

# Fundamentos básicos para manejo de

# R

en  
Estadística  
Descriptiva

Julio Fernando Suárez Cifuentes  
Ricardo Alfredo Rojas Medina



UNIVERSIDAD  
NACIONAL  
DE COLOMBIA  
SEDE MANIZALES  
FACULTAD DE ADMINISTRACIÓN





Fundamentos  
básicos  
para  
manejo  
de

A large, stylized letter 'R' in a serif font, centered on the page. Behind the 'R' is a complex geometric design consisting of several overlapping triangles in various shades of gray. The triangles are arranged in a way that creates a sense of depth and movement, with some pointing upwards and others downwards. The overall effect is a modern, minimalist logo for the R programming language.

R

en Estadística  
Descriptiva



Fundamentos  
básicos  
para  
manejo  
de

R

en Estadística  
Descriptiva

Julio Fernando Suárez Cifuentes  
Ricardo Alfredo Rojas Medina



UNIVERSIDAD NACIONAL DE COLOMBIA

SEDE MANIZALES

FACULTAD DE ADMINISTRACIÓN

Manizales, Colombia Mayo 2013



© UNIVERSIDAD NACIONAL DE COLOMBIA  
SEDE MANIZALES  
FACULTAD DE ADMINISTRACIÓN

ISBN 978-958-761-511-1

#### AUTORES



JULIO FERNANDO SUÁREZ CIFUENTES  
jfsuarezc@unal.edu.co



RICARDO ALFREDO ROJAS MEDINA  
rarojasm@unal.edu.co

DISEÑO Y EDICIÓN  
Sección Publicaciones e Imagen

Primera edición, 2013  
Prohibida la reproducción total o parcial por cualquier medio sin la autorización  
escrita del titular de los derechos patrimoniales



# Contenido



## Capítulo I

1.	Generalidades del paquete	11
1.1	Instalación	12
1.2.	Revisión de las principales opciones de la barra de herramientas	15
1.3	Contribuciones	19
1.4	Manuales	20
1.5	Ayudas en R	21
1.6	Cargar paquetes	23
1.7	Acceso a la información	25
1.8	Exportar datos	35
1.9	Otras instrucciones a tener en cuenta	37



## II. Estructuras de datos

1.	Vectores	39
1.1.	Vector numérico	39
1.2.	Vector lógico	40
1.3	Vector de caracteres	41
1.4	Asignación de nombres a los elementos de un vector	41
1.5	Creación de patrones	42
1.6	Extracción de elementos de un vector	43
1.7	Valores faltantes	45
1.8	Otros elementos a tener en cuenta	46
1.9	Resultados estadísticos a partir de un vector	47
2.	Factores	55
2.1	Factores ordenados	57
3.	Matrices	61
4.	Arrays	74
5.	Listas	78
5.1	Aplicación de Funciones	80
6.	Data.frames	81



## III. Gráficos en R

1.	Plot(): Gráficos de dispersión (nube de puntos)	92
2.	Hist(): Histograma	100



3.	Polygon(): Polígono	103
4.	Barplot(): Gráficos de barras	105
5.	Pie(): Gráficos circulares	109
6.	Boxplot(): Gráficos de caja	112
7.	Par(mfrow): Varios gráficos en una página	114
8.	Stem(): Gráfico de tallos y hojas	116
9.	Opciones para guardar gráficos	117
10.	Comandos especiales	119



## Introducción

Los cursos tradicionales de estadística han sido soportados por una diversidad de paquetes que agilizan el proceso de información y permiten realizar diversos análisis para facilitar la toma de decisiones, sin embargo el egresado al aplicar los conocimientos adquiridos queda limitado cuando debe analizar información, por no contar con paquetes que procesen y generen los resultados requeridos. Adicionalmente el alto costo que tienen los programas especializados en este campo hace que muy pocas empresas puedan acceder a ellos, originando ésta situación, que los análisis estadísticos queden limitados a las aulas de clase, o al empleo de las escasas opciones que traen las hojas electrónicas.

Hay una verdadera comunidad académica alrededor de R, en la cual participan no sólo usuarios sino personas que aportan nuevos desarrollos dando la posibilidad de atraer profesionales de diferentes áreas del conocimiento como sicólogos, ingenieros, administradores, entre otros, quienes se hacen partícipes. Surgen blogs y páginas de internet, en los cuales se ofrecen nuevos elementos que elaboran muchos de los que trabajan en R, y que lo potencializan, estas contribuciones vienen dadas en idiomas diferentes tales como: mandarín, japonés, español, alemán, inglés.

Ante esta situación y con el ánimo de ofrecer a nuestros estudiantes y a la comunidad en general una alternativa con la que puedan desarrollar y aplicar todos los conocimientos adquiridos en los cursos de estadística de su pregrado y postgrado, se ha desarrollado este documento, con el cual se persigue lo siguiente:

- Ofrecer una guía clara, comprensible y accesible a todas las personas interesadas en el manejo de R.
- Existe en la actualidad documentos sobre el programa, que se encuentran en internet, los cuales no son claros y/o presentan muy poca aplicación práctica.
- Apoyar la difusión del programa R y con ello reducir costos en la adquisición o mantenimiento de licencias de paquetes estadísticos.
- Combatir la piratería y defender los derechos de autor.

El documento ha sido desarrollado para que el lector domine la lógica del programa y cree sus propias soluciones a través de las habilidades que pueda desarrollar con la lectura y la aplicación de los diferentes ejemplos que se encuentran a lo largo de todo el documento. Para esto se dispone de una serie de instrucciones que son útiles para solucionar muchos problemas numéricos y que a su vez son soporte para avanzar y profundizar sobre el manejo del programa.

En el primer capítulo se da una noción muy general sobre lo que es el programa y como se instala, se indica el concepto de librería que son las contribuciones, se profundiza sobre el acceso de información bien sea digitándola directamente o por medio de una base de datos.

En el segundo capítulo se ofrecen diferentes estructuras de datos que posee R, y se manejan instrucciones claves que facilitan la obtención de los indicadores estadísticos más importantes y la forma de realizar diferentes representaciones tabulares.

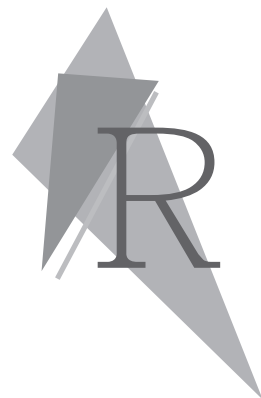
El tercer capítulo está orientado a las representaciones gráficas que apoyen la difusión de los resultados estadísticos para que sean de fácil interpretación, en los cuales el programa R presenta una gran riqueza.

Es importante señalar que en el documento se encuentra un conjunto de instrucciones que están resaltadas, esto se hace con el fin de mostrar al lector cual es la instrucción que debe ser corrida para obtener los resultados que también aparecen en el documento.

Las diferentes etapas del manual han sido desarrolladas usando ejemplos que incluyen las respuestas, a través de los cuales se logra tener mayor conocimiento sobre los comandos usados.

Este documento se entrega con un CD, que incluye archivos con los cuales se han generado los diferentes ejemplos o que contienen los elementos necesarios para desarrollar algunos ejercicios propuestos y se suministra en formato texto cada una de las instrucciones que hacen parte de este libro, esto con el fin de facilitar la ejecución y evitar errores en la digitalización.

# 1. Generalidades del paquete



R es un lenguaje de programación con el que se puede analizar un conjunto de información estadística referida a diferentes áreas del conocimiento como finanzas, mercados, control de calidad, investigación de operaciones, vías y transporte, etc., además permite el análisis de datos que van desde la estadística descriptiva, hasta llegar a estudios mucho más complejos como: análisis multivariado; series de tiempo, regresión múltiple, diseño de experimentos, estadística no paramétrica, estimación bayesiana, optimización, finanzas y muchos otros temas más.

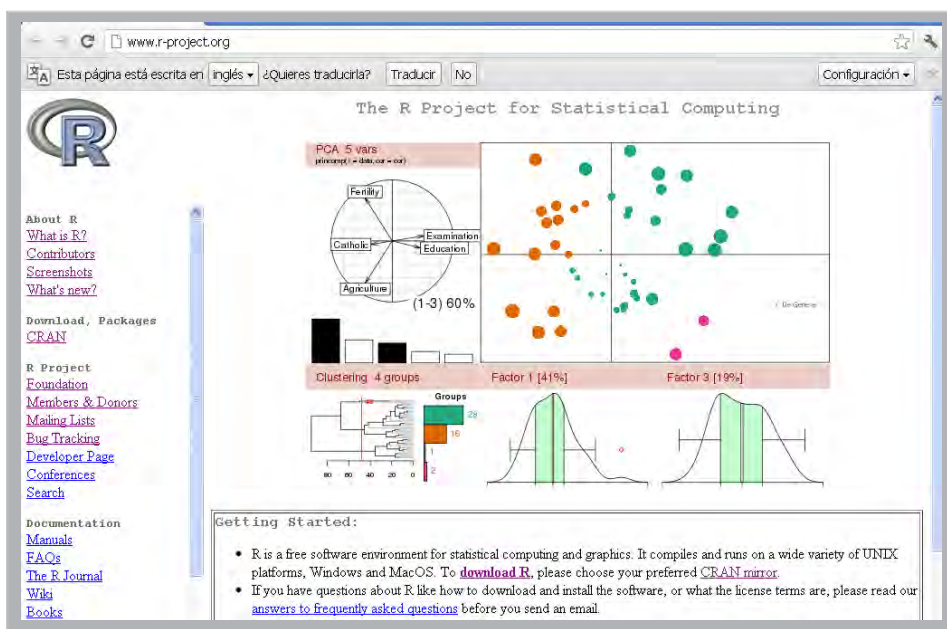
Las ventajas que presenta R frente a otros programas son las siguientes:

- Posibilita la realización de gráficos excelentes.
- Es muy flexible y existen gran cantidad de librerías hechas por usuarios de todo el mundo que al ser cargadas permiten realizar procedimientos específicos.
- Se pueden programar procedimientos y aplicaciones propias.
- R está disponible en varias formas: el código fuente escrito principalmente en C (y algunas rutinas en Fortran), esencialmente para máquinas Unix y Linux, o como archivos binarios precompilados para Windows y Linux.
- R posee muchas funciones para análisis estadísticos y gráficos; estos últimos pueden ser visualizados de manera inmediata en su propia ventana y ser guardados en varios formatos (jpg, png, bmp, ps, pdf, emf, pictex, xfig).
- Una de las características más sobresalientes de R es su enorme flexibilidad, ya que R guarda los resultados como un "objeto", de tal manera que se puede hacer un análisis sin necesidad de mostrar su resultado inmediatamente.
- R es un lenguaje interpretado y no compilado, lo cual significa que los comandos escritos son ejecutados directamente sin necesidad de construir ejecutables.
- Es gratuito.

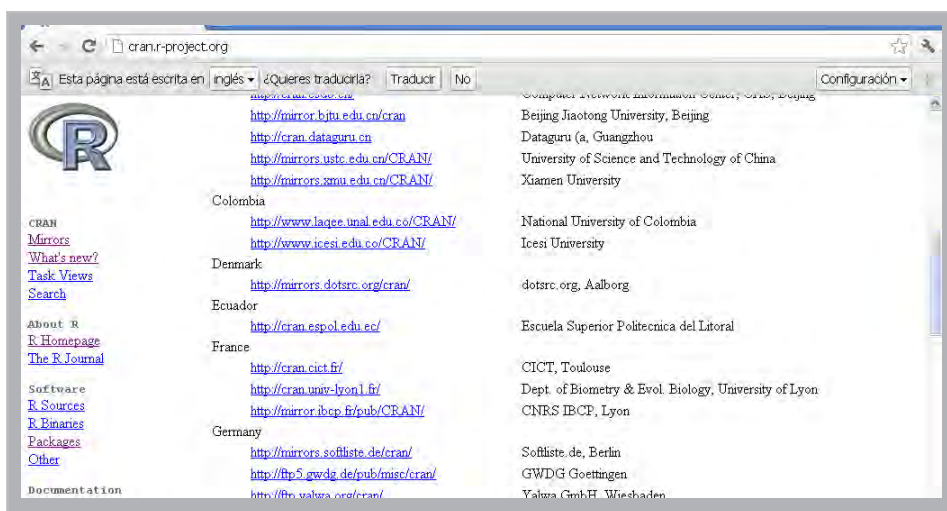
## 1.1 Instalación

Para instalar R se debe seguir el siguiente proceso:

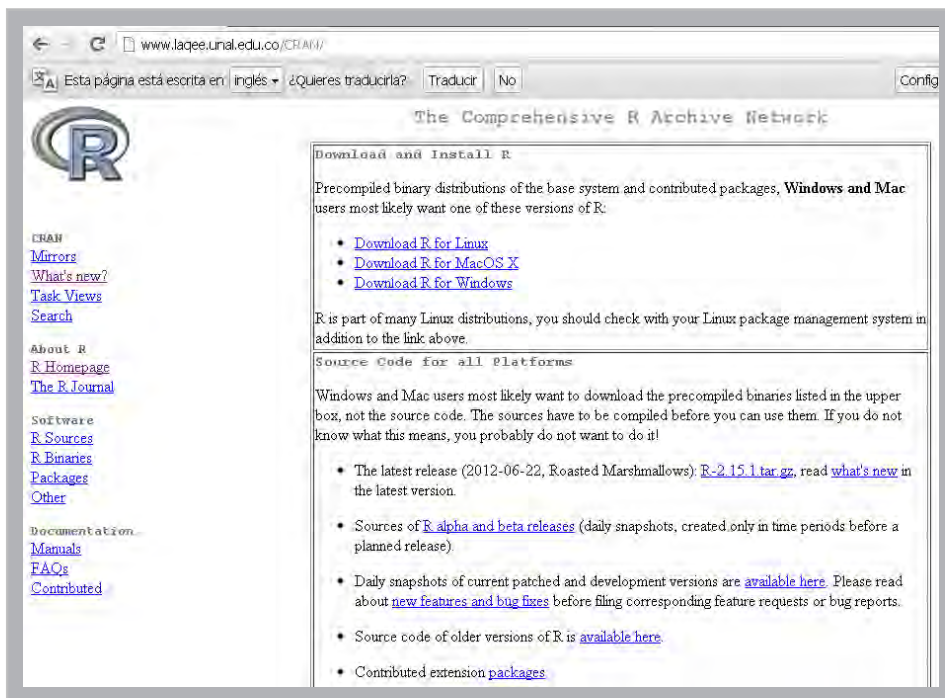
Ir a la página principal cuya dirección es: <http://www.r-project.org/>  
Situado allí dar clic en download.



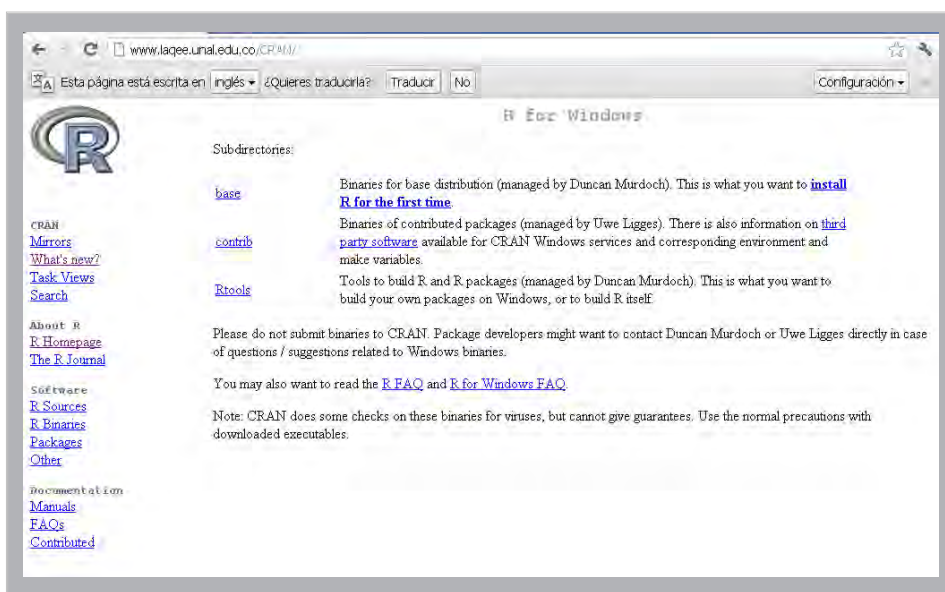
Aparece: CRAN Mirrors, allí ubique Colombia (CRAN es una red de servidores ftp y web de todo el mundo que almacenan idénticas versiones actualizadas de R junto con su documentación. Se utiliza el espejo CRAN más cercano con el fin de minimizar la carga de la red).



Escoja el sistema operativo apropiado y de clic en él.



En caso de que el sistema operativo que se usa sea de 64 bits, dar clic en here, sino en base.



Dar clic en Download R 2.15.1 for Windows (última versión disponible al escribir estas notas).

The screenshot shows the CRAN website at [www.laqee.unal.edu.co/CRAN/](http://www.laqee.unal.edu.co/CRAN/). The page title is "R-2.15.1 for Windows (32/64 bit)". The main content area has a box with the link "Download R 2.15.1 for Windows (47 megabytes, 32/64 bit)" and sub-links for "Installation and other instructions" and "New features in this version". Below this, there is a section for "Frequently asked questions" with links like "How do I install R when using Windows Vista?". A sidebar on the left contains links for "CRAN", "Mirrors", "What's new?", "Task Views", "Search", "About R", "R Homepage", "The R Journal", "Software", "R Sources", "R Binaries", "Packages", "Other", "Documentation", "Manuals", "FAQs", and "Contributed". At the bottom, there is a note to webmasters about a stable link to the current Windows binary release.

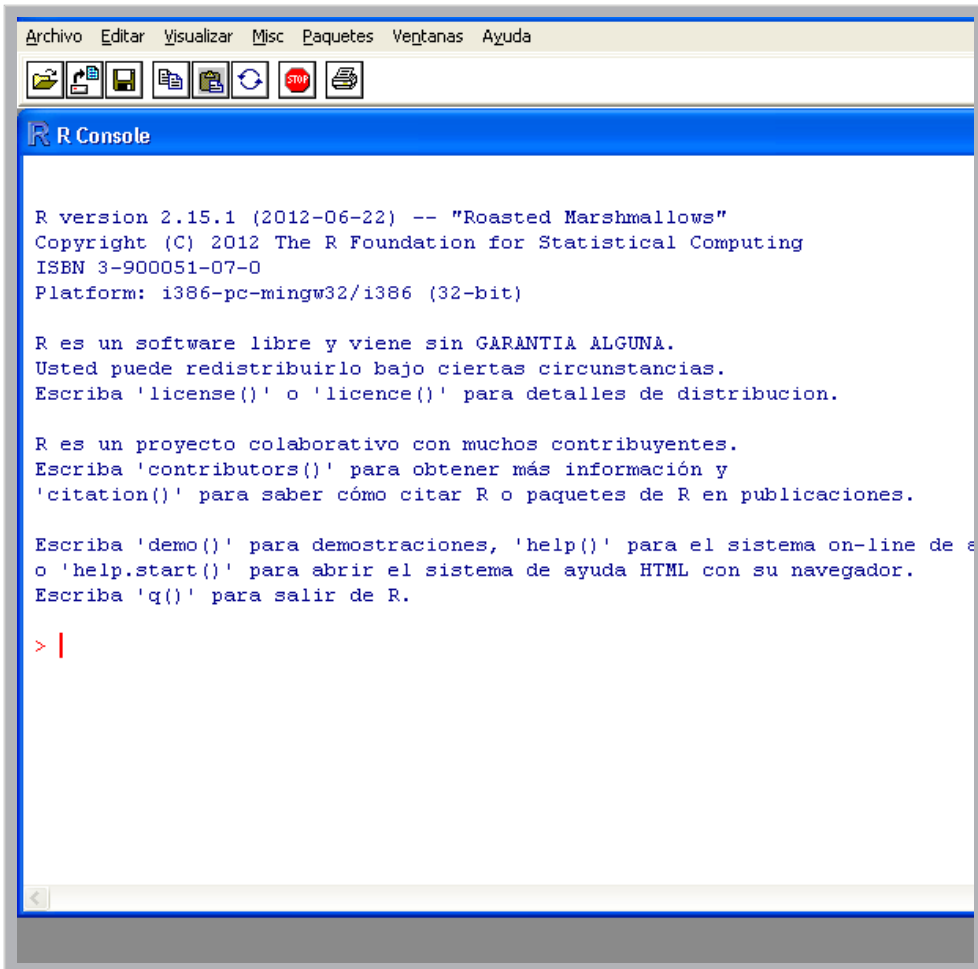
Tome la opción guardar y después vaya al sitio donde almacenó el programa y realice la ejecución. Para verificar que esta se hizo, debe aparecer un icono R en el escritorio.

This screenshot shows the same CRAN website as before, but with a Windows security warning dialog box overlaid. The dialog box is titled "Abrir archivo - Advertencia de seguridad" and contains the text: "No se puede comprobar el fabricante. ¿Está seguro de que desea ejecutar este software?". It lists the file name as "R-2.15.1-win.exe", the manufacturer as "Fabricante desconocido", and the type as "Application". The file is located at "C:\Documents and Settings\usuario\Mis documentos...". There are "Ejecutar" and "Cancelar" buttons. A checkbox "Preguntar siempre antes de abrir este archivo" is checked. The background website content is partially visible through the dialog box.

Una vez instalado R, simultáneamente se instalaron un conjunto de paquetes básicos que traen consigo gran cantidad de funciones para la realización de análisis básicos estadísticos como: representaciones gráficas, ajuste del modelo de regresión lineal general, operaciones con matrices, solución de sistemas de ecuaciones, evaluación y cálculo de probabilidades para variables aleatorias con diversa función de probabilidad. La importancia de R está basada en el hecho de poder cargar diversos paquetes, los cuales indudablemente amplían la capacidad de R para efectuar análisis de diversa índole, cada uno de los cuales trae consigo un manual para su uso.

## 1.2. Revisión de las principales opciones de la barra de herramientas

Para dar inicio se carga el paquete básico dando clic en el icono R en el escritorio, apareciendo entonces la siguiente pantalla:

The image shows a screenshot of the R Console application window. The window has a menu bar with options: Archivo, Editar, Visualizar, Misc, Paquetes, Ventanas, and Ayuda. Below the menu bar is a toolbar with icons for file operations (open, save, print, etc.) and a red 'STOP' button. The main area of the window is titled 'R Console' and contains the following text:

```
R version 2.15.1 (2012-06-22) -- "Roasted Marshmallows"
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)

R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribucion.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

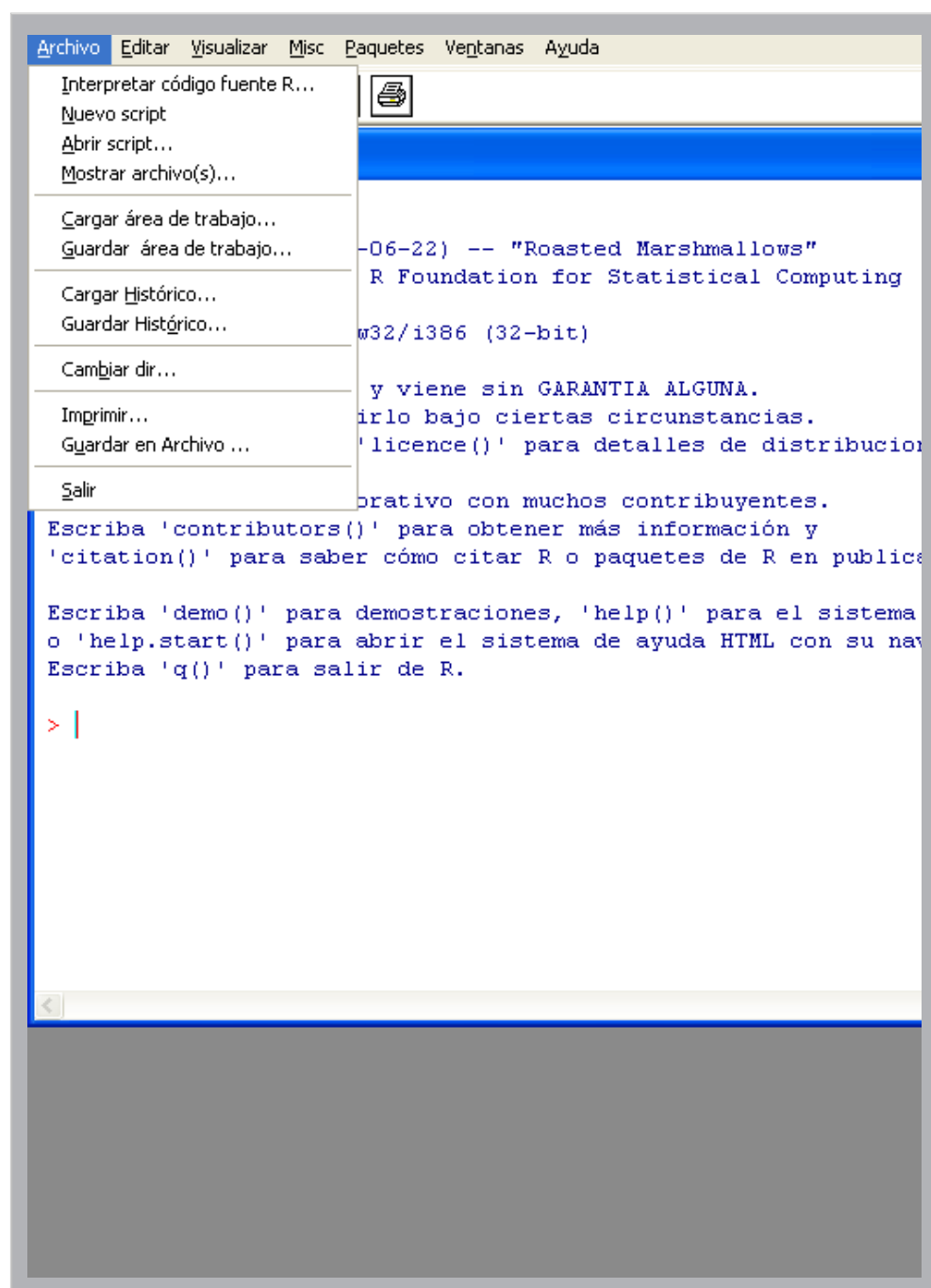
Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> |
```

The text is displayed in a monospaced font. At the bottom of the console, there is a red prompt character '>' followed by a vertical bar '|'. The window has a scroll bar on the right side.



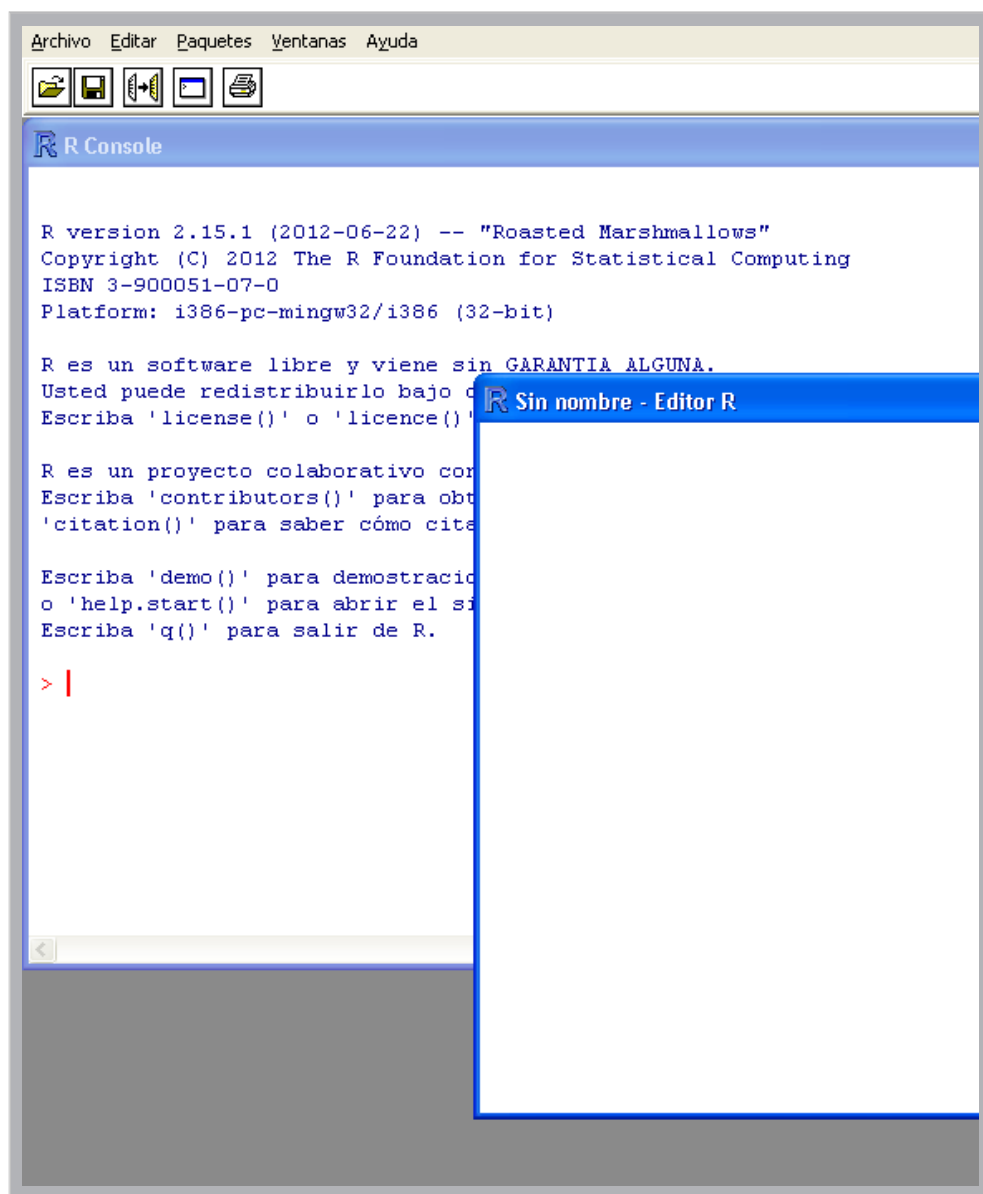
- Al ubicarse en la opción Archivo, se despliegan varias opciones, cada una de las cuales tiene diferente función según se detalla enseguida.



## ■ Nuevo script:

Al ser utilizada, aparece una nueva ventana y se generan dos áreas en R así:

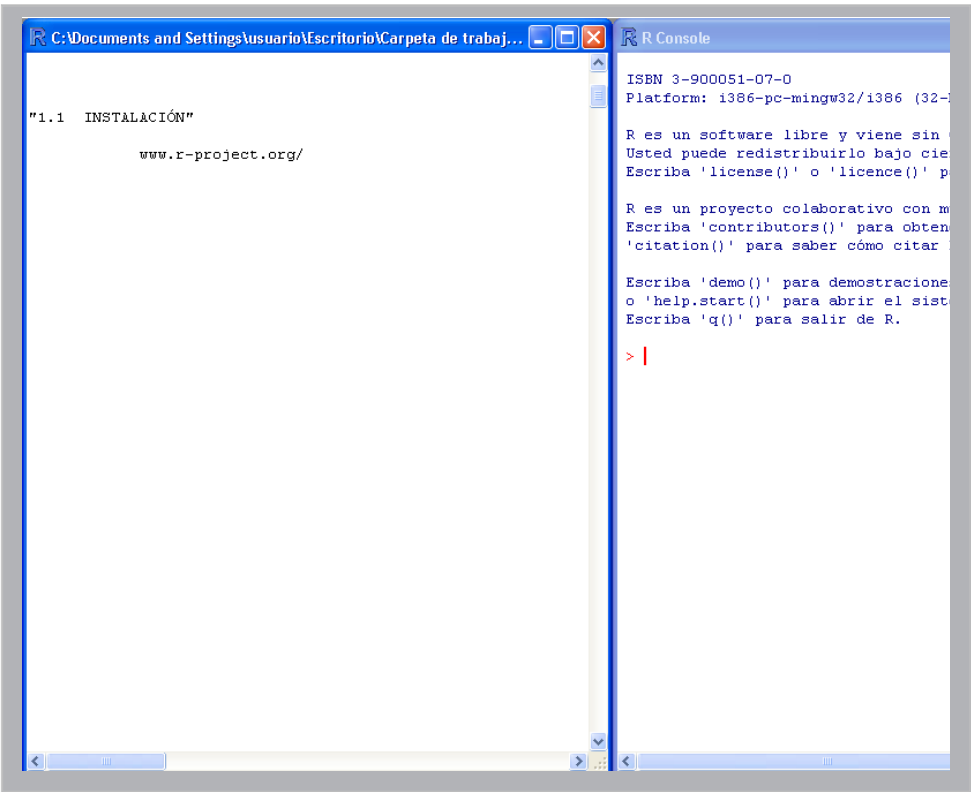
- La ventana nueva que es el editor, donde se escriben las diferentes instrucciones para correr el programa y en la que se puede hacer correcciones, copiar borrar, pegar y otras alternativas.
- La otra ventana es la consola, donde aparecen las instrucciones que fueron corridas y los resultados que ellas generan.



Para visualizar mejor el trabajo que se está ejecutando, se puede ir a la opción Ventanas en el panel superior, allí aparecen las opciones:

- Dividida verticalmente
- Dividida horizontalmente

Estas alternativas mejoran la presentación y hacen mucho más agradable el ambiente de trabajo, quedando en la forma como se muestra a continuación.



Nótese como en la ventana de la izquierda (Editor) aparecen las instrucciones y en la ventana derecha (Consola) los resultados obtenidos. Debe observarse también como es factible efectuar cambios y arreglos en el editor pero no en la consola. Es importante notar también que las opciones que se obtienen al desplegar la ventana de archivos son diferentes y dependerá de donde se esté situado.

Si se está en el editor las opciones son:

- |                     |                                                                |
|---------------------|----------------------------------------------------------------|
| <i>Nuevo script</i> | <i>Abre nueva ventana para instrucciones</i>                   |
| <i>Abrir script</i> | <i>Abre un script hecho con anterioridad</i>                   |
| <i>Guardar</i>      | <i>Guarda el script de las instrucciones</i>                   |
| <i>Guardar como</i> | <i>Guarda el script de las instrucciones bajo nuevo nombre</i> |

Cuando se está en la consola las instrucciones resultantes son:

<i>Nuevo script</i>	<i>Abre nueva ventana para instrucciones</i>
<i>Abrir script</i>	<i>Abre un script hecho con anterioridad</i>
<i>Mostrar archivos</i>	<i>Muestra los archivos según la extensión que se indique</i>
<i>Cargar área de trabajo</i>	<i>Carga el archivo en el cual se encuentran las instrucciones y que aparecerán en el editor</i>
<i>Guardar área de trabajo</i>	<i>Guarda el archivo en el cual se encuentran las instrucciones y que aparecerán en el editor</i>
<i>Cargar histórico</i>	<i>Carga el archivo R en donde se encuentran instrucciones en extensión R (*.R)</i>
<i>Guardar histórico</i>	<i>Guarda el archivo en extensión R (*.R)</i>
<i>Cambiar directorio</i>	<i>Define el lugar de trabajo</i>

R es un lenguaje orientado a objetos, es decir, los objetos a los que se les aplican los comandos de R cuentan con determinadas características y atributos, por lo que un mismo comando aplicado a diferentes objetos puede hacer diferentes cosas o no funcionar.

La asignación de objetos se hace por medio del siguiente símbolo (`<-`), también se puede con el signo igual, aunque este último en R es un operador matemático y esto puede generar confusiones. Para ejemplificar la situación se define un objeto A, al cual se le asigna un valor numérico que para el caso es 6.78, esto se hace así:

```
A <- 6.78
```

Para correr la instrucción se tienen las siguientes alternativas:

- i. Sombrear con el mouse la instrucción que se va a correr y luego se da clic derecho en el mouse, aparece la expresión correr línea y se da nuevamente clic en ella.
- ii. Ir a la segunda línea del panel y al ubicar el mouse sobre el tercer icono aparece la opción: correr línea, se da clic en ella y con esto se corre la instrucción.
- iii. Sombrear la instrucción con el mouse y pulsar simultáneamente las teclas (control, R).
- iv. Ubicar el mouse en algún sitio de la línea de la instrucción y oprimir simultáneamente las teclas control y R.



## 1.3 Contribuciones

R fue escrito inicialmente por Robert Gentleman y Ross Ihaka del Departamento de Estadística de la Universidad de Auckland en Nueva Zelanda y actualmente es el resultado del esfuerzo de colaboración de personas de todo el mundo. A finales de la década de los 90 se conformó el núcleo de desarrollo de R, quienes tienen la posibilidad de realizar modificaciones al código fuente.

Ante la anterior situación se han efectuado un gran número de contribuciones cuyo fin primordial es generar un paquete que contenga una funcionalidad en R de forma que al ser cargado mejora la capacidad, potencialidad y amplía las opciones para efectuar análisis de diferente índole. Estos paquetes se pueden visualizar fácilmente si se llega a la página principal de R, lo que se puede hacer si se despliega el icono de ayuda y se da clic en la opción página principal; situados allí, en el costado izquierdo aparece la opción: Packages, se señala ésta dando clic, al hacerlo se llega a la página denominada Contributed Packages, donde se encuentran ordenados en forma alfabética más de 4038 paquetes, cada uno de los cuales tiene un desarrollo particular en R y en el costado derecho se encuentra una breve reseña sobre lo que cada uno hace.



## 1.4 Manuales

En la página principal de R y a la cual se llega al desplegar la ventana de ayuda, se observa en el costado inferior izquierdo la opción [Manuales](#), al emplearla aparecen distintos manuales y la opción para bajarlos en formato PDF, tal como se muestra enseguida:

- **An Introduction to R** is based on the former "Notes on R", gives an introduction to the language and how to use R for doing statistical analysis and graphics. [[browse HTML](#) | [download PDF](#)]
- A draft of **The R language definition** documents the language *per se*. That is, the objects that it works on, and the details of the expression evaluation process, which are useful to know when programming R functions. [[browse HTML](#) | [download PDF](#)]
- **Writing R Extensions** covers how to create your own packages, write R help files, and the foreign language (C, C++, Fortran, ...) interfaces. [[browse HTML](#) | [download PDF](#)]
- **R Data Import/Export** describes the import and export facilities available either in R itself or via packages which are available from CRAN. [[browse HTML](#) | [download PDF](#)]
- **R Installation and Administration** [[browse HTML](#) | [download PDF](#)]
- **R Internals**: a guide to the internal structures of R and coding standards for the core team working on R itself. [[browse HTML](#) | [download PDF](#)]
- **The R Reference Index**: contains all help files of the R standard and recommended packages in printable form. [[download PDF, 16MB, approx. 3100 pages](#)]

En el penúltimo párrafo de esta página aparece la siguiente leyenda: contributed documentation, al hacer uso de ella se llega a una nueva página en la que figuran los manuales y tutoriales provistos por usuarios de R de todo el mundo. Esta página presenta la ventaja de ofrecer diversos manuales en distintas lenguas que vienen clasificados según el idioma.



## 1.5 Ayudas en R

### 1.5.1 Algunas direcciones importantes

Hay varias direcciones en la cuales se pueden encontrar diversas ayudas, algunas de importancia son las siguientes:

<http://cran.es.r-project.org/faqs.html>

En esta dirección encuentra una página en la que se dispone de tres colecciones de respuestas a las preguntas más frecuentes en R.

El FAQ R es la colección general y contiene información útil para los usuarios en todas las plataformas (Linux, Mac, Unix, Windows). Además hay dos específicos de la plataforma que son:

- El R MacOS X : Preguntas para todos los usuarios de sistemas operativos de Apple.
- El R Windows FAQ para todos los usuarios de los sistemas operativos de Microsoft.

Tenga en cuenta que estas dos últimas son complementarias a las preguntas generales de R.

El CRAN Espejo COMO explica cómo configurar un nuevo espejo en su institución.

<http://cran.es.r-project.org/search.html>

<http://finzi.psych.upenn.edu/search.html>

Las dos direcciones permiten hacer las búsquedas no sólo sobre las listas de e-mail sino también sobre la documentación (incluyendo paquetes).

### 1.5.2 Ayudas dentro del programa

R contiene un conjunto de instrucciones con las cuales se puede obtener ayuda sobre alguna función en particular, algunas de las cuales se mencionan enseguida:

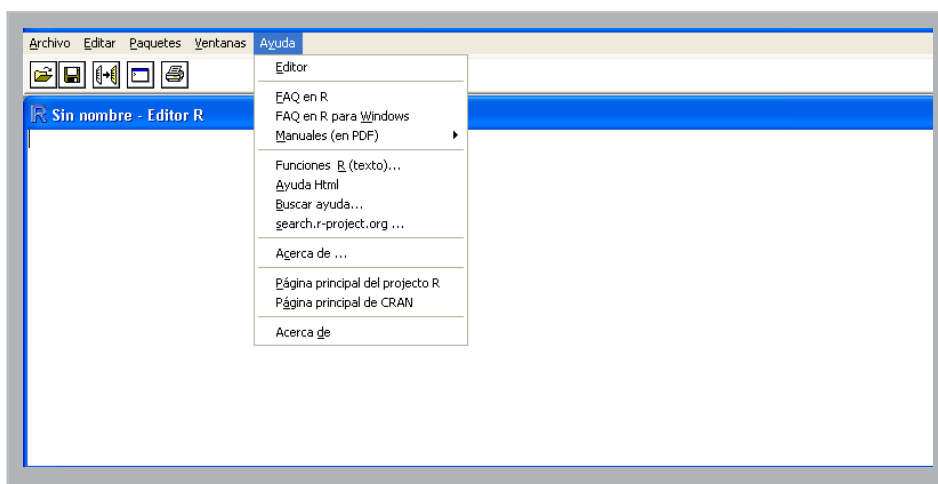
La opción ? seguida de la expresión con la que R lo reconoce, permite en forma rápida dar una idea de lo que hace la instrucción y su manejo en R. Por ejemplo para buscar ayuda sobre la distribución normal, la instrucción es:

`?norm`

Si se desea buscar ayuda para iniciar en R, aplicar:

```
help.start()
help.search()
apropose()
find()
```

Otra alternativa es ir al icono ayuda del panel superior, al desplegarlo muestra las siguientes opciones:



■ Funciones R (texto). Opción en la que se le debe ubicar el nombre de la función en el recuadro que aparece cuando se llama.

■ Ayuda Html. Se llega en forma rápida a una página de R, en donde se localiza información sobre los siguientes aspectos:

- Manuales: Una introducción a R, Escribiendo extensiones con R, Importar y exportar datos en R, definición del lenguaje R, instalación y administración
- Referencias: Paquetes, búsqueda de palabras claves
- Material Misceláneo de R: Acerca de R, licencias, autores, preguntas frecuentes, etc.

■ Buscar ayuda: En el recuadro que aparece cuando se llama la opción, se debe colocar un nombre que puede hacer referencia a: comando; instrucción; parte de una instrucción, y de la cual se desea ayuda.

■ Search.r-project.org: Busca palabras en lista de archivos de ayuda y documentación.



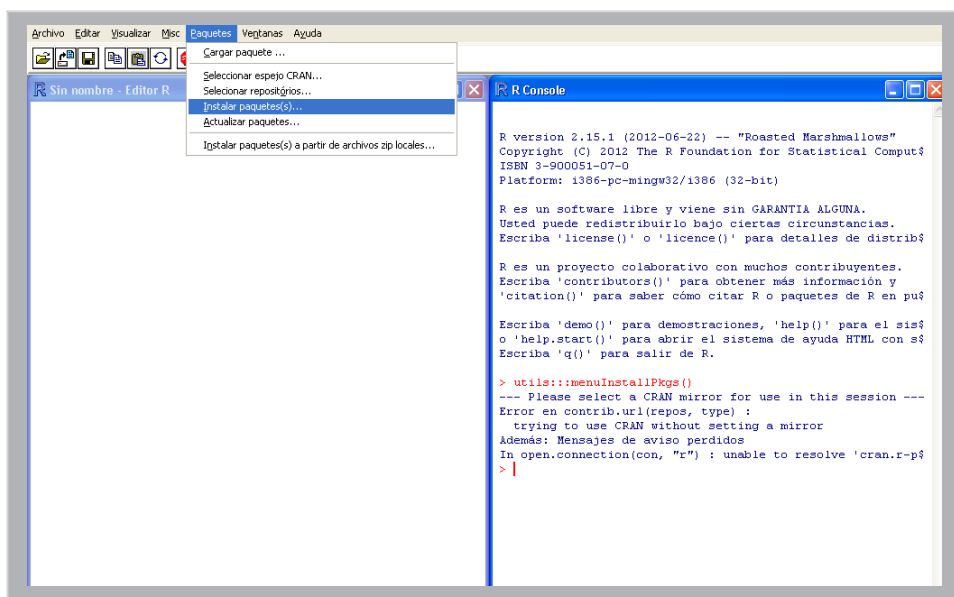
## 1.6 Cargar paquetes

Debe tener presente que al instalar el programa R, se cargan por defecto un conjunto de paquetes que es posible visualizar al desplegar la opción que lleva este mismo nombre y luego dar clic en cargar. Si se requiere de alguno en especial para desarrollar cálculos particulares y no se tiene aún, se puede obtener usando alguna de las dos opciones siguientes:

### 1.6.1 Instalar y cargar el paquete

Se deben seguir los siguientes pasos:

- Ubicarse en R y desplegar la ventana de paquetes.
- Dar clic en la opción instalar paquetes.



- En instalar paquetes, seleccione el espejo cran (Colombia) y busque en el listado el paquete que desea, una vez localizado de doble clic y observará como el sistema lo baja.
- Hecho lo anterior, en la opción paquetes dar clic en el instructivo cargar paquetes, aparece un listado en donde se localizan los paquetes que trae R por defecto junto con los que acaba de bajar. Al tener cargado el paquete, para ser usado se debe correr la siguiente instrucción:

**library(nombre del paquete)**

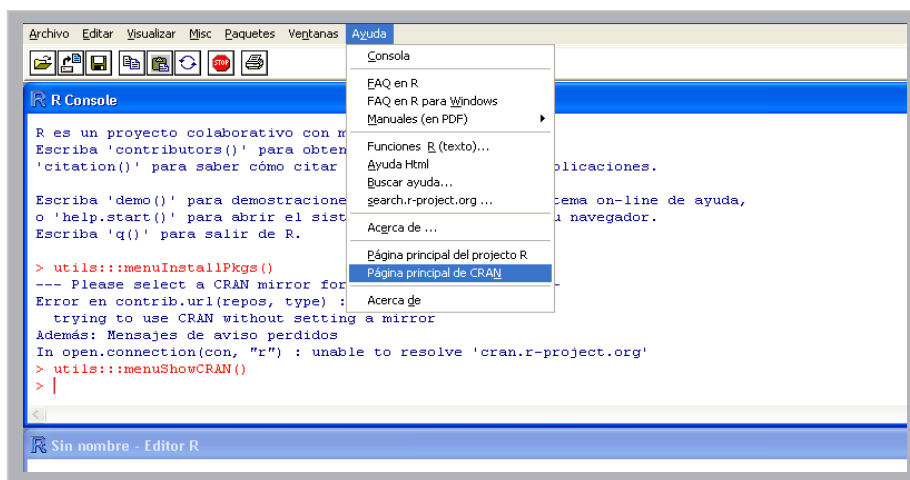


## 1.6.2. Instalar paquetes a partir de archivos zip locales

Mediante este proceso el paquete de interés se baja de la red como un archivo comprimido y se tiene en el sistema para usos posteriores.

Antes de señalar el proceso que se debe seguir para cargar paquetes de la página principal R, es importante indicarle a R el lugar o sitio de trabajo en el cual se encuentra la información que será utilizada y donde se instalarán los diferentes paquetes que se emplearán para desarrollar los procesos según el siguiente detalle:

- Lo primero que se debe hacer es definir y denominar una carpeta, que para el caso se denomina carpeta de trabajo, esta deberá estar ubicada en un lugar del sistema donde sea fácil el acceso y en el cual usted tenga sus preferencias para trabajar (puede ser mis documentos).
- Definido el sitio, se debe cambiar el directorio de R, para lo cual se debe estar en la consola y allí dirigirse al comando archivo, donde se encuentra la opción cambiar de directorio, al dar clic en ella, se despliega una ventana en la cual se tiene la posibilidad de crear la carpeta de trabajo o de indicar el lugar donde está la carpeta, hecha la anterior operación se debe dar aceptar.
- Con lo anterior siempre que grabe información ya sea script o resultados, estos quedarán en la carpeta archivo de trabajo; también deberá ubicar en esta carpeta la información que va a ser cargada o los paquetes que deberán ser leídos por el programa.
- Al estar interesado en un paquete particular se debe seguir el siguiente proceso para cargarlo:
  - Ir a la página CRAN: The Comprehensive R Archive Network ([cran.r-project.org/](http://cran.r-project.org/)). También puede hacerlo por medio de la opción ayuda que aparece en el panel superior, tal como se indica en la figura inferior



- Una vez ubicado en la página principal, en el costado izquierdo observe el icono denominado [Packages](#), al dar clic en la opción le aparece una página en la cual se encuentra en la parte final los 4038 paquetes disponibles a la fecha y a los cuales puede acceder fácilmente al solicitar que sean presentados en forma ordenada, bien sea por el nombre o fecha de publicación.
- Desplegada la lista de opciones de paquetes, ubique el de interés y de clic en él. Al hacer esto, aparece una página en la que se encuentra el manual de referencia en formato PDF y el paquete que deberá descargarlo según sea el sistema operacional en el cual este trabajando, aunque por lo general es Windows binary.
- El archivo una vez descargado debe ser llevado a la carpeta de trabajo y en ningún momento requiere ser desempaquetado, ya que R trabaja con paquetes comprimidos.
- Retire el paquete del sitio donde lo guardó el sistema y ubíquelo en la carpeta de trabajo.
- Estando en el ambiente R, vaya a paquetes y utilice la opción instalar paquetes a partir de archivos zip locales. Al hacerlo, debe aparecer en la consola el mensaje: package 'Nombre del paquete' successfully unpacked and MD5 sums checked, que indica que todo está correcto.
- Siga la siguiente instrucción para cargar el paquete: library(nombre del paquete).
- Con esto, los paquetes quedan activos y listos para ser utilizados en la sesión de trabajo.



## 1.7 Acceso a la información

Aunque en las situaciones en las cuales la fuente de información es primaria (obtenida directamente por el investigador) es necesario ingresar los datos obtenidos haciendo que alguien los digite directamente a R, o a una hoja electrónica, o base de datos, o mediante alguna otra alternativa. En algunos casos diferentes los datos ya existen, ya que una fuente externa los produjo.

### 1.7.1 Digitando directamente

- Usando alguna estructura de datos (vectores, matrices, etc.)

#### ★ Ejemplo 1.1

```
Variable<- c (0,1,2,3,4)
```

- Creando un vector y usando el editor de datos para digitar la información.

Inicialmente se crea un vector sin datos de longitud abierta

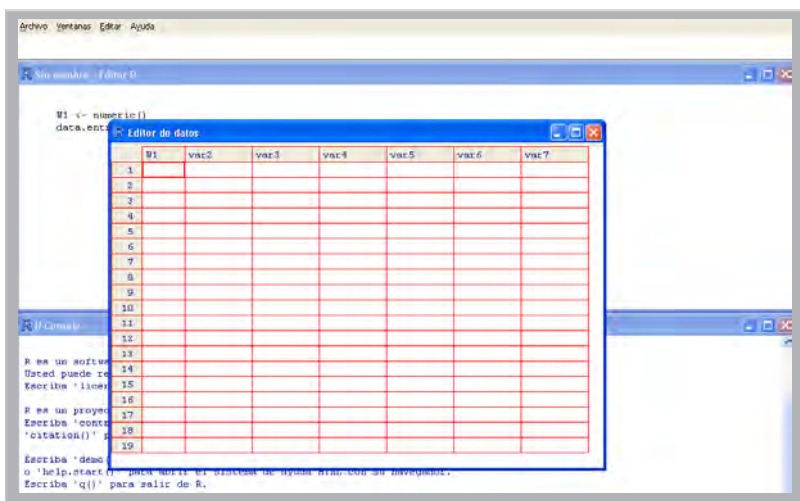
```
W <-numeric() # (también se puede fijar, si se da un valor.)
```

Se ingresa la información de w usando la instrucción: `data.entry(w)`: Esto implica la utilización del editor de datos con que cuenta R, donde se introduce la información. Para una mejor ilustración, observe con detenimiento el siguiente ejemplo.

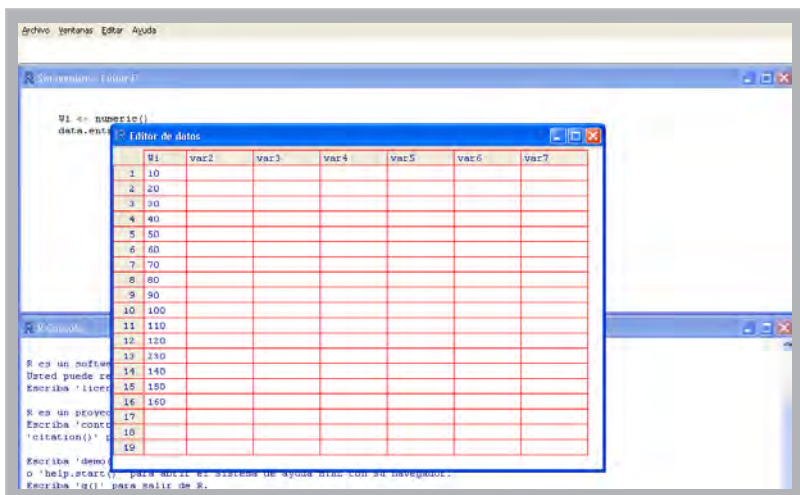
```
w1<-numeric()
```

```
data.entry(w1)
```

Al correr las instrucciones aparece el editor, como se indica enseguida.



Se ingresa la información en el editor y luego se cierra



Finalmente, llame al vector de datos `w1` que se creó y verá que este contiene la información.

- Modificando o ingresando nueva información desde la consola y utilizando el editor de datos.

En este caso se tiene un objeto que contiene cierta información. Estando en la consola, se emplea la opción editar que aparece en la parte superior de la pantalla, allí dé clic en el editor de datos y encuentra un recuadro solicitando el nombre del objeto que contiene los datos que desea ampliar o modificar, al ingresar el nombre del objeto y dar ok, aparece el editor con la información la cual podrá ampliar, modificar o reducir. Cuando termine el proceso cierre la ventana y luego dígame al sistema que desea salvar los cambios.

- Ingresando dato por dato desde la consola.

Es posible ingresar los datos desde la consola, para lo cual se debe utilizar la instrucción `scan()` así:

```
B <- scan()
```

Al correr la instrucción anterior el cursor se desplaza a la consola y queda a la espera para que usted desde allí empiece a ingresar la información. Los datos se van ingresando uno por uno y cuando finalice oprima dos veces enter.

Otra forma de ingresar información con esta opción es ejecutar el comando y luego copiar la columna de datos sin títulos y pegándola en la consola al lado derecho donde aparece el número 1.

Otras formas están en el empleo de comandos o instrucciones que comunican a R con diferentes formatos para realizar la lectura. Se verán a continuación algunas de ellas.

## 1.7.2 Información en formato txt

Para leer archivos de datos que están en formato texto, se debe hacer el siguiente proceso:

- Coloque el archivo que desea importar en la carpeta de trabajo, la cual usted debió de definir al iniciar la sesión en R. Si no lo ha hecho aún, realice el proceso indicado en el inciso anterior.
- Dé a R la instrucción de leer la información así (preferiblemente asignándola a un objeto):

```
read.table("nombre del archivo en comillas.txt",header = T)
```

Se pueden presentar varias situaciones según las columnas estén o no estén tituladas, sin embargo, para cualquiera de los casos siempre irá la instrucción indicada anteriormente y se deben tener presente las situaciones siguientes:

Si las columnas están tituladas se indica a R con la opción `header = T (TRUE)`, si no lo están, se señala con la opción `F (FALSE)`, en este último caso R titula cada columna denominándolas `v1, v2, etc.`

Observe y corra los ejemplos que se ofrecen enseguida.

En el CD que acompaña el texto en la carpeta No. 1 encuentra un archivo en formato txt que se identifica con el nombre `Créditos bancarios`, allí se encuentra localizada información sobre créditos entregados por una entidad bancaria.

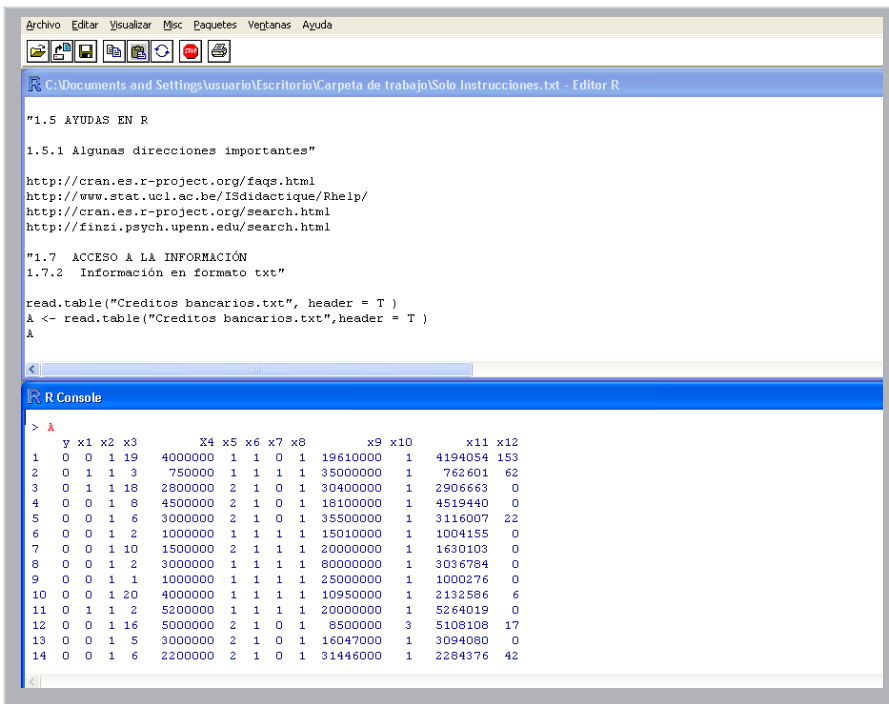
Para leer la información la instrucción es:

```
read.table("Creditos bancarios.txt", header = T )
```

o se le identifica con el objeto `A` y la instrucción es entonces:

```
A <- read.table("Creditos bancarios.txt",header = T )  
A
```

El resultado obtenido es el siguiente:



The screenshot shows an R Editor window with a menu bar (Archivo, Editar, Visualizar, Misc, Paquetes, Ventanas, Ayuda) and a toolbar. The script editor contains the following text:

```
"1.5 AYUDAS EN R  
1.5.1 Algunas direcciones importantes"  
  
http://cran.es.r-project.org/faqs.html  
http://www.stat.ucl.ac.be/ISdidactique/Rhelp/  
http://cran.es.r-project.org/search.html  
http://finzi.psych.upenn.edu/search.html  
  
"1.7 ACCESO A LA INFORMACIÓN  
1.7.2 Información en formato txt"  
  
read.table("Creditos bancarios.txt", header = T )  
A <- read.table("Creditos bancarios.txt",header = T )  
A
```

The R Console at the bottom displays the output of the script, showing a data frame with 14 rows and 12 columns. The columns are labeled `y`, `x1`, `x2`, `x3`, `x4`, `x5`, `x6`, `x7`, `x8`, `x9`, `x10`, `x11`, and `x12`. The data is as follows:

	y	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12
1	0	0	1	19	4000000	1	1	0	1	19610000	1	4194054	153
2	0	1	1	3	750000	1	1	1	1	35000000	1	762601	62
3	0	1	1	18	2800000	2	1	0	1	30400000	1	2906663	0
4	0	0	1	8	4500000	2	1	0	1	18100000	1	4519440	0
5	0	0	1	6	3000000	2	1	0	1	35500000	1	3116007	22
6	0	0	1	2	1000000	1	1	1	1	15010000	1	1004155	0
7	0	0	1	10	1500000	2	1	1	1	20000000	1	1630103	0
8	0	0	1	2	3000000	1	1	1	1	80000000	1	3036784	0
9	0	0	1	1	1000000	1	1	1	1	25000000	1	1000276	0
10	0	0	1	20	4000000	1	1	1	1	10950000	1	2132586	6
11	0	1	1	2	5200000	1	1	1	1	20000000	1	5264019	0
12	0	0	1	16	5000000	2	1	0	1	8500000	3	5108108	17
13	0	0	1	5	3000000	2	1	0	1	16047000	1	3094080	0
14	0	0	1	6	2200000	2	1	0	1	31446000	1	2284376	42

Los datos que integran la información con formato txt, puede estar separada por comas espacios, o por tabulaciones, situaciones en las cuales la instrucción será:

Para datos separados por comas ( , )

```
read.table("Creditos bancarios.txt", header = T, sep = ",")
```

Para datos separados por tabulaciones o espacios en blanco

```
read.table("Creditos bancarios.txt", header = T, "\t")
```

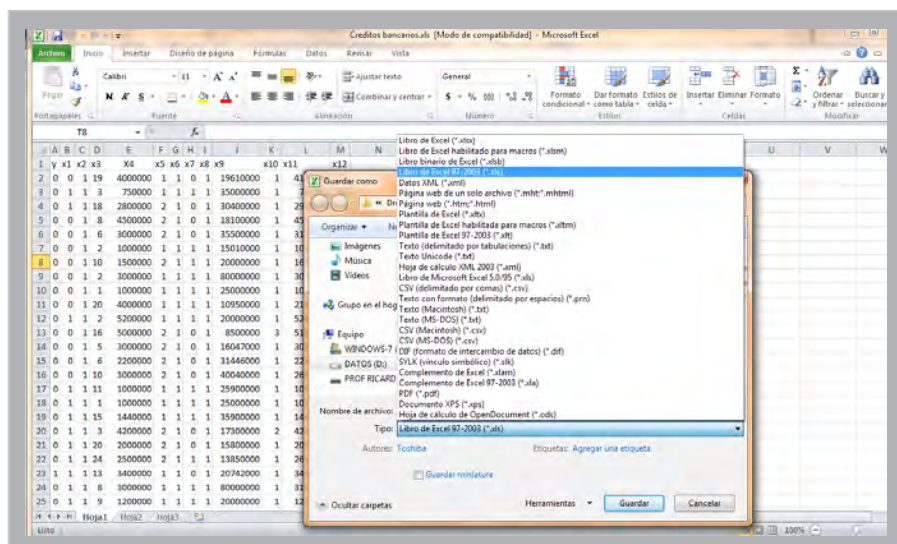
### 1.7.3 Información en hojas electrónicas. Formato EXCEL con extensión xls

Para cargar información desde una hoja electrónica de Excel en formato xls, se debe cargar el paquete RODBC, pero antes de hacer esto, es indispensable que se hagan unos ajustes a la información que reposa en la hoja de cálculo para que pueda ser leída por R. Estos ajustes son ilustrados tomando nuevamente el ejemplo de créditos bancarios que ahora viene ofrecido en formato xls, los pasos son:

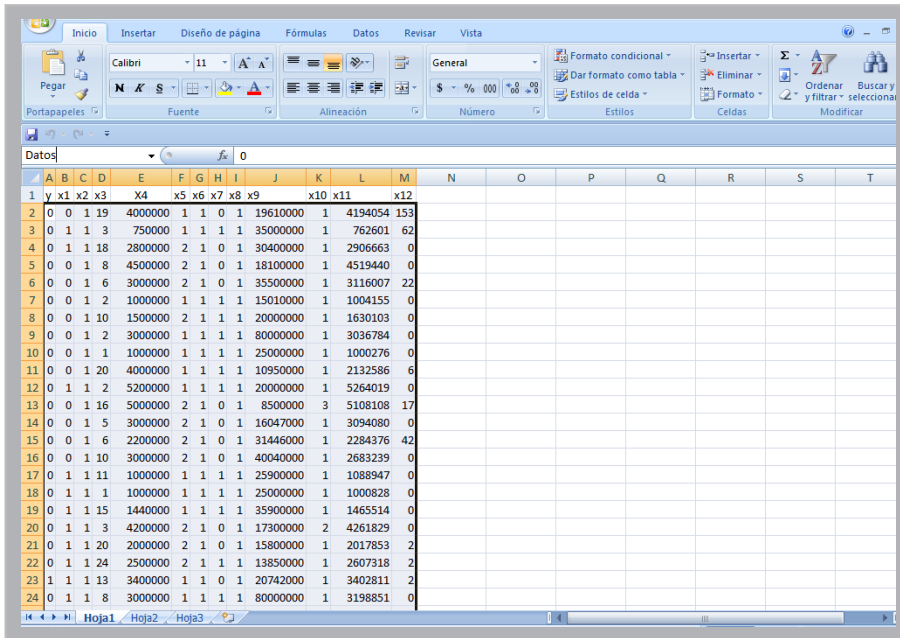
- Verifique que las columnas estén tituladas y que estén conformados por una sola palabra y libres de caracteres especiales (\*, /, -).
- Confirme que la información tienen títulos en la primera fila y de allí hacia abajo solo datos.
- Vaya al botón office y emplee la opción guardar como. En la parte inferior de la ventana que se despliega, aparece la leyenda

*guardar como tipo*

asuma esta opción y localice la alternativa: Libro de EXCEL 97-2003. Guárdelo en este formato para lo cual solo le basta dar clic en esta opción y renombre el archivo.



- Abra el archivo EXCEL en extensión xls. Resalte con el mouse las celdas donde reposa la información que desea importar, teniendo cuidado de no marcar los títulos de las columnas y en el cuadro de nombres identifíquelo colocando cualquier calificativo, que para el caso corresponde a Datos y oprima la tecla enter.



- Cierre el archivo de Excel.

Dé las siguientes instrucciones en R:

- Para establecer la comunicación entre R con EXCEL digite  

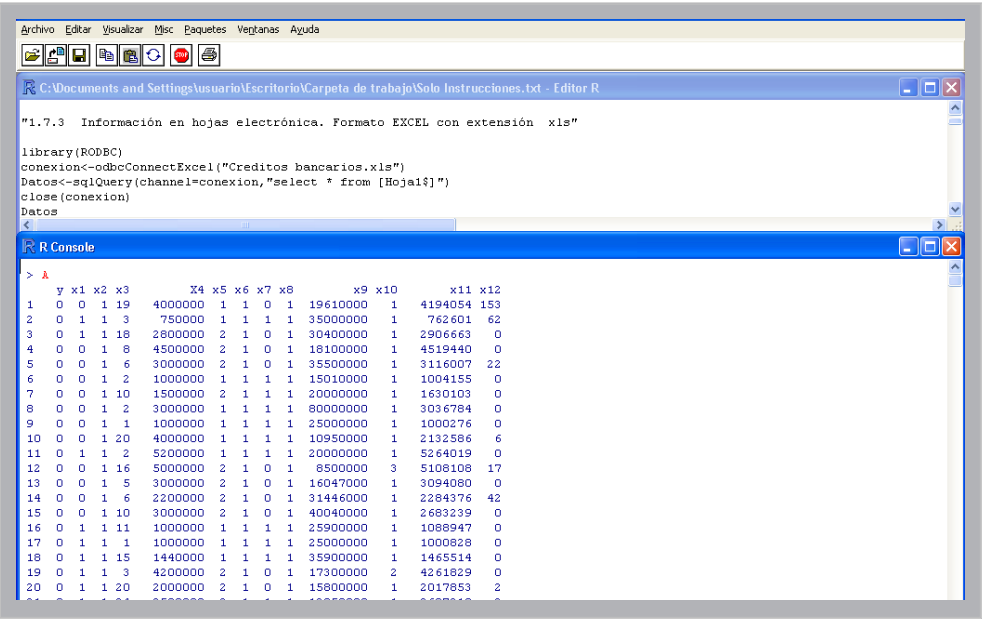
```
library(RODBC)
conexion<-odbcConnectExcel("Creditos bancarios.xls")
```
- Se importan los datos de la hoja en EXCEL donde reposa la información, para el caso Hoja1, usando  

```
Datos<-sqlQuery(channel=conexion,"select * from [Hoja1$]")
```
- Se cierra la conexión  

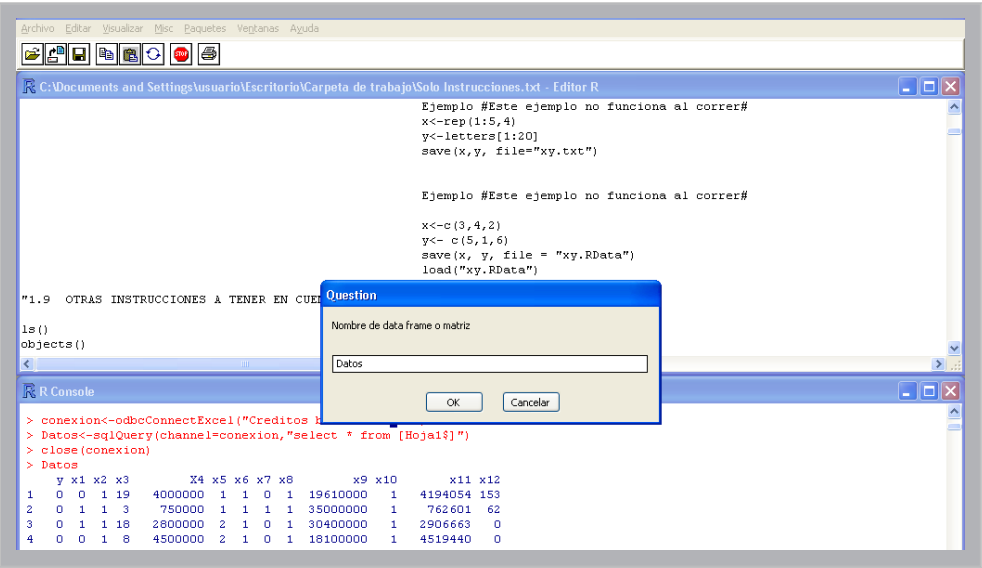
```
close(conexion)
```
- Finalmente, se pide a R que muestre la información  

```
Datos
```

El siguiente es el resultado obtenido.



Es posible hacer modificaciones a la base de datos que ha sido cargada desde EXCEL. Para esto debe ubicarse en la consola y desde allí dirigirse al comando Editor de datos, situado allí, se despliega una ventana donde debe localizar la opción Editor de datos, al hacerlo y dar clic en ella, obtiene un recuadro en el que debe colocar el calificativo con el cual identificó la información al guardarla en EXCEL en extensión xls (Datos, para el caso que se está ilustrando). Al dar OK se obtiene el editor de datos, al cual le puede quitar, modificar o anexar información.





Estando en la hoja del editor de datos, ubíquese en la última fila y oprima enter, al hacerlo salen nuevas filas en las cuales se podrá acceder la información que desee, cuando termine, cierre el editor y la información quedará incluida.

R C:\Documents and Settings\usuario\Escritorio\Carpeta de trabajo\Solo Instrucciones.txt - Editor R

A <- read.table("Creditos bancarios.txt",header = T )

A

1.7.3 Informa

library(RODBC)

conexion<-odbc

Datos<-sqlQuer

Datos

close(conexion)

"Donde los nom

X1 1 Zonal

X2 1 Obligac

X3

R Console

375 0 1 1 6

376 0 1 1 15

377 1 0 1 20

378 0 1 1 9

379 0 1 1 19

380 0 0 1 18

381 0 0 0 1

382 0 0 0 1

383 0 0 1 11

384 1 0 1 19

385 0 0 1 13

386 0 0 1 5

387 0 0 1 1

388 5555 5555 5555 5555

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

1001

1002

1003

1004

1005

1006

1007

1008

1009

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1020

1021

1022

1023

1024

1025

1026

1027

1028

1029

1030

1031

1032

1033

1034

1035

1036

1037

1038

1039

1040

1041

1042

1043

1044

1045

1046

1047

1048

1049

1050

1051

1052

1053

1054

1055

1056

1057

1058

1059

1060

1061

1062

1063

1064

1065

1066

1067

1068

1069

1070

1071

1072

1073

1074

1075

1076

1077

1078

1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

1092

1093

1094

1095

1096

1097

1098

1099

1100

1101

1102

1103

1104

1105

1106

1107

1108

1109

1110

1111

1112

1113

1114

1115

1116

1117

1118

1119

1120

1121

1122

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

1151

1152

1153

1154

1155

1156

1157

1158

1159

1160

1161

1162

1163

1164

1165

1166

1167

1168

1169

1170

1171

1172

1173

1174

1175

1176

1177

1178

1179

1180

1181

1182

1183

1184

1185

1186

1187

1188

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

1201

1202

1203

1204

1205

1206

1207

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

1226

1227

1228

1229

1230

1231

1232

1233

1234

1235

1236

1237

1238

1239

1240

1241

1242

1243

1244

1245

1246

1247

1248

1249

1250

1251

1252

1253

1254

1255

1256

1257

1258

1259

1260

1261

1262

1263

1264

1265

1266

1267

1268

1269

1270

1271

1272

1273

1274

1275

1276

1277

1278

1279

1280

1281

1282

1283

1284

1285

1286

1287

1288

1289

1290

1291

1292

1293

1294

1295

1296

1297

1298

1299

1300

1301

1302

1303

1304

1305

1306

1307

1308

1309

1310

1311

1312

1313

1314

1315

1316

1317

1318

1319

1320

1321

1322

1323

1324

1325

1326

1327

1328

1329

1330

1331

1332

1333

1334

1335

1336

1337

1338

1339

1340

1341

1342

1343

1344

1345

1346

1347

1348

1349

1350

1351

1352

1353

1354

1355

1356

1357

1358

1359

1360

1361

1362

1363

1364

1365

1366

1367

1368

1369

1370

1371

1372

1373

1374

1375

1376

1377

1378

1379

1380

1381

1382

1383

1384

1385

1386

1387

1388

1389

1390

1391

1392

1393

1394

1395

1396

1397

1398

1399

1400

1401

1402

1403

1404

1405

1406

1407

1408

1409

1410

1411

1412

1413

1414

1415

1416

1417

1418

1419

1420

1421

1422

1423

1424

1425

1426

1427

1428

1429

1430

1431

1432

1433

1434

1435

1436

1437

1438

1439

1440

1441

1442

1443

1444

1445

1446

1447

1448

1449

1450

1451

1452

1453

1454

1455

1456

1457

1458

1459

1460

1461

1462

1463

1464

1465

1466

1467

1468

1469

1470

1471

1472

1473

1474

1475

1476

1477

1478

1479

1480

1481

1482

1483

1484

1485

1486

1487

1488

1489

1490

1491

1492

1493

1494

1495

1496

1497

1498

1499

1500

1501

1502

1503

1504

1505

1506

1507

1508

1509

1510

1511

1512

1513

1514

1515

1516

1517

1518

1519

1520

1521

1522

1523

1524

1525

1526

1527

1528

1529

1530

1531

1532

1533

1534

1535

1536

1537

1538

1539

1540

1541

1542

1543

1544

1545

1546

1547

1548

1549

1550

1551

1552

1553

1554

1555

1556

1557

1558

1559

1560

1561

1562

1563

1564

1565

1566

1567

1568

1569

1570

1571

1572

1573

1574

1575

1576

1577

1578

1579

1580

1581

1582

1583

1584

1585

1586

1587

1588

1589

1590

1591

1592

1593

1594

1595

1596

1597

1598

1599

1600

1601

1602

1603

1604

1605

1606

1607

1608

1609

1610

1611

1612

1613

1614

1615

1616

1617

1618

1619

1620

1621

1622

1623

1624

1625

1626

1627

1628

1629

1630

1631

1632

1633

1634

1635

1636

1637

1638

1639

1640

1641

1642

1643

1644

1645

1646

1647

1648

1649

1650

1651

1652

1653

1654

1655

1656

1657

1658

1659

1660

1661

1662

1663

1664

1665

1666

1667

1668

1669

1670

1671

1672

1673

1674

1675

1676

1677

1678

1679

1680

1681

1682

1683

1684

1685

1686

1687

1688

1689

1690

1691

1692

1693

1694

1695

1696

1697

1698

1699

1700

1701

1702

1703

1704

1705

1706

1707

1708

1709

1710

1711

1712

1713

1714

1715

1716

1717

1718

1719

1720

1721

1722

1723

1724

1725

1726

1727

1728

1729

1730

1731

1732

1733

1734

1735

1736

1737

1738

1739

1740

1741

1742

1743

1744

1745

1746

1747

1748

1749

1750

1751

1752

1753

1754

1755

1756

1757

1758

1759

1760

1761

1762

1763

1764

1765

1766

1767

1768

1769

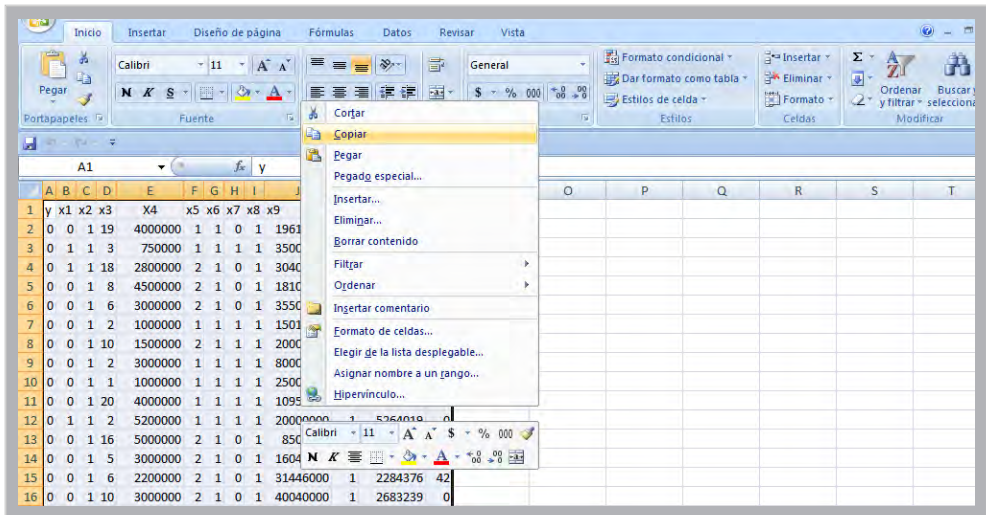
1770

1771

1772

1773

1774



Diríjase a R y en el editor defina un objeto de la siguiente manera:

```
Objeto <- read.delim("clipboard")
Objeto
```

Instrucción que debe ser ejecutada de la siguiente manera:

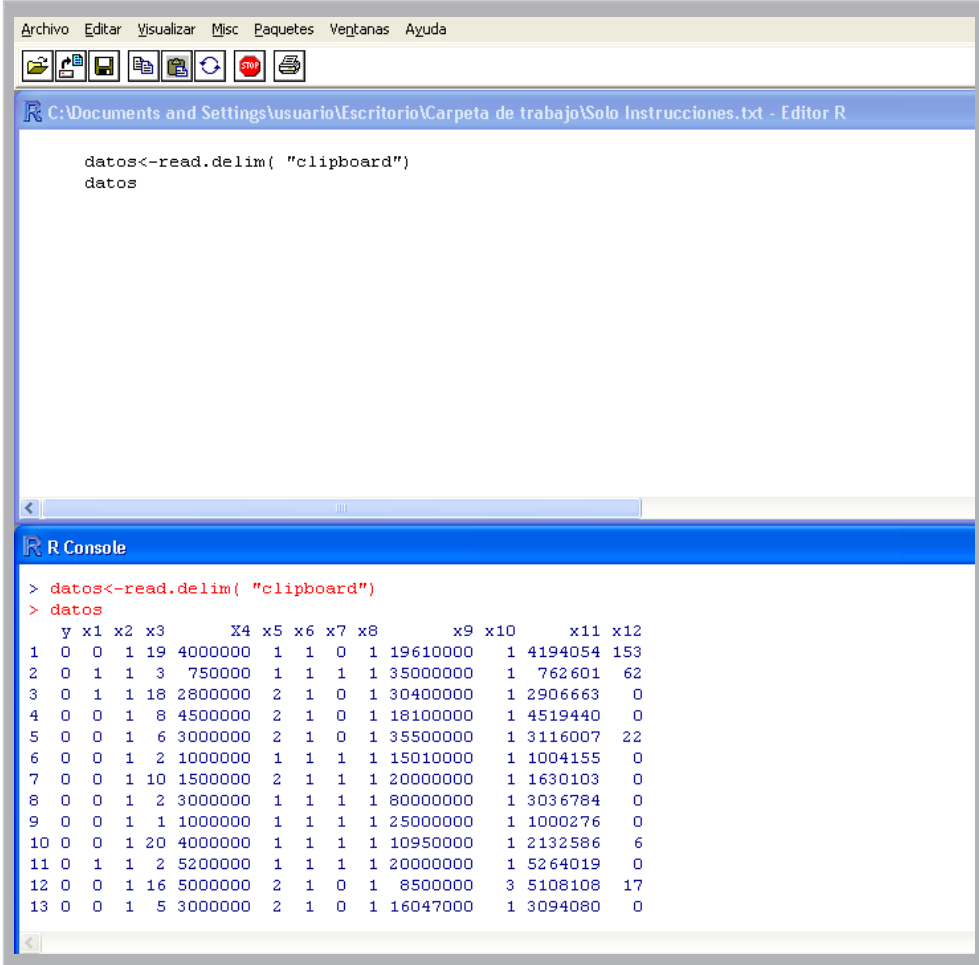
- Ubique el cursor al final de la primera línea de la instrucción y de clic derecho.
- Solicite a R que le muestre la información corriendo la segunda línea. En la consola le aparecerán los datos bajo el nombre objeto, ya que usted así los definio desde un principio.

Para el ejemplo que se ilustra enseguida, usando el archivo créditos bancarios.xls, que es la base donde se localiza la información y que se desea copiar. Para esto se define un objeto llamado datos y la forma para importarlos a R se ilustra enseguida:

```
datos<-read.delim("clipboard")
datos
```



Obteniendo entonces lo siguiente:



The screenshot shows the R Editor window with a menu bar (Archivo, Editar, Visualizar, Misc, Paquetes, Ventanas, Ayuda) and a toolbar. The title bar indicates the file path: C:\Documents and Settings\usuario\Escritorio\Carpeta de trabajo\Solo Instrucciones.txt - Editor R. The editor contains the R code: `datos<-read.delim( "clipboard")` and `datos`. Below the editor is the R Console, which shows the execution of the code and the resulting data table.

```
> datos<-read.delim( "clipboard")
> datos
```

	y	x1	x2	x3	X4	x5	x6	x7	x8	x9	x10	x11	x12
1	0	0	1	19	4000000	1	1	0	1	19610000	1	4194054	153
2	0	1	1	3	750000	1	1	1	1	35000000	1	762601	62
3	0	1	1	18	2800000	2	1	0	1	30400000	1	2906663	0
4	0	0	1	8	4500000	2	1	0	1	18100000	1	4519440	0
5	0	0	1	6	3000000	2	1	0	1	35500000	1	3116007	22
6	0	0	1	2	1000000	1	1	1	1	15010000	1	1004155	0
7	0	0	1	10	1500000	2	1	1	1	20000000	1	1630103	0
8	0	0	1	2	3000000	1	1	1	1	80000000	1	3036784	0
9	0	0	1	1	1000000	1	1	1	1	25000000	1	1000276	0
10	0	0	1	20	4000000	1	1	1	1	10950000	1	2132586	6
11	0	1	1	2	5200000	1	1	1	1	20000000	1	5264019	0
12	0	0	1	16	5000000	2	1	0	1	8500000	3	5108108	17
13	0	0	1	5	3000000	2	1	0	1	16047000	1	3094080	0

## 1.7.5 Otros

El comando `data()` permite leer archivos de otras librerías (paquetes), el paquete debe estar instalado, de lo contrario se debe cargar.

### ★ Ejemplo 1.2

- `data(iris, package="datasets")`  
`data(iris) # como opcional a la anterior`  
`iris`
- `library(MASS)`  
`data(Traffic, package = "MASS")`  
`Traffic`



## 1.8 Exportar datos

R ofrece la posibilidad de exportar un conjunto de datos para que pueda ser leído por cualquier programa. La forma más sencilla es el formato texto.

### ■ Primera alternativa

```
write.table(a,file="fichero.txt",sep="\\",na="NA",dec=".",row.names=TRUE, col.names=TRUE)
```

- a: Nombre del objeto en R donde reposan los datos que se desean exportar
- fichero.txt: Nombre que tendrá el archivo donde estarán los datos exportados y que viene dado en extensión txt
- Sep: "t" quiere decir que los datos estarán separados por una tabulación. También es posible colocar una coma o un espacio
- Na: "NA" Si hay datos faltantes estos aparecen como NA. También es posible dejar los espacios en blanco, caso en el cual se coloca na = " "
- Dec: "." indica que el carácter con el cual se separan los decimales (.)
- row.names: Se emplea para que coloque los títulos de las filas
- col.names: Se utiliza para que coloque los nombres de las variables

Nota: Las COMILLAS deben ser escritas empleando el formato texto.

Para indicar el proceso de manejo de datos en R, se toma como base el archivo Frecuencias.xls, este se lleva a R, allí se duplica la información para luego ser pasada a formato texto bajo el título: Frecuenciasampliado. El proceso es el siguiente:

Se pasa la información de Excel a R

```
library(RODBC)  
conexion<-odbcConnectExcel("Creditos bancarios.xls")  
Datos<-sqlQuery(channel=conexion,"select * from [Hoja1$]")  
close(conexion)  
Datos
```

Se duplica la información, para lo cual se renombra el objeto Datos bajo el título A.

```
A <- Datos  
B <- Datos
```

Se une la información de los objetos A, B a través de la instrucción

```
Z <- cbind(A,B)
```

Se exporta la información

```
write.table(Z,file="fichero.txt",sep="\t",na="NA",dec=".",row.names =FALSE,col.names=TRUE)  
Z
```

Vaya a la carpeta de trabajo y allí debe aparecer el archivo fichero en formato txt. Si desea ubicarla en hoja electrónica, debe sombrear con el mouse toda la información en el formato texto, utilice la opción copiar, puede ser con CTRL C, luego sitúese en cualquier celda de la Hoja EXCEL y allí ordené pegar, CTRL V.

### ■ Segunda alternativa

Con el comando `save()`, y considerando que los archivos guardados con `save()` se cargan con el comando `load()`.

#### ★ Ejemplo 1.3

```
save(pas, file="pasajeros mov.txt")  
load("pasajeros mov.txt")
```

#### ★ Ejemplo 1.4

```
x<-rep(1:5,4)  
y<-letters[1:20]  
save(x,y, file="xy.RData")  
load("xy.RData")  
x  
y
```

### ■ Tercera alternativa

Implica la aplicación de la extensión `.Rdata`, teniendo en cuenta que este tipo de archivo se carga al iniciarse R.

```
save(x, y, file = "xy.RData")
```

#### ★ Ejemplo 1.5

```
x<-c(3,4,2)  
y<- c(5,1,6)  
save(x, y, file = "xy.RData")  
load("xy.RData")  
x  
y
```



## 1.9 Otras instrucciones a tener en cuenta

Durante una sesión de trabajo con R, los objetos que se crean se almacenan por nombre. Para la obtención de los nombres de todos los objetos almacenados en R, se emplea cualquiera de las siguientes instrucciones:

**ls()**  
**objects()**

- |                                           |                                                |
|-------------------------------------------|------------------------------------------------|
| ■ Para borrar un objeto en particular     | <i>rm(Nombre del objeto)</i>                   |
| ■ Para borrar todos los objetos           | <i>rm(list = ls())</i>                         |
| ■ Para salir del programa                 | <i>q()</i>                                     |
| ■ Para observar los gráficos en R         | <i>demo(graphics)</i>                          |
| ■ Lista de archivos del área de trabajos  | <i>list.files()</i>                            |
| ■ Busca ayuda sobre X                     | <i>help(X)</i>                                 |
| ■ Para observar el contenido de un objeto | <i>print()</i>                                 |
| ■ Limpiar la pantalla                     | <i>Presione las teclas ctrl y l, al tiempo</i> |



# 2. Estructuras de Datos



Quien desee trabajar información estadística con R, debe conocer primero como trabaja el programa, de forma que logre una mayor eficiencia. Para esto, debe empezar por conocer las estructuras de datos en las cuales puede apoyarse y adicionalmente conocer como mínimo las aplicaciones estadísticas básicas de uso frecuente.

Se reconocen en R estructuras tipo: vectores, matrices, array, factores, data.frame, listas.

## 1. VECTORES

Los hay de tipo numérico, lógico o de carácter.



### 1.1. Vector numérico

Como su nombre lo indica son conjuntos de números separados por comas, y la representación para su acceso al sistema es `c()`. Dentro del paréntesis, se colocan los valores para la variable cuantitativa, cuyo nombre identifica al vector, el cual se reconoce en general como un objeto.

Los objetos se pueden clasificar en dos grandes grupos:

- **atomics:** todos los elementos que los componen son del mismo modo, como por ejemplo los vectores, matrices, series temporales.
- **recursivos:** pueden combinar una colección de otros objetos de diferente tipo, como son los data.frame, y las listas.



Algunos ejemplos de vectores son:

### ★ Ejemplo 1.1

Crear un vector de 10 elementos y definirlo como un objeto llamado *notas*

```
notas <- c(3.7, 3.4, 2.8, 4.0, 2.5, 3.0, 3.5, 3.7, 3.7, 4)
```

**notas**: es la variable en la que se almacenan los 10 valores, que en R se denomina objeto.

Para hacer la asignación se utilizan los símbolos mayor que y menor que, de la siguiente manera:

<- : son dos símbolos unidos "<" y "-", juntos indican una flecha que se entiende como la asignación de los datos de la derecha en la izquierda.

Este símbolo puede ser reemplazado por "=", eventualmente se encuentra también como "->", es decir, el objeto *notas* puede ser creado mediante la siguiente forma:

```
c(3.7, 3.4, 2.8, 4.0, 2.5, 3.0, 3.5, 3.7, 3.7, 4) -> notas
```

Para leer el resultado puede ser ejecutada la orden

```
print(notas)
```

o simplemente digitando **notas**, sombreándolo y dando la orden con la primera opción de la ventana que aparece al oprimir el botón derecho del ratón (igualmente con el logo ejecutar de la barra de herramientas, o en su defecto, colocando el cursor al final de la línea y oprimiendo las teclas, (Ctrl, + R) obteniendo como respuesta en la ventana de la consola.

```
[1] 3.7 3.4 2.8 4.0 2.5 3.0 3.5 3.7 3.7 4.0
```



## 1.2 Vector lógico

El conjunto de valores proviene de una variable cualitativa, del tipo dicotómico, que toma dos valores Verdadero, Falso, que corresponden a las palabras en inglés True (T), False (F).

### ★ Ejemplo 1.2

Crear un vector de 5 elementos y asignarlo a la variable (objeto) *Y*.

```
Y <- c(T,F,T,T,F)
```

Para hacer la lectura se digita y ejecuta el nombre del objeto

**Y**

```
[1] TRUE FALSE TRUE TRUE FALSE
```



## 1.3 Vector de caracteres

Al igual que en el caso anterior, el vector de caracteres obedece a una variable nominal, pero con una gama de valores no numéricos más amplia.

### ★ Ejemplo 1.3

Crear un vector para el municipio de origen de 4 estudiantes.

```
Z<- c("Manizales", "Chinchiná", "Neira", "Aguadas")
```

**Z**

```
[1] "Manizales" "Chinchiná" "Neira" "Aguadas"
```

NOTA

Respetar las mayúsculas y minúsculas de los nombres, usualmente los comandos e instrucciones van escritas con estas últimas.



## 1.4 Asignación de nombres a los elementos de un vector

Se usa la instrucción **names()**

### ★ Ejemplo 1.4

Sea el vector

i. **w<-c(3,8,9)**

**names(w)<- c("a","b","c")** # se asignan los nombres usando un vector de caracteres, de la misma dimensión de w.

**w**

a b c

3 8 9

ii. **names(notas)<- c("a","b","c","d","e","f","g","h","i","j")** # según el ejemplo 1.1.

**notas**

a	b	c	d	e	f	g	h	i	j
3.7	3.4	2.8	4.0	2.5	3.0	3.5	3.7	3.7	4.0



## 1.5 Creación de patrones

Para simplificar la escritura de vectores existen varias órdenes con las cuales se crean de un modo automático:

a. **from:to** # Desde un Número a otro.

### ★ Ejemplo 1.5

Crear la secuencia de 1 a 5

**1:5**

[1] 1 2 3 4 5 # resultado inmediato en la consola

b. **seq ()** # Cumple el mismo papel del comando anterior, adicionalmente puede llevar un elemento que indique el incremento deseado.

### ★ Ejemplo 1.6

i. **seq(1,6)** # Genera los números entre 1 y 6.

[1] 1 2 3 4 5 6

ii. **seq(1,6,by=0.5)** # Suma al número anterior 0.5 hasta que se llega a 6, empezando en 1.

[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0

iii. **seq(1,6, by= pi)** # Inicia en uno, suma el valor pi hasta llegar a seis#

[1] 1.000000 4.141593

c. **rep()** # Para repetir patrones una cierta cantidad de veces.

### ★ Ejemplo 1.7

i. **rep(1,5)** # Repetir el 1 cinco veces.

[1] 1 1 1 1 1

ii. **rep(c(1,2),5)** # Repetir el patrón (1,2) cinco veces.

[1] 1 2 1 2 1 2 1 2 1 2

iii. `rep(1:4,2)` # Repetir el patrón de 1 a 4, dos veces

```
[1] 1 2 3 4 1 2 3 4
```

iv. `rep(1:3, c(1,3,5))` # Repetir la secuencia 1,2,3 de acuerdo al vector (1,3,5), esto es, 1 vez el 1, 3 veces el 2 y 5 veces el 3.

```
[1] 1 2 2 2 3 3 3 3 3
```

v. Sea `w<-c(3,8,9)`

```
rep(w,c(1,2,3))
```

```
3 8 8 9 9 9
```



## 1.6 Extracción de elementos de un vector

Hay cuatro maneras de seleccionar elementos de un vector, y son:

1. Especificar los índices de los elementos a extraer:

### ★ Ejemplo 1.8

La orden extrae los elementos 1, 3 y 6 del vector x.

```
x <- c(18,11,12,10,7,6,17)
```

```
x[c(1,3,6)] : #El resultados es
```

```
[1] 18 12 6
```

2. Un número negativo precediendo al índice significa exclusión. Con el vector x creado anteriormente:

### ★ Ejemplo 1.9

```
x[-3] # Extrae todos los elementos del vector x menos el tercero.
```

```
[1] 18 11 10 7 6 17
```

### ★ Ejemplo 1.10

```
x[-c(1,2)] # Extrae los elementos del vector x menos el primero y el segundo.
```

```
[1] 12 10 7 6 17
```

3. Especificar una condición lógica.

#### ★ Ejemplo 1.11

En el caso del vector  $x$  creado anteriormente:

- a. Calificar como verdadero (mayor que 10) o falso (menor o igual de 10) cada uno de los elementos del vector  $x$ .

```
x > 10 # se genera un vector lógico
[1] TRUE TRUE TRUE FALSE FALSE FALSE TRUE
```

- b. Extraer todos los elementos que cumplan la condición.

```
x[x > 10]
[1] 18 11 12 17
```

4. En el caso de un vector de variables cualitativas, se utilizan los nombres de las variables para extraer los elementos:

#### ★ Ejemplo 1.12

- a. A partir del vector  $w$  definido como

```
w = c(3,8,9)
wc = c("a", "b", "c")
```

leer o extraer el primer elemento.

```
wc[1] # equivale a wc["a"]
a
3
```

- b. Leer los dos últimos:

```
wc[2:3]
b c
```

#### ★ Ejemplo 1.13

- a. Crear tres variables  $A$ ,  $B$  y  $C$  con los valores 1, (3, 5) y 7 respectivamente.  
b. A continuación definir un vector formado por dichas variables.  
c. Extraer el valor referenciado por la variable  $B$ .

- a. `A <- 1`  
`B <- c(3,5)`  
`C <- 5`
- b. `ABC <- c(A,B,C)`  
`ABC`  
`[1] 1 3 5 5`
- c. `B`  
`[1] 3 5`

#### ★ Ejemplo 1.14

Seleccionar las letras de la a hasta la c y de la A a la D de los vectores `letters` y `LETTERS`, y definir un nuevo vector.

```
letras <- c(letters[1:3], LETTERS[1:4])
letras

[1] "a" "b" "c" "A" "B" "C" "D"
```

## 1.7 Valores faltantes

El símbolo de valor faltante es NA (sigla del inglés Not Available). Cualquier operación aritmética que involucre a un NA da por resultado un NA. Esto se aplica también a los operadores lógicos tales como `<`, `<=`, `>`, `>=`, `=` (Empleado para comprobar si dos objetos son iguales), `!=` (comprueba si dos objetos son distintos).

#### ★ Ejemplo 1.15

Suponiendo el vector

```
x = c(1,2,3,NA,4,5)
x

[1] 1 2 3 NA 4 5
```

- a. `is.na(x)` # Pregunta ¿qué elementos de x son faltantes? Resulta un vector lógico
- ```
[1] FALSE FALSE FALSE TRUE FALSE FALSE
```

b. `x[x>2]` # Pregunta ¿qué valores de x superan a 2? Resulta un vector numérico.

```
[1] 3 NA 4 5
```

c. `x*2` # Multiplica cada elemento de x por 2

```
[1] 2 4 6 NA 8 10
```

d. `x1<-x[!is.na(x)]` # Selecciona los valores de x no faltantes y asigna el resultado al vector x1. De esta manera se eliminan los valores faltantes del vector.

**x1**

```
[1] 1 2 3 4 5
```



## 1.8 Otros elementos a tener en cuenta

a. Verificar el tipo de vector. Se tiene una serie de funciones `is.algo()`, que responde a un valor lógico:

### ★ Ejemplo 1.16

Suponga el vector `x<- c(1:10)`

i. **`is.numeric(x)`** # Es x numérico?

```
[1] TRUE
```

ii. **`is.vector(x)`** # Es x un vector?

```
[1] TRUE
```

iii. **`is.complex(x)`**

```
[1] FALSE
```

iv. **`is.character(x)`** #Es x un carácter?

```
[1] FALSE
```

b. Convertir un tipo de vector a otro tipo. Las funciones que apoyan esta alternativa son `as.algo()`, que fuerza al tipo de datos deseado.

### ★ Ejemplo 1.17

- i. `x<-c(1:10)`
- ii. `x<-as.character(x)`  
`x`

```
[1] "1" "2" "3" "4" "5" "6" "7" "8" "9" "10"
```

c. Identificar la estructura de un objeto

Mediante `str(objeto)` es posible obtener información sobre su estructura.

### ★ Ejemplo 1.18

- i. `x<-1:12`  
`str(x)`

```
int [1:12] 1 2 3 4 5 6 7 8 9 10 ...
```

- ii. `origen<- c("Neira","Marmato","Manizales","Manizales","Marmato",  
"Marquetalia","Neira","Marquetalia","Manizales","Marmato",  
"Chinchina","Salamina","Marmato","Neira","Marquetalia")`  
`str(origen)`

```
chr [1:15] "Neira" "Marmato" "Manizales" "Manizales" "Marmato" ...
```



## 1.9 Resultados estadísticos a partir de un vector

Los resultados de tipo estadístico que se pueden obtener de un vector son dados en términos de indicadores, representaciones tabulares, y representaciones gráficas. A continuación se muestran algunas alternativas.

### 1. Indicadores Estadísticos

Los indicadores resumen el comportamiento de una variable, dentro de estos se conciben las medidas de tendencia central, dispersión, asociación, forma, coeficientes, índices, tasas, etc. Se dará una mirada a los indicadores más usuales de carácter univariado y bivariado.



## ■ Univariado

Resumen de una variable cuantitativa.

### ★ Ejemplo 1.19

Calcular la media, varianza, desviación estándar, coeficiente de variación, cuantiles del vector.

```
notas <- c(3.7, 3.4, 2.8, 4.0, 2.5, 3.0, 3.5, 3.7, 3.7, 4)
```

Habría dos alternativas: aplicar a un vector las definiciones y las cuatro operaciones matemáticas fundamentales, o bien usar directamente los comandos que posee R.

a. Media (promedio) aritmética (o)

- i. **N<-length(notas)** # identificar el tamaño del vector
- ii. **suma<-sum(notas)**
- iii. **media<-suma/N** # definición de la media aritmética
- iv. **mean(notas)** # R ofrece este comando para obtener la media  
[1] 3.43

b. Varianza

- i. **varianzacorregida<-sum((notas-media)^2)/(N-1)** # definición
- ii. **var(notas)** # R ofrece este comando para obtener la varianza corregida.  
[1] 0.2578889

c. Desviación estándar

- i. **desviación<-sqrt(varianzacorregida)**
- ii. **sd(notas)** # R ofrece este comando para obtener la desviación estándar (a partir de la varianza corregida)  
[1] 0.5078276

d. Coeficiente de variación

```
CV<-desviación*100/media # Coeficiente de variación.
```

e. Cuantiles (Medidas de posición)

Para el trabajo con cuantiles, bien sean cuartiles, deciles, percentiles, etc., R ofrece las opciones siguientes:

- i. **quantile(notas,1/4)**
- ii. **quantile(notas,1/2)**

- iii. `quantile(notas,3/4)`
- iv. `quantile(notas,c(0.25,0.5,0.75))`

25% 50% 75%  
3.1 3.6 3.7

- v. `median(notas)` # para obtener solo la mediana#

f. Resumen utilizando algunos indicadores

**summary(notas)** # produce seis valores correspondientes a: mínimo, primer cuartil, mediana, media, tercer cuartil, y el valor máximo del vector.

| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
|------|---------|--------|------|---------|------|
| 2.50 | 3.10    | 3.60   | 3.43 | 3.70    | 4.00 |

## ■ Bivariado

Se trabajan dos variables que pueden ser de igual tipo, o una mezcla. Usualmente se manejan asociadas a un espacio y/o un periodo de tiempo determinado, constituyendo series temporales.

Algunos indicadores se definen a partir de Razones (cociente de dos cantidades), suelen aparecer como medidas de intensidad, con nombres como proporciones, índices, tasas, coeficientes, etc.

Para los siguientes ejemplos abrir el archivo Vias1, del anexo.

```
vias<- read.table("Vias1.txt",header=TRUE)
vias
```

a. Proporciones

### ★ Ejemplo 1.20

```
Proporcion_estudiantes<-"T_Estudiantes"/"Población"
```

La instrucción anterior no corre ya que hace referencia a nombres y no corresponde a objetos.

```
w1<-vias$T_Estudiantes
```

```
w2<-vias$Poblacion
```

```
raz<-w1/w2 # Proporción de estudiantes por año según la población esti-  
mada anualmente.
```

```
raz
```

```
[1] 0.2352356 0.2578832 0.2649165 0.2773447 0.2965087 0.3137868 0.3451846
[8] 0.3329103 0.3262571 0.3122549 0.3197886 0.3241763 0.3241396 0.3151841
[15] 0.3516710 0.3511165 0.3473633 0.3515827 0.3361889 0.3318084 0.3422652
[22] 0.3525314 0.3457089 0.3314257 0.3407183 0.2935365 0.3008823 0.2979183
[29] 0.3001950 0.3024408 0.3046587 0.3068830
```

`round(raz,3)` # igual al resultado anterior, pero con solo 3 decimales.

```
[1] 0.235 0.258 0.265 0.277 0.297 0.314 0.345 0.333 0.326 0.312 0.320 0.324
[13] 0.324 0.315 0.352 0.351 0.347 0.352 0.336 0.332 0.342 0.353 0.346 0.331
[25] 0.341 0.294 0.301 0.298 0.300 0.302 0.305 0.307
```

## b. Porcentajes

### ★ Ejemplo 1.21

```
porc<-100*round(raz,3)
porc
```

```
[1] 23.5 25.8 26.5 27.7 29.7 31.4 34.5 33.3 32.6 31.2 32.0 32.4 32.4 31.5 35.2
[16] 35.1 34.7 35.2 33.6 33.2 34.2 35.3 34.6 33.1 34.1 29.4 30.1 29.8 30.0 30.2
[31] 30.5 30.7
```

## c. Índices

### ★ Ejemplo 1.22

#### i. Índices de base fija

`w3<-vias$Pasaj_mov` # se usa el signo \$ después del nombre del objeto `vías` y antes del nombre de la variable a utilizar, con el fin de acceder a la variable, en este caso: `Pasaj_mov`

```
Base_Fija<-round(100*w3/w3[1],2)
```

```
Base_Fija
```

```
[1] 100.00 105.32 116.56 116.61 140.63 172.81 185.21 210.19 194.88 235.21
[11] 263.12 191.65 264.06 289.72 315.39 341.06 366.72 265.79 286.76 297.74
[21] 366.99 436.23 379.52 374.28 384.32 382.03 351.70 374.57 396.08 417.60
[31] 439.11 460.62
```

#### ii. Índices de base variable

```
t<-2:32
```

```
Base_Variable1<-round(100*w3[t]/w3[t-1],2)
```

```
Base_Variable1
```

```
[1] 105.32 110.67 100.05 120.59 122.88 107.18 113.49 92.71 120.69 111.87
[11] 72.84 137.78 109.72 108.86 108.14 107.53 72.48 107.89 103.83 123.26
[21] 118.87 87.00 98.62 102.68 99.41 92.06 106.50 105.74 105.43 105.15
[31] 104.90
```

```
Variacion1<-round(100*w3[t]/w3[t-1],2)-100
```

```
Variacion1
```

```
[1] 5.32 10.67 0.05 20.59 22.88 7.18 13.49 -7.29 20.69 11.87
[11] -27.16 37.78 9.72 8.86 8.14 7.53 -27.52 7.89 3.83 23.26
[21] 18.87 -13.00 -1.38 2.68 -0.59 -7.94 6.50 5.74 5.43 5.15
[31] 4.90
```

- iii. **Variacion<-c(0,Variacion1)** # corrección de la serie (le faltaba el primer dato)

```
Variacion
```

```
[1] 0.00 5.32 10.67 0.05 20.59 22.88 7.18 13.49 -7.29 20.69
[11] 11.87 -27.16 37.78 9.72 8.86 8.14 7.53 -27.52 7.89 3.83
[21] 23.26 18.87 -13.00 -1.38 2.68 -0.59 -7.94 6.50 5.74 5.43
[31] 5.15 4.90
```

```
Base_Variable<-c(0,Base_Variable1)
```

```
Base_Variable
```

```
[1] 0.00 105.32 110.67 100.05 120.59 122.88 107.18 113.49 92.71 120.69
[11] 111.87 72.84 137.78 109.72 108.86 108.14 107.53 72.48 107.89 103.83
[21] 123.26 118.87 87.00 98.62 102.68 99.41 92.06 106.50 105.74 105.43
[31] 105.15 104.90
```

d. Correlación

```
cor(w1,w2) # Por defecto se usa el método de Pearson.
```

### ★ Ejemplo 1.23

```
cor(vias$Cons_combust,vias$VEH_PUB)
```

## 2. Representaciones tabulares

¿Cómo construir tablas de frecuencias con R? Aquí se da una respuesta que debe ser editada posteriormente a criterio del usuario para ser incluida en su informe. El comando clave es `table()`.

En los siguientes ejemplos, se recomienda ejecutar las instrucciones y observar las diferentes implicaciones de las mismas.

★ Ejemplo 1.24

Con base en el vector

```
notas <- c(3.7, 3.4, 2.8, 4.0, 2.5, 3.0, 3.5, 3.7, 3.7, 4)
```

i. `m<- table(notas)` # Tabla de frecuencias absolutas.  
`m`

|     |     |   |     |     |     |   |
|-----|-----|---|-----|-----|-----|---|
| 2.5 | 2.8 | 3 | 3.4 | 3.5 | 3.7 | 4 |
| 1   | 1   | 1 | 1   | 1   | 3   | 2 |

ii. `prop.table(m)` # Tabla de frecuencias relativas.  
`notas`

|     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|
| 2.5 | 2.8 | 3   | 3.4 | 3.5 | 3.7 | 4   |
| 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.3 | 0.2 |

iii. `100*table(notas)/length(notas)` # Tabla de frecuencias relativas expresadas en porcentajes.  
`100*prop.table(m)` # Igual al caso anterior.  
`notas`

|     |     |    |     |     |     |    |
|-----|-----|----|-----|-----|-----|----|
| 2.5 | 2.8 | 3  | 3.4 | 3.5 | 3.7 | 4  |
| 10  | 10  | 10 | 10  | 10  | 30  | 20 |

iv. `addmargins(100*prop.table(m))` # Incluye marginales

|     |     |    |     |     |     |    |     |
|-----|-----|----|-----|-----|-----|----|-----|
| 2.5 | 2.8 | 3  | 3.4 | 3.5 | 3.7 | 4  | Sum |
| 10  | 10  | 10 | 10  | 10  | 30  | 20 | 100 |

v. `y<- c(T,F,T,T,F)`  
`table(y)`

vi. `addmargins(table(y))` # Equivale a la anterior pero con el vector lógico

|       |      |     |
|-------|------|-----|
| FALSE | TRUE | Sum |
| 2     | 3    | 5   |

vii. `tf <- cut(notas,4)` # Se establecen 4 intervalos.  
`table(tf)`

|             |             |             |          |
|-------------|-------------|-------------|----------|
| (2.5, 2.87] | (2.87,3.25] | (3.25,3.63] | (3.63,4] |
| 2           | 1           | 2           | 5        |

- ix. `tf1<-cut(notas,4,include.lowest = T)` # El primer intervalo es cerrado
- ```
table(tf1)
```
- |            |             |             |          |
|------------|-------------|-------------|----------|
| [2.5,2.87] | (2.87,3.25] | (3.25,3.63] | (3.63,4] |
| 2          | 1           | 2           | 5        |
- x. `tf2<-cut(notas,4,right = TRUE, include.lowest = T)` # Igual al caso anterior
- ```
table(tf2)
```
- |            |             |             |          |
|------------|-------------|-------------|----------|
| [2.5,2.87] | (2.87,3.25] | (3.25,3.63] | (3.63,4] |
| 2          | 1           | 2           | 5        |
- xi. `Acum <- cumsum (table(tf))` # Produce las frecuencias acumuladas
- ```
Acum
```
- |            |             |             |          |
|------------|-------------|-------------|----------|
| (2.5,2.87] | (2.87,3.25] | (3.25,3.63] | (3.63,4] |
| 2          | 3           | 5           | 10       |
- xii. `table(y<-cut(notas,breaks= 0.5*4:8))` # Genera los intervalos según el commando break

### ★ Ejemplo 1.25

Resumir el trabajo realizado anteriormente en una tabla con los títulos correspondientes.

Para esta tarea son opcionales los siguientes dos comandos y un tercero obligatorio.

*cat* : admite salidas seleccionadas por el usuario.  
*print* : imprime los elementos propuestos.  
*cbind* : permite reunir vectores en forma de columnas.

```
cat("\n Índice de Pasajeros Movilizados (%)\n")
print(cbind(vias[1],Base_Fija,Base_Variable,Variacion))
```

Para construir una tabla bivariada se procede dependiendo de los tipos de variables que se vayan a tabular:

- Dos variables continuas

### ★ Ejemplo 1.26

Tomando la información del archivo vías

```
consumo1<-round(vias$Cons_combust/1000,1) #Se redondea las cifras
para mayor facilidad de lectura.
claseconsumo<-cut(consumo1,4, include.lowest = T)
claseperiodo<-cut(vias$periodo,c(1970,1980,1990,2000,2001), include.
lowest = T)
```

```
T1<-table(claseperiodo,claseconsumo) # Es propiamente la tabla doble
addmargins(T1)
```

```
T2<-round(100*prop.table(T1),1)
addmargins(T2)
```

```
T3<-round(100*prop.table(T1,1),1) # Porcentajes por filas.
addmargins(T3,2)
```

```
T4<-round(100*prop.table(T1,2),1) # Porcentajes por columnas.
addmargins(T4,1)
```

b. Dos variables cualitativas

### ★ Ejemplo 1.27

Se tiene un grupo de personas que practican los deportes; b=basket, f=fútbol y v=voleibol; con una determinada antigüedad y según género.

```
deporte<-c("f","b","f","v","b","v","b","f","v","b","v","b")
antigüedad<-c(1,2,1.5,3,2,1,2,5,3,1,2,3)
genero<-c("m","h","m","h","h","m","m","h","m","m","h","h")
```

Se quiere saber quiénes y cuántos practican cada uno de los deportes. La opción table(x,y) permite aproximarnos a la respuesta:

```
table(deporte, genero)
```

siendo ésta la tabla más importante, se procede como en el ejemplo anterior para construir tablas con porcentajes, con respecto al total, por fila, o por columna.

```
h1<-table(deporte,genero)
addmargins(h1)
```

```
h2<-round(100*prop.table(h1),1)
addmargins(h2)
```

```
h3<-round(100*prop.table(h1,2),1)
addmargins(h3,1)
```

```
h4<-round(100*prop.table(h1,1),1)
addmargins(h4,2)
```

c. Dos variables de diferentes tipos

Se sigue un procedimiento mixto entre a. y b.

### ★ Ejemplo 1.28

Se aprovecha este ejemplo, para construir tablas de tres variables, siguiendo con los vectores definidos antes, con deporte, antigüedad, y género.

```
a1<-cut(antigüedad,c(1,3,5),include.lowest=T)
# se agrupa en categorías la variable antigüedad.
```

```
a2<-table(a1,deporte,genero) # Se genera una tabla tridimensional
addmargins(a2)
```

```
a3<-round(100*prop.table(a2),1)
addmargins(a3)
```

## 2. FACTORES

Un factor es un vector que se usa para especificar una clasificación discreta de los componentes de otros vectores de la misma longitud. Más precisamente, corresponden al manejo de variables de tipo cualitativo o vectores numéricos que representen características discretas o continuas.

### ★ Ejemplo 2.1

a. Sea un vector de caracteres que incluye el municipio de origen de 15 estudiantes.

```
origen<- c("Neira", "Marmato", "Manizales", "Manizales", "Marmato",
"Marquetalia", "Neira", "Marquetalia", "Manizales", "Marmato",
"Chinchina", "Salamina", "Marmato", "Neira", "Marquetalia")
```

```
mode(origen)
```

```
[1] "character"
```



```
length(origen) # Longitud del vector.
```

```
[1] 15
```

- b. Se crea una variable de tipo factor, a partir de la existente, dándose un cambio del tipo de vector de carácter a numérico:

```
festudiantes <- as.factor(origen)
festudiantes
```

```
[1] Neira    Marmato  Manizales Manizales Marmato  Marquetalia
[7] Neira    Marquetalia Manizales Marmato  Chinchina Salamina
[13] Marmato  Neira    Marquetalia
Levels: Chinchina Manizales Marmato Marquetalia Neira Salamina
```

```
mode(festudiantes)
```

```
[1] "numeric"
```

Aunque la respuesta del tipo de vector es numérica, R, para ciertos procedimeinetos que impliquen cálculos como la media, varianza, no opera, entonces su uso conviene para realizar conteos de las diferentes alternativas de respuesta observadas, o para tratar variables cuantitativas por niveles de respuesta contenidos en la variable tipo factor. Note la diferencia de los dos tipos de estructuras:

```
summary(festudiantes) # Genera una tabla de frecuencias que muestra los
niveles del factor (las poblaciones de origen), junto con el número de estudian-
tes correspondiente a tales niveles.
```

Chinchiná	Manizales	Marmato	Marquetalia	Neira	Salamina
1	3	4	3	3	1

```
summary(origen)
```

Length	Class	Mode
15	character	character

- c. Aunque en b, el resultado incluye los niveles o alternativas de respuesta diferentes de la variable origen, estos se pueden solicitar con la orden:

```
levels(festudiantes)
```

```
[1] "Chinchina" "Manizales" "Marmato" "Marquetalia" "Neira"
[6] "Salamina"
```

d. Para identificar los atributos del vector `festudiantes` se utiliza:

```
attributes(festudiantes)
```

```
$levels
```

```
[1] "Chinchina" "Manizales" "Marmato" "Marquetalia" "Neira"
```

```
[6] "Salamina"
```

## ★ Ejemplo 2.2

a. Si ahora se dispone de las estaturas de cada uno de los estudiantes del ejemplo anterior:

```
estudiantes.estaturas <- c(1.83, 1.71, 1.79, 1.64, 1.74, 1.81, 1.62,  
1.84, 1.68, 1.81, 1.82, 1.74, 1.84, 1.61, 1.84)
```

b. Se calcula ahora la estatura promedio de los estudiantes de cada municipio. Para hacer esto se utiliza la función:

```
tapply(x, INDEX, FUN = NULL, ...)
```

*INDEX: lista de factores, de la misma longitud.*

*FUN: la función a ser aplicada*

```
tapply(estudiantes.estaturas, festudiantes, mean)
```

Chinchina	Manizales	Marmato	Marquetalia	Neira	Salamina
1.820000	1.703333	1.775000	1.830000	1.686667	1.740000

```
tapply(estudiantes.estaturas, festudiantes, sd)
```

Chinchina	Manizales	Marmato	Marquetalia	Neira	Salamina
NA	0.07767453	0.06027714	0.01732051	0.12423097	NA



## 2.1. Factores ordenados

Son factores cuyos niveles guardan un determinado orden. Para crear un factor ordenado o para transformar un factor en ordenado se usa la función `ordered()`.

### ★ Ejemplo 2.3

a. Sea un vector con el nivel de inglés de 10 estudiantes:

```
nivel.ingles <- c("medio", "medio", "bajo", "medio", "bajo", "medio",  
"alto", "alto", "bajo", "bajo")
```

Enseguida se crea un factor ordenado, dado por el usuario, según el nivel de inglés de los estudiantes:

```
fnivel.ingles <- ordered(nivel.ingles, levels=c("bajo", "medio", "alto"))  
fnivel.ingles
```

```
[1] medio medio bajo medio bajo medio alto alto bajo bajo  
Levels: bajo < medio < alto
```

b. Si ahora se desea saber cuántos estudiantes tienen un nivel de inglés por debajo de "medio":

```
fnivel.ingles < "medio"
```

```
[1] FALSE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE
```

c. Para conocer las frecuencias de los diferentes niveles de inglés:

```
summary(fnivel.ingles)
```

```
bajo medio alto  
4 4 2
```

d. Otra alternativa:

```
fnivel.ingles <- ordered(nivel.ingles, levels=c("bajo", "medio", "alto"))  
fnivel.ingles
```

```
[1] "medio" "medio" "bajo" "medio" "bajo" "medio" "alto" "alto" "bajo"  
[10] "bajo"
```

```
table(fnivel.ingles)
```

```
fnivel.ingles  
bajo medio alto  
4 4 2
```

#### ★ Ejemplo 2.4

i. `x<-factor(c("a","b","a","c","b"))`  
`x`

```
[1] a b a c b
```

```
Levels: a b c
```

```
str(x)
```

```
Factor w/ 3 levels "a","b","c": 1 2 1 3 2
```

ii. `y<-as.vector(x)`  
`y`

```
[1] "a" "b" "a" "c" "b"
```

```
str(y)
```

```
chr [1:5] "a" "b" "a" "c" "b"
```

iii. `z<-as.numeric(x)`  
`str(z)`

```
num [1:5] 1 2 1 3 2
```

#### ★ Ejemplo 2.5

Hay que tener cuidado en la definición de los factores pues se puede perder información.

i. `x<-rep(0:8,5)`  
`x`

```
[1] 0 1 2 3 4 5 6 7 8 0 1 2 3 4 5 6 7 8 0 1 2 3 4 5 6 7 8 0 1  
[39] 2 3 4 5 6 7 8
```

```
length(x)
```

```
[1] 45
```

ii. `table(y<-cut(x,bre=2*0:4))`

```
(0,2] (2,4] (4,6] (6,8]
```

```
10 10 10 10
```

```
str(y)
```

Factor w/ 4 levels "(0,2]","(2,4]",...: NA 1 1 2 2 3 3 4 4 NA ... 2

Se observa que el objeto y es una estructura tipo factor, pero además que hay cinco datos perdidos.

iii. **table(y1<-cut(x,bre=2\*0:4,right=F))**

```
[0,2) [2,4) [4,6) [6,8)
  10    10    10    10
```

Usando el factor y1 ocurre algo similar a lo anterior.

iv. **which(is.na(y))** # ¿cuáles datos están perdidos?

```
[1] 1 10 19 28 37
```

v. **x[is.na(y)]** # ¿a qué valores corresponden los datos perdidos?

```
[1] 0 0 0 0 0
```

vi. **which(is.na(y1))**

```
[1] 9 18 27 36 45
```

vii. **x[is.na(y1)]**

```
[1] 8 8 8 8 8
```

## ★ Ejemplo 2.6

Del Ejemplo 1.26, se tienen algunos objetos que son del tipo factor. Observe:  
`consumo1<-round(vias$Cons_combust/1000,1)` #para mayor facilidad

i. **claseconsumo<-cut(consumo1,4, include.lowest = T)** # Primer factor  
**str(claseconsumo)**

Factor w/ 4 levels "[210,422]","(422,634]",...: 1 1 1 1 1 2 2 2 2 ...

ii. **claseperiodo <- cut(vias\$periodo, c(1970,1980,1990,2000,2001), include.lowest= T)**  
**str(claseperiodo)**

Factor w/ 4 levels "[1970,1980]",...: 1 1 1 1 1 1 1 1 1 ...

iii. `T1<-table(claseperiodo,claseconsumo) # tabla de doble entrada`

`addmargins(T1)`

`T2<-round(100*prop.table(T1),1)`

`addmargins(T2)`

claseperiodo	[210,422]	(422,634]	(634,846]	(846,1.06e+03]	Sum
[1970,1980]	15.6	18.8	0.0	0.0	34.4
(1980,1990]	0.0	15.6	15.6	0.0	31.2
(1990,2000]	0.0	0.0	0.0	31.2	31.2
(2000,2001]	0.0	0.0	3.1	0.0	3.1
Sum	15.6	34.4	18.7	31.2	99.9

### 3. MATRICES

Una MATRIZ en R, es un conjunto de objetos indizados por filas y columnas. Equivale a decir que es un arreglo de vectores. Los datos que contiene una matriz deben ser todos del mismo tipo: todos numéricos, o de tipo carácter o lógico, pero no mezclados.

La sintaxis general de la orden para crear una matriz es la siguiente:

`matrix(data, nrow, ncol, byrow=F)`

donde:

*data*        *# datos que forman la matriz*  
*nrow*       *# número de filas de la matriz*  
*ncol*       *# número de columnas de la matriz*  
*byrow*      *# los datos se colocan por filas o por columnas según se van leyendo.*

#### ★ Ejemplo 3.1

a. `matrix(1:6) #` Crea una matriz con 6 elementos. Al no especificarse nada, se entiende que se desea crear un vector columna y así los coloca por defecto.

b. `matrix(1:6,nrow=2) #` se indica el número de filas

```

      [,1] [,2] [,3]
[1,]  1   3   5
[2,]  2   4   6

```

c. `matrix(1:6,nrow=2,byrow=T)` # `byrow=T` :señala que se construya por filas

```
[,1] [,2] [,3]
[1,]  1  2  3
[2,]  4  5  6
```

Funciones sobre matrices

En la siguiente tabla se tienen algunas de las funciones usadas en R para matrices :

FUNCION	APLICACION
dim	Devuelve las dimensiones de una matriz
dimnames	Devuelve el nombre de las dimensiones de una matriz
colnames	Devuelve el nombre de las columnas de una matriz
rownames	Devuelve el nombre de las filas de una matriz
mode	Devuelve el tipo de datos de los elementos de una matriz
length	Devuelve el número total de elementos de una matriz
is.matrix	Devuelve T si el objeto es una matriz, F si no lo es
[ , ]	Accede a elementos dentro de la matriz
apply	Aplica una función sobre las filas o columnas
cbind	Añade una columna a una matriz dada
rbind	Añade una fila a una matriz dada

★ Ejemplo 3.2

- a. Sea el vector:
- ```
notas<- c(3.7,3.4,2.8,4.0,2.5,3.0,3.5,3.7,3.7,4.0)
```
- b. Se crea una fila
- ```
fila <- rbind(notas)
```

```
Se lee
fila
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
notas 3.7 3.4, 2.8 4.0 2.5 3.0 3.5 3.7 3.7 4.0
```

c. Se crea una columna

```
columna <- cbind(notas)
columna
      notas
[1,]  3.7
[2,]  3.4
[3,]  2.8
[4,]  4.0
[5,]  2.5
[6,]  3.0
[7,]  3.5
[8,]  3.7
[9,]  3.7
[10,] 4.0
```

d. Lista de objetos creados:

```
d.1 objects() # alternativa 1,
      [1] "columna" "fila" "notas"

d.2 ls() # alternativa 2,
      [1] "columna" "fila" "notas"
```

### ★ Ejemplo 3.3

- i. Crea una matriz 3 x 2  
**x = matrix(1:6,nrow=3)**
- ii. Número de elementos de x  
**length(x)**  
[1] 6
- iii. Tipo de datos de la matriz x  
**mode(x)**  
[1] "numeric"
- iv. Dimensiones de la matriz x  
**dim(x)**  
[1] 3 2
- v. Nombre de las dimensiones de la matriz  
**dimnames(x)**  
NULL



- vi. Nombre de las filas de la matriz  
**rownames(x)**  
NULL
- vii. Nombre de las columnas de la matriz  
**colnames(x)**  
NULL
- viii. El objeto x, ¿es una matriz?  
**is.matrix(x)**  
[1] TRUE

### ★ Ejemplo 3.4

Crear un vector de dos palabras

```
y <- c("blanco", "negro")
is.matrix(y) # El objeto y, ¿es una matriz?

[1] FALSE
```

### ★ Ejemplo 3.5

Sea la matriz

```
datos <- matrix(c(20,65,174,22,70,180,19,68,170), nrow=3, byrow=T)
```

- i. Asignar nombres a las columnas  
**colnames(datos) <- c("edad", "peso", "altura")**

datos	edad	peso	altura
juan	20	65	174
josé	22	70	180
julian	19	68	170

- ii. Se asignan nombres a las filas  
**rownames(datos) <- c("juan", "josé", "julian")**  
**datos**

	edad	peso	altura
juan	20	65	174
josé	22	70	180
julian	19	68	170

- iii. Alternativamente usando la función `dimnames`

```
dimnames(datos) <- list(c("juan", "josé", "julian"), c("edad", "peso",  
"altura"))
```

**datos**

	edad	peso	altura
juan	20	65	174
josé	22	70	180
julian	19	68	170

donde `list()` es una estructura de datos denominada lista, e incluye dos vectores con los nombres de las filas y columnas, respectivamente.

- iv. Se agrega una fila

```
jorge<-c(20,63,165)
```

```
datos1<-rbind(datos,jorge)
```

- v. Se agrega una columna

```
hermanos<-c(2,1,0,3)
```

```
datos2<-cbind(datos1,hermanos)
```

- vi. Acceder a los elementos de la matriz usando los nombres:

- f.1 Edades de todas las personas

```
datos[, "edad"] # es equivalente a: datos[, 1]
```

juan	josé	julian
20	22	19

- f.2 Variables del individuo "josé"

```
datos["josé",] # es equivalente a: datos[ 2, ]
```

edad	peso	altura
22	70	180

- f.3 Edad y altura de todas las personas

```
datos[,c("edad", "altura")]
```

	edad	altura
juan	20	174
josé	22	180
julian	19	170

- vii. Para ordenar que muestre los nombres de filas y columnas.

**dimnames(datos)**

```
[[1]]  
[1] "juan" "josé" "julian"  
[[2]]  
[1] "edad" "peso" "altura"
```

- h. Halle la media de las variables edad, peso y altura. Se usa la orden:

`apply(X, MARGIN, FUN, ...)`

X	Matrix o Array (está estructura de datos se verá posteriormente).
MARGIN	un vector que da los subíndices sobre los cuales la función será aplicada: 1 indica filas, 2 indica columnas, c (1,2) indica filas y columnas.
FUN	Función a ser aplicada
...	Argumentos opcionales de FUN

**apply(datos2, 2, mean)**

```
edad  peso  altura  hermanos  
20.25 66.50 172.25  1.50
```

La función `apply`, reemplaza las siguientes cuatro instrucciones:

```
mean(datos2[,1]); mean(datos2[,2]) ; mean(datos2[,3]) ; mean(datos2[,4])
```

### ★ Ejemplo 3.6

Para restar o sumar en filas o columnas una determinada cantidad expresada a través de un indicador, se usa la función `sweep()`, dada en forma general como:

`sweep (x,MARGIN, STATS, FUN="-",...)`

**MARGIN=1:** para filas

**MARGIN=2:** para columnas

**STATS:** medida estadística deseada

**FUN=:** toma los valores "-" (por defecto), "+", "\*", "/", etc.

i. `A <- matrix(1:8, 4,2)`  
A

	[,1]	[,2]
[1,]	1	5
[2,]	2	6
[3,]	3	7
[4,]	4	8

ii. `sweep(A, 1, 5)`

	[,1]	[,2]
[1,]	-4	0
[2,]	-3	1
[3,]	-2	2
[4,]	-1	3

iii. `sweep(A, 1, 5,FUN="+")`

	[,1]	[,2]
[1,]	6	10
[2,]	7	11
[3,]	8	12
[4,]	9	13

iv. `apply(A,2,mean)`

[1] 2.5 6.5

`centrar<- sweep(A, 2, apply(A,2,mean))` # resta la media de cada columna a cada elemento de la columna respectiva de la matriz A.

	[,1]	[,2]
[1,]	-1.5	-1.5
[2,]	-0.5	-0.5
[3,]	0.5	0.5
[4,]	1.5	1.5

v. `apply(A,2,sd)`

[1] 1.290994 1.290994

`reducir<-sweep(centrar, 2, apply(A,2,sd))` # se estandariza la matriz A.

	[,1]	[,2]
[1,]	-2.7909	-2.7909
[2,]	-1.7909	-1.7909
[3,]	-0.7909	-0.7909
[4,]	-0.2090	-0.2090

### ★ Ejemplo 3.7

Además de la opción iv. y v. del ejemplo anterior, para estandarizar una matriz se puede trabajar brevemente con la opción `scale()`, considerando las variables como columnas. Al aplicarla, se obtiene como respuesta además de la matriz estandarizada, los atributos valores medios aplicados para centrar y las desviaciones estándar para la reducción:

```
A <- matrix(1:8, 4, 2)
scale(A)

      [,1]      [,2]
[1,] -1.1618950 -1.1618950
[2,] -0.3872983 -0.3872983
[3,]  0.3872983  0.3872983
[4,]  1.1618950  1.1618950

attr(,"scaled:center")

[1] 2.5 6.5

attr(,"scaled:scale")

[1] 1.290994 1.290994
```

### ★ Ejemplo 3.8

En este ejemplo se presentan otras maneras de trabajar con matrices:

- a. Para construir una matriz de 8 números, en 2 filas y 4 columnas:

```
M<-array(c(2,7,5,9,3,1,6,4), dim=c(2,4))
M

      [,1] [,2] [,3] [,4]
[1,]  2   5   3   6
[2,]  7   9   1   4
```

- b. Para acceder a algunos elementos de M,

```
b1.    M[2,3]

      [1] 1
```

```
b2.    M[2,3:4]

      [1] 1 4
```

```

b3.  M[1,c(1,3,4)]
      [1] 2 3 6

b4.  M[1,]
      [1] 2 5 3 6

b5.  M[,4]
      [1] 6 4

b6.  M[2,c(2,4)]
      [1] 9 4

b7.  M[2,c(2,4)]=c(17,18) ; M
      [,1] [,2] [,3] [,4]
[1,]  2    5    3    6
[2,]  7   17    1   18

```

c. Para conocer el número de filas, columnas, y dimensiones de la matriz

```

c1.  nrow(M)
      [1] 2

c2.  ncol(M)
      [1] 4

c3.  dim(M)
      [1] 2 4

```

### ★ Ejemplo 3.9

a. Operaciones algebraicas con matrices:

Sea la matriz

```
M<-array(c(2,7,5,9,3,1,6,4), dim=c(2,4))
```

i.  $M+3$

```

      [,1] [,2] [,3] [,4]
[1,]  5    8    6    9
[2,] 10   20    4   21

```

ii. **M\*2**

```
      [,1] [,2] [,3] [,4]  
[1,]  4  10   6  12  
[2,] 14  34   2  36
```

b. Traspuesta de una matriz

**Mt<- t(M) ; Mt**

```
      [,1] [,2]  
[1,]  2   7  
[2,]  5  17  
[3,]  3   1  
[4,]  6  18
```

c. Suma de matrices

i. **M1<- M\*2**

**M1**

```
      [,1] [,2] [,3] [,4]  
[1,]  4  10   6  12  
[2,] 14  18   2   8
```

ii. **Suma<- M+M1**

**Suma**

```
      [,1] [,2] [,3] [,4]  
[1,]  6  15   9  18  
[2,] 21  27   3  12
```

d. Producto de matrices (operador %\*% para producto):

i. **Producto<-M1%\*% Mt**

**Producto**

```
      [,1] [,2]  
[1,] 148 172  
[2,] 172 294
```

ii. **Y<-c(2,3,1)**

**Y**

```
[1] 2 3 1
```

```
A<- matrix(c(3,2,5,7,8,2.3,6,9.1,12), 3,3)
A
```

```
      [,1] [,2] [,3]
[1,]  3   7.0  6.0
[2,]  2   8.0  9.1
[3,]  5   2.3 12.0
```

```
H<-A%*%Y
H
```

```
      [,1]
[1,] 33.0
[2,] 37.1
[3,] 28.9
[1]  2  3  1
```

e. Productos cruzados

A través de la opción **crossprod()** se logra obtener vectores, y matrices, y con **drop()** escalares.

i. **z <- crossprod(1:4)** # esto implica un escalar y equivale a  $\text{sum}(1 + 2^2 + 3^2 + 4^2)$

```
      [,1]
[1,] 30
```

ii. **drop(z)** # scalar

```
[1] 30
```

1. **x <- 1:4**

```
names(x)<- letters[1:4]
```

```
x
a b c d
1 2 3 4
```

```
tcrossprod(as.matrix(x))
```

```
      a  b  c  d
a     1  2  3  4
b     2  4  6  8
c     3  6  9 12
d     4  8 12 16
```



f. Matriz diagonal

La función **diag(x)** extrae la diagonal de una matriz o crea una matriz diagonal.

★ Ejemplo 3.10

Retomando el objeto A,

```
A<- matrix(c(3,2,5,7,8,2.3,6,9.1,12), 3,3)
```

i. **diag(A)** # genera un vector con los elementos de la diagonal de A

```
[1] 3 8 12
```

ii. **diag(diag(A))** # produce una matriz diagonal con los elementos de A

```
[,1] [,2] [,3]
```

```
[1,] 3 0 0
```

```
[2,] 0 8 0
```

```
[3,] 0 0 12
```

g. Solución de un sistema de ecuaciones lineales

Se usa la función genérica **solve**, permite solucionar ecuaciones del tipo  $a \%*\% x = b$ , con b una matriz o un vector.

```
solve(a, b, tol, LINPACK = FALSE, ...)
```

a: Puede ser una matriz cuadrada numérica o compleja que contiene los coeficientes del sistema lineal.

b: Puede ser una matriz cuadrada numérica o compleja dada al lado derecho del sistema (si se omite, se asume como la matriz idéntica, y la función **solve** dará como resultado la inversa de a).

tol: Tolerancia para detectar dependencia lineal en las columnas de a.

LINPACK: Elemento lógico. Se da por defecto, para que exista compatibilidad con versiones de R inferiores a 1.7.0, de otra manera se utiliza LAPACK.

...: Argumentos adicionales según algunos métodos que lo requieran.

### ★ Ejemplo 3.11

**Sol<-solve(A,H)**

**Sol**

```
      [,1]
[1,]  2
[2,]  3
[3,]  1
```

h. Inversión matricial

### ★ Ejemplo 3.12

**solve(A)**

```
      [,1]      [,2]      [,3]
[1,] 0.4596779 -0.42985733 0.09613618
[2,] 0.1316515 0.03673994 -0.09368685
[3,] -0.2167657 0.17206540 0.06123324
```

**solve(A)%\*%A** # da como resultado la matriz idéntica

```
      [,1]      [,2]      [,3]
[1,] 1.000000e+00 1.591744e-16 1.665335e-16
[2,] -1.249001e-16 1.000000e+00 -1.665335e-16
[3,] 3.469447e-17 5.082198e-18 1.000000e+00
```

i. Autovalores y Autovectores de matrices SIMÉTRICAS

Los devuelve en una estructura tipo lista, con un primer elemento nombre \$values que tiene el vector de autovalores y un segundo nombre \$vectors con la matriz de autovectores por columnas

### ★ Ejemplo 3.13

**S=A%\*%t(A)** (t(A): transpuesta de A)

**S**

```
      [,1]      [,2]      [,3]
[1,]  94.0    116.60   103.10
[2,]  116.6   150.81   137.60
[3,]  103.1   137.60   174.29
```

**AvalAvec=eigen(S);AvalAvec**

\$values

```
[1] 383.606262 33.412964 2.080774
```

\$vectors

```
      [,1]      [,2]      [,3]  
[1,] -0.4722270 -0.4672215  0.7474662  
[2,] -0.6116804 -0.4368868 -0.6595279  
[3,] -0.6347037  0.7686573  0.0794805
```

**AvalAvec**\$vectors[,2] :Segundo autovector

```
[1] -0.4672215 -0.4368868  0.7686573
```

**Avalsolo=eigen(S,only.values=TRUE)\$values;Avalsolo** :calcula sólo autovalores

```
[1] 383.606262 33.412964 2.080774
```

j. Producto interno

#### ★ Ejemplo 3.14

```
x <- 1:4  
z <- x %*% x  
z  
      [,1]  
[1,] 30
```

## 4. ARRAYS

Un ARRAY en R es lo mismo que una matriz, salvo que puede tener más de dos dimensiones. Se puede decir que es la generalización de una matriz de dos dimensiones al caso multidimensional. Su definición general es de la forma:

**array(datos, dimensiones) # dimensiones: c(filas, columnas,matrices)**

Los comandos para manejar arrays son similares a los que manejan matrices.

#### ★ Ejemplo 4.1

Generar un arreglo con los números del 1 al 12, creando tres matrices de dos filas por tres columnas.

```
array(1:12, c(2,3,3))
```

```
., 1
```

```
      [,1] [,2] [,3]  
[1,]   1   3   5  
[2,]   2   4   6
```

```
., 2
```

```
      [,1] [,2] [,3]  
[1,]   7   9  11  
[2,]   8  10  12
```

```
., 3
```

```
      [,1] [,2] [,3]  
[1,]   1   3   5  
[2,]   2   4   6
```

#### ★ Ejemplo 4.2

- a. Crear un arreglo con la edad media, el peso medio y la estatura media para hombres y mujeres de dos poblaciones: VillaMaria y Neira:

```
z<- array(c(45,46,65,55,170,167,48,49,68,56,169,165),c(2,3,2))  
dimnames(z) = list(c("hombres","mujeres"),c("edad","peso","altura"),  
c("VillaMaria","Neira"))  
z
```

VillaMaria

	edad	peso	altura
hombres	45	65	170
mujeres	46	55	167

Neira

	edad	peso	altura
hombres	48	68	169
mujeres	49	56	165

- b. Para acceder a los nombres de las dimensiones del array:

b1. De todos los elementos

**dimnames(z)**

```
[[1]]
```

```
[1] "hombres" "mujeres"
```

```
[[2]]
```

```
[1] "edad" "peso" "altura"
```

```
[[3]]
```

```
[1] "VillaMaria" "Neira"
```

b2. Datos para la población "VillaMaria "

**z[,,"VillaMaria"]**

	edad	peso	altura
hombres	45	65	170
mujeres	46	55	167

b3. Datos de todos los hombres

**z["hombres",,]**

	VillaMaria	Neira
edad	45	48
peso	65	68
altura	170	169

b4. Edades de las personas

**z[,,"edad",]**

	VillaMaria	Neira
hombres	45	48
mujeres	46	49

c. Aplicar funciones a los elementos del array, utilizando la función `apply`, del mismo modo que para matrices:

c1. Media de las variables edad, peso y altura para hombres y para mujeres, sin distinguir población

**apply(z,2,mean)**

edad	peso	altura
47.00	61.00	167.75

c2. Media de las variables para cada una de las poblaciones

i. `apply(z[, "VillaMaria" ], 2, mean)`

```
edad peso altura
45.5  60.0 168.5
```

ii. `apply(z[, 2], 2, mean)`

```
edad peso altura
48.5  62.0 167.0
```

### ★ Ejemplo 4.3

Para definir un arreglo de 'ceros' y otra con un mismo valor, 1:

i. `Z<-array(0,c(4,3))`

```
Z
      [,1] [,2] [,3]
[1,]  0  0  0
[2,]  0  0  0
[3,]  0  0  0
[4,]  0  0  0
```

ii. `U<-array(1,c(4,3))`

```
U
      [,1] [,2] [,3]
[1,]  1  1  1
[2,]  1  1  1
[3,]  1  1  1
[4,]  1  1  1
```

### ★ Ejemplo 4.4

Con la función `table()` se puede crear un array como sigue:

i. `x<-factor(rep(1:2,50))`

```
table(x)
```

```
x
1  2
50 50
```

```

ii. y<-factor(rep(1:4,25))
    table(y)

    y
    1  2  3  4
    25 25 25 25

iii. z<-factor(c(rep(1:3,33),1))
     table(z)

     z
     1  2  3
    34 33 33

iv.  w3<-table(x,y,z)
     w3

     , , z = 1
       y
     x  1 2 3 4
     1  9 0 8 0
       2  0 8 0 9

     , , z = 2
       y
     x  1 2 3 4
     1  8 0 8 0
     2  0 9 0 8, , z = 3
       y
     x  1 2 3 4
     1  8 0 9 0
     2  0 8 0 8

v.   is.array(w3)
     [1] TRUE

```

## 5. LISTAS

Las listas sirven para vincular objetos donde cada uno puede tener una estructura distinta. Esto no ocurre en algunos casos, como en los arrays, donde los elementos deben tener valores del mismo tipo (números, caracteres, lógicos), y de la misma longitud.

Una lista tiene una serie de componentes, a los que se les debe asignar un nombre, puede considerarse un objeto consistente en una colección ordenada de objetos conocidos como componentes:

```
Objeto<- list(nombre1=objeto1, nombre2=objeto2,.....)
```

#### ★ Ejemplo 5.1

```
familia <- list(padre="jose",madre="maria",numero.hijos=3,nombre.hijos =c("jesus", "chucho", "eva"),edades.hijos=c(7,5,3),ciudad="Neira")
familia
```

```
familia$padre
```

```
[1] "jose"
```

```
familia$madre
```

```
[1] "maria"
```

```
familia$numero.hijos
```

```
[1] 3
```

```
familia$nombre.hijos
```

```
[1] "jesus" "chucho" "eva"
```

```
familia$edades.hijos
```

```
[1] 7 5 3
```

```
familia$ciudad
```

```
[1] "Neira"
```

Para acceder a componentes concretos, se usa el nombre de la lista seguido del número que tiene el componente en la lista entre corchetes:

#### ★ Ejemplo 5.2

- i. Si se desea ver los nombres de los hijos

```
familia[4]
```

```
nombre.hijos
```

```
[1] "jesus" "chucho" "eva"
```

equivale lo anterior a

```
familia[[4]]
```



[1] "jesus" "chucho" "eva"

Al querer el nombre del segundo hijo

**familia[[4]][2]**

[1] "chucho"

ii. **familia[1]**

\$padre

[1] "jose"

### ★ Ejemplo 5.3

Para ver los nombres de los objetos dentro de la lista:

i. **names(familia)**

[1] "padre" "madre" "numero.hijos" "nombre.hijos" "edades.hijos" "ciudad"

ii. **familia\$padre**

[1] "jose"

Equivalente a:

**familia[[1]]**

[1] "jose"

## 5.1 Aplicación de funciones

Aplicar una función a todas las componentes de una lista:

**lapply(x, fun, ...)**

### ★ Ejemplo 5.4

**familia\$padre**

[1] jose

**familia\$madre**

[1] maria

```
familia$numero.hijos
```

```
[1] 3
```

```
familia$nombre.hijos
```

```
[1] jesus, chucho, eva
```

```
familia$edades.hijos
```

```
[1] 7 5 3
```

```
familia$ciudad
```

```
[1] Neira
```

Desde luego que la respuesta de la aplicación `mean` a las variables cualitativas es NA no aplicable ( Not available).

Para aplicar una función a todas las componentes de una lista y simplificar el resultado en una estructura de vector, se tiene la función

```
sapply(x, fun, ..., simplify = TRUE)
```

#### ★ Ejemplo 5.5

```
sapply(familia, mean)
```

padre	madre	numero.hijos	nombre.hijos	edades.hijos	ciudad
NA	NA	3	NA	5	NA

Por la misma razón al caso anterior ocurren los NA.

## 6. DATA.FRAMES

Los `data.frames` son una estructura de datos que generaliza a las matrices, en el sentido en que las columnas (variables a menudo) pueden ser de diferente tipo entre sí (no todas numéricas, o no todas tipo nominales). Sin embargo, todos los elementos de una misma columna deben ser del mismo tipo. Al igual que las filas y columnas de una matriz, todos los elementos de un `data.frame` deben ser de la misma longitud.

Esta estructura de `data.frame` es el objeto que utiliza R para contener un banco de datos típico, rectangular, de individuos×variables.

También se puede ver como una lista con las siguientes restricciones:

- Las componentes deben ser vectores de cualquier tipo, factores, matrices numéricas, u otros data.frames.
- Puesto que un vector representa una variable del banco de datos y las columnas de una matriz representarán varias variables de ese mismo banco, la longitud de los vectores debe ser la misma y coincidir con el número de filas de las matrices.

Para crear un banco de datos data.frame se requiere la función:

**data.frame(..., row.names = NULL, check.rows = FALSE,...)**

en donde se deben destacar:

...: van las variables como en una lista cuyos componentes son nombre1=objeto1, nombre2=objeto2,..., etc.

check.rows: elemento lógico, si es **TRUE**, entonces se verifica la consistencia de la longitud y nombres.

Los datos de un data.frame pueden ser accedidos como elementos de una matriz o de una lista. De este modo, pueden usarse funciones tales como dimnames, dim, nrow, etc., sobre un data frame como si se tratara de una matriz.

### ★ Ejemplo 6.1

- datos <- matrix(c(20,65,174,22,70,180,19,68,170),nrow=3,byrow=T)**  
**dimnames(datos)<-list(c("juan","josé","julian"),**  
**c("edad","peso","altura"))**
- Se añade una columna a la matriz datos para que contenga el lugar de origen de cada persona:  
**origen = c("Manizales","Chinchina","Salamina")**  
**datos2 = cbind(datos,origen)**  
**datos2**  

	edad	peso	altura	origen
juan	"20"	"65"	"174"	"Manizales"
josé	"22"	"70"	"180"	"Chinchina"
julian	"19"	"68"	"170"	"Salamina"
- Como todas las variables han sido convertidas a tipo carácter, lo que no conviene, porque en algún tipo de cálculo se obtiene un error:

**mean(datos[, "edad"]) # Calcular la media de la variable edad**  
**[1] 20.33333**

```
mean(datos2[, "edad"]) # Hacer lo mismo, pero con datos2
```

Aviso en `mean.default(datos2[, "edad"])` :

```
argument is not numeric or logical: returning NA
```

```
[1] NA
```

- iv. Se ve qué tipo de datos hay en ambas matrices:

```
mode(datos)
```

```
[1] "numeric"
```

```
mode(datos2) # Los datos han sido convertidos a tipo carácter
```

```
[1] "character"
```

- v. Para poder añadir la columna de tipo carácter sin causar problemas se hace:

```
datos3 = data.frame(datos, origen)
```

```
datos3
```

	edad	peso	altura	origen
juan	20	65	174	Manizales
josé	22	70	180	Chinchina
julian	19	68	170	Salamina

```
mean(datos3[, "edad"]) # Ahora el cálculo no da problemas
```

```
[1] 20.33333
```

- vi. Sin embargo, a la hora de utilizar ciertas funciones hay que tener presente que no todas las variables son del mismo tipo:

```
apply(datos3, 2, mean)
```

Aviso en `mean.default(newX[, i], ...)` :

```
argument is not numeric or logical: returning NA
```

Aviso en `mean.default(newX[, i], ...)` :

```
argument is not numeric or logical: returning NA
```

Aviso en `mean.default(newX[, i], ...)` :

```
argument is not numeric or logical: returning NA
```

Aviso en `mean.default(newX[, i], ...)` :

```
argument is not numeric or logical: returning NA
```

```
edad peso altura origen
```

```
NA NA NA NA
```

No funciona porque se ha intentado hallar la media aritmética de cada columna de `datos3`. El error lo provoca la variable "provincia", que no es numérica.

```
apply(datos3[,1:3],2,mean)
```

```
      edad      peso      altura  
20.33333 67.66667 174.66667
```

En este caso no ha habido error, porque la función `apply` se ha aplicado sobre las variables numéricas.

- vii. Para acceder a los datos se procede indistintamente como si se tratase de una matriz o de una lista:

```
datos2[,2] # Acceso en modo matriz  
65 70 68
```

```
datos2[, "edad"] # Acceso en modo matriz  
20 22 19
```

- viii. Lo mismo para la variable "origen":

```
datos2[,4]  
[1] Manizales Chinchina Salamina  
Levels: Manizales Chinchina Salamina
```

```
datos2[, "origen"]  
[1] Manizales Chinchina Salamina  
Levels: Manizales Chinchina Salamina
```

### ★ Ejemplo 6.2

Sea un `data.frame` con 2 columnas, a partir de un vector numérico `v`, que será una columna de nombre `valor` y otro vector que será otra columna de nombre `calificación`, formado con datos cualitativos.

- i. 

```
v<-c(2.1, 1.4, 6, 3.5, 8)  
DatosEstruc<-data.frame(valor=v, calificación=c("alto", "bajo", "medio", "alto", "medio"))  
DatosEstruc
```

```
      valor calificación  
1      2.1      alto  
2      1.4      bajo  
3      6.0      medio  
4      3.5      alto  
5      8.0      medio
```

- ii. Acceder separadamente a cada columna.

Se aplica el signo \$ entre el nombre del data.frame y el de la columna,

**DatosEstruc\$valor**

[1] 2.1 1.4 6.0 3.5 8.0

**DatosEstruc\$calificación**

[1] alto bajo medio alto medio

Levels: alto bajo medio

- iii. Conocer la dimensión de un objeto, en este caso el data.frame DatosEstruc, se puede emplear la función dim:

**dim(DatosEstruc)**

[1] 5 2 # Es decir, tiene 2 columnas, y cada una 5 filas.

- iv. La función de R summary() aplicada al data frame dá el resumen estadístico de sus variables (columnas)

**summary(DatosEstruc)**

	valor	calificación
Min.	:1.4	alto :2
1st Qu.	:2.1	bajo :1
Median	:3.5	medio:2
Mean	:4.2	
3rd Qu.	:6.0	
Max.	:8.0	

- v. Ir a un valor específico

**DatosEstruc[1,2]**

[1] alto

Levels: alto bajo medio

**DatosEstruc[1,]**

	valor	calificación
1	2.1	alto

```
DatosEstruc[1:3,]
```

	Valor	Calificación
1	2.1	alto
2	1.4	bajo
3	6.0	medio

```
DatosEstruc[1:3,2]
```

```
[1] alto bajo medio
```

```
Levels: alto bajo medio
```

vi. Generar ordenamientos

```
nuevo_orden<-c(2,1,4,3,5)
```

```
nuevo_orden
```

```
[1] 2 1 4 3 5
```

```
DatosEstruc_ordenado=DatosEstruc[nuevo_orden,]
```

```
DatosEstruc_ordenado
```

	Valor	Calificación
2	1.4	bajo
1	2.1	alto
4	3.5	alto
3	6.0	medio
5	8.0	medio

### ★ Ejemplo 6.3

Se tiene una lista numerada de personas que practican diferentes deportes; b=basket, f=fútbol y v=voley. Se quiere saber quiénes y cuántos practican cada uno de los deportes.

```
deporte <- c("f","b","f","v","b","v","b","f","v","b","v","b")
```

```
lista<-data.frame(Id=1:12,deporte)
```

```
basket <-lista[,2]=="b"
```

```
lista[basket,] # extraer los que juegan basket
```

	Id	deporte
2	2	b
5	5	b
7	7	b
10	10	b
12	12	b

```
summary(lista[, "deporte"])
```

```
b f v  
5 3 4
```

```
nrow(lista[basket,])
```

```
[1] 5
```

#### ★ Ejemplo 6.4

Leer el archivo Vías1 e identificar la estructura del mismo.

```
vias<- read.table("Vias1.txt",header=T)
```

```
str(vias)
```

```
'data.frame'      : 32 obs. of 7 variables:  
$ orden           : int  1 2 3 4 5 6 7 8 9 10 ...  
$ periodo         : int  1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 ...  
$ Cons_combust    : int   210876 253965 297054 340143 383232 426321 469410  
                  497681 535781 573880 ...  
$ Poblacion       : int  232214 233819 235595 237304 239025 240759 242505  
                  244264 246036 247820 ...  
$ VEH_PUB        : int  1411 1536 1661 1786 1911 2036 2161 2286 2411 2590...  
$ Pasaj_mov       : int  25944 27325 30240 30254 36484 44833 48050 54532  
                  50559 61022...  
$ T_Estudiantes   : int  54625 60298 62413 65815 70873 75547 83709 81318  
                  80271 77383...
```

#### ★ Ejemplo 6.5

Suponga la siguiente estructura:

```
banco<-data.frame(cbind(x=1, y=1:10), ch=sample(LETTERS[1:3], 10,  
repl=TRUE))
```

```
banco
```

	x	y	ch
1	1	1	C
2	1	2	A
3	1	3	C
4	1	4	C
5	1	5	C
6	1	6	C
7	1	7	A
8	1	8	B
9	1	9	B
10	1	10	A



La función **sample()** utilizada en este ejemplo obtiene una muestra (con o sin reemplazamiento).

```
str(banco)
```

```
'data.frame': 10 obs. of 3 variables:
```

```
$ x : num 1 1 1 1 1 1 1 1 1 1
```

```
$ y : num 1 2 3 4 5 6 7 8 9 10
```

```
$ ch: Factor w/ 3 levels "A","B","C": 3 1 3 3 3 3 1 2 2 1
```

### ★ Ejemplo 6.6

Para seleccionar determinadas filas y columnas de un data.frame, incluimos en la opción `select=`, dentro de la orden `subset()`. Esto permite crear otro data.frame con las características deseadas :

`subset(data.frame, condición lógica, select=vector con nombres de variables)`

```
banco3<-subset(banco,y>5,select=c(x,ch))
```

```
banco3
```

```
   x  ch
6  1   C
7  1   B
8  1   B
9  1   B
10 1   A
```

### ★ Ejemplo 6.7

Es posible construir una hoja de datos para ingresar información interactivamente.

```
misdatos <- data.frame(edad=numeric(0), genero=character  
(0), peso=numeric(0))
```

```
misdatos <- edit(misdatos)
```

Al correr la última instrucción se activa el editor de datos para proceder a la digitación de la información.

RGui

Archivo Ventanas Editor Ayuda

R Console

```
> banco.2
  x ch
6 1 C
7 1 B
8 1 B
9 1 B
10 1 A
> banco3<-subs
> banco3
  x ch
6 1 C
7 1 A
8 1 B
9 1 B
10 1 A
> data(bees)
Mensajes de av
In data(bees)
> matplot((-4:
> mydata <- da
> mydata <- e
> mydata <- e
> misdatos <-
> misdatos <-
```

R Editor de datos

	edad	genero	peso	var4	var5	var6	var7	var8
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								

l=TRUE))

eric(0))

Inicio Windows Live ... 2 Internet Ex... 2011 8 RGui Cursillo R 1. El... Manual R para ... ES



# 3. Gráficos en R



R ofrece una inmensa variedad de gráficos y sería imposible detallar las distintas posibilidades, ya que existen diversas funciones gráficas cada una de las cuales presenta gran cantidad de opciones permitiendo esto una enorme flexibilidad para la elaboración de las representaciones.

Las órdenes gráficas se dividen en tres grupos básicos:

- Alto nivel. Son funciones que crean un nuevo gráfico, con ejes, etiquetas, títulos, etc.
- Bajo nivel. Son funciones que añaden información a un gráfico existente, tales como puntos adicionales, líneas y etiquetas.
- Interactivas. Son funciones que permiten interactuar con un gráfico, aumentando o disminuyendo información, utilizando un dispositivo apuntador, como un ratón.

El programa cuenta con dos demos, en los cuales se puede observar la versatilidad y la calidad en los gráficos. Para correr los demos, tome cada una de las siguientes instrucciones y llévelas al editor de R, enseguida corralas y luego oprima return para observar las diferentes representaciones.

```
demo(graphics)
demo(persp)
demo(lm.glm)
```

Una vez hecho lo anterior, se despliega una ventana a la cual se le debe dar clic para que vaya mostrando los gráficos. Es importante resaltar la calidad de los colores y la variedad de gráficos que hay, sobre todo la posibilidad de generar algunos especializados que requieren paquetes particulares como sucede con los gráficos topográficos.

Otra manera de conocer la forma como funciona una determinada función gráfica es emplear la opción: `example` (Nombre del gráfico o función a conocer). Al hacerlo se activa la ventana de gráficos y sobre ella se da clic con el mouse para ir cambiando de gráfico, en la ventana de la consola aparecen las instrucciones que los generan. En caso de estar interesado en una función los resultados aparecen en la consola.

Así, si se desea averiguar sobre el gráfico de barras o una función en especial, las opciones a emplear son:

**example(barplot)**

Si se desea dibujar el histograma de unos datos y no se sabe cómo hacerlo, se puede pedir ayuda con la instrucción:

**help.search("histogram")**

El resultado genera una serie de funciones que contienen la palabra histogram y que pertenecen a distintos paquetes. Algunas funciones harán referencia al gráfico, mientras que otras, tal vez, sean funciones estadísticas o de otro tipo.

R posee diferentes funciones para crear y modificar gráficos, las cuales pueden ser aplicadas en las distintas representaciones básicas que se desean. Algunos de los gráficos con los que cuenta R son: plot, hist, barplot, boxplot, pie, entre otros.



## 1. plot():

### GRÁFICOS DE DISPERSIÓN (Nube de puntos)

La instrucción general viene dada por:

plot(x, type, col, lwd, main, xlab, ylab, abline, lty, ...) y los argumentos son:

plot:	Instrucción para realizar gráfico
x:	Nombre del objeto
type	Tipo de gráfico que dependerá de la letra que se coloque, así: "l" <i>Si se desea un gráfico de línea</i> "s" <i>Para hacer una representación en escalera</i> "p" <i>Para puntos</i> "b" <i>Para puntos y líneas</i> "c" <i>Líneas sin puntos</i> "h" <i>Solo puntos que son unidos al eje de las abscisas con una línea</i>
col	Establece el color al gráfico
lwd	Esta instrucción se utiliza para darle grosor a la línea que genera la representación gráfica (solamente se emplea si el gráfico que se va a realizar posee líneas)
main	Instrucción que se utiliza para colocarle el título principal al gráfico
xlab	Titula el eje X, el nombre debe estar en comillas
ylab	Titula el eje Y, el nombre debe estar en comillas

abline Esta instrucción se coloca con el objetivo de introducir a la gráfica una línea ya sea horizontal (h) o vertical (v), la ubicación se pondrá después de la instrucción (h o v)

lty Este comando es un carácter que controla el tipo de las líneas

1	Sólida
2	Quebrada
3	Punteada
4	Punto-línea
5	Línea larga-corta
6	Dos líneas cortas

xlim Especifica el límite inferior y superior del eje x

ylim Especifica el límite inferior y superior del eje y

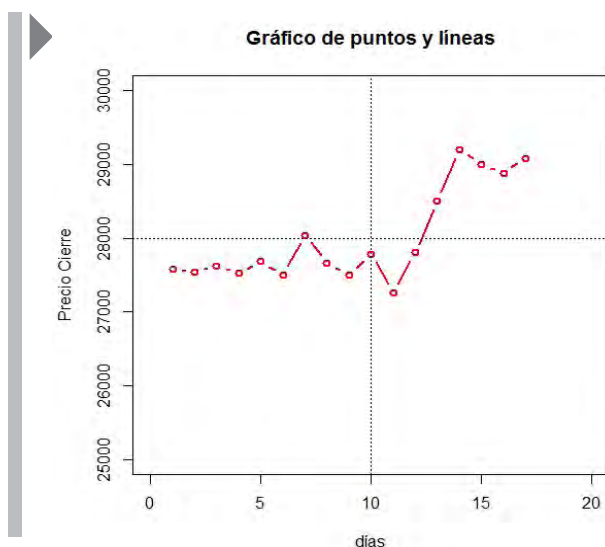
### ★ Ejemplo 1.1

Se lee la información del precio de cierre de las acciones de Bancolombia

```
Bancolombia <-read.delim("clipboard")
Bancolombia
```

Se realiza el gráfico

```
plot (Bancolombia, type="b", col = "red", lwd=2,xlab = "días", ylab =
"Precio Cierre", main="Gráfico de puntos y líneas", xlim=c(0,20),
ylim=c(25000,30000))
abline(h= 28000,lty=3)
abline(v= 10, lty=3)
```



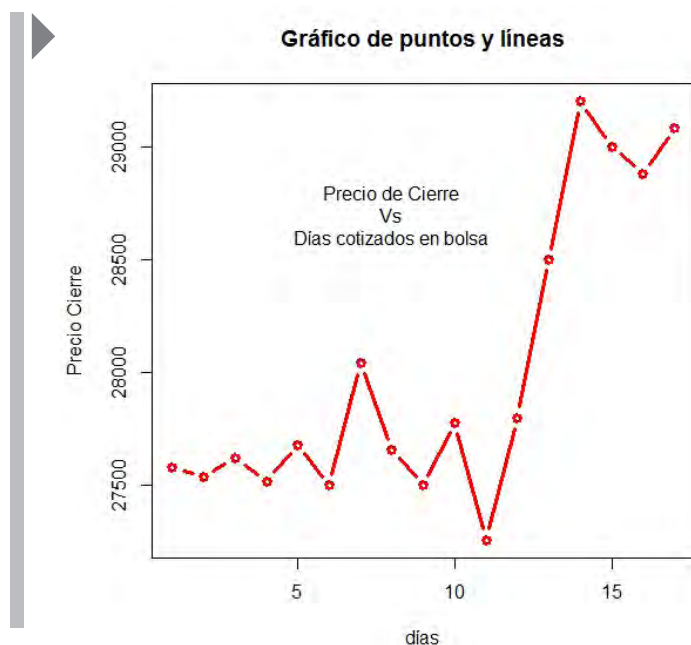
Modificaciones para plot

Para colocarle texto al gráfico, se utiliza el siguiente comando: `text(x, y, "nombre deseado")`

Donde:

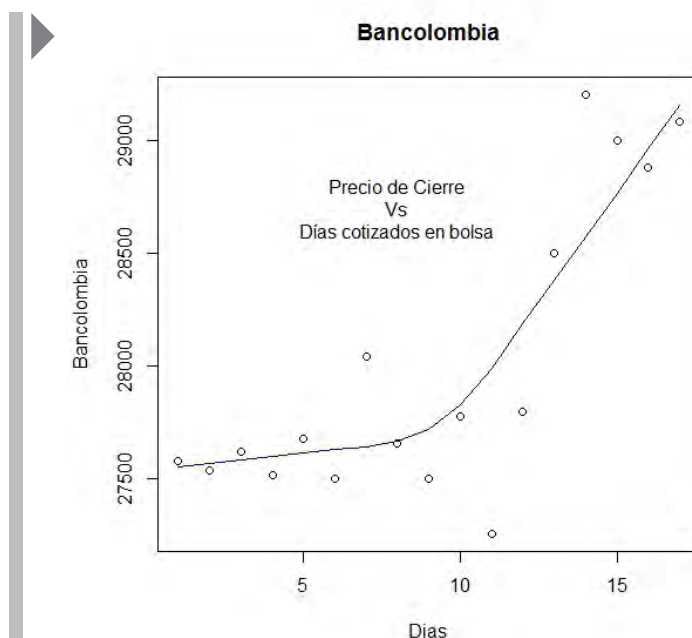
- x Indica la posición del texto en el eje x
- y Indica la posición del texto en el eje y
- \n Establece diferentes niveles al colocar el texto

```
plot(Bancolombia, type="b", col = "red", lwd=3, xlab = "días",  
ylab = "Precio Cierre", main="Gráfico de puntos y líneas")  
text(8, 28700, "Precio de Cierre\nVs\nDías cotizados en bolsa")
```



Para ubicar una línea que se ajuste al conjunto de información

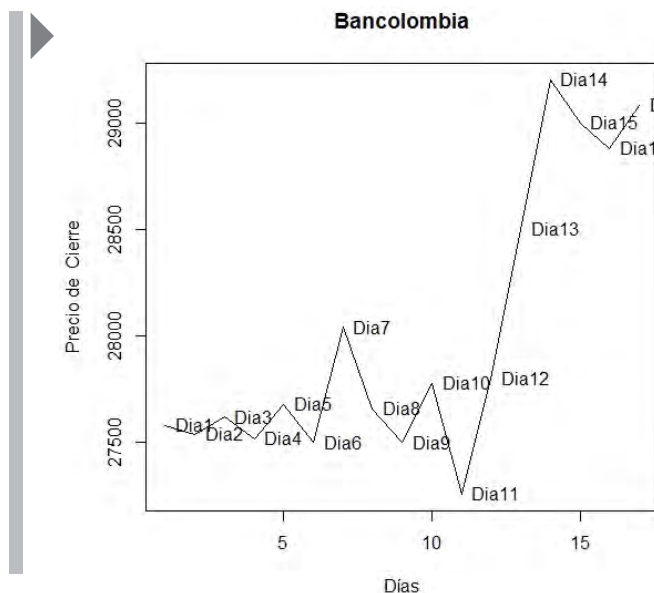
```
plot(Bancolombia, main="Bancolombia", type="p")  
text(8, 28700, "Precio de Cierre\nVs\nDías cotizados en bolsa")  
lines (lowess(Bancolombia), col="blue", lty = 1)
```



Para asignar nombres a cada uno de los puntos que generan el gráfico

```
plot(Bancolombia)
h4<-Bancolombia[,"Dias"]
h4
h5<-Bancolombia[,"Bancolombia"]
h5
nomes=c("Dia1","Dia2","Dia3","Dia4","Dia5","Dia6","Dia7","Dia
8","Dia9","Dia10","Dia11","Dia12","Dia13","Dia14","Dia15","Dia1
6","Dia17")
xy.dat=data.frame(h4,h5,row.names=nomes)
plot(xy.dat, type = "l", xlab = "Días", ylab = "Precio de Cierre",
main = "Bancolombia")
text( xy.dat, labels = nomes, pos = 4)
```





Otros comandos son:

**font:** Este comando permite cambiar el estilo del texto (1:normal, 2:cursiva, 3:negrilla, 4:negrilla cursiva)

*font.main=* Cambia el estilo del texto del título principal

*font.sub=* Cambia el estilo de texto del subtítulo

*font.lab=* Cambia el estilo de texto de los títulos de los ejes

*font.axis=* Cambia el estilo de los números de los ejes

**adj:** Es la instrucción para justificar el texto (0: justificado a la izquierda, 0.5 centrado, 1 justificado a la derecha)

**lwd:** Da grosor a la línea VALORES

**pch:** Gráfico de un carácter o símbolo a utilizar. Para esto se asigna un número, el cual hace referencia a un dibujo, como se muestra en la siguiente representación:



- bg:** Se utiliza para especificar el color del fondo de la figura, pero en otros casos para dar color al fondo de la imagen
- bty:** Controla el tipo de caja que se dibuja alrededor del gráfico: "ö", "l", "7", "ü", "j", "n"; la caja se parece a su respectivo carácter
- lty:** Indica un carácter que controla el tipo de las líneas (1:sólida, 2:quebrada, 3:punteada, 4:punto-línea, 5:línea larga-corta, 6: dos líneas cortas)
- tcl:** Se encarga de alargar las líneas que se encuentran al lado de cada dato
- las:** Un entero que controla la orientación de los caracteres de los ejes (0:paralelo a los ejes, 1: horizontal, 2: perpendicular a los ejes, 3: vertical)
- cex:** Este comando controla el tamaño del texto y los símbolos
- cex.axis:** La misma instrucción anterior se puede usar para cambiar el tamaño de los números de los ejes
- cex.lab:** Esta instrucción controla el tamaño de los títulos de los ejes
- cex.main:** Este comando se utiliza para cambiar el tamaño del título principal
- cex.sub:** Se usa para cambiar el tamaño del subtítulo
- col:** Controla el color de los títulos y de los símbolos
- col.axis:** Controla el color de los números de los ejes
- col.lab:** Controla el color de los títulos de los ejes
- col.main:** Controla el color del título principal
- col.sub:** Controla el color del subtítulo
- labels:** Representa rótulos que pueden ser consignados por medio de vectores de caracteres.
- día <- c("..","..."...)=** se debe crear un vector, el cual será usado para dar nombre a cada uno de los puntos.
- color <- c("..","..."...)=** se debe crear un vector, el cual será usado para dar color a cada uno de los puntos si se desea diferenciarlos.

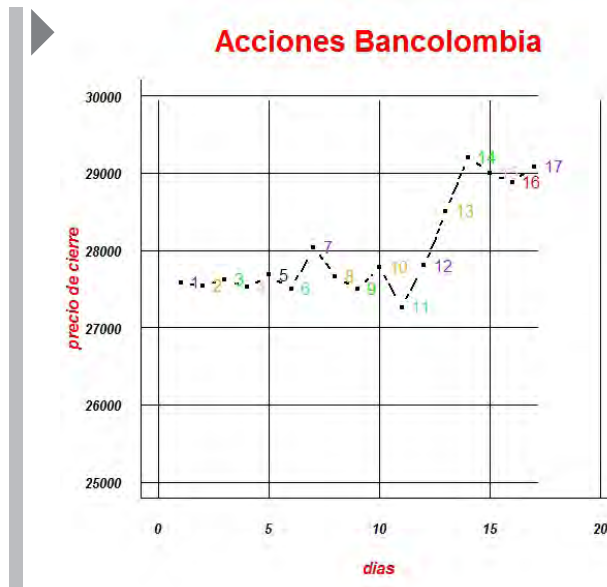
Tomando nuevamente el valor de cierre para las acciones del Banco de Colombia y aplicando las nuevas instrucciones se tiene:

```
plot(Bancolombia, type= "b", xlab="dias", ylab="precio de cierre", font.lab=4, xlim=c(0,20), ylim=c(25000,30000), pch=19, col="black", bg="black", main= "Acciones Bancolombia", font.main=2, adj=0.5,font.sub=3, bty="t", tcl=20, lty=6, las=1, cex=0.5,cex.axis=0.8,cex.lab=1, lwd=2, cex.main =1.8, cex.sub="1.5", col.axis="black", font.axis=4, col.lab= "red", col.main="red", col.sub="red")
```

```
día <- c("1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17")
```

```
color <- c("purple","orange","green","pink","black","turquoise","purple","orange","green","orange","turquoise","purple","orange","green","pink","red")
```

```
text(Bancolombia, labels=día, pos=4,col=color)
```



## ★ Ejemplo 1.2

Observar diferentes gráficas en una sola imagen.

Para hacer la ilustración se carga la información referente al rendimiento de las acciones de Bancolombia, Tablemac y Ecopetrol. Con ellas se realizará la representación del rendimiento de las acciones en el mismo gráfico.

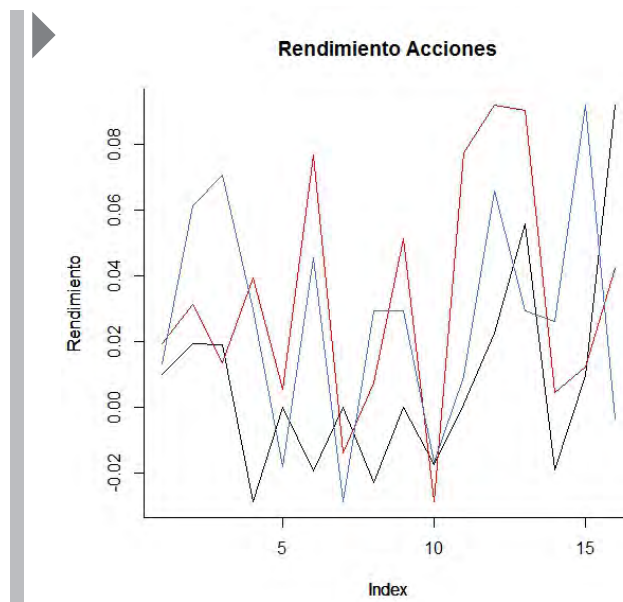
```
Rentablemac <- c(0.0099,0.0194,0.0190,-0.0287,0.0000,-0.0196,0,-0.0230,0,-0.0174, 0.0010,0.0224,0.0558,-0.0192,0.0097,0.0918)
```

```
Renbancolom <- c(-0.0015,0.0029,-0.0036,0.0058,-0.0065,0.0194,-
0.0136,-.0058,0.0101, -0.0189,0.0196,0.0249,0.0243,-0.0069,-
0.0041,0.0069)
```

```
Renecopetrol <- c(-0.0076,0.0113,0.0149,-0.0012,-0.0199,0.0050,-
0.0241,-0.0013,-.0013, -0.0195, -0.0092,0.0131,-0.0013,-
0.0026,0.0233,-0.0142)
```

La instrucción que permite generar el gráfico es:

```
plot (Rentablemac,type="l",col="black",bty="l",main="Rendimien
to Acciones", ylab = "Rendimiento")
par(new=TRUE)
plot (Renbancolom,type="l",col="red",axes=FALSE,ylab = "Rendi
miento")
par(new=TRUE)
plot (Renecopetrol,col="royalblue",type="l",axes=FALSE,ylab =
"Rendimiento")
par(new=TRUE)
```





## 2. hist(): HISTOGRAMA

El formato general para la realización del gráfico viene dada por la siguiente expresión:

```
hist(x,breaks=... , freq=T/F, col= ....., border=T/F, main="...", axes=T/F,  
labels=T/F)
```

Donde:

hist	Efectúa el histograma
x	Nombre del objeto que para el cual se construye el histograma
nclass	Número aproximado de clases
ylim	Establece límite a la escala del eje y
freq	Argumento lógico; si se especifica como TRUE (T), se representan las frecuencias absolutas, para las relativas se coloca FALSE (F)
Col	Controla el color de las barras del gráfico
border	Demarca la base del histograma, si es TRUE (T) marca la línea horizontal sobre la cual descansa la figura. Si es FALSE (F) no muestra la línea
main	Coloca el título a la representación
Sub	Asigna subtítulo al gráfico y lo sitúa abajo del eje X
ylab	Titula el eje y
xlab	Titula el eje x
labels	Señala el valor que presenta cada clase, localizándolo en la parte superior del rectángulo. Si se coloca TRUE (T) se quiere que aparezcan los valores, si se utiliza FALSE (F) no parecen estos valores

### ★ Ejemplo 2.1

Tomando la información de créditos bancarios e importando a R la columna correspondiente a X4, se procede a realizar el gráfico con el siguiente proceso:

Se lee la información de la base Excel y se importa a R

```
X4 <-read.delim("clipboard")  
X4
```

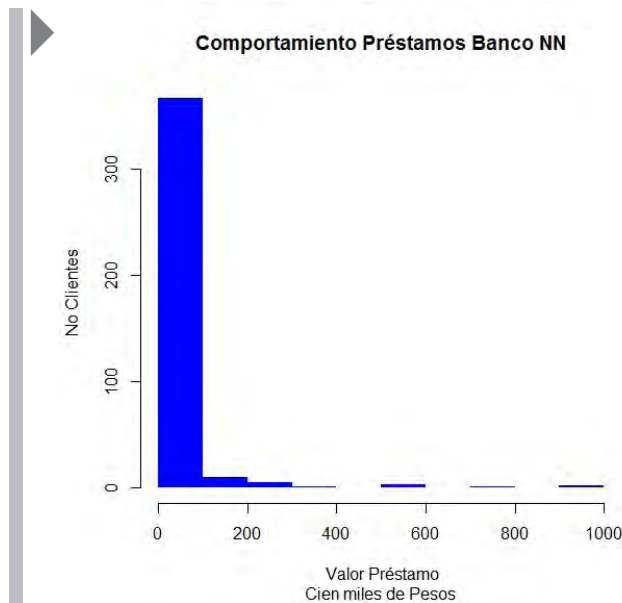
Se efectúa el gráfico empleando las opciones señaladas anteriormente, pero primero se debe dar a R la instrucción de que tome los 387 valores que integran la variable : Valor del desembolso (X4), ya que en su memoria aparece como un solo dato, que es la información referente al número de columnas que fueron accedadas.

```
Vrdesembolso <- X4$X4
```

Se pasa la información a cientos de miles de pesos

```
Vrdesembolso <- Vrdesembolso/100000
```

```
hist(Vrdesembolso,nclass= 12, freq=T, col= "BLUE", border=F, main =
"Comportamiento Préstamos Banco NN", sub = "Cien miles de
Pesos", axes=T,labels=F, ylab = "No Clientes", xlab = " Valor Préstamo")
```



## ★ Ejemplo 2.2

También es posible realizar el histograma y polígono de frecuencias. Para ilustrar el proceso se toma la información referente a la cantidad de onzas de jugo de concentrado de mora de una muestra de frascos. Los datos son definidos como un objeto y se da la instrucción de hacer el histograma como se detalla enseguida.

Se introduce la información

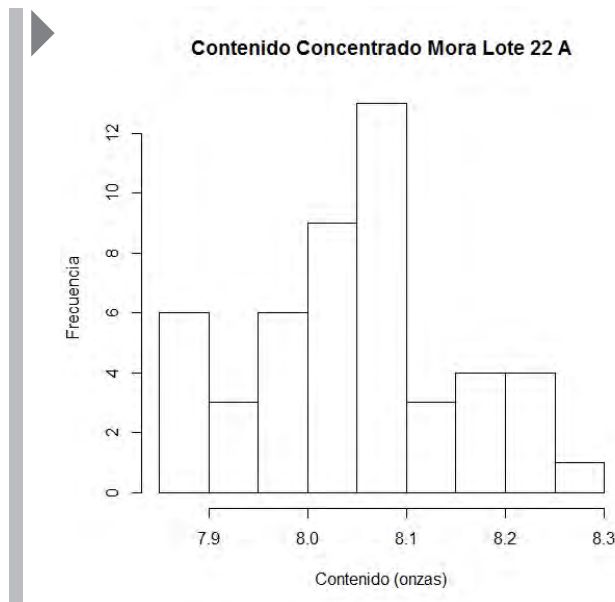
```
Frascos <-c(7.85,7.86,7.87,7.87,7.88,7.89,7.92,7.94,7.95,7.96,7.97,7.9
7,7.98,7.99, 7.99,8.01,8.03,8.03,8.04,8.05,8.05,8.05,8.05,8.06,8.06
,8.06,8.07, 8.07,8.07, 8.08, 8.09, 8.09,8.09,8.10,8.10,8.10,8.11,8.11,8.1
2,8.16,8.16,8.17,8.19,8.21,8.21, 8.22, 8.24, 8.26)
```

Se da la instrucción para hacer el histograma.

```
hist(Frascos,main="Contenido Concentrado Mora Lote 22 A",xlab="Contenido (onzas)", ylab = "Frecuencia")
```

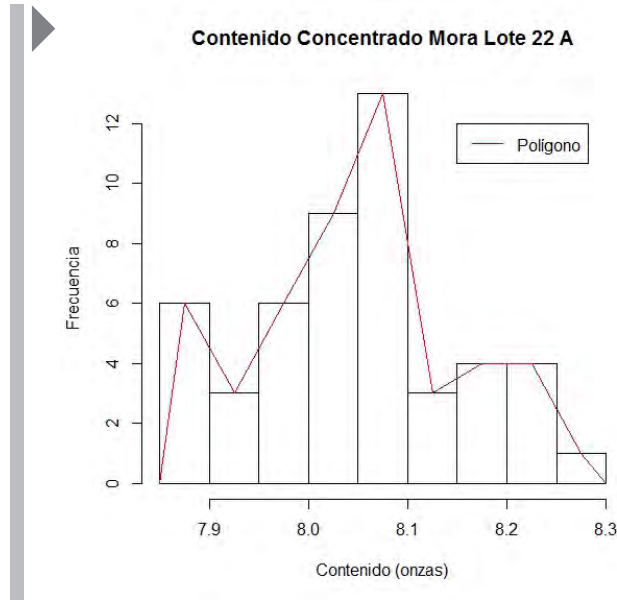
Se toma la anterior instrucción y se define como un objeto para hacer el polígono de frecuencias.

```
A <- hist (Frascos, main= "Contenido Concentrado Mora Lote 22 A", xlab = " Contenido (onzas) ", ylab = "Frecuencia")
```



Se genera el polígono de frecuencias

```
lines(c(min(A$breaks),A$mids,max(A$breaks)),c(0,A$counts,0),type="l",col="red")  
legend(8.15,12,"Polígono",lty=1,col="red")
```



### 3. polygon(): POLÍGONO

La forma general del comando viene dada por:

```
polygon (x, y = NULL, density = NULL, border = NULL, col = NA, lty = par("lty"),
...,)
```

- x,y      Vectores que contienen las coordenadas de los vértices del polígono
- density    La densidad de líneas de sombreado. El valor predeterminado NULL, significa que no hay líneas de sombreado. El valor cero de la densidad significa que no hay sombras ni relleno, mientras que valores negativos (y NA) suprimir la sombra (y permitir así que el color de relleno)
- col        El color de relleno del polígono. El valor por defecto es NA, que deja sin llenar los polígonos
- border    Color del borde de la frontera. El valor predeterminado, NULL, significa usar par ("fg"). border = NA , omite las fronteras
- lty        Tipo de línea a utilizar



### ★ Ejemplo 3.1

Se define el objeto con los valores de la variable independiente

```
x <- c(1,2,3,4,5,6,7,8,9,10,11,12,13,15)
```

Se define el objeto con los valores de la variable dependiente

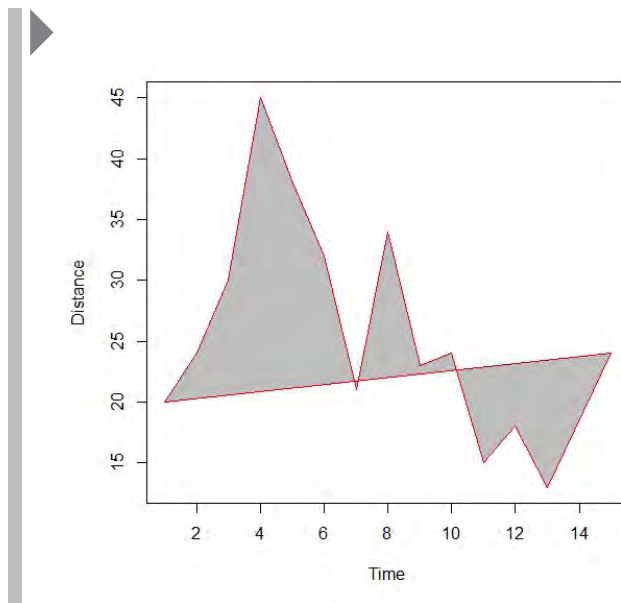
```
y <- c(20,24,30,45,38,32,21,34,23,24,15,18,13,24)
```

Se hace el gráfico

```
plot(x, y, type="n", xlab="Time", ylab="Distance")
```

Se emplea el comando polygon

```
polygon(x,y, col="gray", border = "red")
```



### ★ Ejemplo 3.2

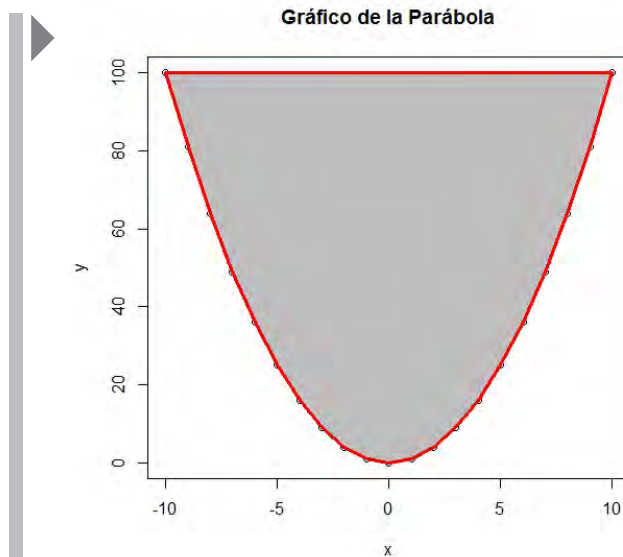
Interesa hacer el gráfico de una parábola que pasa por el origen y su recorrido es el intervalo (-10,10)

```
x <- = seq(-10,10)
```

```
y <- x^2
```

```
plot(x,y, main = "Gráfico de la Parábola")
```

```
polygon(x,y, col="gray", border = "red", lwd = 3)
```



## 4. barplot(): GRÁFICOS DE BARRAS

La instrucción general viene dada por:

```
barplot(height, width = 1, names.arg = NULL, beside = FALSE, horiz = FALSE,
density = NULL, angle = 45,col = NULL, border = par("fg"),main = NULL, sub
= NULL, xlab = NULL, ylab = NULL, xlim = NULL, ylim = NULL, xpd = TRUE,
log = "",axes = TRUE, axisnames = TRUE, cex.axis = par("cex.axis"),cex.names=
par("cex.axis"),
inside = TRUE, plot = TRUE, axis.lty = 0, offset = 0,add = FALSE, args.legend
= NULL, ...)
```

Donde

width	Vector opcional para determinar el ancho de la barra
main	Título del gráfico
sub	Subtítulo del gráfico
xlab	Nombre del eje x
ylab	Nombre del eje y
xlim	Límites del eje x
ylim	Límite del eje y
names.arg	Nombre que se da a cada una de las barras del gráfico, se debe ingresar como un vector
beside	Un valor lógico. Si es FALSO, las columnas de altura se presentan como barras apiladas, y si es verdad, las columnas se representan como barras yuxtapuestas

### ★ Ejemplo 4.1

Para ilustrar el proceso que se debe seguir para realizar el gráfico, se toma el estado civil de un grupo de personas así:

Soltero	8
Casado	12
Separado	11
Union libre	20
Viudo	4
Divorciado	3

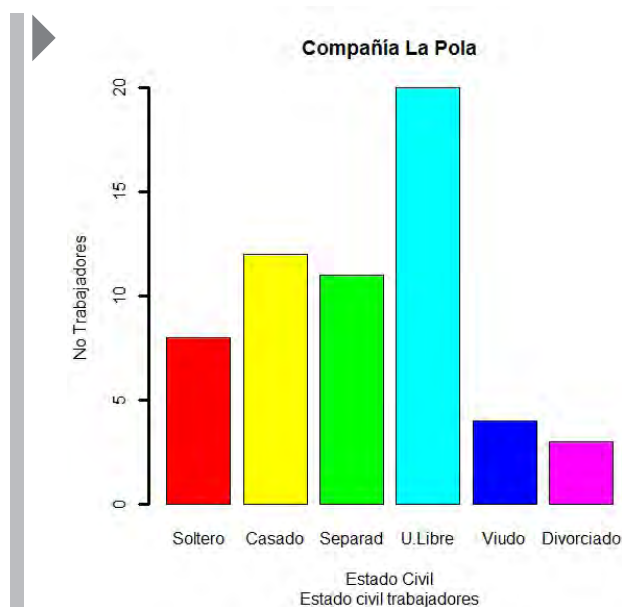
Se ingresa las frecuencias para cada una de las clases y se define como un objeto llamado Estadocivil

```
Estadocivil <- c(8,12,11,20,4,3)
```

Se da la instrucción para generar el gráfico de barra

```
Titulobarras <- c("Soltero","Casado","Separad","U.Libre", "Viudo",  
"Divorciado")
```

```
barplot( Estadocivil, width = 1,col=rainbow(6),lwd=3,  
main="Compañía La Pola", sub = "Estado civil trabajadores", xlab  
= "Estado Civil", ylab = "No Trabajadores", names.arg = Tituloba-  
rras)
```



## ★ Ejemplo 4.2

Supongamos que la Compañía La Pola desea mostrar el valor de las ventas durante los últimos 6 años a nivel global y para cada una de las seis sucursales que tiene a lo largo de todo el país. La información que posee se muestra en el siguiente cuadro.

Compañía La Pola Ventas Netas por Sucursal 2007-2010				
Sucursal	Año			
	2007	2008	2009	2010
1	23	32	39	23
2	26	31	40	24
3	28	34	42	25
4	23	34	43	26
5	25	36	35	27
6	23	38	38	30

El proceso que se debe seguir para realizar la representación es el siguiente:

Se define un objeto y se ingresan las ventas para cada una de las sucursales en forma matricial. El ingreso de la información se hace por fila.

```
VTASSUCURSAL <- matrix (c(23,32,39,23,26,31,40,24,28,34,42,25,
23,34,43,26,25, 36, 35 ,27,23,38, 38,30), byrow = TRUE, nrow = 6,
ncol = 4)
```

Se titula cada una de las filas que conforman la matriz

```
rownames(VTASSUCURSAL) <- c("sucursal 1","sucursal 2","sucurs-
sal3" ,"sucursal 4","sucursal 5","sucursal 6")
```

Se genera el título a la información que esta sobre el eje X

```
Titulobarras <- c("2001","2002","2003","2004")
```

Se corre la instrucción para hacer el gráfico con los siguientes comandos:

**barplot**

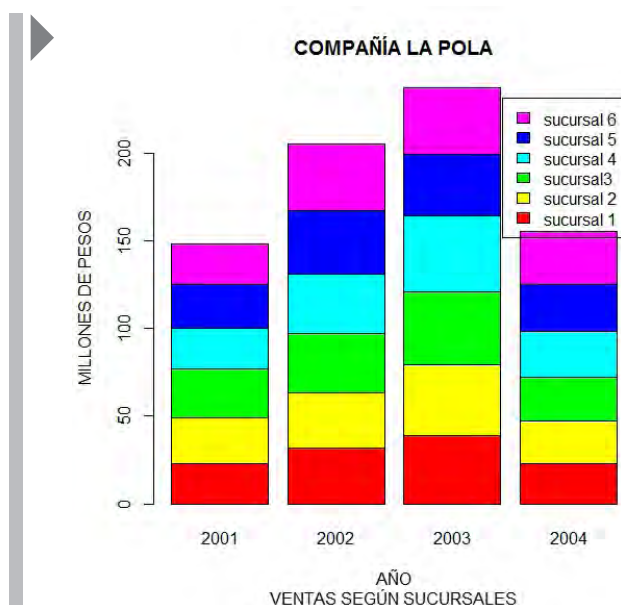
Hace el gráfico de barras

**VTASSUCURSAL**

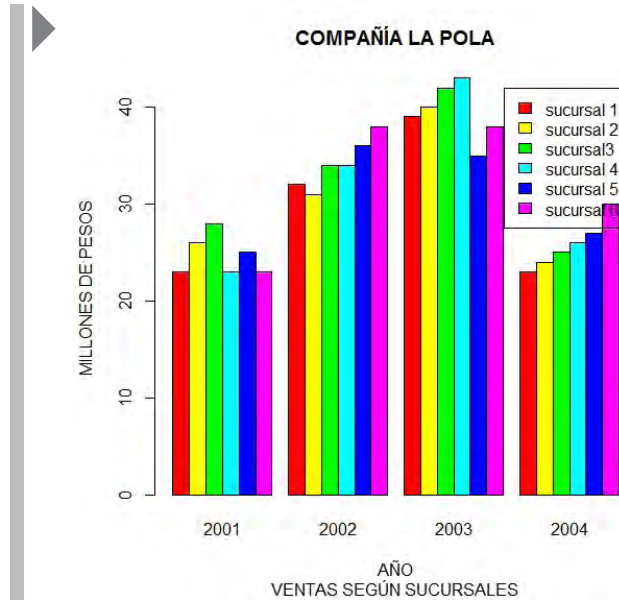
Nombre del objeto donde está la información que va a ser representada

<code>col=rainbow(6)</code>	Genera la cantidad de colores distintos según el número que se introduzca
<code>main</code>	Coloca el título principal al gráfico
<code>sub</code>	Coloca subtítulo
<code>xlab</code>	Titula el eje X
<code>ylab</code>	Titula el eje Y
<code>names.arg</code>	Ubica los títulos a cada una de las barras que integran el gráfico
<code>beside</code>	Operador lógico, si es FALSE(F) = Hace un gráfico en barras de partes componentes. Si es TRUE (T) hace el gráfico de barras corrientes
<code>legend</code>	Introduce el cuadro de convenciones

```
ACUMULADO <- barplot(VTASSUCURSAL, col=rainbow(6) ,
  main= "COMPAÑÍA LA POLA", sub = "VENTAS SEGÚN SUCURSALES", xlab = "AÑO", ylab = "MILLONES DE PESOS", names.arg = Titulobarras, beside = F, legend = rownames(VTASSUCURSAL))
```



Tomando la misma información anterior pero utilizando el comando `beside = T`, se genera el siguiente gráfico



## 5. pie(): GRÁFICOS CIRCULAR

La instrucción general viene dada por:

**pie(x, labels = names(x), radius = 0.8, clockwise = FALSE, col = NULL, main = NULL, ...)**

**x** Nombre del objeto que se desea representar. Se define como un vector numérico en el cual aparecen los valores porcentuales en que participan cada una de las categorías que son objeto de representación

**labels** : Nombre de cada una de las categorías que están siendo representadas. Se deben definir como un vector, en el cual los nombres deben estar ubicadas en el mismo orden en que se colocaron los valores al definir el objeto x

**radius** : Determina el radio del círculo, por lo tanto, es el determinante del tamaño que se desea para la representación. El radio del gráfico se dibuja en un cuadrado cuyos lados van de -1 a 1

**clockwise:** Operador lógico, si es TRUE (T) se hace la representación siguiendo las manecillas del reloj. Si es FALSE (F), no se sigue las manecillas del reloj

**col:** Define los colores para cada una de las categorías del gráfico. Los colores pueden ser definidos de diversas maneras así:

Definiéndolos como un objeto e indicando cada uno de los colores con los que se quiere hacer el gráfico. Ejemplo: `col = c("purple", "violetred1", "green3", "cornsilk", "cyan", "red")`

Puede ser definido así: `col = 1: n`. Donde `n` es el número de categorías que integran el gráfico

Puede hacerse con el comando `col=rainbow(n)`, donde `n` es el número de categorías a graficar

**main:** Título principal

### ★ Ejemplo 5.1

Tomando nuevamente el ejemplo de la Compañía La Pola y haciendo la representación porcentual de las ventas por sucursal para el año 2010, se tiene lo siguiente:

Compañía La Pola Ventas Netas por Sucursal Año 2010		
Sucursal	Año	
	Ventas	Porcentaje
1	23	14,84
2	24	15,48
3	25	16,13
4	26	16,77
5	27	17,42
6	30	19,35
Total	155	100,00

Se ingresa la participación de ventas para cada una de las sucursales en el año 2010

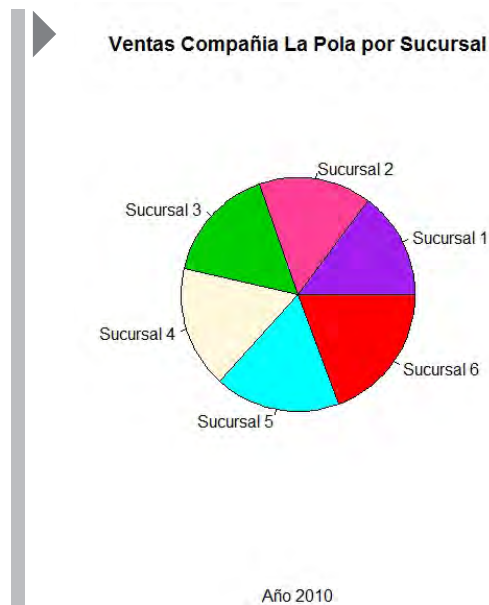
```
vtas2010 <- c(0.1484, 0.1548, 0.1613, 0.1677, 0.1742, 0.1936)
```

Se dan los nombres a las categorías

```
names(vtas2010) <- c("Sucursal 1", "Sucursal 2", "Sucursal 3", "Sucursal 4", "Sucursal 5", "Sucursal 6")
```

Se corre la instrucción para hacer el gráfico

```
pie(vtas2010, col = c("purple", "violetred1", "green3", "cornsilk", "cyan", "red"), labels = names(vtas2010), radius = 0.6, main = "Ventas Compañía La Pola por Sucursal", sub = "Año 2010", clockwise = FALSE)
```



## ★ Ejemplo 5.2

Se define un objeto con 200 elementos

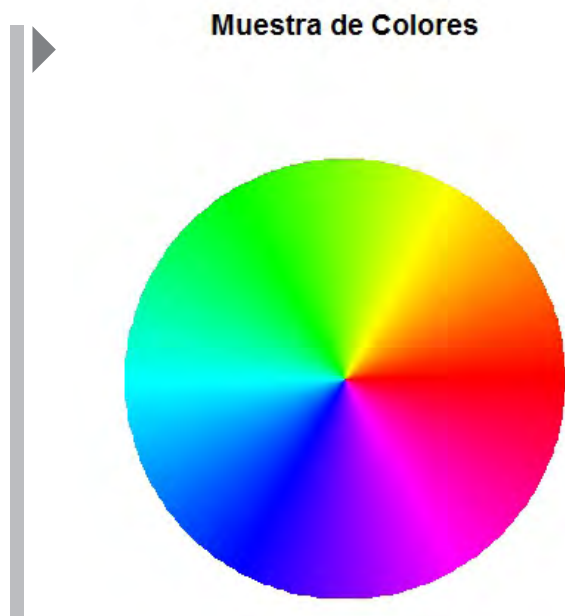
```
n <- 200
```

Se desea realizar un gráfico de pastel para 200 categorías cada una de ellas con una observación para lo cual se utiliza la instrucción siguiente: `rep(1,n)`.

Se realiza el gráfico

```
pie(rep(1,n), labels="", col=rainbow(n), border=NA, main = "Muestra de Colores")
```





## 6. boxplot(): GRÁFICOS DE CAJA

El formato general para la realización del gráfico viene dado por la siguiente expresión:

```
boxplot (x , main="xxx", col= xxx(4), range= xxx,  horizontal=F/T, boxwax= xxx,
names= xxx, outline=T, notch=F)
```

boxplot	Indica que se va a realizar el gráfico de caja
x	Nombre del objeto en el que reposa la información a graficar
nn	Vector que se crea para darle nombre a cada boxplot
main	Instrucción para titular la gráfica
col	Genera el color que se desea dar al gráfico, que puede ser determinado por su nombre al cual se le puede agregar un número de 1 a 4. Donde 1 es el color más claro y 4 el más oscuro.
horizontal	Elemento lógico, si se coloca TRUE (T) hace el gráfico en forma horizontal, si es FALSE (F) genera el gráfico en forma vertical
boxwax	Ancho de la caja

names	Nombre de los boxplots
outline	Elemento lógico, si se coloca TRUE (T) permite ver los datos que quedan por fuera del boxplot. Si se emplea FALSE (F) no permite ver los datos
Notch	Comando lógico empleado para crear una cintura al boxplot. Si es TRUE (T) se genera la cintura, si es FALSE (F) queda en forma rectangular

### ★ Ejemplo 6.1

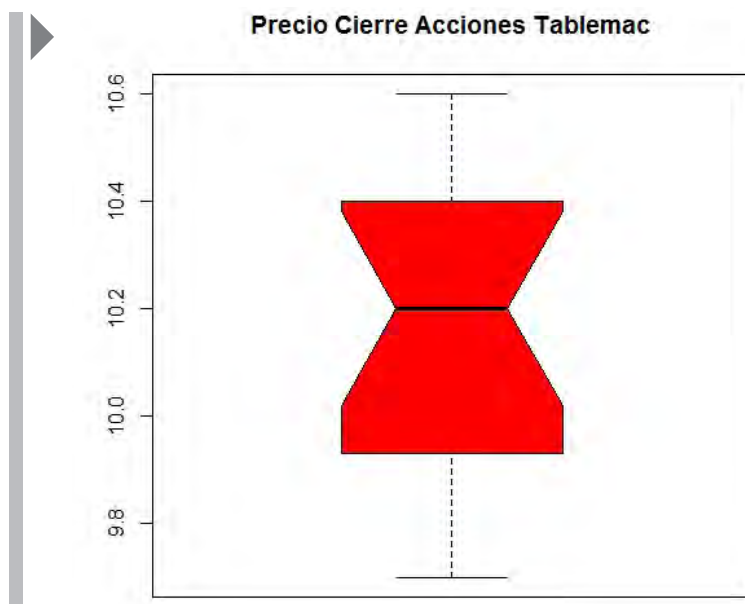
Tomando el precio de cierre de las acciones de Tablemac y con el ánimo de ilustrar el manejo de las instrucciones para generar el gráfico, se tiene lo siguiente:

Se lee la información Tablemac de la hoja Excel y se importa a R

```
Tablemac <-read.delim("clipboard")
```

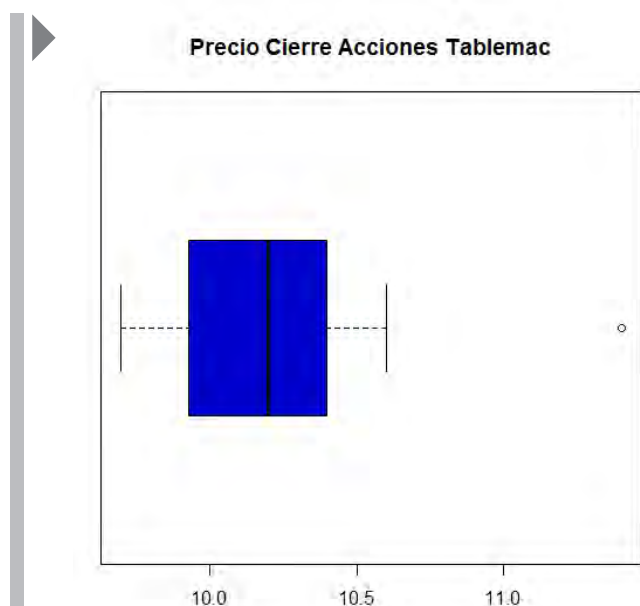
```
Tablemac
```

```
boxplot(Tablemac, main="Precio Cierre Acciones Tablemac",  
col=rainbow(4), range=1.5, horizontal=F, boxwax=50, outline=F,  
notch=T)
```



Ahora se hace el gráfico en forma horizontal (horizontal = T) sin cintura (Notch = F), se muestran los datos que quedan por fuera del boxplot (outline = T) y se cambia el color a azul oscuro (col="blue3")

```
Tablemac <-read.delim("clipboard")
Tablemac
boxplot(Tablemac,main="Precio Cierre Acciones Tablemac",
col="blue3", range=1.0, horizontal=T, boxwax=50, outline=T,
notch=F)
```



## 7. par(mfrow):

### VARIOS GRÁFICOS EN UNA PÁGINA

Es posible realizar en una misma página varios gráficos, para lograr esto se debe emplear la opción:

```
Par(mfrow = c(filas,columnas))
```

**par(mfrow=c(filas,columnas)).** Está instrucción divide la pantalla gráfica en tantas filas y columnas como se indica. Puede también ser mfcol para indicar que las gráficas aparezcan en el orden de las columnas y no de las filas.

Así, si se utiliza

```
par(mfrow=c(1,1))
```

Realiza un solo gráfico por ventana, que es la opción por defecto

<code>par(mfrow=c(2,1))</code>	Efectúa una matriz de gráficos de orden 2x1, un gráfico debajo de otro
<code>par(mfrow=c(2,3))</code>	Genera una matriz de gráficos de orden 2 x 3, dos filas por tres Columnas

### ★ Ejemplo 7.1

Se hace el gráfico del rendimiento de las acciones de Tablemac, Bancolombia, y Ecopetrol. Lo primero que se hace es calcular el rendimiento para las acciones en mención y se incorporan como objeto en R.

Rendimiento = Logaritmo Natural (Vr acción periodo t/ Vr acción periodo t-1)

Los resultados obtenidos son:

```
Rentablemac <- c(0.0099,0.0194,0.0190,-0.0287,0.0000,-
0.0196,0,-0.0230,0,-0.0174, 0.0010,0.0224,0.0558,-
0.0192,0.0097,0.0918)
```

```
Renbancolom <- c(-0.0015,0.0029,-0.0036,0.0058,-
0.0065,0.0194,-0.0136,-.0058,0.0101,
-0.0189,0.0196,0.0249,0.0243,-0.0069,-0.0041,0.0069)
```

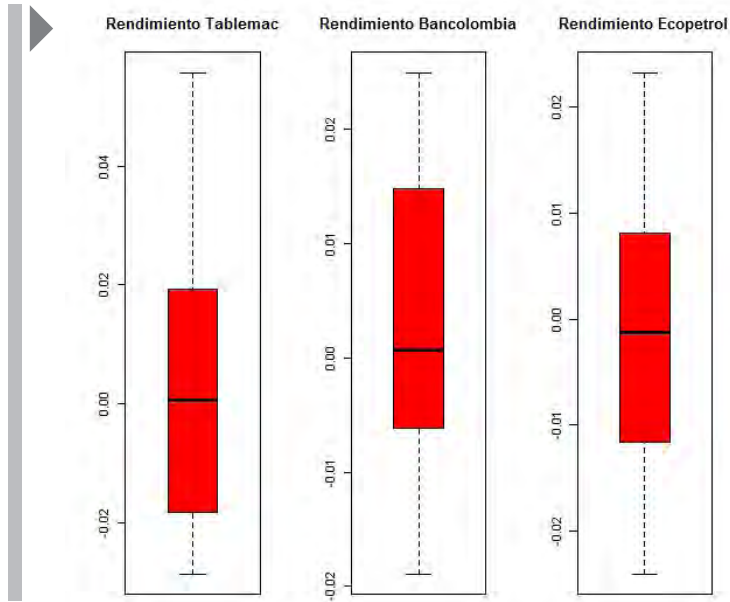
```
Renecopetrol <- c(-0.0076,0.0113,0.0149,-0.0012,-
0.0199,0.0050,-0.0241,-0.0013,-.0013, -0.0195, -0.0092,0.0131,-
0.0013,-0.0026,0.0233,-0.0142)
```

Se indica a R cuantos gráficos se van a hacer y la forma como se desean representar

```
par(mfrow=c(1,3))
```

Se generan las instrucciones para los gráficos

```
boxplot(Rentablemac,main="Rendimiento Tablemac",
col="RED", range=1.7, horizontal=F, boxwax=50, outline=F,
notch=F)
boxplot(Renbancolom,main="Rendimiento Bancolombia",
col="RED", range=1940, horizontal=F, boxwax=10, outline=F,
notch=F)
boxplot(Renecopetrol, main="Rendimiento Ecopetrol",
col="RED", range=280, horizontal=F, boxwax=10, outline=F,
notch=F)
```



## 8. stem():

### GRÁFICOS DE TALLOS Y HOJAS

La instrucción general viene dada por el siguiente formato:

```
stem(x, scale = 1, width = 80, atom = 1e-08)
```

x            Nombre del objeto que se desea representar  
scale        Controla la longitud  
width        Controla el ancho



#### Ejemplo 8.1

Tomando el ejemplo de las ventas de la Compañía La Pola por sucursales ofrecida en el ejemplo 4.2 y haciendo la representación de las ventas totales mediante un gráfico de tallos y hojas, se tiene:

```
VTASSUCURSAL <- matrix( c(23,32,39,23,26,31,40,24,28,34,4
2,25,23,34,43, 26,25, 36, 35 ,27,23, 38,38,30), byrow = TRUE,
c(6,4))
stem(VTASSUCURSAL, scale = 1, width = 80)
```

El resultado es:

The decimal point is 1 digit(s) to the right of the |

2 | 33334

2 | 556678

3 | 01244

3 | 56889

4 | 023

## 9. OPCIONES PARA GUARDAR GRÁFICOS

Los gráficos generados con el uso del programa R, pueden ser guardados y existen distintas maneras para hacerlo. Las más corrientes son formato JPEG y PDF, cada uno de los cuales se ilustrará enseguida.

Para guardar la representación en formato JPEG, la instrucción que se debe utilizar es:

<code>jpeg(Nombre del archivo)</code>	Nombre del archivo.jpeg")
<code>dev.off()</code>	Es el comando que le indica al programa el fin de las instrucciones, es decir, cierra el dispositivo gráfico.

### Ejemplo 9.1

Para ilustrar el proceso que se debe seguir, se toma el ejemplo de las ventas de la Compañía La Pola según sucursales que fue ofrecido en el ejemplo 4.2. Lo primero que se debe realizar es el gráfico, para lo cual se siguen las instrucciones que se indican enseguida.

Se guarda la representación, lo primero que se debe hacer es definir el formato en el que se desea guardar y el nombre en que se hará.

```
jpeg(file="Ventas por Sucursal.jpeg")
```

Se realiza el Gráfico

```
VTASSUCURSAL <- matrix (c(23,32,39,23,26,31,40,24,28,34,42,25,  
23,34,43,26,25, 36, 35 ,27,23,38, 38,30), byrow = TRUE, nrow = 6,  
ncol = 4)
```

```
rownames(VTASSUCURSAL) <- c("sucursal 1","sucursal 2","sucur-  
sal3" ,"sucursal 4","sucursal 5","sucursal 6")
```

```
VTASSUCURSAL
```

```
Titulobarras <- c("2001","2002","2003","2004")
```

```
ACUMULADO <- barplot(VTASSUCURSAL,col=rainbow(6),main= "COMPAÑÍA LA POLA", sub = "VENTAS SEGÚN SUCURSA-  
LES", xlab = "AÑO", ylab = "MILLONES DE PESOS", names.arg =  
Titulobarras, beside = F, legend = rownames(VTASSUCURSAL))  
dev.off()
```

El gráfico aparece en formato de imagen en la carpeta de trabajo, puede dirigirse allí y encontrará el archivo: VTASSUCURSAL en extensión JPEG.

Si se desea guardar la representación en formato PDF, el proceso que se debe seguir es el siguiente:

Se define el formato en que se quiere guardar el gráfico y bajo que nombre se hará

```
pdf(file="Vtas La Pola por Sucursal.pdf")
```

Se genera el gráfico

```
VTASSUCURSAL <- matrix (c(23,32,39,23,26,31,40,24,28,34,42,25,  
23,34,43,26,25, 36, 35 ,27,23,38, 38,30), byrow = TRUE, nrow = 6,  
ncol = 4)  
rownames(VTASSUCURSAL) <- c("sucursal 1","sucursal 2","sucur-  
sal3" ,"sucursal 4","sucursal 5","sucursal 6")  
VTASSUCURSAL  
Titulobarras <- c("2001","2002","2003","2004")  
ACUMULADO <- barplot(VTASSUCURSAL, col=rainbow(6) ,  
main= "COMPAÑÍA LA POLA", sub = "VENTAS SEGÚN SUCUR-  
SALES", xlab = "AÑO", ylab = "MILLONES DE PESOS", names.arg  
= Titulobarras, beside = F, legend = rownames(VTASSUCURSAL))
```

Se cierran las instrucciones y se guarda

```
dev.off()
```

El gráfico aparece en la carpeta de trabajo, puede dirigirse allí y encontrará el archivo: VTASSUCURSAL en formato pdf

Al igual que se puede elaborar una ventana donde se muestren diferentes gráficos, también se puede elaborar un archivo con una imagen que posea varios gráficos, lo cual permite analizarlas y realizar comparaciones. Para esto las instrucciones que permite guardarlos en formato PDF o JPEG según la instrucción que se emplee son las siguientes:

- `jpeg(file="Nombre del archivo.jpeg", quality=100)`
- Se dan las instrucciones para realizar los diferentes gráficos
- `par(mfrow=c(X,Y))`
- `dev.off()`

Donde

<code>jpeg=</code>	Se indica el tipo de archivo que se quiere guardar
<code>file=</code>	Comando que permite dar nombre al archivo
<code>quality=</code>	Cifra porcentual que determina la calidad con que se quiere hacer la representación y donde 100 es la máxima
<code>par(mfrow=c(X,Y))</code>	Hace un arreglo matricial, de X columnas y Y filas, y las graficas van apareciendo en orden de las filas
<code>dev.off()</code>	Este comando indica al programa el fin de las instrucciones, es decir, cierra el dispositivo gráfico

Como en los casos anteriores, el archivo es encontrado en la carpeta de trabajo, bajo el nombre en que fue guardado.

## 10. COMANDOS ESPECIALES

### 10.1 Comando `fourfoldplot`

El comando permite graficar tablas de contingencia para variables dicotómicas, su forma general es la siguiente:

```
fourfoldplot(x, color = ("color 1","color 2"),conf.level = 0.95, margin = c(1, 2),
space = 0.2, main = NULL)
```

Donde

<code>x</code>	Tabla de contingencia de orden 2 por 2
<code>color</code>	Vector de longitud 2 que especifica los colores a utilizar
<code>conf.level</code>	Nivel de confianza, utilizado para los anillos de confianza en la razón de momios. Debe ser un número no negativo menor de 1, si se establece en 0, los anillos de confianza son suprimidas
<code>margin</code>	Debe ser de 1, 2, o c (1, 2) (por defecto), que corresponde a la fila, columna o márgenes
<code>space</code>	Fija el tamaño de las letras
<code>main</code>	Título del gráfico



Ejemplo 10.1

Se tienen los estudiantes de Administración de Empresas de la Universidad Nacional clasificados según género y jornada a la que pertenecen. La información se suministra en el siguiente cuadro:

Universidad Nacional de Colombia Estudiantes de Administración de Empresas 2011 - 01			
Concepto	Genero		Total
Jornada	Diurna	Nocturna	
Masculino	220	310	530
Femenino	170	280	450
Total	390	590	980

Se ingresa la información en forma matricial

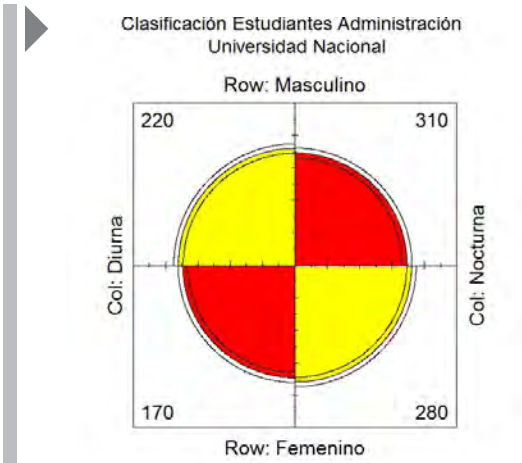
```
Administración <- matrix(c(220,310,170,280), nrow = 2, byrow = T)
```

Se titulan las filas y las columnas de la matriz

```
dimnames(Administración) <- list ( c ( "Masculino" ,"Femenino" ), c ("Diurna",  
"Nocturna"))
```

Se realiza el gráfico

```
fourfoldplot((Administración),color = c("red", "yellow"),conf.level = 0.99,  
space = 0.2,main="Clasificación Estudiantes Administración \n Universidad  
Nacional")
```



## 10.2 layout(). Visualización de gráficos en una misma ventana.

Para poder visualizar diferentes gráficos en una misma ventana, además de las opciones `par(mfcol)` y `par(mfrow)`, R también ofrece la posibilidad de crear una matriz personalizada y así poder agregar los gráficos que se deseen y en el orden que interese. Para esto, se tiene la instrucción `layout()`, en la cual se crea una matriz que es la que va a contener los gráficos.

```
mat <- matrix(1:4, 2, 2)
mat
layout(mat)
```

**mat** Crea una matriz de dos columnas por dos filas

**layout(mat)** Se le indica al programa que las gráficos van a tener el orden que se ha dado en la matriz `mat`, por lo tanto, los gráficos aparecerán en el orden dado

### ★ Ejemplo 10.2

Para ilustrar el proceso, se toman las instrucciones de los últimos ejemplos a fin de realizar el gráfico con la opción `layout`.

Se crea la matriz donde se ubicarán los gráficos. Para el caso, el espacio a graficar se divide en cuatro zonas de igual tamaño en la que se distinguen dos filas y dos columnas.

```
mat <- matrix(1:4, 2, 2)
mat
layout(mat)
```

Se generan los gráficos

■ **Gráfico 1.** Se ubica en la posición 1.1

```
VTASSUCURSAL <- matrix (c(23,32,39,23,26,31,40,24,28,34,42,25,
23,34,43,26,25, 36, 35 ,27,23,38, 38,30), byrow = TRUE, nrow = 6,
ncol = 4)
```

```
rownames(VTASSUCURSAL) <- c("sucursal 1","sucursal 2","sucurs-
sal3" ,"sucursal 4","sucursal 5","sucursal 6")
```

```
Titulobarras <- c("2001","2002","2003","2004")
```

```
ACUMULADO <- barplot(VTASSUCURSAL, col=rainbow(6) ,
main= "COMPAÑÍA LA POLA", sub = "VENTAS SEGÚN SUCUR-
```

```
SALES", xlab = "AÑO", ylab = "MILLONES DE PESOS", names.arg
= Titulobarras, beside = F)
```

■ Gráfico 2. Se ubica en la posición 1.2

```
VTASSUCURSAL <- matrix (c(23,32,39,23,26,31,40,24,28,34,42,25,
23,34,43,26,25, 36, 35 ,27,23,38, 38,30), byrow = TRUE, nrow = 6,
ncol = 4)
```

```
rownames(VTASSUCURSAL) <- c("sucursal 1","sucursal 2","sucur-
sal3","sucursal 4","sucursal 5","sucursal 6")
```

```
Titulobarras <- c("2001","2002","2003","2004")
```

```
ACUMULADO <- barplot(VTASSUCURSAL, col=rainbow(6) ,
main= "COMPAÑÍA LA POLA", sub = "VENTAS SEGÚN SUCUR-
SALES", xlab = "AÑO", ylab = "MILLONES DE PESOS", names.arg
= Titulobarras, beside =T)
```

■ Gráfico 3. Se localiza en la posición 2.1

```
vtas2010 <- c(0.1484, 0.1548, 0.1613, 0.1677, 0.1742, 0.1936)
```

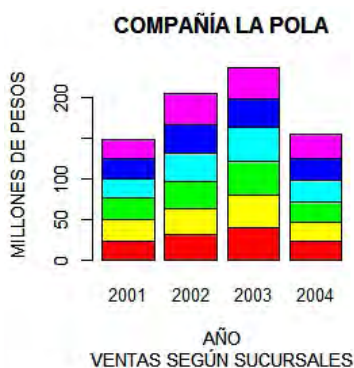
```
names(vtas2010) <- c("Sucursal 1", "Sucursal 2","Sucursal 3","Sucur-
sal 4", "Sucursal 5","Sucursal 6")
```

```
pie(vtas2010, col = c("purple", "violetred1", "green3","cornsilk",
"cyan", "red"), labels =names(vtas2010), radius =0.6,main = "Ven-
tas Compañía La Pola por Sucursal", sub = "Año 2010",clockwise =
FALSE )
```

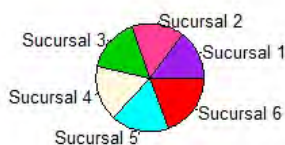
■ Gráfico 4. Se sitúa en la posición 2.2

```
n <- 200
```

```
pie(rep(1,n), labels="", col=rainbow(n), border=NA, main = "Mues-
tra de Colores")
```



Ventas Compañía La Pola por Sucur:

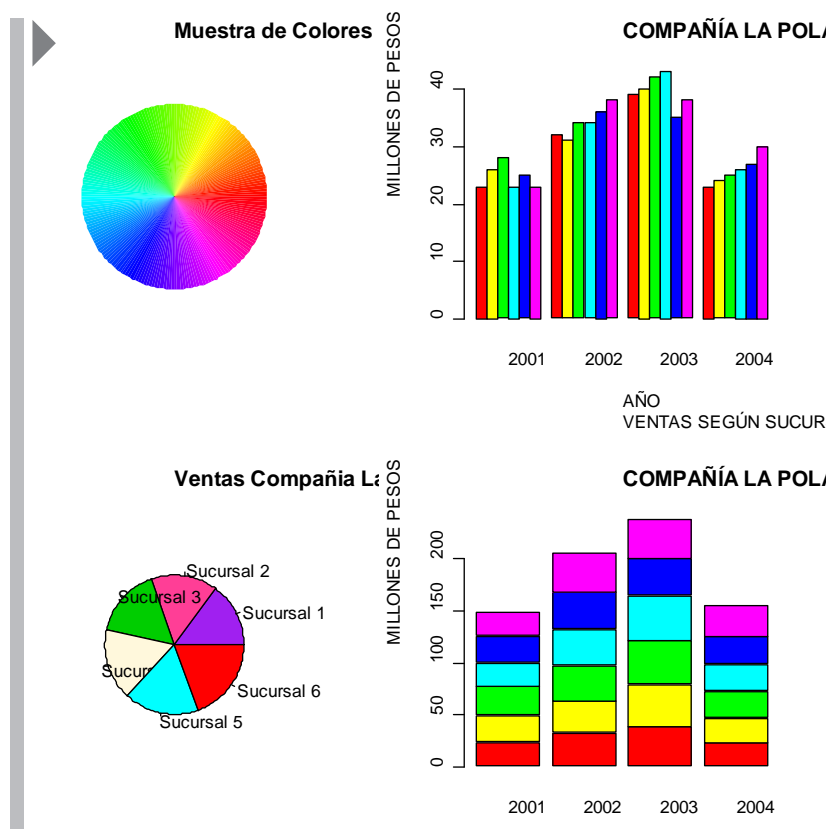


Año 2010



También es posible ubicar los gráficos en un orden deseado, digamos 4, 3, 2,1, haciendo el arreglo por filas y colocando la siguiente instrucción antes de dar las especificaciones de las diferentes representaciones que se desean hacer.

```
layout(matrix(c(4,3,2,1), 2,2))
layout.show(4)
```



No necesariamente los espacios para la realización de los gráficos deben ser de igual tamaño, R presenta la posibilidad de hacer modificaciones a los mismos con las siguientes instrucciones:

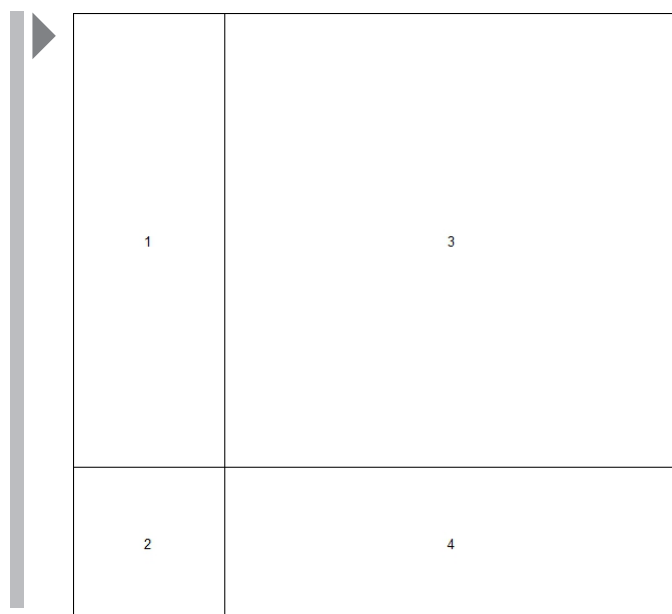
`widths=(X,Y)`. Hace referencia al ancho del espacio para hacer la representación. Cuando los valores X, Y son iguales, el ancho también lo es. Si toman valores diferentes por ejemplo 1,5; quiere decir que el espacio en Y es cinco veces más grande que el espacio en X.

`heights=()`. Hace referencia a la altura para hacer la representación. Cuando los valores X, Y son iguales, la altura también lo es. Si toman valores diferentes por ejemplo 1,5; quiere decir que el espacio en Y es cinco veces más grande que el espacio en X.

### ★ Ejemplo 10.3

Para clarificar el proceso a seguir y la forma como se generan los espacios para hacer las representaciones en R, enseguida se ofrece un ejemplo utilizando las instrucciones anteriores.

```
m <- matrix(1:4, 2, 2)
layout(m, widths=c(1, 3), heights=c(3, 1))
layout.show(4)
```





## Bibliografía

ARRIAZA Gómez, A., FERNÁNDEZ Palacín, J., LÓPEZ Sánchez, M., MUÑOZ Márquez, M., PÉREZ Plaza, S., & SÁNCHEZ Navas, A. (Febrero de 2008). knuth.uca.es. (S. d. Cádiz, Ed.) Recuperado el 9 de Septiembre de (H, 2008)2011, de Estadística Básica con R y R-Commander:

*<http://knuth.uca.es/moodle/course/view.php?id=37>*

CORREA, J. C., & GONZÁLEZ, N. (2011). Gráficos Estadísticos con R. Recuperado el 5 de 10 de 2011, de The R Project for Statistical Computing:

*<http://cran.r-project.org/doc/contrib/grafi3.pdf>*

RAMÓN, D. U. (2003). cran.r-project.org. Recuperado el 5 de 10 de 2011, de Unidad de Bioinformática, Centro Nacional de Investigaciones Oncológicas (CNIO): *<http://cran.r-project.org/doc/contrib/curso-R.Diaz-Uriarte.pdf>*

The R Reference Index. (2011). The Comprehensive R Archive Network. Recuperado el 2 de 09 de 2011

MAINDONALD, J.(2008). "Using R for Data Analysis and Graphics .Introduction, Code and Commentary". Disponible en línea en: (H., 2008)

PARADIS, Emmanuel.(2003). "R para Principiantes". Disponible en línea en: (H, 2008) Programa R. Disponible en *<http://www.cran.r-project.org>*

SAEZ, Antonio C. (2009) "Métodos Estadísticos con R y R Commander". Disponible en línea en: *<http://cran.r-project.org/doc/contrib/Saez-Castillo-RRCmdrv21.pdf>*

VENABLES; Bill; SMITH Dave. (2000). "Notas sobre R: Un entorno de programación para Análisis de Datos y Gráficos". Disponible en línea en: *<http://www.et.bs.ehu.es/~etptupaf/pub/R/sp-R.pdf>*







Fundamentos básicos para el manejo de R en estadística descriptiva  
Impreso en la Sección de Publicaciones e Imagen  
de la Universidad Nacional de Colombia, Sede Manizales, mayo de 2013.  
Tamaño: 170 x 240 mm. Tiraje: 200 ejemplares.  
Fuentes tipográficas: Adobe Ming Std, Gothic720 Lt BT, Gothic 720 BT,  
Schneider Md BT, Schneider Md BT, Optima.  
Páginas interiores impresas en propalmate 90 gramos  
Carátula propalcote 240 gramos.

as", ylab="font", font="10",  
main="2, adj=0.5, font",  
axis="black",  
4", "5", "6", "7", "8", "9", "10",  
pink", "black", "turquoise", "purple", "orange", "green", "orange", "tur",  
labels = dia,  
23,32,39,23,26,31,40,24,21  
SAL) <- c("sucursal",  
S <-  
VTASSUCURSAL,col=rainbow(6),  
LONES DE PESOS", names.arg,  
ont.lab=4, xlim=c(0,20), ylim=c(2  
nt.sub=3, bty="l", tcl=20, lty=6, la  
k", font.axis=4, col.  
2", "3", "4", "5", "6", "7", "8",  
ange", "green", "pink", "black", "turquoise", "purple",  
ombia,  
matrix(c(23,32,39,23,26,31,40,24,28,34,42,25,23,  
SSUCURSAL) <- c("sucursal 1", "sucurs  
A S U  
plot(VTASSUCURSAL,col=rainbow(6),main="COM  
ILLONES DE PESOS", names.arg = Titulo  
lab="dias", ylab="precio de cierre", font.lab=4, xli  
38,7.89,7.92,7.94,7.95,7.96,7.97,7.97,7.98,7.99, 7.99,8.01,8.03,8.03,8.04,8.05,8.05  
9,8.09,8.10,8.10,8.10,8.11,8.11,8.11,8.12,8.16,8.16,8.17,8.19,8.  
23,26,31,40,24,28,34,42,25,23,34,43,26,25, 36, 35 ,27,23,38, 38,30),  
c("sucursal 1", "sucursal 2", "sucursal3", "sucursal  
S U C (" 2 0 0 1", " 2 0  
col=rainbow(6), main="COMPAÑIA LA POLA", sub="V  
OS", names.arg = Titulobarras, beside = F, legen  
40,24,28,34,42,25,23,34,43,26,25, 36, 35 ,27,23,38, 38,3  
sursal 1", "sucursal 2", "sucursal3", "suc  
c (" 2 0 0 1",  
main="COMPAÑIA LA POLA", sub  
S", names.arg =  
dim=c