para ocasiones en las cuales se desee eliminar un paquete previamente instalado en R tenemos dos alternativas: la primera consiste básicamente en eliminar la carpeta que contiene dicho paquete del repositorio del computador mientras que la segunda alternativa consiste en emplear el siguiente comando:

```
remove.packages("nombre_paquete", "directorio")
```

donde:

- **nombre\_paquete:** Nombre o vector de nombres de los paquetes a ser eliminados.
- **directorio:** Es un vector de caracteres que proporcionan la dirección del directorio del cual se eliminará el paquete. Si se omite dicho parámetro se asume por default la dirección obtenida por `.libPaths()`.

### 1.4.2. Repositorios externos

Habíamos comentado sobre la existencia de varios repositorios adicionales al CRAN de entre los cuales destacan R-Forge<sup>2</sup>, BioConductor<sup>3</sup> y Omegahat<sup>4</sup>, dentro de estos repositorios se pueden encontrar paquetes que no se encuentran en el CRAN, o bien, versiones más actualizadas de los mismos.

Para instalar los paquetes desde los repositorios externos antes mencionados basta la siguiente sentencia:

```
install.packages("pckgname", repos="http://r-forge.r-project.org")
install.packages("pckgname", repos="http://www.omegahat.org/R")
install.packages("pckgname", repos="http://www.bioconductor.org/
packages/2.10/bioc")
```

Algunos de los paquetes se encuentran en formato binario, para su instalación basta adicionar el parámetro `type="source"`.

### 1.4.3. Github

GitHub es un repositorio de archivos y proyectos el cual emplea un sistema de control de versiones conocido como Git. Github fue lanzado a inicios del 2010, el código alojado en el repositorio se almacena de forma pública aunque también se puede almacenar de forma privada siempre y cuando se tenga una cuenta de pago.

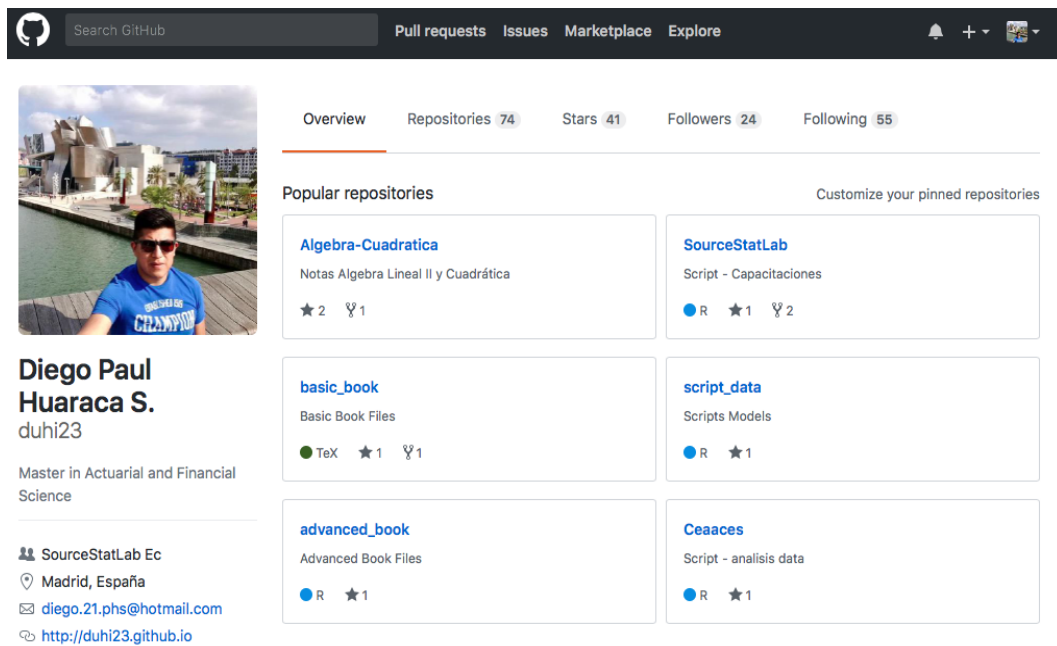


Figura 1.23: Página de Github

La instalación y sincronización de Git con RStudio se realiza de la siguiente manera:

1. Descargar e instalar la plataforma específica de Git, misma que puede ser obtenida desde la página web <http://www.git-scm.com/>. Su instalación se realiza con todas las opciones por *default*.

<sup>2</sup>Sistema para el desarrollo y versionamiento de paquetes en R.

<sup>3</sup>Subproyecto dedicado a la investigación en bioestadística.

<sup>4</sup>Subproyecto dedicado al desarrollo de interfaces para la programación estadística basado en Java.

2. Configurar Git con los comandos globales a través de la versión bash, esto permitirá que Git tenga conexión con los repositorios de Github para el envío/recepción de archivos, teclear lo siguiente:

```
git config --global user.name "nombre de la cuenta de Github"
git config --global user.email "Github-email@something.com"
```

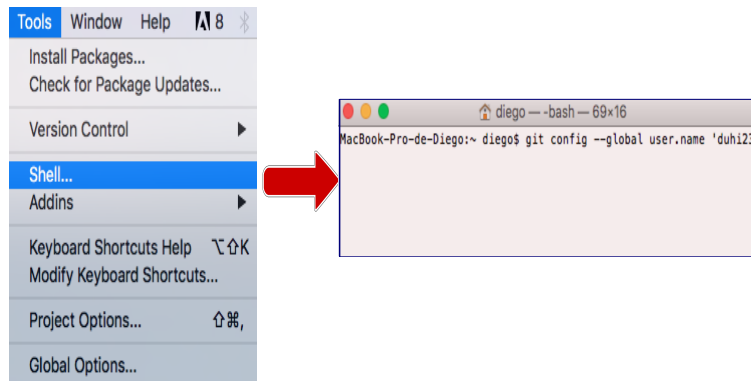


Figura 1.24: Configuración acceso a Git

3. Por último, abrir RStudio y configurar la ruta del ejecutable de Git:  
Ir a Tools > Options > Git/SVN.

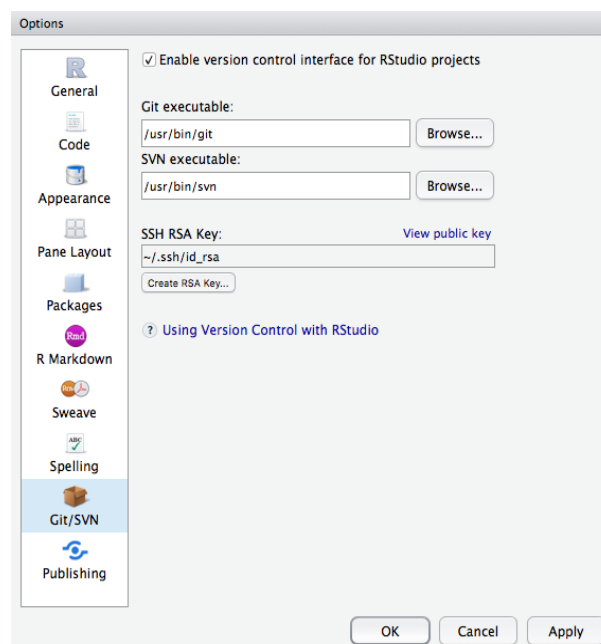
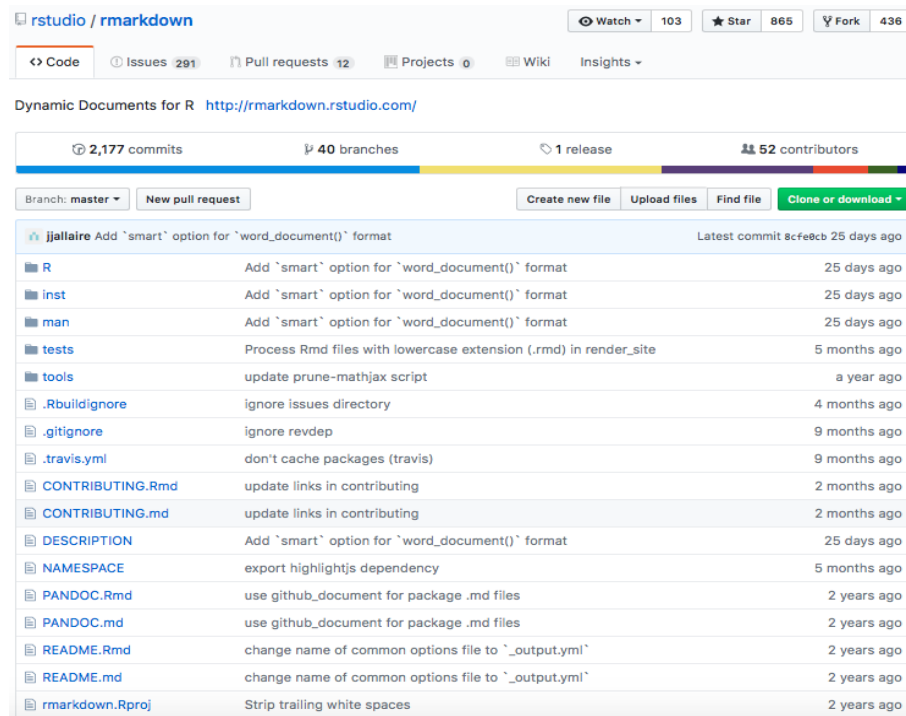


Figura 1.25: Configuración ruta de Git

En la actualidad, gran cantidad de usuarios R en todo el mundo han alojado o se encuentran desarrollando sus proyectos en el sistema de control de versiones GitHub, Inc<sup>5</sup>. por lo cual se hace necesario contar con una solución al momento de instalar algún paquete alojado en dicho sistema.

<sup>5</sup><https://github.com/>

Figura 1.26: Repositorio en Github del paquete **rmarkdown**

Nuestra solución consiste en lo siguiente:

```
install.packages('devtools', dependencies=TRUE)
devtools::install_github("rstudio/rmarkdown")
```

Iniciamos instalando el paquete **devtools**, el cual contiene la función **install\_github**, misma que se encargará de acceder al sistema Github e instalar el paquete deseado, para esta ocasión hemos seleccionado el paquete **rmarkdown** alojado en el repositorio Github dentro del proyecto **rstudio**.

## 1.5. Información de paquetes

En el caso que se requiera información acerca de un paquete '**pckgname**', previamente instalado podemos recurrir al comando:

```
library(help='pckgname')
```

```

Information on package 'stats'

Description:

Package:      stats
Version:      3.4.0
Priority:      base
Title:        The R Stats Package
Author:       R Core Team and contributors worldwide
Maintainer:   R Core Team <R-core@r-project.org>
Description:  R statistical functions.
License:      Part of R 3.4.0
Imports:      utils, grDevices, graphics
Suggests:     MASS, Matrix, SuppDists, methods, stats4
NeedsCompilation: yes
Built:        R 3.4.0; x86_64-apple-darwin15.6.0; 2017-04-21 20:26:55 UTC; unix

Index:

.checkMFClasses      Functions to Check the Type of Variables passed
                      to Model Frames
AIC                   Akaike's An Information Criterion
ARMAacf               Compute Theoretical ACF for an ARMA Process
ARMAtoMA              Convert ARMA Process to Infinite MA Process
Beta                  The Beta Distribution
Binomial              The Binomial Distribution
Box.test              Box-Pierce and Ljung-Box Tests
C                     Sets Contrasts for a Factor

```

Figura 1.27: Información del paquete **stats**

Una vez instalado un paquete, podemos enlistar todas las funciones que han sido implementadas dentro del mismo como se muestra en el siguiente ejemplo:

```

library(foreign)
ls('package:foreign')

## [1] "data.restore" "lookup.xport" "read.arff"    "read.dbf"
## [5] "read.dta"     "read.epiinfo" "read.mtp"     "read.octave"
## [9] "read.S"       "read.spss"    "read.ssd"     "read.systat"
## [13] "read.xport"   "write.arff"   "write.dbf"    "write.dta"
## [17] "write.foreign"

```

Para conocer los parámetros de cada una de las funciones implementadas dentro de un paquete podemos recurrir al siguiente comando:

```
lsf.str('package:foreign')
```

Por último, cuando un paquete es empleado dentro de una investigación científica o académica surge la necesidad de citar a los autores del paquete, esto lo solventamos mediante el comando `citation`.

```

citation("foreign")

## To cite package foreign in publications use:
##
## R Core Team (2014). foreign: Read Data Stored by Minitab, S, SAS,
## SPSS, Stata, Systat, Weka, dBase, .... R
## package version 0.8-61. http://CRAN.R-project.org/package=foreign
##
## A BibTeX entry for LaTeX users is
##

```

```
## @Manual{,
##   title = {foreign: Read Data Stored by Minitab, S, SAS, SPSS,
##   Stata, Systat, Weka, dBase, ...},
##   author = {{R Core Team}},
##   year = {2014},
##   note = {R package version 0.8-61},
##   url = {http://CRAN.R-project.org/package=foreign},
## }
```

## 1.6. Actualización de paquetes

En el caso que se desee actualizar todos los paquetes previamente instalados contamos con el comando `update.packages`, el mismo que revisa las actualizaciones de los paquetes. El comando anterior preguntará al usuario si desea actualizar uno por uno los paquetes, lo cual es tedioso pues si contamos con un gran número de paquetes nos demandará bastante tiempo aprobar la actualización de los mismos. Para subsanar lo anterior basta añadir el parámetro `ask = TRUE`.

```
update.packages(ask=TRUE)
```

En el caso que se desee conocer únicamente los paquetes que fueron instalados previamente y en la actualidad constan con un versionamiento emplearemos lo siguiente:

```
old.packages()
```

El comando anterior nos mostrará la versión del paquete que ha sido instalada y la última versión disponible del mismo.

Para finalizar la sección dejamos un comando útil para los desarrolladores que deseen conocer información de contacto de los desarrolladores de paquetes, así como también información de las personas que dan mantenimiento a los mismos:

```
lapply(dir(R.home("library")), packageDescription)
```

## 1.7. Actualización de R

Mantener actualizado R es relativamente fácil a través de los comandos:

```
install.packages('installr', dependencies = TRUE)
library('installr')
updateR()
```

Iniciamos instalando el paquete **installr**, acto seguido procedemos a cargar el paquete a través del comando `library`, y finalmente, mediante el comando `updateR` verificamos la versión más reciente del programa y descargamos la misma.

En el caso que se desee conocer la versión de R que se encuentra instalada podemos teclear en la consola el comando:



```
R.version
##
## platform      x86_64-apple-darwin15.6.0
## arch          x86_64
## os            darwin15.6.0
## system        x86_64, darwin15.6.0
## status
## major         3
## minor         4.0
## year          2017
## month         04
## day           21
## svn rev       72570
## language      R
## version.string R version 3.4.0 (2017-04-21)
## nickname      You Stupid Darkness
```

## 1.8. Obteniendo ayuda

R nos facilita la vida al momento de indagar sobre la funcionalidad de ciertos comandos con tan sólo tener disponible una conexión de Internet. Por ejemplo, si el usuario se encuentra interesado en obtener información acerca de la función `seq()`, tecleamos en la consola lo siguiente:

```
help(seq)
```

Una forma alternativa para acceder a la ayuda es:

```
?seq
```

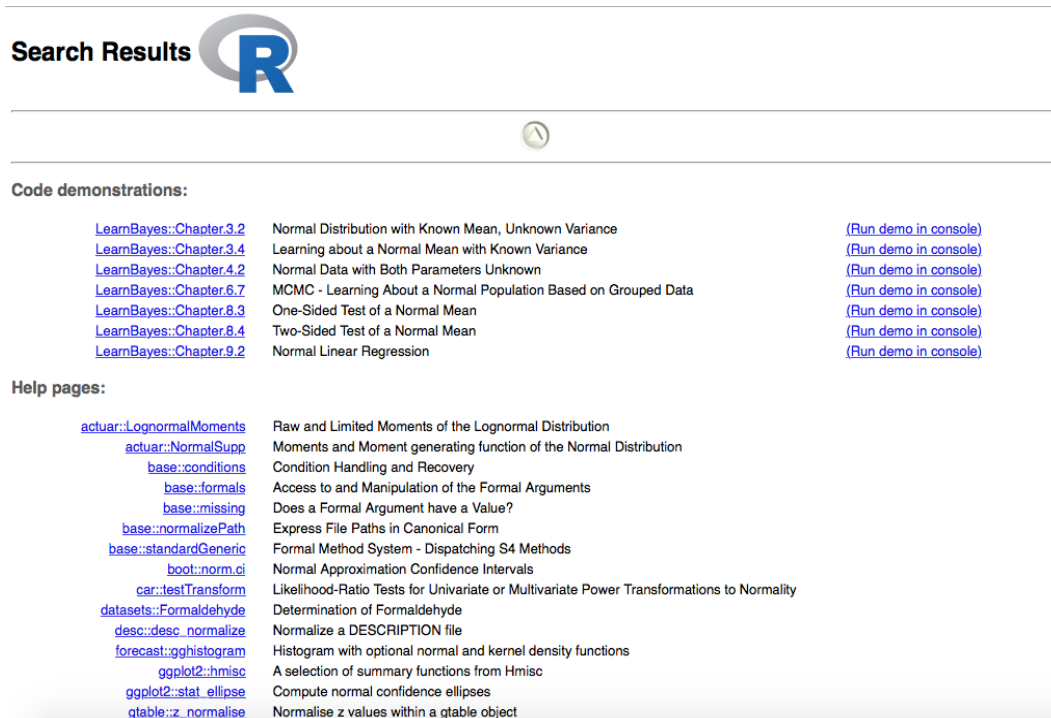
Para el caso que se encuentren trabajando con funciones especificadas por caracteres especiales, el argumento debería ir entre comillas, con el fin de transformarlo en una *cadena de caracteres*:

```
help("[")
```

Se pueden utilizar tanto comillas simples como dobles sin que esto conlleve problemas a posteriori.

Adicionalmente, si nos encontramos interesados en buscar información relacionada a un término en específico podemos teclear lo siguiente:

```
help.search("termino_especifico")
```

Figura 1.28: Resultado de la búsqueda: `help.search("normal")`

En el caso que se requiera un manual de ayuda completo en formato HTML, empleamos el comando:

```
help.start()
```

Por último, existen circunstancias en las cuales se hace indispensable conocer todas las funciones relacionadas a un nombre en específico. El comando `apropos` realiza la búsqueda en el listado de funciones que R maneja. A continuación, mostramos su uso por medio de ejemplos:

Primero, obtenemos todas las funciones relacionadas con el término **mean**.

```
apropos("mean")

## [1] ".colMeans"      ".rowMeans"      "colMeans"      "kmeans"
## [5] "mean"           "mean.Date"      "mean.default"  "mean.difftime"
## [9] "mean.POSIXct"   "mean.POSIXlt"   "rowMeans"      "weighted.mean"
```

Seguido, obtenemos todas las funciones que poseen la letra w al final de sus nombres.

```
apropos("w$")

## [1] "all.equal.raw"    "ar.yw"          "as.data.frame.raw"
## [4] "as.raw"           "charToRaw"      "dev.new"
## [7] "file.show"        "grepRaw"        "is.raw"
## [10] "layout.show"     "n2mfrow"        "new"
## [13] "nrow"             "NROW"           "pat_brew"
## [16] "pat_rnw"          "plot.new"       "plot.window"
## [19] "rainbow"          "raw"            "row"
## [22] "show"             "tryNew"         "url.show"
## [25] "View"             "window"
```

Ahora, obtenemos todas las funciones que poseen la letra x al inicio de sus nombres.

```
apropos("^x")

## [1] "x11" "X11"
## [3] "X11.options" "X11Font"
## [5] "X11Fonts" "xedit"
## [7] "xemacs" "xfig"
## [9] "xinch" "xor"
## [11] "xor.hexmode" "xor.octmode"
## [13] "xpdrows.data.frame" "xspline"
## [15] "xtabs" "xtfrm"
## [17] "xtfrm.AsIs" "xtfrm.Date"
## [19] "xtfrm.default" "xtfrm.difftime"
## [21] "xtfrm.factor" "xtfrm.numeric_version"
## [23] "xtfrm.POSIXct" "xtfrm.POSIXlt"
## [25] "xtfrm.Surv" "xy.coords"
## [27] "xyinch" "xyTable"
## [29] "xyz.coords" "xzfile"
```

Para finalizar, obtenemos las funciones que tiene un número entre el 4 y 8 en sus nombres.

```
apropos("[4-8]")

## [1] ".S4methods" "asS4" "enc2utf8" "fixPre1.8"
## [5] "Harman74.cor" "intToUtf8" "isS4" "seemsS4Object"
## [9] "state.x77" "utf8ToInt" "validUTF8"
```

## 1.9. Mostrando ejemplos

La mayoría de funciones incluyen ejemplos que pueden ser ejecutados y facilitan al usuario comprender de mejor manera su uso. Para ejecutar los ejemplos usaremos el comando `example`, como se muestra a continuación:

```
example(seq)

##
## seq> seq(0, 1, length.out = 11)
## [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
##
## seq> seq(stats::rnorm(20)) # effectively 'along'
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
##
## seq> seq(1, 9, by = 2) # matches 'end'
## [1] 1 3 5 7 9
##
## seq> seq(1, 9, by = pi) # stays below 'end'
## [1] 1.000000 4.141593 7.283185
##
## seq> seq(1, 6, by = 3)
## [1] 1 4
```

```
##
## seq> seq(1.575, 5.125, by = 0.05)
## [1] 1.575 1.625 1.675 1.725 1.775 1.825 1.875 1.925 1.975 2.025 2.075
## [12] 2.125 2.175 2.225 2.275 2.325 2.375 2.425 2.475 2.525 2.575 2.625
## [23] 2.675 2.725 2.775 2.825 2.875 2.925 2.975 3.025 3.075 3.125 3.175
## [34] 3.225 3.275 3.325 3.375 3.425 3.475 3.525 3.575 3.625 3.675 3.725
## [45] 3.775 3.825 3.875 3.925 3.975 4.025 4.075 4.125 4.175 4.225 4.275
## [56] 4.325 4.375 4.425 4.475 4.525 4.575 4.625 4.675 4.725 4.775 4.825
## [67] 4.875 4.925 4.975 5.025 5.075 5.125
##
## seq> seq(17) # same as 1:17, or even better seq_len(17)
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17

example(range)

##
## range> (r.x <- range(stats::rnorm(100)))
## [1] -2.435493 2.199566
##
## range> diff(r.x) # the SAMPLE range
## [1] 4.635058
##
## range> x <- c(NA, 1:3, -1:1/0); x
## [1] NA 1 2 3 -Inf NaN Inf
##
## range> range(x)
## [1] NA NA
##
## range> range(x, na.rm = TRUE)
## [1] -Inf Inf
##
## range> range(x, finite = TRUE)
## [1] 1 3
```

## 2

### Entornos de desarrollo

Un entorno de desarrollo integrado, también conocido como IDE (Integrated Development Environment) es un programa informático compuesto por un conjunto de herramientas de programación que contiene: un editor, un compilador, un depurador y un constructor de interfaz gráfica, el mismo que viene empaquetado como una **aplicación** que facilita de sobremanera la realización de operaciones al usuario mediante una serie de menús o mediante interacción con los objetos gráficos que aparecen en pantalla, a través de periféricos como: el ratón y teclado.

En este capítulo analizaremos dos IDE's de gran importancia y utilidad en el mundo de los usuarios de R:

- RStudio
- R Analytic Flow

#### 2.1. RStudio

RStudio es un entorno IDE de código abierto lanzado en Febrero 2011 el cual incluye una consola, un editor resaltado de texto que admite la ejecución directa, así como herramientas para gráficos, historial, depuración y gestión del espacio de trabajo. RStudio se encuentra disponible para todas las plataformas (Windows, Mac, Linux), además puede ser ejecutado a través de un navegador web<sup>1</sup>.

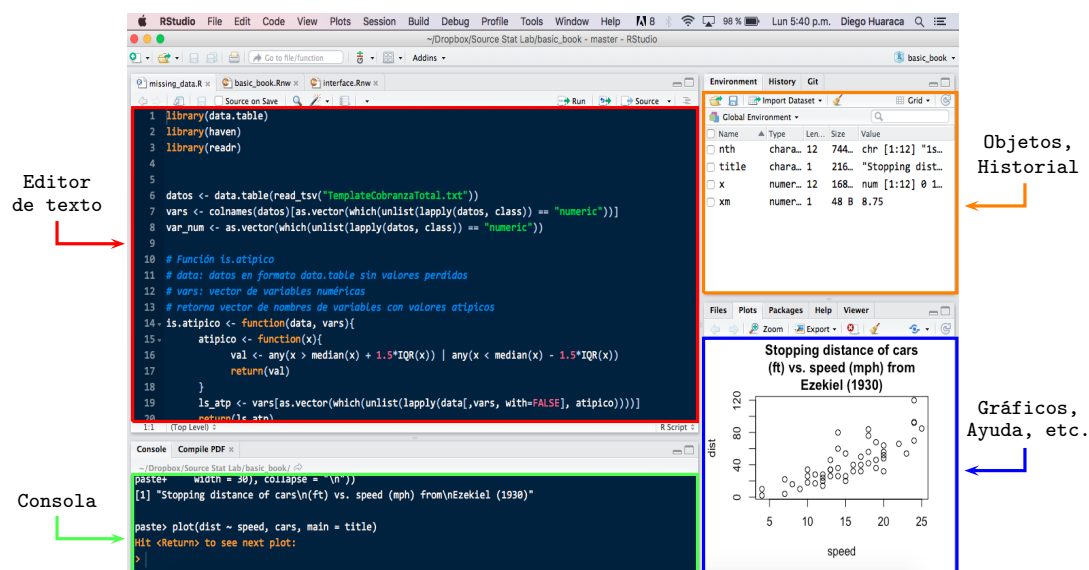


Figura 2.1: Entorno IDE de RStudio

<sup>1</sup>Opción válida para la versión **server**.

Con el propósito que el usuario realice un trabajo rápido y eficiente, RStudio incorpora atajos de teclado que reducen la dependencia del ratón, se recomienda a los nuevos usuarios hacer uso de los mismos y adquirir este buen hábito. Para acceder al listado de atajos se debe presionar la combinación de teclas<sup>2</sup>:

| SISTEMA         | COMBINACIÓN        |
|-----------------|--------------------|
| Windows / Linux | Alt + Shift + K    |
| Mac OS          | Option + Shift + K |

### 2.1.1. Instalación y actualización

RStudio puede ser obtenido libremente desde su página web <http://www.rstudio.org/>. Una vez obtenido el archivo ejecutable la instalación se la realiza de manera simple e intuitiva.

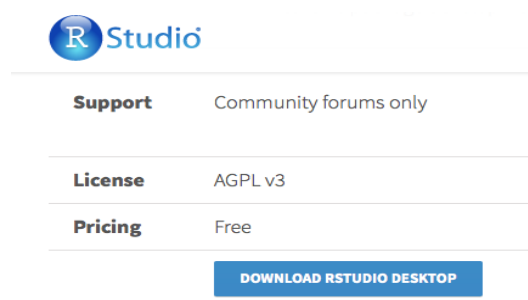


Figura 2.2: Descarga RStudio

Una forma de descarga e instalación más técnica se realiza a partir de los siguientes comandos:

```
# descargar e instalar el paquete installr
install.packages("installr")
# cargar el paquete installr
library(installr)
# instalamos RStudio IDE
install.RStudio()
```

La actualización del programa se realiza desde el menú: Help > Check for Updates.

### 2.1.2. Funcionamiento

RStudio ofrece una amplia integración con ficheros de diversos formatos: R scripts (.R), Mark-down (.md), LaTeX (.Rnw) entre otros. La facilidad en la generación de documentos dinámicos con RStudio y knitr han hecho que el programa se convierta en la IDE preferida por muchos usuarios de R.

El programa se encuentra organizado en cuatro ventanas de trabajo distintas:

- **Editor de código fuente:** Se encuentra en la zona superior izquierda, esta ventana nos permite abrir y editar ficheros con código R.
- **Consola:** Se ubica en la zona inferior izquierda, esta ventana es también conocida como consola y nos permite ejecutar comandos de R.

<sup>2</sup>En caso de requerir el listado completo de atajos recomendamos visitar la página: <https://support.rstudio.com/hc/en-us/articles/200711853-Keybaord-Shortcuts>.

- **Navegador de objetos:** La zona superior derecha posee dos ventanas auxiliares:
  - **Workspace:** En esta ventana se enlistan todos los objetos creados en memoria.
  - **History:** En esta ventana se almacena el histórico de las líneas de código que han sido ejecutadas en R.
- **Visualización e información:** Esta última ventana ubicada en la zona inferior derecha se encuentra conformada por 4 ventanas auxiliares:
  - **Files:** Provee el acceso al árbol de directorios y ficheros del disco duro.
  - **Plots:** Ventana auxiliar en la cual aparecen los gráficos creados en la consola.
  - **Packages:** Esta ventana facilita la administración de los paquetes de R instalados en el computador.
  - **Help:** Esta última ventana nos ayuda en la búsqueda de información respecto a un comando en específico.

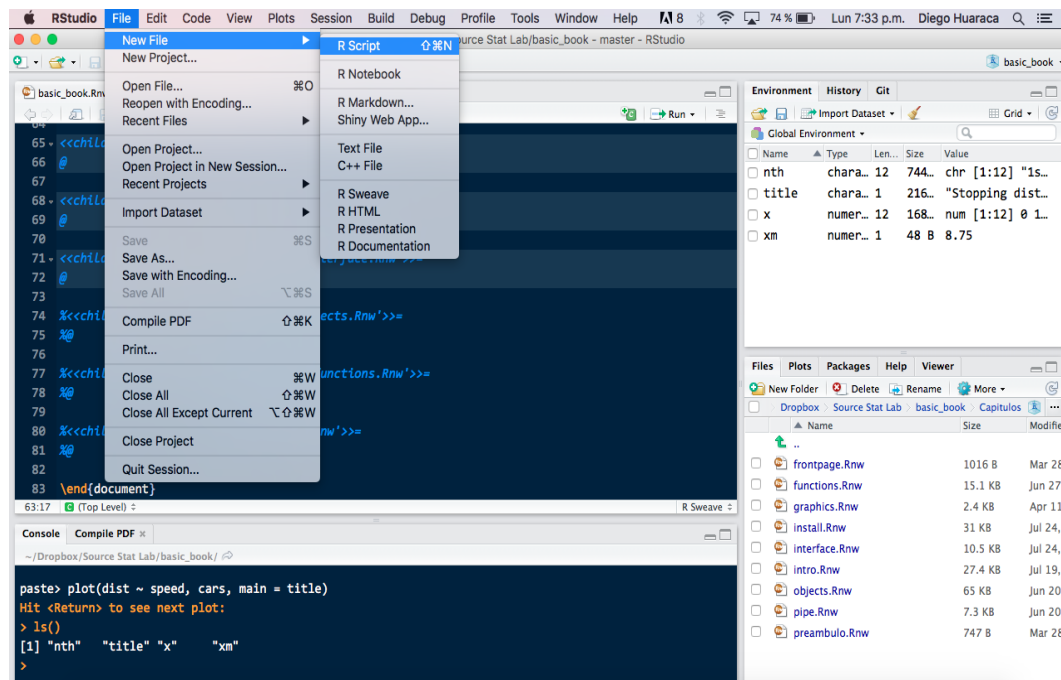


Figura 2.3: Formatos permitidos en RStudio

RStudio ofrece varios mecanismos para controlar varios aspectos de la evaluación durante una sesión. La función `options()` es empleada para compartir los valores de parámetros entre las funciones.

### 2.1.3. Ancho de impresión

Existen ocasiones en las cuales el usuario desea controlar el ancho de impresión de los resultados que se muestran en la pantalla, como primer paso para modificar el ancho de impresión debemos obtener el parámetro actual mediante:

```
getOption("width")
```

```
## [1] 75
```

Una vez conocido el ancho de pantalla actual procedemos a modificar el mismo cambiando el valor del parámetro `width`, de la siguiente manera:

```
options(width=40)
rnorm(10)

## [1] 0.6714312 0.9850095 0.1952725
## [4] 0.9658726 -0.3239620 -2.7130319
## [7] 1.0865178 0.9463704 -0.4491579
## [10] -1.2791995
```

```
options(width=55)
rnorm(10)

## [1] 1.47615095 -0.02445285 -0.53329803 0.29598874
## [5] -1.03588016 0.20675179 -0.28847083 0.70194712
## [9] 2.09013787 -0.50014789
```

```
options(width=70)
rnorm(10)

## [1] -0.5963025 -1.4479531 -2.0437566 1.2247703 -0.4182759 0.5951789
## [7] -0.2676930 1.2807380 0.3799919 2.1697036
```

#### 2.1.4. Prompt

Para los usuarios que deseen cambiar el símbolo `>` del prompt o interpretador por otro símbolo diferente como: `— >` o por un nombre, tenemos el siguiente código:

```
options(prompt="—>")
options(prompt="diego >")
```

#### 2.1.5. Decimales

Una preocupación adicional para los usuarios es la cantidad de decimales con la cual se muestran los resultados, dicha cantidad de decimales puede ser modificada y debe encontrarse en el rango de 1 a 22.

```
getOption("digits")

## [1] 7
```

R por default muestran los resultados con 7 decimales, sin embargo los mismos pueden ser modificados como se muestra a continuación:

```
options(digits=2)
rnorm(3)

## [1] 0.039 -0.751 -1.426
```



```
options(digits=5)
rnorm(3)

## [1] -0.11837  0.43202 -1.81202
```

```
options(digits=10)
rnorm(3)

## [1]  2.0947402348 -0.4062529089  0.2639165715
```

Existen opciones adicionales que pueden ser modificadas de acuerdo a las necesidades que tenga el usuario, para ver el listado completo de opciones podemos teclear en la consola el comando:

```
help(options)
```

### 2.1.6. Respaldo de información

Un tema importante dentro del análisis de datos es el respaldo de información que se pueda dar sobre ciertos resultados obtenidos, en este punto R consta de dos comandos muy útiles: `save()` & `load()`.

El primero de ellos permite almacenar en disco los objetos que desee el usuario (almacenamiento parcial), dicho comando puede ser configurado de tal manera que almacene todos los objetos que se encuentra válidos en el área de trabajo.

```
# si deseamos guardar el objeto "datos_banco" con el nombre "base"
save(datos_banco, file = "base.RData")
# para el caso que se desee almacenar todos los objetos con el nombre "info"
save(list = ls(all = TRUE), file = "info.RData")
```

El segundo comando nos va a permitir cargar los objetos guardados en el área de trabajo actual o en un ambiente determinado.

```
# cargamos el objeto base en el area de trabajo
load("base.RData")
# ahora cargamos "info" en un ambiente determinado "env"
load("info.RData", envir = env)
```

## 2.2. R Analytic Flow

R Analytic Flow (RAF) es una interfaz gráfica de usuario desarrollado por Ryota Suzuki<sup>3</sup>, que facilita el análisis de datos a través de diagramas de flujo. El software se encuentra bajo licencia BSD & GPL, por lo cual puede obtenerse de forma gratuita a través de la página web de Ef-prime, Inc. <http://www.ef-prime.com> para las plataformas: Windows, Mac OS X y Linux.

<sup>3</sup>Ryota Suzuki es un desarrollador de software orientado al análisis de datos, fundó con sus amigos la empresa Ef-prime, Inc. en Tokyo, además es el creador del paquete *pvc* de R.

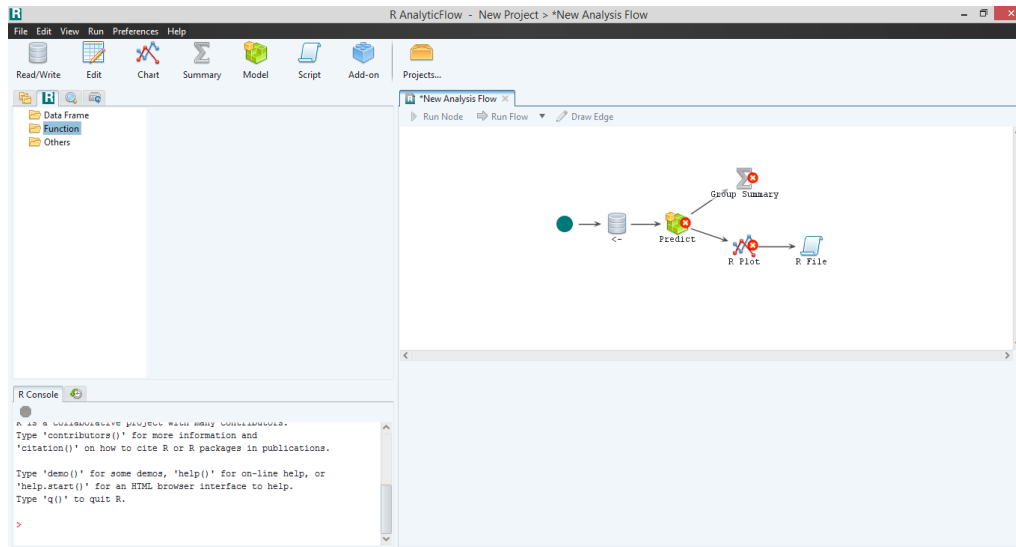


Figura 2.4: R Analytic Flow

R Analytic Flow permite insertar código de R enmarcado dentro de nodos con la capacidad de ejecutar diferentes rutinas a partir de determinadas conexiones entre nodos.

### 2.2.1. Ventajas

A continuación enumeramos algunas de las ventajas que posee R Analytic Flow:

1. Facilita ejecutar procesos a través de flujos.
2. Fácil implementación de tareas en cada nodo.
3. Reduce la complejidad a la hora de programar varias funciones que se relacionen entre sí.
4. El número de usuarios que usan R Analytic Flow va en aumento debido a las facilidades que presenta.

### 2.2.2. Desventajas

Algunas de las desventajas por las cuales los usuarios no usan R Analytic Flow:

1. Escasa documentación sobre el manejo de la interfaz.
2. El código fuente se encuentra administrado únicamente por Ef-prime, Inc. Esto impide que se pueda seguir optimizando la interfaz con mayor rapidez.