

Capítulo 1

Obtención R y descarga de paquetes

Este capítulo está escrito para el usuario que nunca ha descargado R en su computadora, mucho menos abrió el programa. El capítulo ofrece algunos antecedentes breves sobre lo que es el programa, luego procede a describir cómo R se puede descargar e instalar de forma totalmente gratuita. Además, el capítulo enumera algunos recursos basados en Internet sobre R que los usuarios deseen consultar siempre que tengan preguntas que no se abordan en este libro.

1.1 Antecedentes e instalación

R es una plataforma para el lenguaje de programación estadística orientado a objetos S. S fue desarrollado inicialmente por John Chambers en Bell Labs, mientras R fue creado por Ross Ihaka y Robert Gentleman. R es ampliamente utilizado en estadística y se ha vuelto bastante popular en Ciencias Políticas durante la última década. El programa también se ha vuelto más utilizado en el mundo empresarial y en el trabajo gubernamental, por lo que la formación como un R el usuario se ha convertido en una habilidad comercializable. R, que es shareware, es similar a S-plus, que es la plataforma comercial para S. Esencialmente R se puede utilizar como matriz lenguaje de programación o como un paquete estadístico estándar que opera mucho como los programas vendidos comercialmente Stata, SAS y SPSS.

Electrónico suplementario material: La en línea versión de esto capítulo (doi: [10.1007 / 978-3-319-23446-5_1](https://doi.org/10.1007/978-3-319-23446-5_1)) contiene usuarios autorizados material, que está disponible para suplementarios.

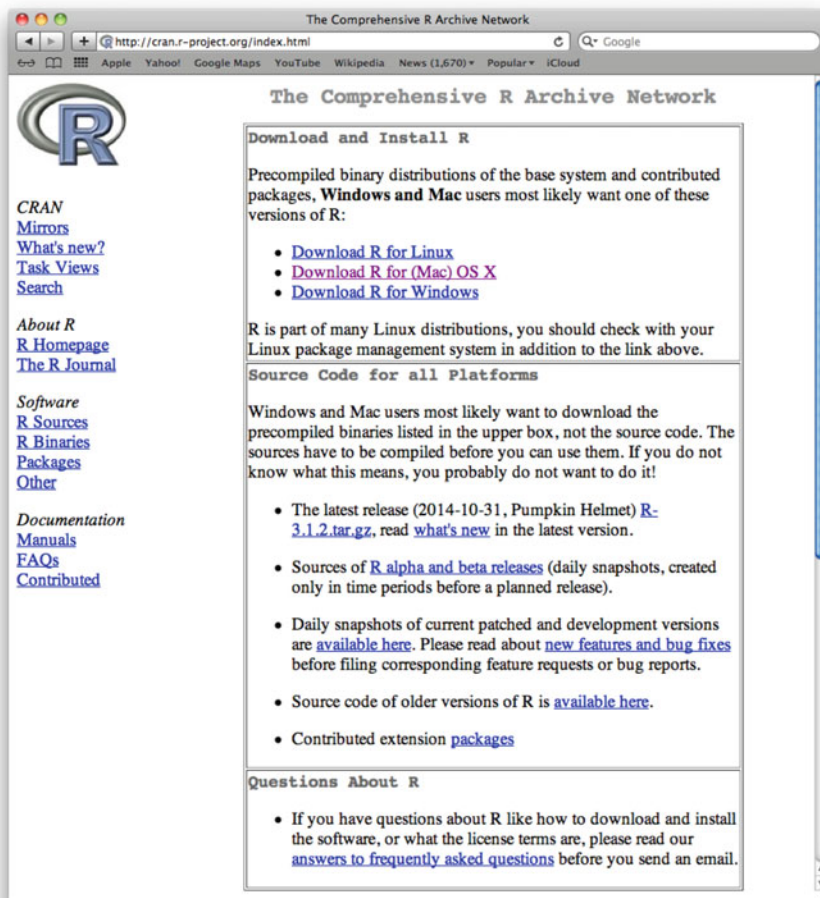


Figura 1.1 El comprensivo R página de inicio de la red de archivos (CRAN). El cuadro superior, "Descargar e instalar R", ofrece enlaces de instalación para los tres sistemas operativos principales.

1.1.1 ¿Dónde puedo conseguirlo? R?

La belleza de R es que es shareware, por lo que es gratis para cualquiera. Para obtener R para Windows, Mac o Linux, simplemente visite el completo R red de archivo (CRAN) en <http://www.cran.r-project.org/>. Figura 1.1 muestra la página de inicio de este sitio web. Como puede verse, en la parte superior de la página hay un recuadro etiquetado *Descargue e instale R*. Dentro de este hay enlaces para la instalación usando los sistemas operativos Linux, Mac OS X y Windows. En el caso de Mac, al hacer clic en el enlace aparecerá un archivo descargable con el paquete sufixo que instalará la última versión. Para Windows, un enlace llamado base se presentará, lo que conduce a una exe archivo para

descarga e instalación. En cada sistema operativo, la apertura del archivo respectivo guiará al usuario a través del proceso de instalación automatizado.¹ En este sencillo procedimiento, un usuario puede instalar R en su máquina personal en cinco minutos.

A medida que pasan los meses y los años, los usuarios observarán el lanzamiento de nuevas versiones de R. No hay parches de actualización para R, por lo tanto, a medida que se lanzan nuevas versiones, debe instalar completamente una nueva versión siempre que desee actualizar a la última edición. Los usuarios no necesitan reinstalar todas las versiones que se lanzan. Sin embargo, a medida que pasa el tiempo, las bibliotecas complementarias (que se describen más adelante en este capítulo) dejarán de admitir versiones anteriores de R. Una posible guía sobre este punto es actualizar a la versión más reciente siempre que no se pueda instalar una biblioteca de interés debido a la falta de soporte. El único inconveniente importante que plantea la reinstalación completa es que las bibliotecas complementarias creadas por el usuario tendrán que reinstalarse, pero esto se puede hacer según sea necesario.

1.2 Introducción: una primera sesión en R

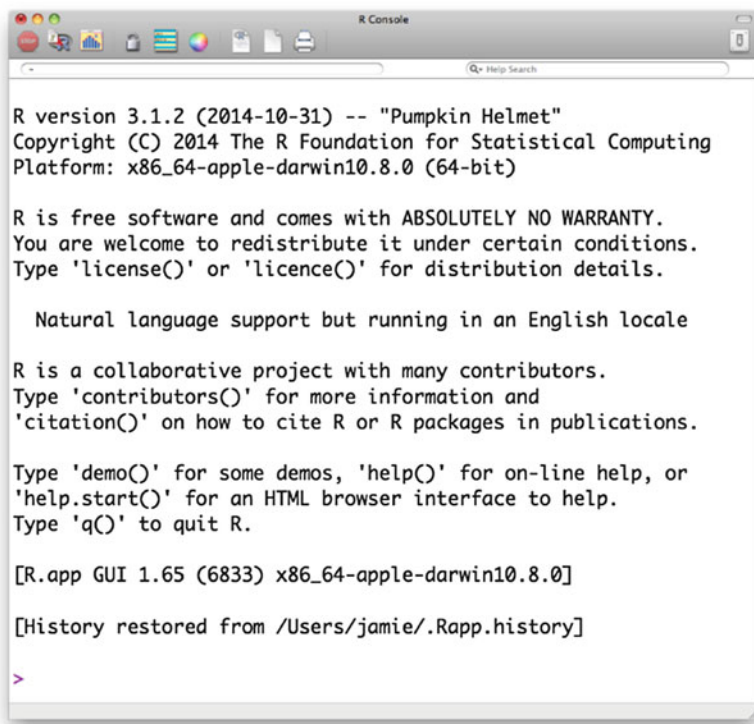
Una vez que haya instalado R, habrá un icono en el menú Inicio de Windows (con la opción de colocar un acceso directo en el Escritorio) o en la carpeta Aplicaciones de Mac (con la opción de mantener un icono en el Dock del área de trabajo). Comenzará haciendo clic o haciendo doble clic en el icono R. Figura 1.2 muestra la ventana asociada con la versión Mac del software. Notarás que R tiene algunas opciones de botones en la parte superior de la ventana. Dentro de Mac, la barra de menú en la parte superior del espacio de trabajo también incluirá algunos menús desplegables. En Windows, los menús desplegables también se presentarán dentro de la R ventana. Sin embargo, con solo un puñado de menús y botones, los comandos en R se ingresan principalmente a través del código de usuario. Los usuarios que deseen la opción alternativa de tener más menús y botones disponibles pueden desear instalar RStudio o un programa similar que agregue una interfaz de apuntar y hacer clic a R, pero el conocimiento de la sintaxis es esencial. Figura 1.3 muestra la ventana asociada con la versión Mac de RStudio.²

Los usuarios pueden enviar su código a través de archivos de script (la opción recomendada, que se describe en la Sección. 1.3) o en la línea de comando que se muestra en la parte inferior de la R consola. En la Fig. 1.2, el mensaje tiene este aspecto:

>

¹Los nombres de estos archivos cambian a medida que las nuevas versiones de R son liberados. En el momento de esta impresión, los archivos respectivos seR-3.1.2-snowleopard.pkg o R-3.1.2-mavericks.pkg para varias versiones de Mac OS X y R-3.1.2-win.exe para ventanas. Los usuarios de Linux lo encontrarán más fácil de instalar desde una terminal. El código de terminal está disponible siguiendo el *Descargar R para Linux* enlace en la página de CRAN, luego elija una distribución de Linux en la página siguiente y use el código de terminal que aparece en la página resultante. En el momento de la impresión, se admiten Debian, varias formas de Red Hat, OpenSUSE y Ubuntu.

²RStudio está disponible en <http://www.rstudio.com>.



```

R version 3.1.2 (2014-10-31) -- "Pumpkin Helmet"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin10.8.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.65 (6833) x86_64-apple-darwin10.8.0]

[History restored from /Users/jamie/.Rapp.history]

>

```

Figura 1.2 R Consola para una nueva sesión

Siempre que escriba código directamente en el símbolo del sistema, si un usuario escribe un solo comando que abarca varias líneas, el símbolo del sistema se convierte en un signo más (+) para indicar que el comando no está completo. El signo más no indica ningún problema o error, solo le recuerda al usuario que el comando anterior aún no está completo. El cursor se coloca automáticamente allí para que el usuario pueda ingresar comandos.

En la edición electrónica de este libro, ingrese la sintaxis y las impresiones de salida de R estará codificado por colores para ayudar a distinguir lo que el usuario debe escribir en un archivo de script de los resultados esperados. El código de entrada se escribirá en **fuentes de teletipo azul**. R la salida se escribirá en **fuentes de teletipo negro**. Mensajes de error que R las devoluciones se escribirán en **fuentes de teletipo rojo**. Estos colores corresponden a la codificación de colores R utiliza para texto de entrada y salida. Si bien es posible que los colores no sean visibles en la edición impresa, el texto del libro también distinguirá las entradas de las salidas. Además, los nombres de las variables se escribirán en **negrita**. Las palabras clave conceptuales de estadística y programación, así como el texto enfatizado, se escribirán en *cursiva*. Finalmente, cuando el significado de un comando no sea evidente, las iniciales identificativas estarán subrayadas y en **negrita** en el texto. Por ejemplo, el comando **significa**

len el oído **metroodel**.

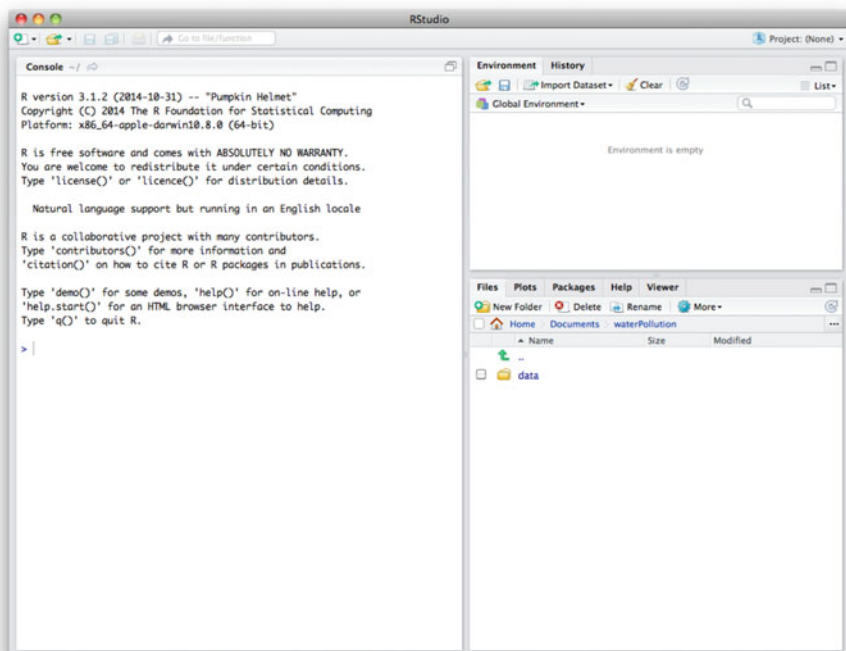


Figura 1.3 Abrir ventana desde una sesión de RStudio

Cuando se escribe R sintaxis en la línea de comandos o en un archivo de script, los usuarios deben tener en cuenta algunos preliminares importantes:

- Expresiones y comandos en R distinguen entre mayúsculas y minúsculas. Por ejemplo, la función `var` devuelve la varianza de una variable: una función simple discutida en el Cap. 4. Por el contrario, `elVAR` comando desde el `vars` Library estima un modelo de autorregresión vectorial, una técnica avanzada que se analiza en el cap. 9. Del mismo modo, si un usuario nombra un conjunto de datos `mis datos`, entonces no se puede llamar con los nombres `MyData`, `MyData`, `MYDATA`, o `mis datos`. R supondría que cada uno de estos nombres indica un significado diferente.
- Las líneas de comando no necesitan estar separadas por ningún carácter especial como un punto y coma como en `Limdep`, `SAS` o `Gauss`. Una simple devolución dura servirá.
- R ignora todo lo que sigue al carácter de almohadilla (`#`) como comentario. Esto se aplica cuando se utiliza la línea de comandos o archivos de secuencia de comandos, pero es especialmente útil cuando se guardan notas en archivos de secuencia de comandos para su uso posterior.
- El nombre de un objeto debe comenzar con un carácter alfabético, pero puede contener caracteres numéricos a partir de entonces. Un punto también puede formar parte del nombre de un objeto. Por ejemplo, `x.1` es un nombre válido para un objeto en R.
- Puede utilizar las teclas de flecha del teclado para volver a los comandos anteriores. Una pulsación de la flecha hacia arriba recupera el comando introducido anteriormente y lo coloca en

la línea de comando. Cada pulsación adicional de la flecha se mueve a un comando anterior al que aparece en la línea de comando, mientras que la flecha hacia abajo llama al comando que sigue al que aparece en la línea de comando.

Aparte de este puñado de reglas importantes, el símbolo del sistema en R tiende a comportarse de manera intuitiva, devolviendo respuestas a los comandos de entrada que podrían adivinarse fácilmente. Por ejemplo, en su nivel más básico R funciona como una calculadora de gama alta. Algunas de las claves *aritméticas* los comandos son: suma (+), resta (-), multiplicación (*), división (/), exponenciación (^), la función módulo (%) y división entera (% /%). Los paréntesis () especifican el orden de las operaciones. Para ejemplo, si escribimos la siguiente entrada:

```
(3 + 5/78) ^ 3 * 7
```

Luego R imprime la siguiente salida:

```
[1] 201.3761
```

Como otro ejemplo, podríamos preguntar R cuál es el resto al dividir 89 entre 13 usando la función módulo:

```
89 %% 13
```

R luego proporciona la siguiente respuesta:

```
[1] 11
```

Si quisiéramos R para realizar la división de enteros, podríamos escribir:

```
89% /% 13
```

Nuestra respuesta de salida a esto es:

```
[dieciséis
```

La opción El comando permite al usuario modificar los atributos de la salida. Por ejemplo, el dígito El argumento ofrece la opción de ajustar cuántos dígitos se muestran. Esto es útil, por ejemplo, al considerar la precisión con la que desea presentar los resultados en una tabla. Otras funciones integradas útiles del álgebra y la trigonometría incluyen: sin (x), cos (x), tan (x), exp (x), log (x), sqrt (x), y Pi. Para aplicar algunas de estas funciones, primero podemos expandir el número de dígitos impresos y luego pedir el valor de la constante:

```
opciones (dígitos = 16)
Pi
```

R en consecuencia imprime el valor de hasta 16 dígitos:

```
[1] 3.141592653589793
```

También podemos usar comandos como Pi para insertar el valor de dicha constante en una función. Por ejemplo, si quisiéramos calcular el seno de radianes $\frac{\pi}{2}$ (90°) ángulo, podríamos escribir:

```
senado (pi / 2)
```

R imprime correctamente ese pecado. /D 1:

```
[1] 1
```

1.3 Guardar entrada y salida

Al analizar datos o programar en R, un usuario nunca se meterá en problemas serios siempre que siga dos reglas básicas:

1. Deje siempre intactos los archivos de datos originales. Cualquier versión revisada de los datos debe escribirse en un *nuevo* archivo. Si está trabajando con un conjunto de datos particularmente grande y difícil de manejar, escriba un programa corto que se adapte a lo que necesita, guarde el archivo limpiado por separado y luego escriba el código que funcione con el nuevo archivo.
2. Escriba todo el código de entrada en una secuencia de comandos que se guarda. Los usuarios generalmente deben evitar escribir código directamente en la consola. Esto incluye código para limpiar, recodificar y remodelar datos, así como para realizar análisis o desarrollar nuevos programas.

Si se siguen estas dos reglas, el usuario siempre puede recuperar su trabajo hasta el punto de algún error u omisión. Entonces, incluso si, en la administración de datos, se pierde o pierde información esencial, o incluso si un revisor de una revista nombra un predictor que un modelo debe agregar, el usuario siempre puede volver sobre sus pasos. Al llamar al conjunto de datos original con el programa guardado, el usuario puede hacer *menor* Ajustes al código para incorporar una nueva función de análisis o recuperar información perdida. Por el contrario, si se sobrescriben los datos originales o no se guarda el código de entrada, es probable que el usuario deba reiniciar todo el proyecto desde el principio, lo cual es una pérdida de tiempo.

A *archivo de script* en R es simplemente texto sin formato, generalmente guardado con el sufijo *.R*. Para crear un nuevo archivo de script en R, simplemente elige *Archivo! Nuevo documento* en el menú desplegable para abrir el documento. Alternativamente, la ventana de la consola que se muestra en la Fig. 1.2 muestra un icono que parece una página en blanco en la parte superior de la pantalla (segundo icono de la derecha). Al hacer clic en esto también se creará un nuevo *.R* archivo de script. Una vez abierto, lo normal *Ahorrary Guardar como* comandos de la *Archivo* se aplica el menú. Para abrir un existente *.R* guión, elige *Archivo! Abrir documento* en el menú desplegable, o haga clic en el icono en la parte superior de la pantalla que parece una página con escritura (tercer icono de la derecha en la Fig. 1.2). Cuando se trabaja con un archivo de secuencia de comandos, cualquier código dentro del archivo se puede ejecutar en la consola simplemente resaltando el código de interés y escribiendo el atajo de teclado Ctrl + R en Windows o Cmd + Retorno en Mac. Además del editor de archivos de script predeterminado, también están disponibles editores de texto más sofisticados como Emacs y RWinEdt.

El producto de cualquier *R* La sesión se guarda en el *directorio de trabajo*. El directorio de trabajo es la ruta de archivo predeterminada para todos los archivos que el usuario desea leer o escribir. El comando `getwd` (significado **obtener Working Directory**) se imprimirá *R* directorio de trabajo actual, mientras que `setwd` (**establecer working Directory**) le permite cambiar el directorio de trabajo como desee. Dentro de una máquina con Windows, la sintaxis para verificar y luego configurar, el directorio de trabajo se vería así:

```
getwd ()
setwd ("C: / temp /")
```

Esto ahora escribe cualquier archivo de salida, ya sean conjuntos de datos, figuras o salida impresa en la carpeta. temperatura en el C: manejar. Observa esoR espera que las barras diagonales designen subdirectorios, lo que contrasta con el uso típico de barras invertidas de Windows. Por lo tanto, especificando C: / temp / como apunta el directorio de trabajo C: \ temp \ en la sintaxis normal de Windows. Mientras tanto, para Mac o Unix, configurar un directorio de trabajo sería similar, y el directorio de ruta se imprime exactamente como estos sistemas operativos los designan con barras diagonales:

```
setwd ("/ Volumes / flashdisk / temp")
```

Tenga en cuenta que setwd se puede llamar varias veces en una sesión, según sea necesario. Además, especificar la ruta completa para cualquier archivo anula el directorio de trabajo.

A *guardar salida* de su sesión en R, Prueba el lavabo mando. Como término de computación general, un **lavabo** es un punto de salida para un programa donde se escriben datos o resultados. En R, en consecuencia, este término se refiere a un archivo que registra toda nuestra salida impresa. Para guardar la salida de su sesión en el archivo Rintro.txt dentro del tipo de directorio de trabajo:

```
fregadero ("Rintro.txt")
```

Alternativamente, si quisiéramos anular el directorio de trabajo, en Windows, por ejemplo, podríamos haber escrito:

```
fregadero ("C: / myproject / code / Rintro.txt")
```

Ahora que hemos creado un archivo de salida, cualquier salida que normalmente se imprimiría en la consola se imprimirá en el archivo. Rintro.txt. (Por esta razón, en una primera ejecución de código nuevo, generalmente es aconsejable permitir que la salida se imprima en la pantalla y luego volver a ejecutar el código más tarde para imprimir en un archivo). impresión El comando es útil para crear resultados que se pueden seguir fácilmente. Por ejemplo, el comando:

```
print ("La media de la variable x es ...")
```

imprimirá lo siguiente en el archivo Rintro.txt:

```
[1] "La media de la variable x es ..."
```

Otro comando de impresión útil es el gato comando (abreviatura de **gato** enate, para conectar las cosas entre sí), lo que le permite mezclar objetos en R con texto. Como vista previa de las herramientas de simulación descritas en el Cap.11, creemos una variable llamada X mediante simulación:

```
x <- normal (1000)
```

A modo de explicación: esta sintaxis se extrae al azar 1000 veces de una distribución normal estándar y asigna los valores al vector X. Observe la flecha (<-), formada con un *menos que* firmar y un *guión*, cual es Roperador de asignación. Cada vez que asignamos algo con la flecha (<-) el nombre de la izquierda (X en este caso) nos permite recordar el resultado de la operación de la derecha (normal (1000))

en este caso).³ Ahora podemos imprimir la media de estos 1000 sorteos (que deberían estar cerca de 0 en este caso) en nuestro archivo de salida de la siguiente manera:

```
cat ("La media de la variable x es ...", mean(x), "\n")
```

Con esta sintaxis, los objetos de R se puede incrustar en la declaración que imprima. El personaje \n introduce un retorno de carro. También puede imprimir cualquier resultado estadístico utilizando elimpresión o gato comandos. Recuerde, su salida no va al archivo de registro a menos que use uno de los comandos de impresión. Otra opción es simplemente copiar y pegar los resultados delR ventana de la consola en Word o un editor de texto. Para desactivar el comando de fregadero, simplemente escriba:

```
lavabo()
```

1.4 Gestión de sesiones de trabajo

Una característica clave de R es que es un *orientado a objetos* lenguaje de programación. Las variables, los marcos de datos, los modelos y las salidas se almacenan en la memoria como *objetos*, o ubicaciones identificadas (y nombradas) en la memoria con características definidas. R almacena en la memoria de trabajo cualquier objeto que cree usando el nombre que defina cada vez que carga datos en la memoria o estima un modelo. Alist los objetos que ha creado en una sesión utilizan cualquiera de los siguientes comandos:

```
objetos()
ls ()
```

A rmmetrossobre todos los objetos en R tipo:

```
rm (lista = ls (todo = VERDADERO))
```

Como regla general, es una buena idea utilizar el rm comando al inicio de cualquier programa nuevo. Si el usuario anterior guardó su espacio de trabajo, es posible que haya usado objetos que comparten el mismo nombre que el suyo, lo que puede crear confusión.

A quit R cierre la ventana de la consola o escriba:

```
q ()
```

En este punto, R le preguntará si desea guardar la imagen del espacio de trabajo. Generalmente, es recomendable no hacer esto, ya que comenzar con una pizarra limpia en cada sesión es más probable que evite errores de programación o confusión en las versiones de los objetos cargados en la memoria.

Finalmente, en muchos R sesiones, necesitaremos cargar *paquetes* o lotes de código y datos que ofrecen funciones adicionales no escritas en Rcódigo base. A lo largo de este libro cargaremos varios paquetes, particularmente en el Cap.8, dónde

³La flecha (<-) es el operador de asignación tradicional, aunque un solo signo igual (=) también puede servir para las asignaciones.

nuestro enfoque estará en paquetes de ejemplo escritos por destacados científicos políticos para implementar métodos de vanguardia. Los comandos necesarios para cargar paquetes son `install.packages`, un comando que descarga e instala automáticamente un paquete en la copia de un usuario de R, y `Biblioteca`, un comando que carga el paquete en una sesión determinada. Supongamos que queremos instalar el paquete `MCMCpack`. Este paquete proporciona herramientas para el modelado bayesiano que usaremos en el Cap. 8. La forma de la sintaxis de estos comandos es:

```
biblioteca install.packages
("MCMCpack") (MCMCpack)
```

La instalación del paquete distingue entre mayúsculas y minúsculas y la ortografía. R probablemente le pedirá en este punto que elija uno de los CRANmirrors desde el cual descargar este paquete: Para una descarga más rápida, los usuarios generalmente eligen el espejo que está más próximo geográficamente. `install.packages` El comando solo necesita ejecutarse una vez por R instalación para que un paquete en particular esté disponible en una máquina. `Biblioteca` El comando debe ejecutarse para cada sesión en la que un usuario desee usar el paquete. Por lo tanto, en la próxima sesión que queremos usar `MCMCpack`, solo necesitamos escribir: `biblioteca (MCMCpack)`.

1.5 Recursos

Dada la amplia gama de funciones básicas que están disponibles en R, mucho menos la gama aún más amplia de funciones creadas por R paquetes, un libro como este no puede abordar todo R es capaz de hacer. Este libro debe servir como un recurso que presenta cómo un investigador puede utilizar R como un programa de estadísticas básico y ofrecen algunos consejos generales sobre el uso de paquetes y funciones de programación. A medida que surgen preguntas sobre temas que no se tratan en este espacio, hay varios otros recursos que pueden ser útiles:

- Dentro R, la *Ayudar* menú desplegable (también disponible escribiendo `help.start()` en la consola) ofrece varios manuales de uso, incluida una "Introducción a R" y escribiendo `R Extensiones` ". Esto también abre un motor de búsqueda basado en HTML de los archivos de ayuda.
- El Instituto de Investigación y Educación Digital de UCLA ofrece varios tutoriales agradables (<http://www.ats.ucla.edu/stat/r/>). El sitio web de CRAN también incluye una variedad de manuales en línea (<http://www.cran.r-project.org/other-docs.html>).
- Algunos buenos tutoriales interactivos incluyen *remolino*, que es un paquete que instalas en tu propia copia de R (más información: <http://www.swirlstats.com/>) y prueba R, que se completa en línea (<http://tryr.codeschool.com/>).
- Dentro de R consola, los comandos `?`, `ayuda()`, y `help.search()` todos sirven para encontrar documentación. Por ejemplo, `?lm` encontraría la documentación para el comando de modelo lineal. Alternativamente, `help.search("modelo lineal")` buscaría en la documentación una frase.

- Para buscar información en Internet, Rbuscar (<http://www.rseek.org/>, con tecnología de Google) es un motor de búsqueda útil que busca solo en sitios web centrados en R.
- Finalmente, los usuarios de Twitter hacen referencia R a través del hashtag #rstats.

En este punto, los usuarios deberían tener R instalados en su máquina, tienen un sentido básico de cómo se ingresan los comandos y se genera la salida, y reconocen dónde encontrar los vastos recursos disponibles para R usuarios. En los próximos seis capítulos, veremos cómo R se puede utilizar para desempeñar el papel de un programa de software de análisis estadístico o econometría.

1,6 Problemas de práctica

Cada El capítulo terminará con algunos problemas de práctica. Si ha probado código todos los ejemplos del capítulo, debería poder completarlos en su Si aún no propio. lo ha hecho, continúe e instale R en su máquina para gratis y pruebe el código del capítulo. Luego intente las siguientes preguntas.

1. Calcule lo siguiente en R:

(a) $7 \cdot 2_3$

(B) $\frac{8}{82C1}$

(c) cpagos

(D) $\frac{81}{81}$

(el N $má$

2. ¿Qué dice el comando `cor` ¿hacer? Busque documentación al respecto y describa qué hace la función.
3. ¿Qué dice el comando `runif` ¿hacer? Busque documentación al respecto y describa qué hace la función.
4. Cree un vector llamado X que consta de 1000 extracciones de una distribución normal estándar, utilizando código tal como se ve en la Sección. 1.3. Crea un segundo vector llamado y del mismo modo. Calcule el coeficiente de correlación entre los dos vectores. ¿Qué resultado obtiene y por qué obtiene este resultado?
5. Familiarícese con cómo decidir cuándo los paquetes complementarios pueden resultarle útiles. Iniciar sesión en <http://www.rseek.org> y busca lo que el stringr el paquete lo hace. ¿Qué tipo de funcionalidad le ofrece este paquete? ¿Cuándo querrías usarlo?