

Prácticas de Estadística con R



Alfredo Sánchez Alberca
asalber@ceu.es
<https://aprendeconalf.es>

Tabla de contenidos

Prefacio

¡Bienvenido a Prácticas de Estadística con R!

Este libro presenta una recopilación de prácticas de Estadística Descriptiva e Inferencial con el lenguaje de programación [R](#), con problemas aplicados a las Ciencias y las Ingenierías.

No es un libro para aprender a programar con R, ya que solo enseña el uso del lenguaje y de algunos de sus paquetes para resolver problemas de Estadística. Para quienes estén interesados en aprender a programar en este lenguaje, os recomiendo leer este [manual de R](#).

Capítulos

1. [Introducción a R](#)
2. [Preprocesamiento de datos](#)
3. [Distribuciones de frecuencias y representaciones gráficas](#)
4. [Estadística descriptiva](#)
5. [Regresión](#)
6. [Distribuciones de probabilidad.qmd](#)
7. [Intervalos de confianza para una población](#)
8. [Intervalos de confianza para la comparación de dos poblaciones](#)
9. [Contrastes de hipótesis](#)
10. [Análisis de la varianza](#)

Licencia

Esta obra está bajo una licencia Reconocimiento – No comercial – Compartir bajo la misma licencia 3.0 España de Creative Commons. Para ver una copia de esta licencia, visite <https://creativecommons.org/licenses/by-nc-sa/3.0/es/>.

Con esta licencia eres libre de:

- Copiar, distribuir y mostrar este trabajo.
- Realizar modificaciones de este trabajo.

Bajo las siguientes condiciones:

- **Reconocimiento.** Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador (pero no de una manera que sugiera que tiene su apoyo o apoyan el uso que hace de su obra).
- **No comercial.** No puede utilizar esta obra para fines comerciales.
- **Compartir bajo la misma licencia.** Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.

Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra.

Estas condiciones pueden no aplicarse si se obtiene el permiso del titular de los derechos de autor.

Nada en esta licencia menoscaba o restringe los derechos morales del autor.

1 Introducción a R

La gran potencia de cómputo alcanzada por los ordenadores ha convertido a los mismos en poderosas herramientas al servicio de todas aquellas disciplinas que, como la Estadística, requieren manejar un gran volumen de datos. Actualmente, prácticamente nadie se plantea hacer un estudio estadístico serio sin la ayuda de un buen programa de análisis de datos.

R es un potente lenguaje de programación que incluye multitud de funciones para la representación y el análisis de datos. Fue desarrollado por Robert Gentleman y Ross Ihaka en la Universidad de Auckland en Nueva Zelanda, aunque actualmente es mantenido por una enorme comunidad científica en todo el mundo.



Figura 1.1: Logotipo de R

Las ventajas de R frente a otros programas habituales de análisis de datos, como pueden ser SPSS, SAS o Matlab, son múltiples:

- Es software libre y por tanto gratuito. Puede descargarse desde la web <http://www.r-project.org/>.
- Es multiplataforma. Existen versiones para Windows, Macintosh, Linux y otras plataformas.
- Está avalado y en constante desarrollo por una amplia comunidad científica distribuida por todo el mundo que lo utiliza como estándar para el análisis de datos.
- Cuenta con multitud de paquetes para todo tipo de análisis estadísticos y representaciones gráficas, desde los más habituales, hasta los más novedosos y sofisticados que no incluyen otros programas. Los paquetes están organizados y documentados en un [repositorio CRAN](#) (Comprehensive R Archive Network) desde donde pueden descargarse libremente.

- Es programable, lo que permite que el usuario pueda crear fácilmente sus propias funciones o paquetes para análisis de datos específicos. Existen multitud de libros, manuales y tutoriales libres que permiten su aprendizaje e ilustran el análisis estadístico de datos en distintas disciplinas científicas como las Matemáticas, la Física, la Biología, la Psicología, la Medicina, etc.

1.1 Instalación de R

R puede descargarse desde el [sitio web oficial de R](#) o desde el repositorio principal de paquetes de R [CRAN](#). Basta con descargar el archivo de instalación correspondiente al sistema operativo de nuestro ordenador y realizar la instalación como cualquier otro programa.

El intérprete de R se arranca desde la terminal, aunque en Windows incorpora su propia aplicación, pero es muy básica. En general, para trabajos serios, conviene utilizar un entorno de desarrollo para R.

1.2 Entornos de desarrollo

Por defecto el entorno de trabajo de R es en línea de comandos, lo que significa que los cálculos y los análisis se realizan mediante comandos o instrucciones que el usuario teclea en una ventana de texto. No obstante, existen distintas interfaces gráficas de usuario que facilitan su uso, sobre todo para usuarios noveles. Algunas de ellas, como las que se enumeran a continuación, son completos entornos de desarrollo que facilitan la gestión de cualquier proyecto:

- [RStudio](#). Probablemente el entorno de desarrollo más extendido para programar con R ya que incorpora multitud de utilidades para facilitar la programación con R.
- [RKWard](#). Es otra otro de los entornos de desarrollo más completos que además incluye a posibilidad de añadir nuevos menús y cuadros de diálogo personalizados.
- [Visual Studio Code](#). Es un entorno de desarrollo de propósito general ampliamente extendido. Aunque no es un entorno de desarrollo específico para R, incluye una extensión con utilidades que facilitan mucho el desarrollo con R.

2 Preprocesamiento de datos

2.1 Ejercicios Resueltos

Para la realización de esta práctica se requieren los paquetes `readr` y `dplyr` de la colección de paquetes `tidyverse`.

```
library(tidyverse)
# Incluye los siguientes paquetes:
# - readr: para la lectura de ficheros csv.
# - dplyr: para el preprocesamiento y manipulación de datos.
```

Ejercicio 2.1. La siguiente tabla contiene los ingresos y gastos de una empresa durante el primer trimestre del año.

Mes	Ingresos	Gastos	Impuestos
Enero	45000	33400	6450
Febrero	41500	35400	6300
Marzo	51200	35600	7100

- a. Crear un data frame con los datos de la tabla.

Solución

```
df <- data.frame(
  Mes = c("Enero", "Febrero", "Marzo"),
  Ingresos = c(45000, 41500, 51200),
  Gastos = c(33400, 35400, 35600)
)
df
```

```
      Mes Ingresos Gastos
1  Enero   45000   33400
2 Febrero   41500   35400
3  Marzo   51200   35600
```

- b. Añadir una nueva columna con los siguientes impuestos pagados.

Mes	Impuestos
Enero	6450
Febrero	6300
Marzo	7100

💡 Solución 1

Con las funciones básicas de R.

```
df$Impuestos <- c(6450, 6300, 7100)
df
```

	Mes	Ingresos	Gastos	Impuestos
1	Enero	45000	33400	6450
2	Febrero	41500	35400	6300
3	Marzo	51200	35600	7100

💡 Solución 2

Con las funciones del paquete dplyr.

```
df <- df |>
  mutate(Impuestos = c(6450, 6300, 7100))
df
```

	Mes	Ingresos	Gastos	Impuestos
1	Enero	45000	33400	6450
2	Febrero	41500	35400	6300
3	Marzo	51200	35600	7100

c. Añadir una nueva fila con los siguientes datos de Abril.

Mes	Ingresos	Gastos	Impuestos
Abril	49700	36300	6850

💡 Solución 1

Con las funciones básicas de R.

```
df <- rbind(df, list(Mes = "Abril", Ingresos = 49700, Gastos = 36300, Impuestos = 6850))
df
```

	Mes	Ingresos	Gastos	Impuestos
1	Enero	45000	33400	6450

2	Febrero	41500	35400	6300
3	Marzo	51200	35600	7100
4	Abril	49700	36300	6850

💡 Solución 2

Con las funciones del paquete dplyr.

```
df <- df |>
  add_row(Mes = "Abril", Ingresos = 49700, Gastos = 36300, Impuestos = 6850)
df
```

	Mes	Ingresos	Gastos	Impuestos
1	Enero	45000	33400	6450
2	Febrero	41500	35400	6300
3	Marzo	51200	35600	7100
4	Abril	49700	36300	6850

- d. Cambiar los ingresos de Marzo por 50400.

💡 Solución

```
df[3, "Ingresos"] <- 50400
df
```

	Mes	Ingresos	Gastos	Impuestos
1	Enero	45000	33400	6450
2	Febrero	41500	35400	6300
3	Marzo	50400	35600	7100
4	Abril	49700	36300	6850

- e. Crear una nueva columna con los beneficios de cada mes (ingresos - gastos - impuestos).

💡 Solución 1

Con las funciones básicas de R.

```
df$Beneficios <- df$Ingresos - df$Gastos - df$Impuestos
df
```

	Mes	Ingresos	Gastos	Impuestos	Beneficios
1	Enero	45000	33400	6450	5150

2	Febrero	41500	35400	6300	-200
3	Marzo	50400	35600	7100	7700
4	Abril	49700	36300	6850	6550

💡 Solución 2

Con las funciones del paquete dplyr.

```
df <- df |>
  mutate(Beneficios = Ingresos - Gastos - Impuestos)
df
```

	Mes	Ingresos	Gastos	Impuestos	Beneficios
1	Enero	45000	33400	6450	5150
2	Febrero	41500	35400	6300	-200
3	Marzo	50400	35600	7100	7700
4	Abril	49700	36300	6850	6550

- f. Crear una nueva columna con el factor **Balance** con dos posibles categorías: **positivo** si ha habido beneficios y **negativo** si ha habido pérdidas.

💡 Solución 1

Con las funciones básicas de R.

```
df$Balance <- cut(df$Beneficios, breaks = c(-Inf, 0, Inf), labels = c("negativo", "positivo"))
df
```

	Mes	Ingresos	Gastos	Impuestos	Beneficios	Balance
1	Enero	45000	33400	6450	5150	positivo
2	Febrero	41500	35400	6300	-200	negativo
3	Marzo	50400	35600	7100	7700	positivo
4	Abril	49700	36300	6850	6550	positivo

💡 Solución 2

Con las funciones del paquete dplyr.

```
df <- df |>
  mutate(Balance = cut(Beneficios, breaks = c(-Inf, 0, Inf), labels = c("negativo", "positivo")))
df
```

	Mes	Ingresos	Gastos	Impuestos	Beneficios	Balance
--	-----	----------	--------	-----------	------------	---------

1	Enero	45000	33400	6450	5150	positivo
2	Febrero	41500	35400	6300	-200	negativo
3	Marzo	50400	35600	7100	7700	positivo
4	Abril	49700	36300	6850	6550	positivo

- g. Filtrar el conjunto de datos para quedarse con los nombres de los meses y los beneficios de los meses con balance positivo.

💡 Solución 1

Con las funciones básicas de R.

```
df[df$Balance == "positivo", c("Mes", "Beneficios")]
```

	Mes	Beneficios
1	Enero	5150
3	Marzo	7700
4	Abril	6550

💡 Solución 2

Con las funciones del paquete dplyr.

```
df |>
  filter(Balance == "positivo") |>
  select(Mes, Beneficios)
```

	Mes	Beneficios
1	Enero	5150
2	Marzo	7700
3	Abril	6550

Ejercicio 2.2. El fichero `colesterol.csv` contiene información de una muestra de pacientes donde se han medido la edad, el sexo, el peso, la altura y el nivel de colesterol, además de su nombre.

- a. Crear un data frame con los datos de todos los pacientes del estudio a partir del fichero `colesterol.csv`.

💡 Solución 1

Con las funciones básicas de R.

```
df <- read.csv("https://raw.githubusercontent.com/asalber/estadística-practicas")
```

💡 Solución 2

Con la función `read_csv` del paquete `readr`.

```
df <- read_csv("https://raw.githubusercontent.com/asalber/estadística-practicas")
```

b. Mostrar el contenido del data frame.

💡 Solución 1

Con las funciones básicas de R.

```
df
```

```
# A tibble: 14 x 6
```

	nombre <chr>	edad <dbl>	sexo <chr>	peso <dbl>	altura <dbl>	colesterol <dbl>
1	José Luis Martínez Izquierdo	18	H	85	1.79	182
2	Rosa Díaz Díaz	32	M	65	1.73	232
3	Javier García Sánchez	24	H	NA	1.81	191
4	Carmen López Pinzón	35	M	65	1.7	200
5	Marisa López Collado	46	M	51	1.58	148
6	Antonio Ruiz Cruz	68	H	66	1.74	249
7	Antonio Fernández Ocaña	51	H	62	1.72	276
8	Pilar Martín González	22	M	60	1.66	NA
9	Pedro Gálvez Tenorio	35	H	90	1.94	241
10	Santiago Reillo Manzano	46	H	75	1.85	280
11	Macarena Álvarez Luna	53	M	55	1.62	262
12	José María de la Guía Sanz	58	H	78	1.87	198
13	Miguel Angel Cuadrado Gutiérrez	27	H	109	1.98	210
14	Carolina Rubio Moreno	20	M	61	1.77	194

💡 Solución 2

Con la función `glimpse` del paquete `dplyr`. Esta función muestra las columnas del data frame en filas, de manera que permite ver todas las columnas de un data frame cuando este tiene muchas columnas.

```
glimpse(df)
```

```
Rows: 14
Columns: 6
$ nombre    <chr> "José Luis Martínez Izquierdo", "Rosa Díaz Díaz", "Javier G~
$ edad      <dbl> 18, 32, 24, 35, 46, 68, 51, 22, 35, 46, 53, 58, 27, 20
$ sexo      <chr> "H", "M", "H", "M", "M", "H", "H", "M", "H", "H", "M", "H", ~
$ peso      <dbl> 85, 65, NA, 65, 51, 66, 62, 60, 90, 75, 55, 78, 109, 61
$ altura    <dbl> 1.79, 1.73, 1.81, 1.70, 1.58, 1.74, 1.72, 1.66, 1.94, 1.85, ~
$ colesterol <dbl> 182, 232, 191, 200, 148, 249, 276, NA, 241, 280, 262, 198, ~
```

- c. Crear una nueva columna con el índice de masa corporal, usando la siguiente fórmula

$$\text{IMC} = \frac{\text{Peso (kg)}}{\text{Altura (cm)}^2}$$

Solución

```
df <- df |>
  mutate(imc = round(peso/altura^2))
df
```

A tibble: 14 x 7

	nombre <chr>	edad <dbl>	sexo <chr>	peso <dbl>	altura <dbl>	colesterol <dbl>	imc <dbl>
1	José Luis Martínez Izquierdo	18	H	85	1.79	182	27
2	Rosa Díaz Díaz	32	M	65	1.73	232	22
3	Javier García Sánchez	24	H	NA	1.81	191	NA
4	Carmen López Pinzón	35	M	65	1.7	200	22
5	Marisa López Collado	46	M	51	1.58	148	20
6	Antonio Ruiz Cruz	68	H	66	1.74	249	22
7	Antonio Fernández Ocaña	51	H	62	1.72	276	21
8	Pilar Martín González	22	M	60	1.66	NA	22
9	Pedro Gálvez Tenorio	35	H	90	1.94	241	24
10	Santiago Reillo Manzano	46	H	75	1.85	280	22
11	Macarena Álvarez Luna	53	M	55	1.62	262	21
12	José María de la Guía Sanz	58	H	78	1.87	198	22
13	Miguel Angel Cuadrado Gutiérrez	27	H	109	1.98	210	28
14	Carolina Rubio Moreno	20	M	61	1.77	194	19

- d. Crear una nueva columna con la variable `obesidad` recodificando la columna `imc` en las siguientes categorías.

Rango IMC	Categoría
Menor de 18.5	Bajo peso
De 18.5 a 24.5	Saludable
De 24.5 a 30	Sobrepeso
Mayor de 30	Obeso

Solución

```
df <- df |>
  mutate(Obesidad = cut(imc, breaks = c(0, 18.5, 24.5, 30, Inf), labels = c("Bajo peso", "Saludable", "Sobrepeso", "Obeso")))
df
```

A tibble: 14 x 8

	nombre <chr>	edad <dbl>	sexo <chr>	peso <dbl>	altura <dbl>	colesterol <dbl>	imc <dbl>	Obesidad <fct>
1	José Luis Martínez Izquierdo	18	H	85	1.79	182	27	Sobrepeso
2	Rosa Díaz Díaz	32	M	65	1.73	232	22	Saludable
3	Javier García Sánchez	24	H	NA	1.81	191	NA	<NA>
4	Carmen López Pinzón	35	M	65	1.7	200	22	Saludable
5	Marisa López Collado	46	M	51	1.58	148	20	Saludable
6	Antonio Ruiz Cruz	68	H	66	1.74	249	22	Saludable
7	Antonio Fernández Ocaña	51	H	62	1.72	276	21	Saludable
8	Pilar Martín González	22	M	60	1.66	NA	22	Saludable
9	Pedro Gálvez Tenorio	35	H	90	1.94	241	24	Saludable
10	Santiago Reillo Manzano	46	H	75	1.85	280	22	Saludable
11	Macarena Álvarez Luna	53	M	55	1.62	262	21	Saludable
12	José María de la Guía Sanz	58	H	78	1.87	198	22	Saludable
13	Miguel Angel Cuadrado Gutierrez	27	H	109	1.98	210	28	Sobrepeso
14	Carolina Rubio Moreno	20	M	61	1.77	194	19	Saludable

- e. Seleccionar las columnas `nombre`, `sexo` y `edad`.

Solución

```
df |>
  select(nombre, sexo, edad)
```

A tibble: 14 x 3

	nombre	sexo	edad
--	--------	------	------

	<chr>	<chr>	<dbl>
1	José Luis Martínez Izquierdo	H	18
2	Rosa Díaz Díaz	M	32
3	Javier García Sánchez	H	24
4	Carmen López Pinzón	M	35
5	Marisa López Collado	M	46
6	Antonio Ruiz Cruz	H	68
7	Antonio Fernández Ocaña	H	51
8	Pilar Martín González	M	22
9	Pedro Gálvez Tenorio	H	35
10	Santiago Reillo Manzano	H	46
11	Macarena Álvarez Luna	M	53
12	José María de la Guía Sanz	H	58
13	Miguel Angel Cuadrado Gutiérrez	H	27
14	Carolina Rubio Moreno	M	20

f. Anonimizar los datos eliminando la columna **nombre**.

Solución

```
df |>
  select(-nombre)
```

A tibble: 14 x 7

	edad	sexo	peso	altura	colesterol	imc	Obesidad
	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<fct>
1	18	H	85	1.79	182	27	Sobrepeso
2	32	M	65	1.73	232	22	Saludable
3	24	H	NA	1.81	191	NA	<NA>
4	35	M	65	1.7	200	22	Saludable
5	46	M	51	1.58	148	20	Saludable
6	68	H	66	1.74	249	22	Saludable
7	51	H	62	1.72	276	21	Saludable
8	22	M	60	1.66	NA	22	Saludable
9	35	H	90	1.94	241	24	Saludable
10	46	H	75	1.85	280	22	Saludable
11	53	M	55	1.62	262	21	Saludable
12	58	H	78	1.87	198	22	Saludable
13	27	H	109	1.98	210	28	Sobrepeso
14	20	M	61	1.77	194	19	Saludable

g. Reordenar las columnas poniendo la columna **sexo** antes que la columna **edad**.

💡 Solución

```
df |>
  select(nombre, sexo, edad, everything())
```

A tibble: 14 x 8

	nombre <chr>	sexo <chr>	edad <dbl>	peso <dbl>	altura <dbl>	colesterol <dbl>	imc <dbl>	Obesidad <fct>
1	José Luis Martínez Izquier~	H	18	85	1.79	182	27	Sobrepe~
2	Rosa Díaz Díaz	M	32	65	1.73	232	22	Saludab~
3	Javier García Sánchez	H	24	NA	1.81	191	NA	<NA>
4	Carmen López Pinzón	M	35	65	1.7	200	22	Saludab~
5	Marisa López Collado	M	46	51	1.58	148	20	Saludab~
6	Antonio Ruiz Cruz	H	68	66	1.74	249	22	Saludab~
7	Antonio Fernández Ocaña	H	51	62	1.72	276	21	Saludab~
8	Pilar Martín González	M	22	60	1.66	NA	22	Saludab~
9	Pedro Gálvez Tenorio	H	35	90	1.94	241	24	Saludab~
10	Santiago Reillo Manzano	H	46	75	1.85	280	22	Saludab~
11	Macarena Álvarez Luna	M	53	55	1.62	262	21	Saludab~
12	José María de la Guía Sanz	H	58	78	1.87	198	22	Saludab~
13	Miguel Angel Cuadrado Gut~	H	27	109	1.98	210	28	Sobrepe~
14	Carolina Rubio Moreno	M	20	61	1.77	194	19	Saludab~

- h. Filtrar el data frame para quedarse con las mujeres.

💡 Solución

```
df |>
  filter(sexo == "M")
```

A tibble: 6 x 8

	nombre <chr>	edad <dbl>	sexo <chr>	peso <dbl>	altura <dbl>	colesterol <dbl>	imc <dbl>	Obesidad <fct>
1	Rosa Díaz Díaz	32	M	65	1.73	232	22	Saludable
2	Carmen López Pinzón	35	M	65	1.7	200	22	Saludable
3	Marisa López Collado	46	M	51	1.58	148	20	Saludable
4	Pilar Martín González	22	M	60	1.66	NA	22	Saludable
5	Macarena Álvarez Luna	53	M	55	1.62	262	21	Saludable
6	Carolina Rubio Moreno	20	M	61	1.77	194	19	Saludable

- i. Filtrar el data frame para quedarse con los hombres mayores de 30 años.

Solución

```
df |>
  filter( sexo == "H" & edad > 30)
```

A tibble: 5 x 8

	nombre	edad	sexo	peso	altura	colesterol	imc	Obesidad
	<chr>	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<fct>
1	Antonio Ruiz Cruz	68	H	66	1.74	249	22	Saludable
2	Antonio Fernández Ocaña	51	H	62	1.72	276	21	Saludable
3	Pedro Gálvez Tenorio	35	H	90	1.94	241	24	Saludable
4	Santiago Reillo Manzano	46	H	75	1.85	280	22	Saludable
5	José María de la Guía Sanz	58	H	78	1.87	198	22	Saludable

- j. Filtrar el data frame para eliminar las filas con datos perdidos en la columna colesterol.

Solución

```
df |>
  filter(!is.na(colesterol))
```

A tibble: 13 x 8

	nombre	edad	sexo	peso	altura	colesterol	imc	Obesidad
	<chr>	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<fct>
1	José Luis Martínez Izquier~	18	H	85	1.79	182	27	Sobrepe~
2	Rosa Díaz Díaz	32	M	65	1.73	232	22	Saludab~
3	Javier García Sánchez	24	H	NA	1.81	191	NA	<NA>
4	Carmen López Pinzón	35	M	65	1.7	200	22	Saludab~
5	Marisa López Collado	46	M	51	1.58	148	20	Saludab~
6	Antonio Ruiz Cruz	68	H	66	1.74	249	22	Saludab~
7	Antonio Fernández Ocaña	51	H	62	1.72	276	21	Saludab~
8	Pedro Gálvez Tenorio	35	H	90	1.94	241	24	Saludab~
9	Santiago Reillo Manzano	46	H	75	1.85	280	22	Saludab~
10	Macarena Álvarez Luna	53	M	55	1.62	262	21	Saludab~
11	José María de la Guía Sanz	58	H	78	1.87	198	22	Saludab~
12	Miguel Angel Cuadrado Gut~	27	H	109	1.98	210	28	Sobrepe~
13	Carolina Rubio Moreno	20	M	61	1.77	194	19	Saludab~

- k. Ordenar el data frame según la columna nombre.

💡 Solución

```
df |>
  arrange(nombre)
```

A tibble: 14 x 8

	nombre <chr>	edad <dbl>	sexo <chr>	peso <dbl>	altura <dbl>	colesterol <dbl>	imc <dbl>	Obesidad <fct>
1	Antonio Fernández Ocaña	51	H	62	1.72	276	21	Saludab~
2	Antonio Ruiz Cruz	68	H	66	1.74	249	22	Saludab~
3	Carmen López Pinzón	35	M	65	1.7	200	22	Saludab~
4	Carolina Rubio Moreno	20	M	61	1.77	194	19	Saludab~
5	Javier García Sánchez	24	H	NA	1.81	191	NA	<NA>
6	José Luis Martínez Izquie~	18	H	85	1.79	182	27	Sobrepe~
7	José María de la Guía Sanz	58	H	78	1.87	198	22	Saludab~
8	Macarena Álvarez Luna	53	M	55	1.62	262	21	Saludab~
9	Marisa López Collado	46	M	51	1.58	148	20	Saludab~
10	Miguel Angel Cuadrado Gut~	27	H	109	1.98	210	28	Sobrepe~
11	Pedro Gálvez Tenorio	35	H	90	1.94	241	24	Saludab~
12	Pilar Martín González	22	M	60	1.66	NA	22	Saludab~
13	Rosa Díaz Díaz	32	M	65	1.73	232	22	Saludab~
14	Santiago Reillo Manzano	46	H	75	1.85	280	22	Saludab~

1. Ordenar el data frame ascendentemente por la columna **sexo** y descendentemente por la columna **edad**.

💡 Solución

```
df |>
  arrange(sexo, desc(edad))
```


A tibble: 14 x 8

	nombre <chr>	edad <dbl>	sexo <chr>	peso <dbl>	altura <dbl>	colesterol <dbl>	imc <dbl>	Obesidad <fct>
1	Antonio Ruiz Cruz	68	H	66	1.74	249	22	Saludab~
2	José María de la Guía Sanz	58	H	78	1.87	198	22	Saludab~
3	Antonio Fernández Ocaña	51	H	62	1.72	276	21	Saludab~
4	Santiago Reillo Manzano	46	H	75	1.85	280	22	Saludab~
5	Pedro Gálvez Tenorio	35	H	90	1.94	241	24	Saludab~
6	Miguel Angel Cuadrado Gut~	27	H	109	1.98	210	28	Sobrepe~
7	Javier García Sánchez	24	H	NA	1.81	191	NA	<NA>
8	José Luis Martínez Izquie~	18	H	85	1.79	182	27	Sobrepe~

9	Macarena Álvarez Luna	53	M	55	1.62	262	21	Saludab~
10	Marisa López Collado	46	M	51	1.58	148	20	Saludab~
11	Carmen López Pinzón	35	M	65	1.7	200	22	Saludab~
12	Rosa Díaz Díaz	32	M	65	1.73	232	22	Saludab~
13	Pilar Martín González	22	M	60	1.66	NA	22	Saludab~
14	Carolina Rubio Moreno	20	M	61	1.77	194	19	Saludab~

Ejercicio 2.3. El fichero `notas-curso2.csv` contiene las notas de las asignaturas de un curso en varios grupos de alumnos.

- a. Crear un data frame con los datos del curso a partir del fichero `notas-curso2.csv`.

 Solución

```
df <- read_csv("https://raw.githubusercontent.com/asalber/estadistica-practicas/master/data/notas-curso2.csv")
df

# A tibble: 120 x 9
  sexo    turno grupo trabaja notaA notaB notaC notaD notaE
<chr> <chr> <chr> <chr>   <dbl> <dbl> <dbl> <dbl> <dbl>
1 Mujer  Tarde   C      N      5.2    6.3    3.4    2.3    2
2 Hombre Mañana  A      N      5.7    5.7    4.2    3.5    2.7
3 Hombre Mañana  B      N      8.3    8.8    8.8     8     5.5
4 Hombre Mañana  B      N      6.1    6.8     4     3.5    2.2
5 Hombre Mañana  A      N      6.2     9     5     4.4    3.7
6 Hombre Mañana  A      S      8.6    8.9    9.5    8.4    3.9
7 Mujer  Mañana  A      N      6.7    7.9    5.6    4.8    4.2
8 Mujer  Tarde   C      S      4.1    5.2    1.7    0.3     1
9 Hombre Tarde   C      N      5      5      3.3    2.7     6
10 Hombre Tarde   C      N      5.3    6.3    4.8    3.6    2.3
# i 110 more rows
```

- b. Convertir el data frame a formato largo.

 Solución

```
df_largo <- df |> pivot_longer(notasA:notaE, names_to = "Asignatura", values_to = "Nota")
df_largo

# A tibble: 600 x 6
  sexo    turno grupo trabaja Asignatura  Nota
<chr> <chr> <chr> <chr>   <chr>     <dbl>
1 Mujer  Tarde   C      N      5.2    6.3    3.4    2.3    2
2 Hombre Mañana  A      N      5.7    5.7    4.2    3.5    2.7
3 Hombre Mañana  B      N      8.3    8.8    8.8     8     5.5
4 Hombre Mañana  B      N      6.1    6.8     4     3.5    2.2
5 Hombre Mañana  A      N      6.2     9     5     4.4    3.7
6 Hombre Mañana  A      S      8.6    8.9    9.5    8.4    3.9
7 Mujer  Mañana  A      N      6.7    7.9    5.6    4.8    4.2
8 Mujer  Tarde   C      S      4.1    5.2    1.7    0.3     1
9 Hombre Tarde   C      N      5      5      3.3    2.7     6
10 Hombre Tarde   C      N      5.3    6.3    4.8    3.6    2.3
# i 590 more rows
```

```

1 Mujer Tarde C N notaA 5.2
2 Mujer Tarde C N notaB 6.3
3 Mujer Tarde C N notaC 3.4
4 Mujer Tarde C N notaD 2.3
5 Mujer Tarde C N notaE 2
6 Hombre Mañana A N notaA 5.7
7 Hombre Mañana A N notaB 5.7
8 Hombre Mañana A N notaC 4.2
9 Hombre Mañana A N notaD 3.5
10 Hombre Mañana A N notaE 2.7
# i 590 more rows

```

- c. Crear una nueva columna con la variable `calificación` que contenga las calificaciones de cada asignatura.

💡 Solución

```

df_largo <- df_largo |>
  mutate(Calificación = cut(Nota, breaks = c(0, 4.99, 6.99, 8.99, 10), labels =
df_largo

```

A tibble: 600 x 7

	sexo	turno	grupo	trabaja	Asignatura	Nota	Calificación
	<chr>	<chr>	<chr>	<chr>	<chr>	<dbl>	<fct>
1	Mujer	Tarde	C	N	notaA	5.2	AP
2	Mujer	Tarde	C	N	notaB	6.3	AP
3	Mujer	Tarde	C	N	notaC	3.4	SS
4	Mujer	Tarde	C	N	notaD	2.3	SS
5	Mujer	Tarde	C	N	notaE	2	SS
6	Hombre	Mañana	A	N	notaA	5.7	AP
7	Hombre	Mañana	A	N	notaB	5.7	AP
8	Hombre	Mañana	A	N	notaC	4.2	SS
9	Hombre	Mañana	A	N	notaD	3.5	SS
10	Hombre	Mañana	A	N	notaE	2.7	SS

i 590 more rows

- d. Filtrar el conjunto de datos para obtener las asignaturas y las notas de las mujeres del grupo A, ordenadas de mayor a menor.

Solución

```
df_largo |>
  filter(sexo == "Mujer", grupo == "A") |>
  select(Asignatura, Nota) |>
  arrange(desc(Nota))

# A tibble: 75 x 2
  Asignatura  Nota
  <chr>      <dbl>
1 notaB      9.2
2 notaE      9.2
3 notaB      8.8
4 notaB      8.6
5 notaB      8.6
6 notaA      8.3
7 notaB      8.2
8 notaB      8.1
9 notaA       8
10 notaB      8
# i 65 more rows
```

2.2 Ejercicios Propuestos

Ejercicio 2.4. La siguiente tabla recoge las notas de los alumnos de un curso con dos asignaturas.

Alumno	Sexo	Física	Química
Carlos	H	6.7	8.1
María	M	7.2	9.5
Carmen	M	5.5	5
Pedro	H		4.5
Luis	H	3.5	5
Sara	M	6.2	4

- Definir cuatro vectores con el nombre, el sexo y las notas de Física y Química.

💡 Solución

```
nombre <- c("Carlos", "María", "Carmen", "Pedro", "Luis", "Sara")
sexo <- c("H", "M", "M", "H", "H", "M")
fisica <- c(6.7, 7.2, 5.5, NA, 3.5, 6.2)
quimica <- c(8.1, 9.5, 5, 4.5, 5, 4)
```

- b. Convertir el sexo en un factor y mostrar sus niveles.

💡 Solución

```
sexo <- factor(sexo)
levels(sexo)
```

```
[1] "H" "M"
```

- c. Crear un data frame con el nombre, sexo y las notas de Física y Química.

💡 Solución

```
df <- data.frame(nombre, sexo, fisica, quimica)
df
```

	nombre	sexo	fisica	quimica
1	Carlos	H	6.7	8.1
2	María	M	7.2	9.5
3	Carmen	M	5.5	5.0
4	Pedro	H	NA	4.5
5	Luis	H	3.5	5.0
6	Sara	M	6.2	4.0

- d. Crear una nueva columna con la nota media de Física y Química.

💡 Solución

```
df$media <- (df$fisica + df$quimica) / 2
df
```

	nombre	sexo	fisica	quimica	media
1	Carlos	H	6.7	8.1	7.40
2	María	M	7.2	9.5	8.35

3	Carmen	M	5.5	5.0	5.25
4	Pedro	H	NA	4.5	NA
5	Luis	H	3.5	5.0	4.25
6	Sara	M	6.2	4.0	5.10

- e. Crear una nueva columna booleana **aprobado** que tenga el valor **TRUE** si la media es mayor o igual que 5 y **FALSE** en caso contrario.

Solución

```
df$aprobado <- df$media >= 5
df
```

	nombre	sexo	fisica	quimica	media	aprobado
1	Carlos	H	6.7	8.1	7.40	TRUE
2	María	M	7.2	9.5	8.35	TRUE
3	Carmen	M	5.5	5.0	5.25	TRUE
4	Pedro	H	NA	4.5	NA	NA
5	Luis	H	3.5	5.0	4.25	FALSE
6	Sara	M	6.2	4.0	5.10	TRUE

- f. Filtrar el data frame para quedarse con el nombre y la media de las mujeres que han aprobado.

Solución

```
df[df$sexo == "M" & df$media >= 5, c("nombre", "media")]
```

	nombre	media
2	María	8.35
3	Carmen	5.25
6	Sara	5.10

Ejercicio 2.5. Se ha diseñado un ensayo clínico aleatorizado, doble-ciego y controlado con placebo, para estudiar el efecto de dos alternativas terapéuticas en el control de la hipertensión arterial. Se han reclutado 100 pacientes hipertensos y estos han sido distribuidos aleatoriamente en tres grupos de tratamiento. A uno de los grupos (control) se le administró un placebo, a otro grupo se le administró un inhibidor de la enzima convertidora de la angiotensina (IECA) y al otro un tratamiento combinado de un diurético y un Antagonista del Calcio. Las variables respuesta final fueron las presiones arteriales sistólica y diastólica.

Los datos con las claves de aleatorización han sido introducidos en una base de datos que

reside en la central de aleatorización, mientras que los datos clínicos han sido archivados en dos archivos distintos, uno para cada uno de los dos centros participantes en el estudio.

Las variables almacenadas en estos archivos clínicos son las siguientes:

- CLAVE: Clave de aleatorización
- NOMBRE: Iniciales del paciente
- F_NACIM: Fecha de Nacimiento
- F_INCLUS: Fecha de inclusión
- SEXO: Sexo (0: Hombre 1: Mujer)
- ALTURA: Altura en cm.
- PESO: Peso en Kg.
- PAD_INI: Presión diastólica basal (inicial)
- PAD_FIN: Presión diastólica final
- PAS_INI: Presión sistólica basal (inicial)
- PAS_FIN: Presión sistólica final

El archivo de claves de aleatorización contiene sólo dos variables.

- CLAVE: Clave de aleatorización
 - FARMACO: Fármaco administrado (0: Placebo, 1: IECA, 2:Ca Antagonista + diurético)
- a. Crear un data frame con los datos de los pacientes del hospital A (fichero de csv [datos-hospital-a.csv](#)).
 - b. Crear un data frame con los datos de los pacientes del hospital B (fichero csv [datos-hospital-b.csv](#)).
 - c. Fusionar los datos de los dos hospitales en un nuevo data frame.

i Ayuda

Para fusionar las filas de dos data frames con las mismas columnas usar la función `rbind`.

- d. Crear un data frame con los datos de las claves de aleatorización (fichero csv [claves-aleatorizacion.csv](#))
- e. Fusionar el data frame con los datos clínicos y el data frame con claves de aleatorización en un nuevo data frame.

i Ayuda

Para fusionar las columnas de dos data frames usando una misma columna como clave en ambos data frames usar la función `left_join` del paquete `dplyr`.

- f. Convertir en un factor el sexo.
- g. Crear una nueva columna para la evolución de la presión arterial diastólica restando las columnas PAS_FIN y PAS_FIN.

3 Distribuciones de frecuencias y representaciones gráficas

3.1 Ejercicios Resueltos

Para la realización de esta práctica se requieren los siguientes paquetes:

```
library(tidyverse)
# Incluye los siguientes paquetes:
# - readr: para la lectura de archivos csv.
# - dplyr: para el preprocesamiento y manipulación de datos.
# - ggplot2: para la representación gráfica.
library(knitr) # para el formateo de tablas.
library(kableExtra) # para personalizar el formato de las tablas.
```

Ejercicio 3.1. En una encuesta a 25 matrimonios sobre el número de hijos que tenían se obtuvieron los siguientes datos:

1, 2, 4, 2, 2, 2, 3, 2, 1, 1, 0, 2, 2, 0, 2, 2, 1, 2, 2, 3, 1, 2, 2, 1, 2

- a. Crear un conjunto de datos con la variable `hijos`.

💡 Solución

```
df <- data.frame(hijos = c(1, 2, 4, 2, 2, 2, 3, 2, 1, 1, 0, 2, 2, 0, 2, 2, 1, 2, 2, 3, 1, 2, 2, 1, 2))
```

- b. Construir la tabla de frecuencias.

💡 Solución 1

Para obtener las frecuencias absolutas se puede usar la función `table`, y para las frecuencias relativas la función `prop.table` ambas del paquete base de R.

```
# Frecuencias absolutas.
ni <- table(df$hijos)
# Frecuencias relativas
fi <- prop.table(ni)
# Frecuencias acumuladas.
Ni <- cumsum(ni)
# Frecuencias relativas acumuladas.
Fi <- cumsum(fi)
# Creación de un data frame con las frecuencias.
tabla_frec <- cbind(ni, fi, Ni, Fi)
tabla_frec
```

```
ni  fi Ni  Fi
0  2 0.08  2 0.08
1  6 0.24  8 0.32
2 14 0.56 22 0.88
3  2 0.08 24 0.96
4  1 0.04 25 1.00
```

💡 Solución 2

Otra alternativa es usar la función `count` del paquete `dplyr`.

```
library(dplyr)
library(knitr)
library(kableExtra)
count(df, hijos) |>
  mutate(fi = n/sum(n), Ni = cumsum(n), Fi = cumsum(n)/sum(n)) |>
  kable()
```

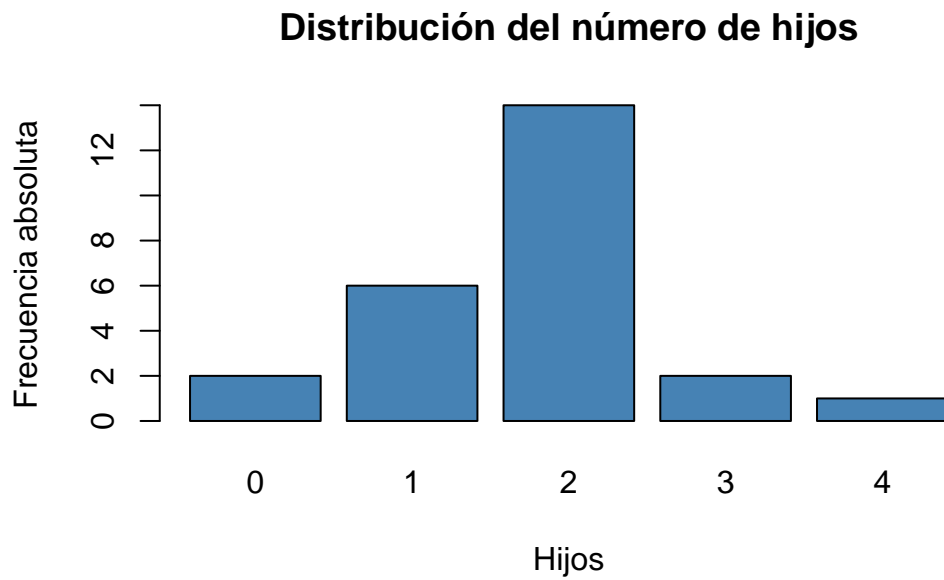
hijos	n	fi	Ni	Fi
0	2	0.08	2	0.08
1	6	0.24	8	0.32
2	14	0.56	22	0.88
3	2	0.08	24	0.96
4	1	0.04	25	1.00

- c. Dibujar el diagrama de barras de las frecuencias absolutas, relativas, absolutas acumuladas y relativas acumuladas.

💡 Solución 1

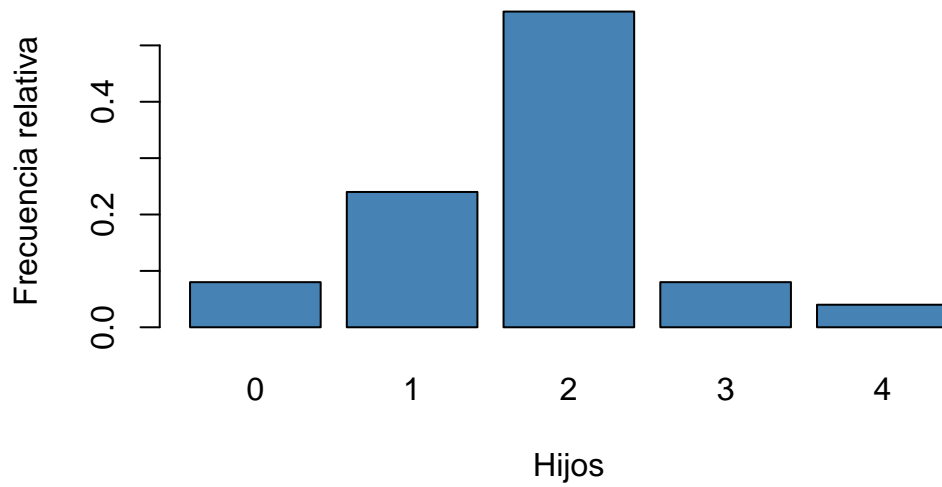
Para dibujar un diagrama de barras se puede usar la función `barplot` del paquete `graphics`.

```
# Diagrama de barras de frecuencias absolutas.  
barplot(ni, col = "steelblue", main="Distribución del número de hijos", xlab="Hijos",
```



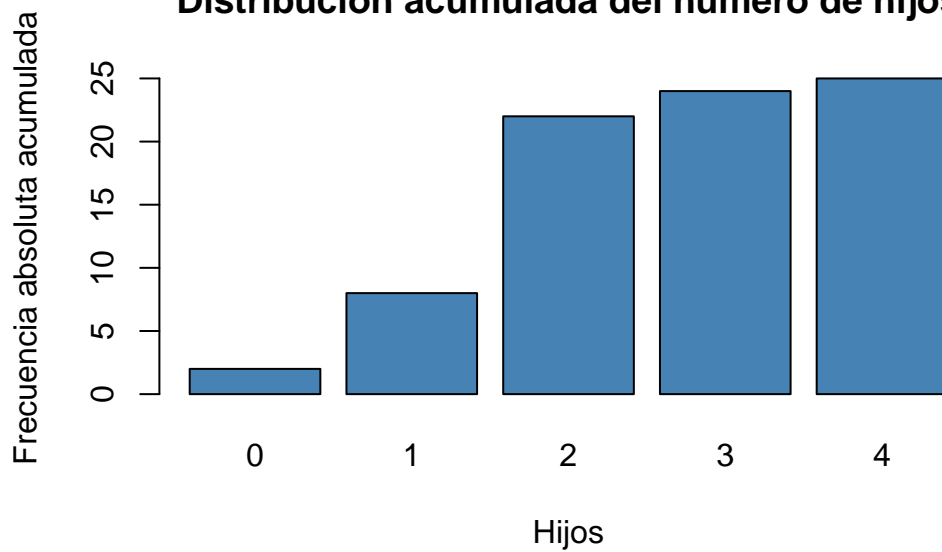
```
# Diagrama de barras de frecuencias relativas.  
barplot(fi, col = "steelblue", main="Distribución del número de hijos", xlab="Hijos",
```

Distribución del número de hijos

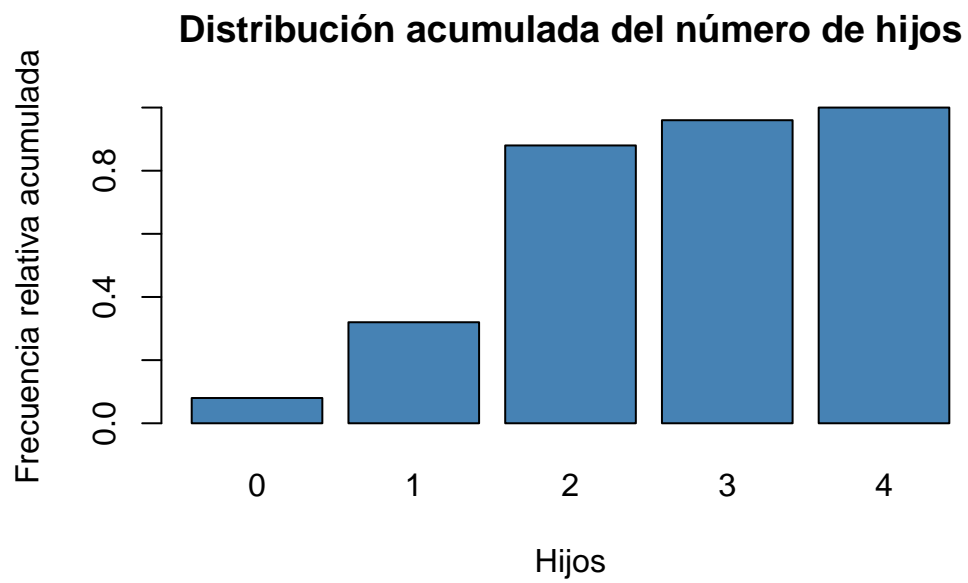


```
# Diagrama de barras de frecuencias absolutas acumuladas.  
barplot(Ni, col = "steelblue", main="Distribución acumulada del número de hijos")
```

Distribución acumulada del número de hijos



```
# Diagrama de barras de frecuencias relativas acumuladas.  
barplot(Fi, col = "steelblue", main="Distribución acumulada del número de hijos")
```

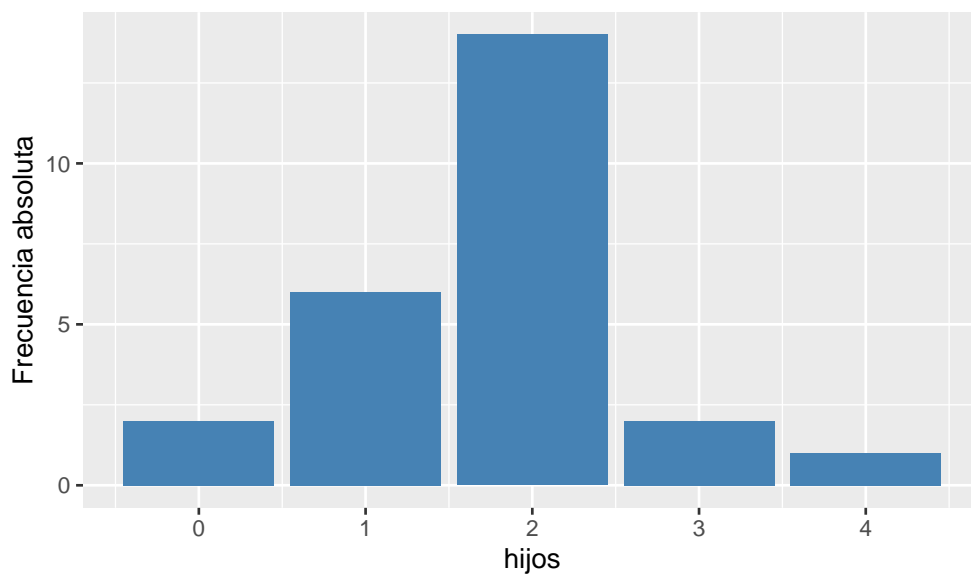


💡 Solución 2

Otra alternativa es usar la función la función `geom_bar` del paquete `ggplot2`.

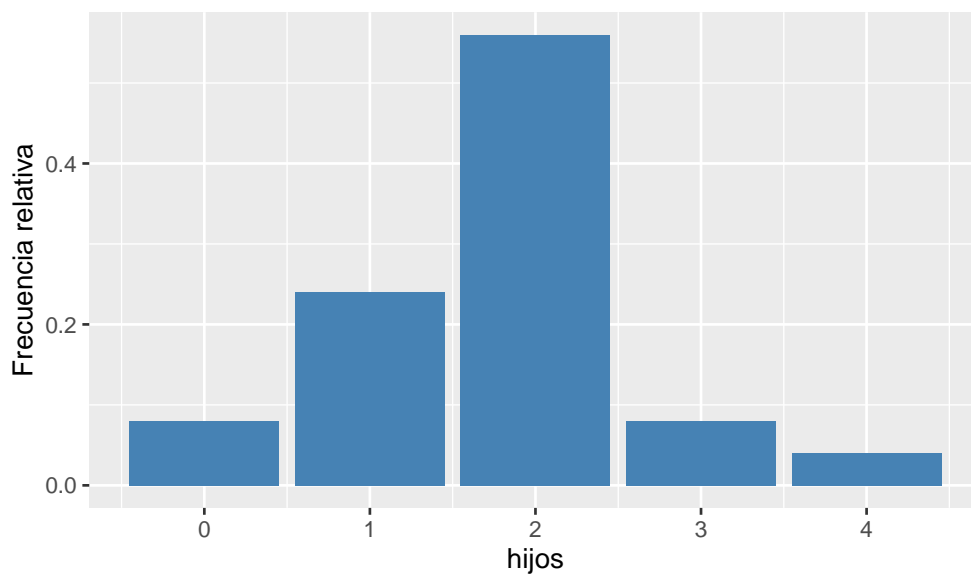
```
library(ggplot2)
# Diagrama de barras de frecuencias absolutas
ggplot(df, aes(x = hijos)) +
  geom_bar(fill = "steelblue") +
  labs(title = "Distribución del número de hijos", y = "Frecuencia absoluta")
```

Distribución del número de hijos

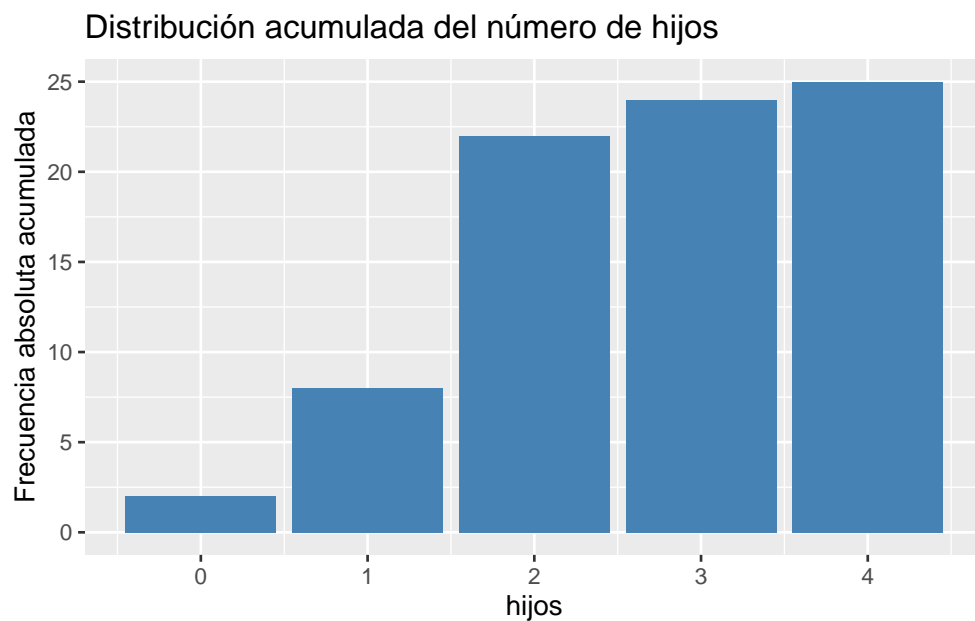


```
# Diagrama de barras de frecuencias relativas
ggplot(df, aes(x = hijos)) +
  geom_bar(aes(y = after_stat(count/sum(count))), fill = "steelblue") +
  labs(title = "Distribución del número de hijos", y = "Frecuencia relativa")
```

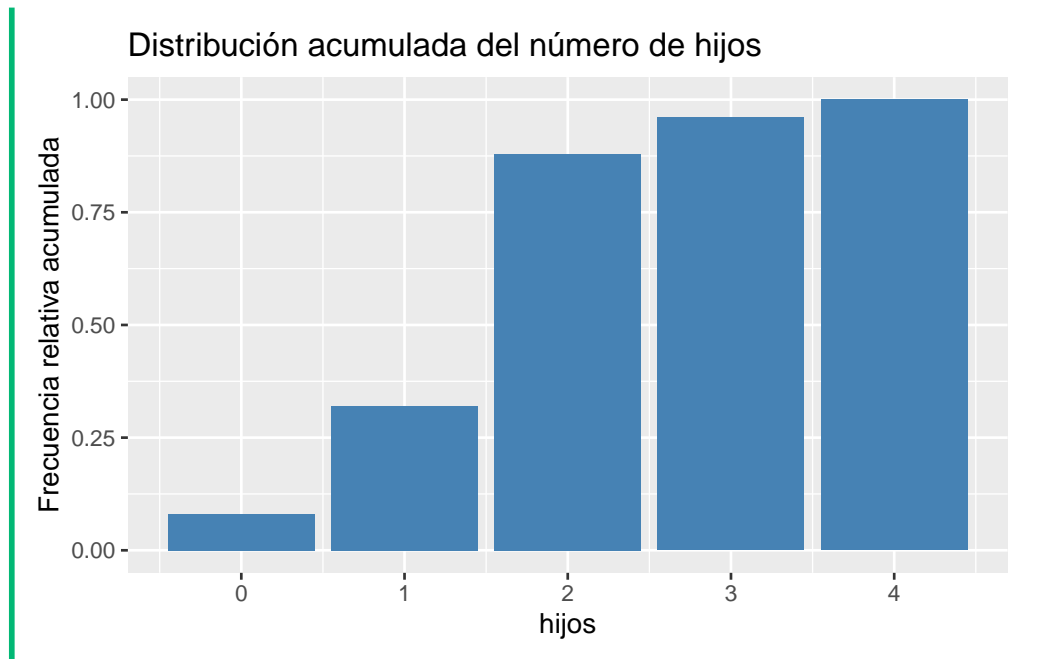
Distribución del número de hijos



```
# Diagrama de barras de frecuencias acumuladas
ggplot(df, aes(x = hijos)) +
  geom_bar(aes(y = after_stat(cumsum(count))), fill = "steelblue") +
  labs(title = "Distribución acumulada del número de hijos", y = "Frecuencia a
```



```
# Diagrama de barras de frecuencias acumuladas
ggplot(df, aes(x = hijos)) +
  geom_bar(aes(y = after_stat(cumsum(count)/sum(count))), fill = "steelblue") +
  labs(title = "Distribución acumulada del número de hijos", y = "Frecuencia a
```

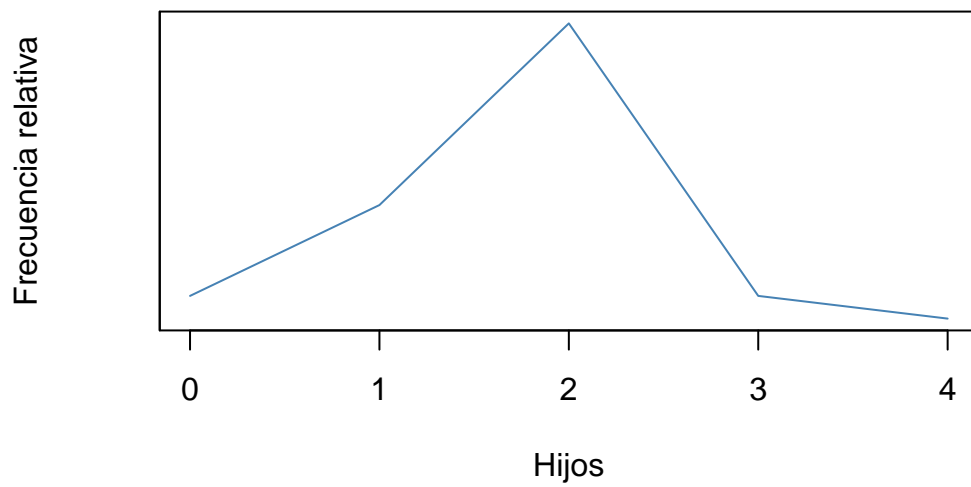
d. Dibujar el polígono de frecuencias relativas.

💡 Solución 1

Para dibujar un diagrama de líneas se puede usar la función `plot` del paquete `graphics`.

```
# Frecuencias relativas.  
plot(names(fi), fi, type = "l", col = "steelblue", main="Distribución del número de hijos")
```

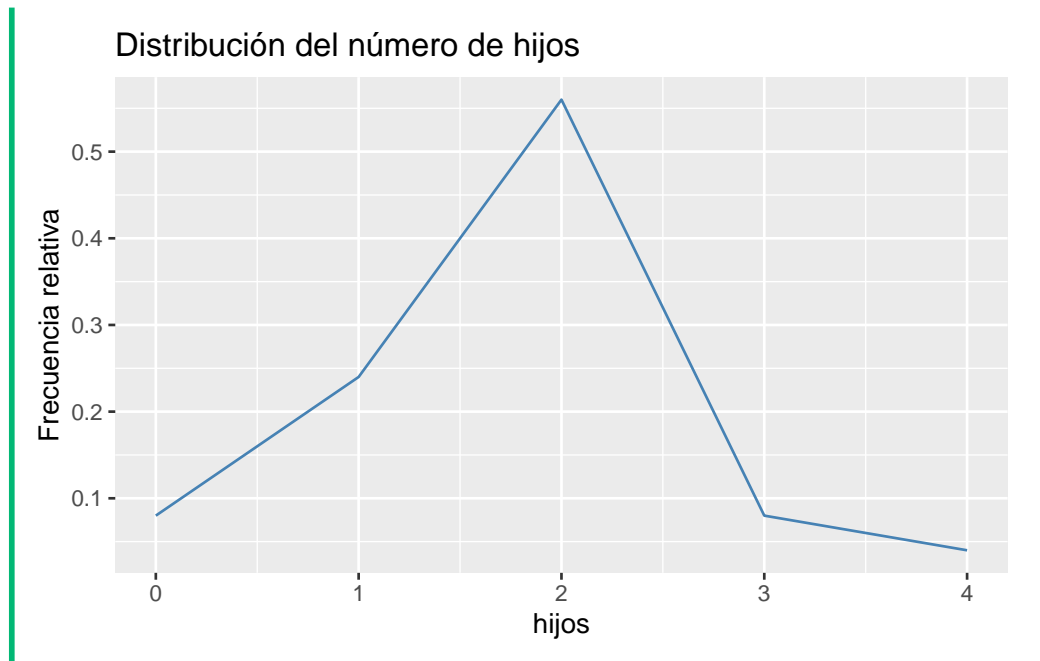
Distribución del número de hijos



💡 Solución 2

Otra alternativa es usar la función la función `geom_line` del paquete `ggplot2`.

```
library(ggplot2)
count(df, hijos) |>
  mutate(fi = n/sum(n)) |>
  ggplot(aes(x=hijos, y=fi)) +
  geom_line(col = "steelblue") +
  labs(title = "Distribución del número de hijos", y = "Frecuencia relativa")
```



Ejercicio 3.2. En un servicio de atención al cliente se han registrado el número de llamadas de clientes cada día del mes de noviembre, obteniendo los siguientes datos:

15, 23, 12, 10, 28, 50, 12, 17, 20, 21, 18, 13, 11, 12, 26, 30, 6, 16, 19, 22, 14, 17, 21, 28, 9, 16, 13, 11, 16, 20

- a. Crear un conjunto de datos con la variable `llamadas`.

💡 Solución

```
df <- data.frame(llamadas = c(15, 23, 12, 10, 28, 50, 12, 17, 20, 21, 18, 13, 11, 12, 26, 30, 6, 16, 19, 22, 14, 17, 21, 28, 9, 16, 13, 11, 16, 20))
```

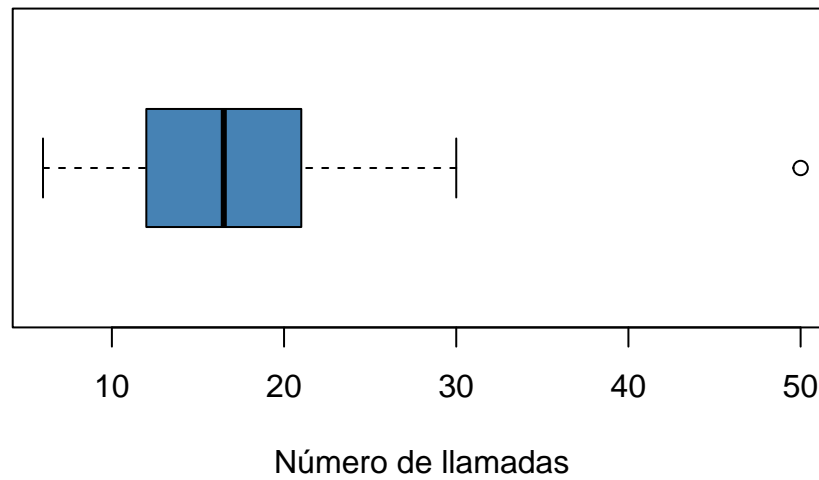
- b. Dibujar el diagrama de cajas. ¿Existe algún dato atípico? En el caso de que exista, eliminarlo y proceder con los siguientes apartados.

💡 Solución 1

Para dibujar un diagrama de cajas se puede usar la función `boxplot` del paquete `graphics`.

```
# Frecuencias relativas.
boxplot(df$llamadas, col = "steelblue", main="Distribución del número de llamadas")
```

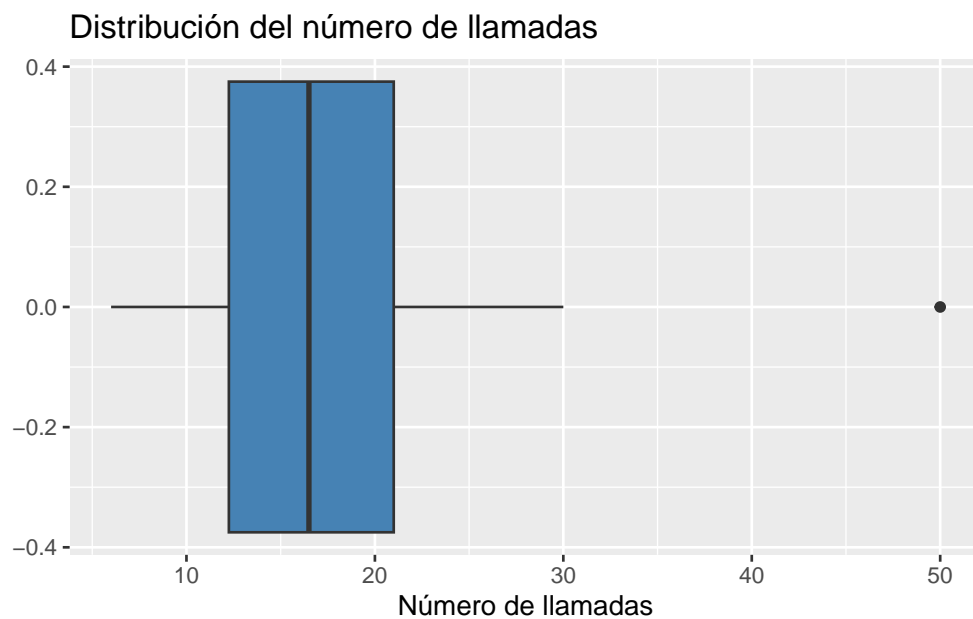
Distribución del número de llamadas



💡 Solución 2

Otra alternativa es usar la función la función `geom_boxplot` del paquete `ggplot2`.

```
library(ggplot2)
ggplot(df, aes(x = llamadas)) +
  geom_boxplot(fill = "steelblue") +
  labs(title = "Distribución del número de llamadas", x = "Número de llamadas")
```



Hay un día con 50 llamadas, que es un valor atípico en comparación con el resto de días.

💡 Solución

La función `cut`

```
# Eliminación del dato atípico.  
df <- df[df$llamadas != 50, , drop = F]
```

- c. Construir la tabla de frecuencias agrupando en 5 clases.

💡 Solución 1

Para agrupar los datos en intervalos se puede utilizar la función `cut` del paquete base de R, y para contar las frecuencias absolutas y relativas las funciones `table`, y `prop.table` respectivamente.

```
# Frecuencias absolutas. Creación automática de 5 clases con intervalos cerrados.
```

```

ni <- table(cut(df$llamadas, breaks = 5, right = F))
# Creación manual de 5 clases.
ni <- table(cut(df$llamadas, breaks = seq(5, 30, 5)))
# Frecuencias relativas
fi <- prop.table(ni)
# Frecuencias acumuladas.
Ni <- cumsum(ni)
# Frecuencias relativas acumuladas.
Fi <- cumsum(fi)
# Creación de un data frame con las frecuencias.
tabla_frec <- cbind(ni, fi, Ni, Fi)
tabla_frec

```

	ni	fi	Ni	Fi
(5,10]	3	0.1034483	3	0.1034483
(10,15]	9	0.3103448	12	0.4137931
(15,20]	9	0.3103448	21	0.7241379
(20,25]	4	0.1379310	25	0.8620690
(25,30]	4	0.1379310	29	1.0000000

💡 Solución 2

Otra alternativa es usar la función `count` del paquete `dplyr`.

```

library(dplyr)
library(knitr)
library(kableExtra)
mutate(df, llamadas_int = cut(llamadas, breaks = seq(5, 30, 5))) |>
  count(llamadas_int) |>
  mutate(fi = n/sum(n), Ni = cumsum(n), Fi = cumsum(n)/sum(n)) |>
  kable()

```

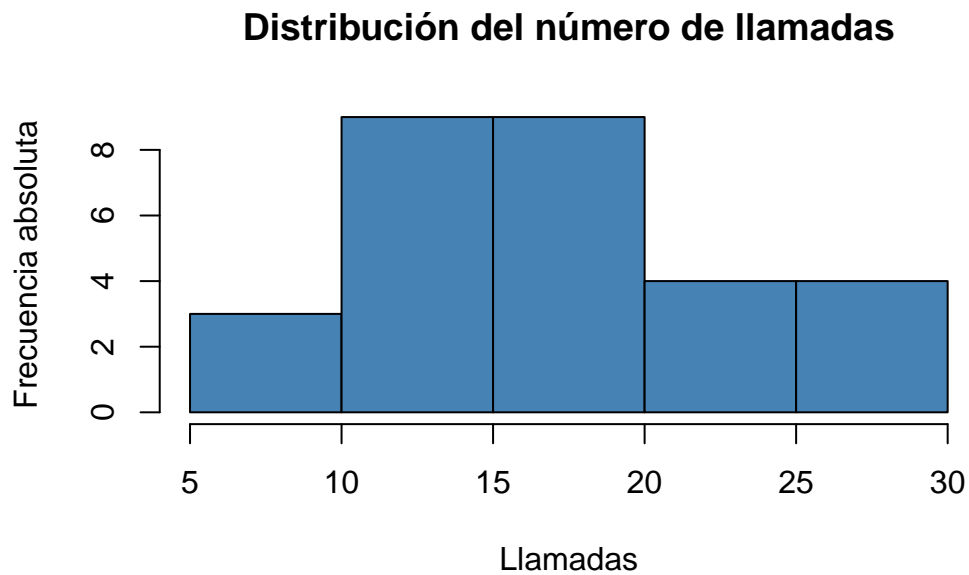
llamadas_int	n	fi	Ni	Fi
(5,10]	3	0.1034483	3	0.1034483
(10,15]	9	0.3103448	12	0.4137931
(15,20]	9	0.3103448	21	0.7241379
(20,25]	4	0.1379310	25	0.8620690
(25,30]	4	0.1379310	29	1.0000000

- d. Dibujar el histograma de frecuencias absolutas, relativas, absolutas acumuladas y relativas acumuladas correspondiente a la tabla anterior.

💡 Solución 1

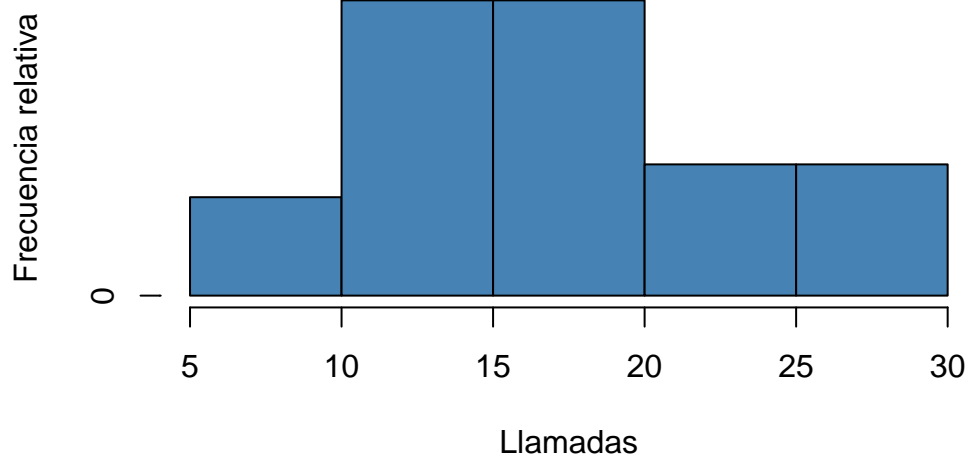
Para dibujar un histograma se puede usar la función `hist` del paquete `graphics`.

```
# Histograma de frecuencias absolutas.  
histo <- hist(df$llamadas, breaks = seq(5, 30, 5), col = "steelblue", main="Dis
```



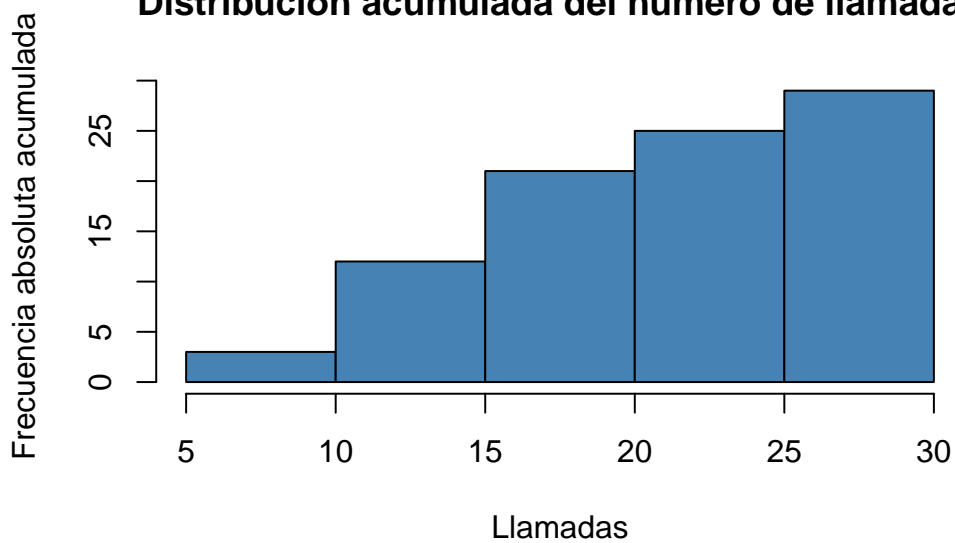
```
ni <- histo$counts  
# Histograma de frecuencias relativas.  
histo$counts <- ni/sum(ni)  
plot(histo, col = "steelblue", main="Distribución del número de llamadas", xlab=
```

Distribución del número de llamadas

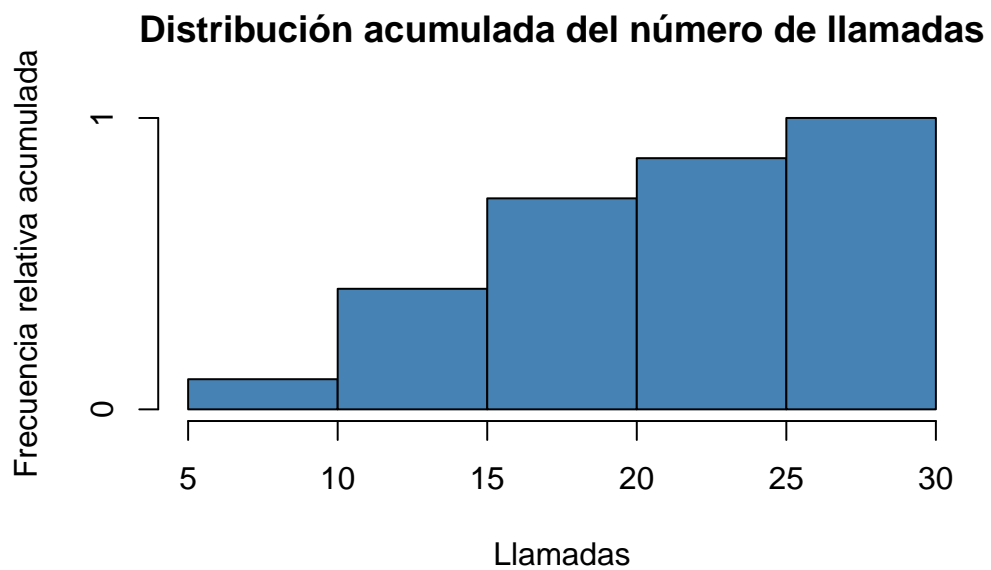


```
# Histograma de frecuencias absolutas acumuladas.  
histo$counts <- cumsum(ni)  
plot(histo, col = "steelblue", main="Distribución acumulada del número de llama
```

Distribución acumulada del número de llamadas



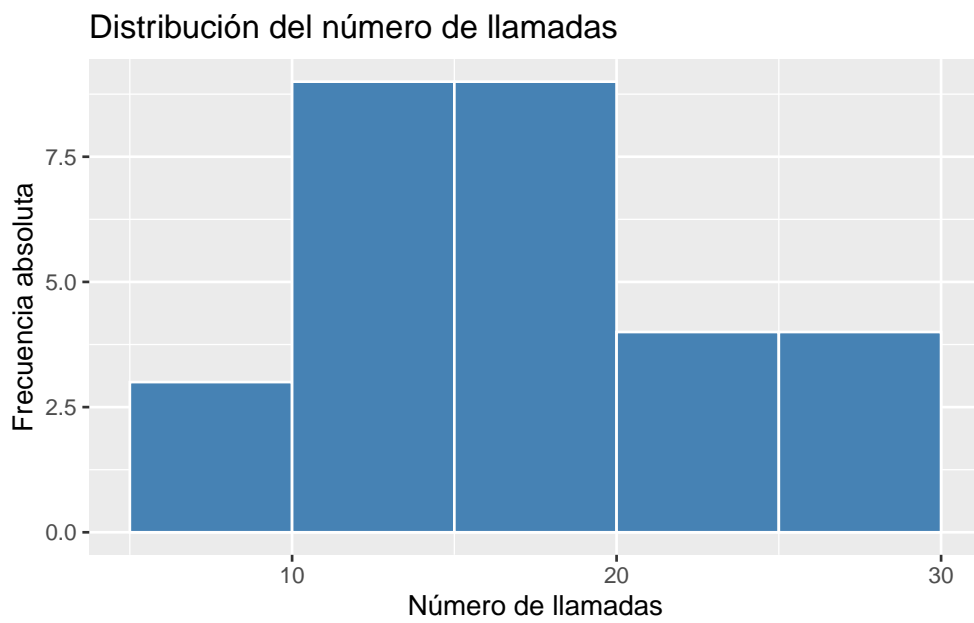
```
# Histograma de frecuencias relativas acumuladas.  
histo$counts <- cumsum(ni)/sum(ni)  
plot(histo, col = "steelblue", main="Distribución acumulada del número de llama
```

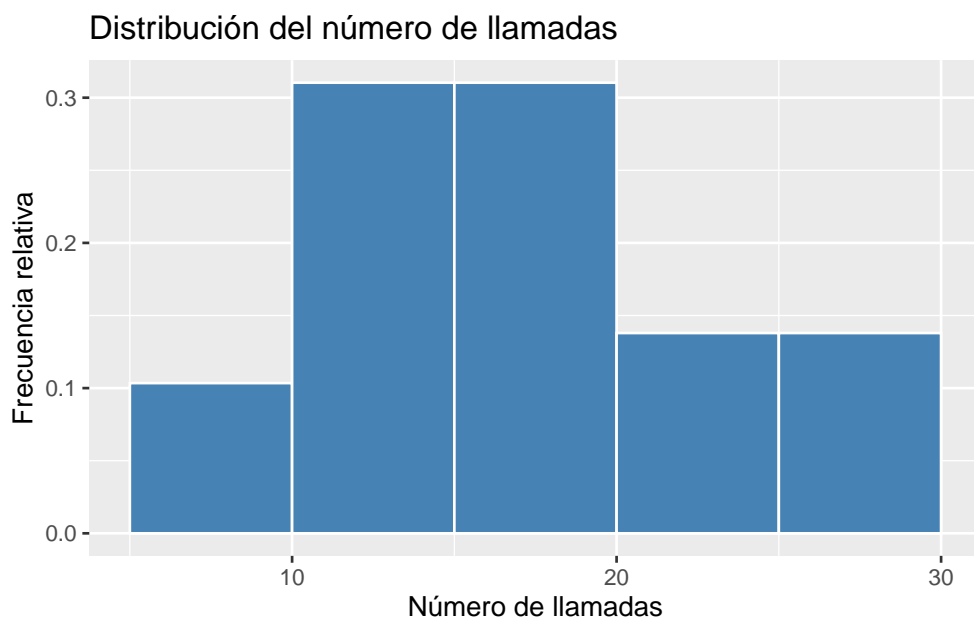
💡 Solución 2

Otra alternativa es usar la función la función `geom_histogram` del paquete `ggplot2`.

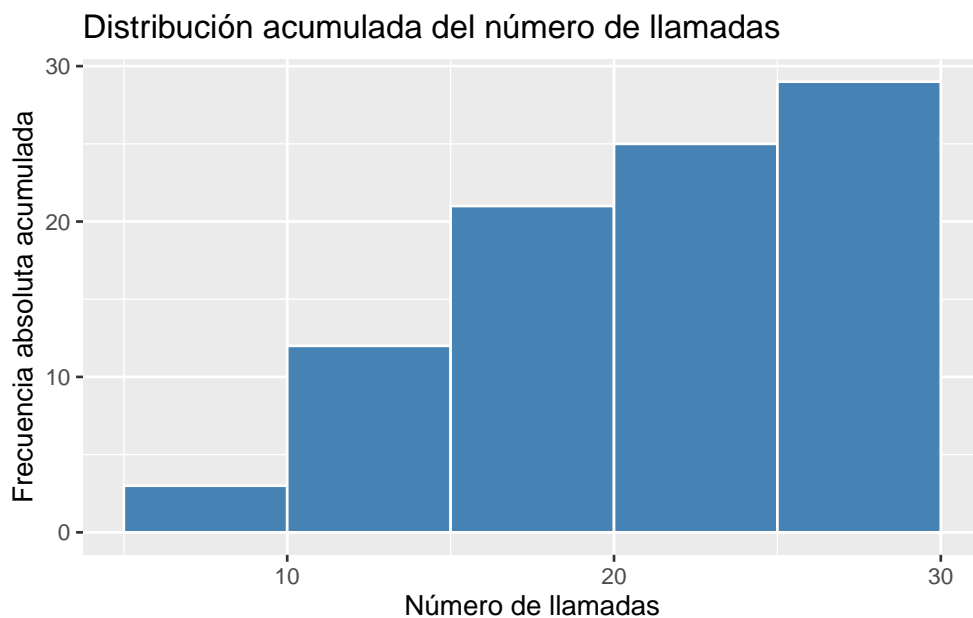
```
library(ggplot2)
# Histograma de frecuencias absolutas
ggplot(df, aes(x = llamadas)) +
  geom_histogram(breaks = seq(5, 30, 5), fill = "steelblue", col = "white") +
  labs(title = "Distribución del número de llamadas", x = "Número de llamadas")
```



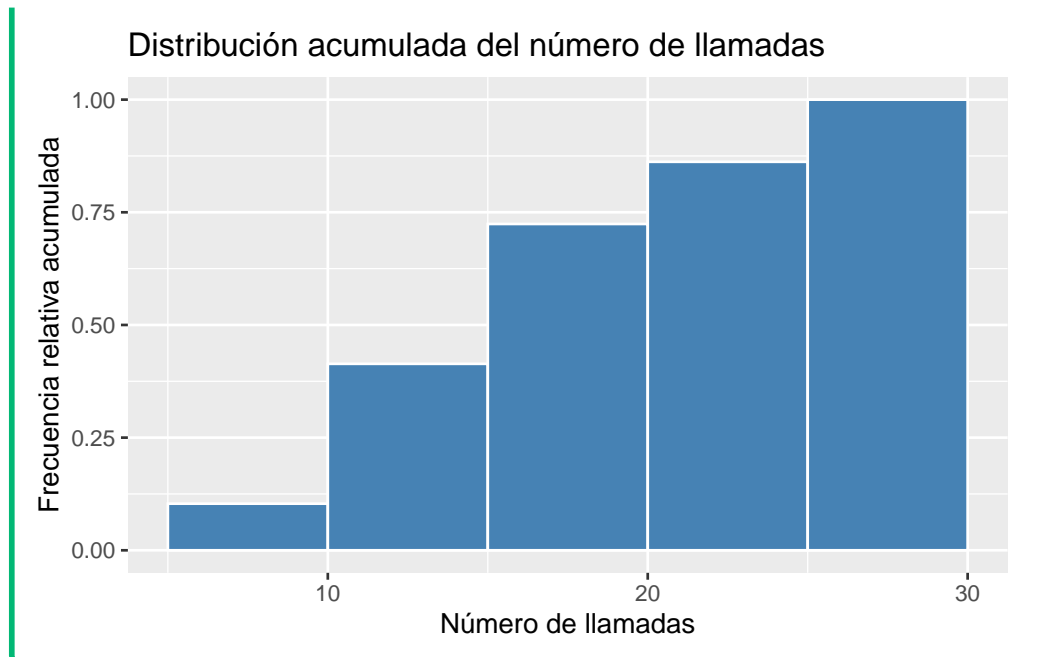
```
# Histograma de frecuencias relativas
ggplot(df, aes(x = llamadas)) +
  geom_histogram(aes(y = after_stat(count/sum(count))), breaks = seq(5, 30, 5),
  labs(title = "Distribución del número de llamadas", x = "Número de llamadas")
```



```
# Histograma de frecuencias acumuladas
ggplot(df, aes(x = llamadas)) +
  geom_histogram(aes(y = after_stat(cumsum(count))), breaks = seq(5, 30, 5),
  labs(title = "Distribución acumulada del número de llamadas", x = "Número d
```



```
# Histograma de frecuencias relativas acumuladas
ggplot(df, aes(x = llamadas)) +
  geom_histogram(aes(y = after_stat(cumsum(count)/sum(count))), breaks = seq
  labs(title = "Distribución acumulada del número de llamadas", x = "Número d
```

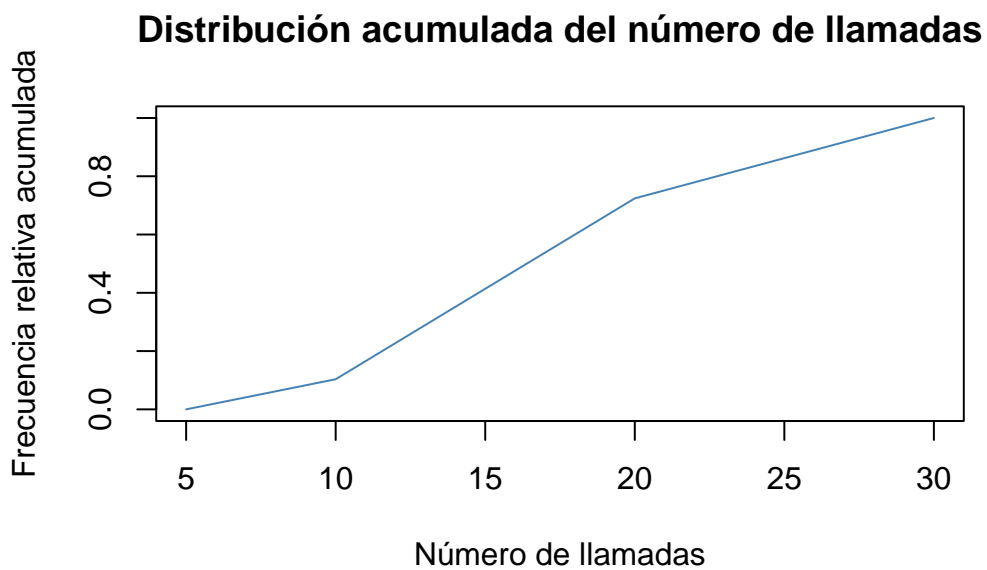


- e. Dibujar el polígono de frecuencias relativas acumuladas (ojiva).

💡 Solución 1

Para dibujar la ojiva se puede usar la función `plot` del paquete `graphics`.

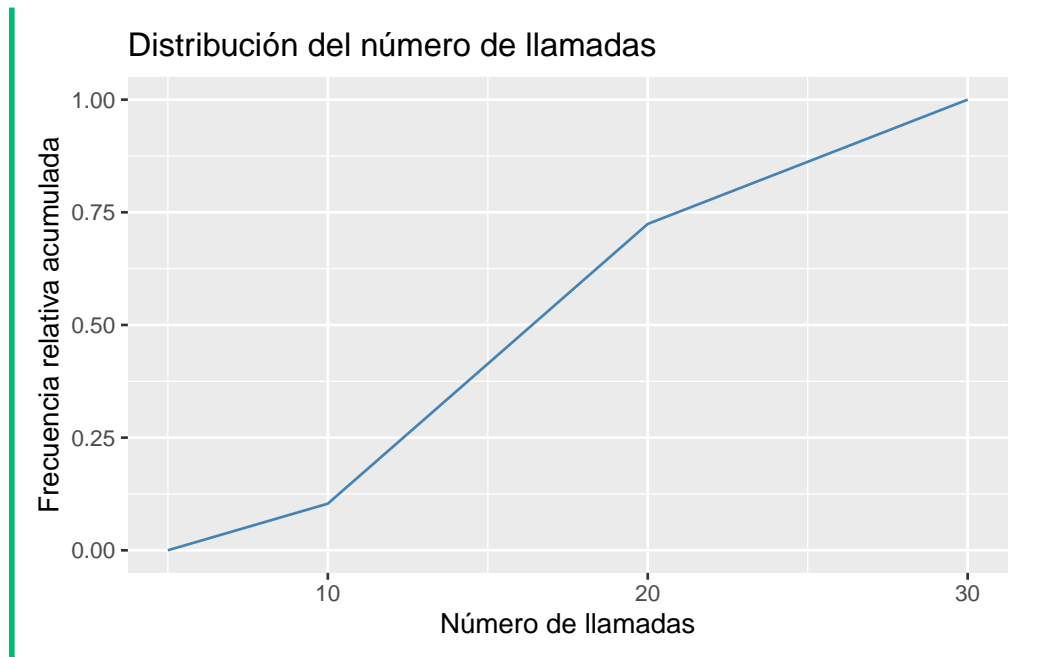
```
# Ojiva
cortes = seq(5, 30, 5)
ni <- table(cut(df$llamadas, breaks = cortes))
Fi <- c(0, cumsum(ni)/sum(ni))
plot(cortes, Fi, type = "l", col = "steelblue", main = "Distribución acumulada")
```



💡 Solución 2

Otra alternativa es usar la función la función `geom_line` del paquete `ggplot2`.

```
library(ggplot2)
# Ojiva
cortes <- seq(5, 30, 5)
tabla_frec <- mutate(df, llamadas_int = cut(df$llamadas, breaks = cortes)) |>
  count(llamadas_int) |>
  mutate(cortes = cortes[-1], Fi = cumsum(n)/sum(n)) |>
  select(cortes, Fi)
tabla_frec <- rbind(data.frame(cortes = cortes[1], Fi = 0), tabla_frec)
ggplot(tabla_frec, aes(x = cortes , y = Fi)) +
  geom_line(col = "steelblue") +
  labs(title = "Distribución del número de llamadas", x = "Número de llamadas")
```



Ejercicio 3.3. Los grupos sanguíneos de una muestra de 30 personas son:

A, B, B, A, AB, 0, 0, A, B, B, A, A, A, A, AB, A, A, A, B, 0, B, B, B, A, A, A, 0, A, AB, 0

- a. Crear un conjunto de datos con la variable `grupo_sanguíneo`.

💡 Solución

```
df <- data.frame(grupo_sanguineo = c("A", "B", "B", "A", "AB", "0", "0", "A", "B", "B", "A", "A", "A", "A", "AB", "A", "A", "A", "B", "0", "B", "B", "B", "A", "A", "A", "0", "A", "AB", "0"))
```

- b. Construir la tabla de frecuencias.

💡 Solución 1

Para obtener las frecuencias absolutas se puede usar la función `table`, y para las frecuencias relativas la función `prop.table` ambas del paquete base de R.

```
# Frecuencias absolutas.
ni <- table(df$grupo_sanguineo)
# Frecuencias relativas
fi <- prop.table(ni)
tabla_frec <- cbind(ni, fi)
tabla_frec
```

```
      ni      fi
0     5 0.1666667
A    14 0.4666667
AB     3 0.1000000
B     8 0.2666667
```

💡 Solución 2

Otra alternativa es usar la función `count` del paquete `dplyr`.

```
library(dplyr)
library(knitr)
library(kableExtra)
count(df, grupo_sanguineo) |>
  mutate(fi = n/sum(n)) |>
  kable()
```

grupo_sanguineo	n	fi
0	5	0.1666667
A	14	0.4666667
AB	3	0.1000000
B	8	0.2666667

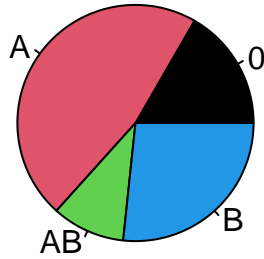
c. Dibujar el diagrama de sectores.

💡 Solución 1

Para dibujar el diagrama de sectores se puede usar la función `pie` del paquete `graphics`.

```
# Diagrama de sectores
pie(ni, col = 1:length(ni), main = "Distribución de los grupos sanguíneos")
```

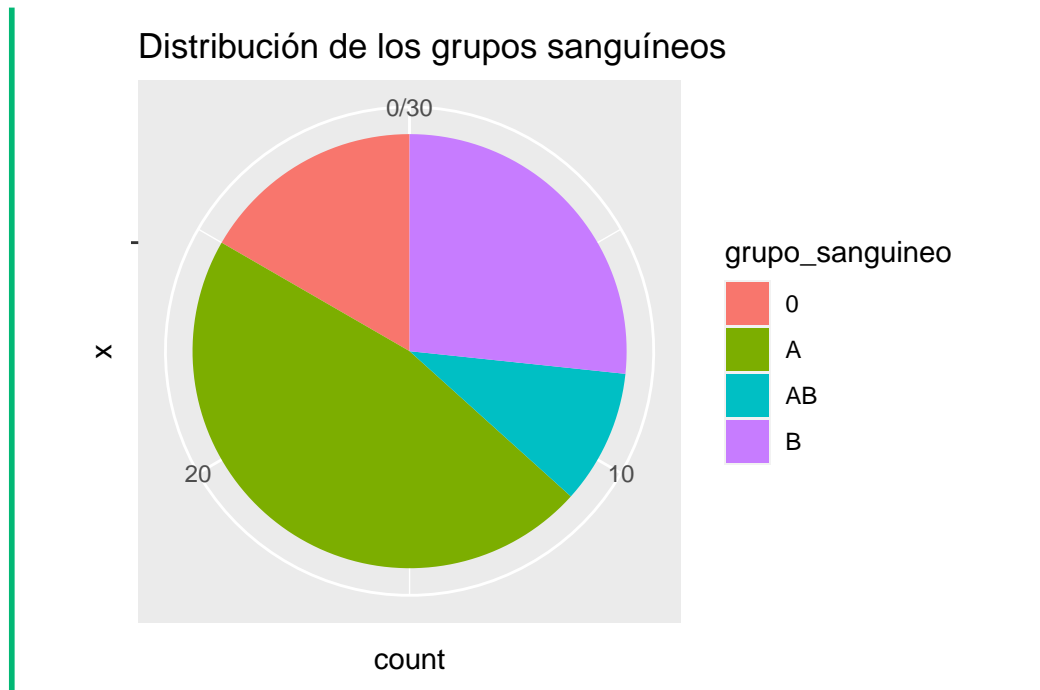
Distribución de los grupos sanguíneos



💡 Solución 2

Otra alternativa es usar las funciones `geom_bar` y `coord_polar` del paquete `ggplot2`.

```
ggplot(df, aes(x = "", fill = grupo_sanguineo)) +  
  # Añadir la capa de las barras.  
  geom_bar() +  
  # Añadir el sistema de coordenadas polares  
  coord_polar(theta = "y") +  
  labs(title = "Distribución de los grupos sanguíneos")
```

Ejercicio 3.4. En un estudio de población se tomó una muestra de 27 personas, y se les preguntó por su edad y estado civil, obteniendo los siguientes resultados:

	Estado civil	Edad
Soltero	31, 45, 35, 65, 21, 38, 62, 22, 31	
Casado	62, 39, 62, 59, 21, 62	
Viudo	80, 68, 65, 40, 78, 69, 75	
Divorciado	31, 65, 59, 49, 65	

- a. Crear un conjunto de datos con la variables `estado_civil` y `edad`.

Solución

```
df <- data.frame(
  edad = c(31, 45, 35, 65, 21, 38, 62, 22, 31, 62, 39, 62, 59, 21, 62, 80, 68, 65, 40, 78, 69, 75, 31, 65, 59, 49, 65),
  estado_civil = rep(c("Soltero", "Casado", "Viudo", "Divorciado"), c(9, 6, 7, 5))
)
```

- b. Calcular los tamaños muestrales según `estado_civil`.

💡 Solución 1

Usando la función `table` del paquete base de R podemos obtener las frecuencias absolutas del estado civil que es el tamaño muestral de cada grupo.

```
table(df$estado_civil)
```

Casado	Divorciado	Soltero	Viudo
6	5	9	7

💡 Solución 2

Usando las funciones `group_by`, `summarise` y `n` del paquete `dplyr`.

```
library(dplyr)
df |> group_by(estado_civil) |>
  summarise(n = n())
```

```
# A tibble: 4 x 2
  estado_civil      n
  <chr>          <int>
1 Casado           6
2 Divorciado       5
3 Soltero          9
4 Viudo            7
```

- c. Construir la tabla de frecuencias de la variable `edad` para cada categoría de la variable `estado_civil`.

💡 Solución

Para dividir la muestra en grupos se puede usar la función `group-by` del paquete `dplyr`.

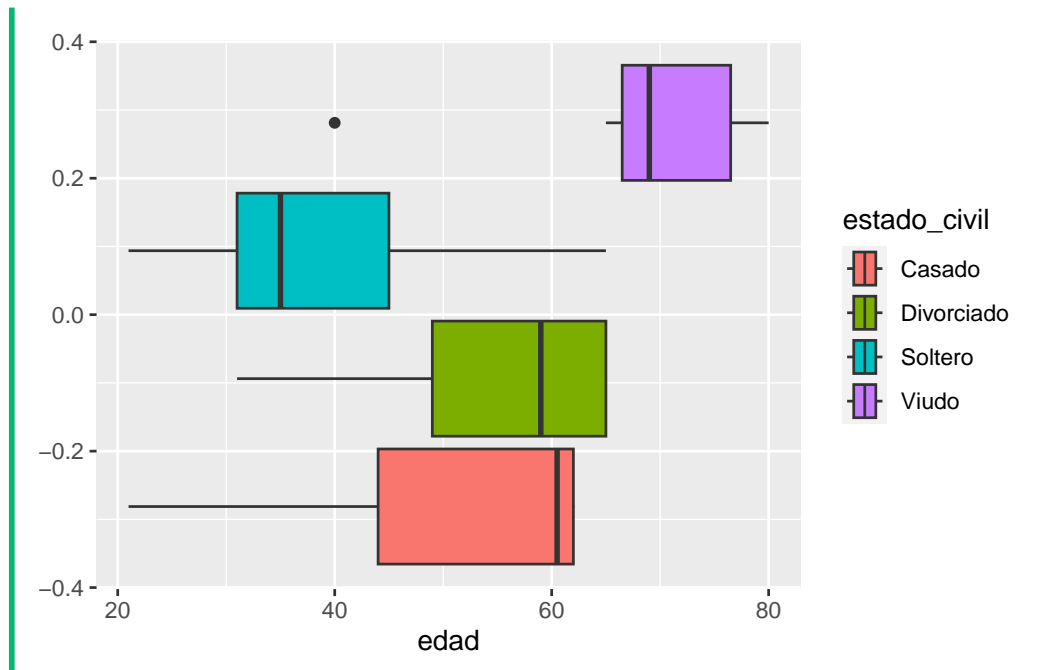
```
library(dplyr)
library(knitr)
library(kableExtra)
mutate(df, edad_int = cut(edad, breaks = seq(20, 80, 10))) |>
  group_by(estado_civil) |>
  count(edad_int) |>
  mutate(fi = n/sum(n), Ni = cumsum(n), Fi = cumsum(n)/sum(n)) |>
  kable()
```

estado_civil	edad_int	n	fi	Ni	Fi
Casado	(20,30]	1	0.1666667	1	0.1666667
Casado	(30,40]	1	0.1666667	2	0.3333333
Casado	(50,60]	1	0.1666667	3	0.5000000
Casado	(60,70]	3	0.5000000	6	1.0000000
Divorciado	(30,40]	1	0.2000000	1	0.2000000
Divorciado	(40,50]	1	0.2000000	2	0.4000000
Divorciado	(50,60]	1	0.2000000	3	0.6000000
Divorciado	(60,70]	2	0.4000000	5	1.0000000
Soltero	(20,30]	2	0.2222222	2	0.2222222
Soltero	(30,40]	4	0.4444444	6	0.6666667
Soltero	(40,50]	1	0.1111111	7	0.7777778
Soltero	(60,70]	2	0.2222222	9	1.0000000
Viudo	(30,40]	1	0.1428571	1	0.1428571
Viudo	(60,70]	3	0.4285714	4	0.5714286
Viudo	(70,80]	3	0.4285714	7	1.0000000

- d. Dibujar los diagramas de cajas de la edad según el estado civil. ¿Existen datos atípicos? ¿En qué grupo hay mayor dispersión?

💡 Solución

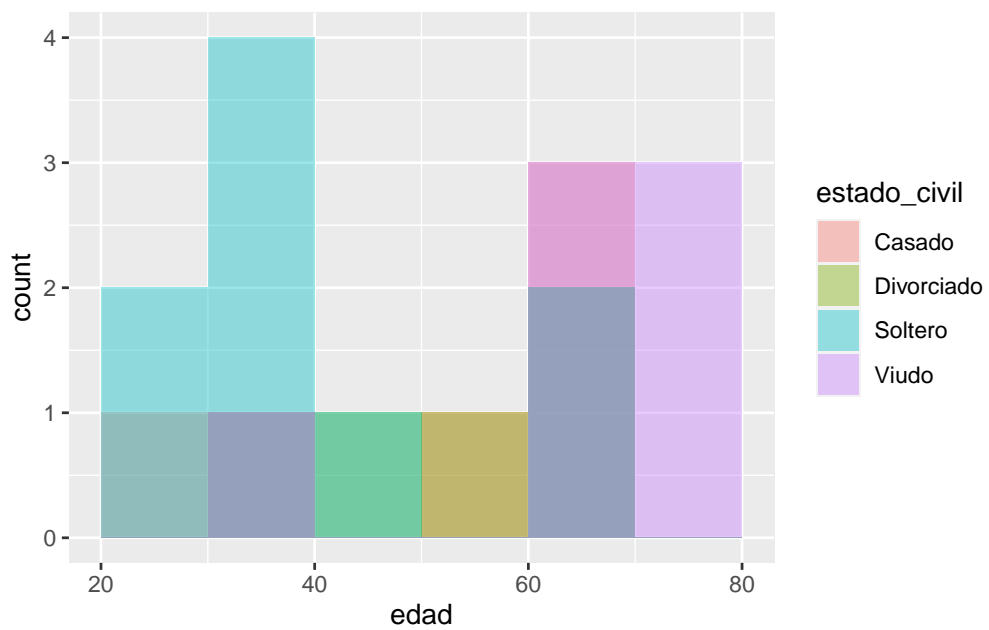
```
library(ggplot2)
ggplot(df, aes(x = edad, fill = estado_civil)) +
  geom_boxplot()
```



e. Dibujar los histogramas de la edad según el estado civil.

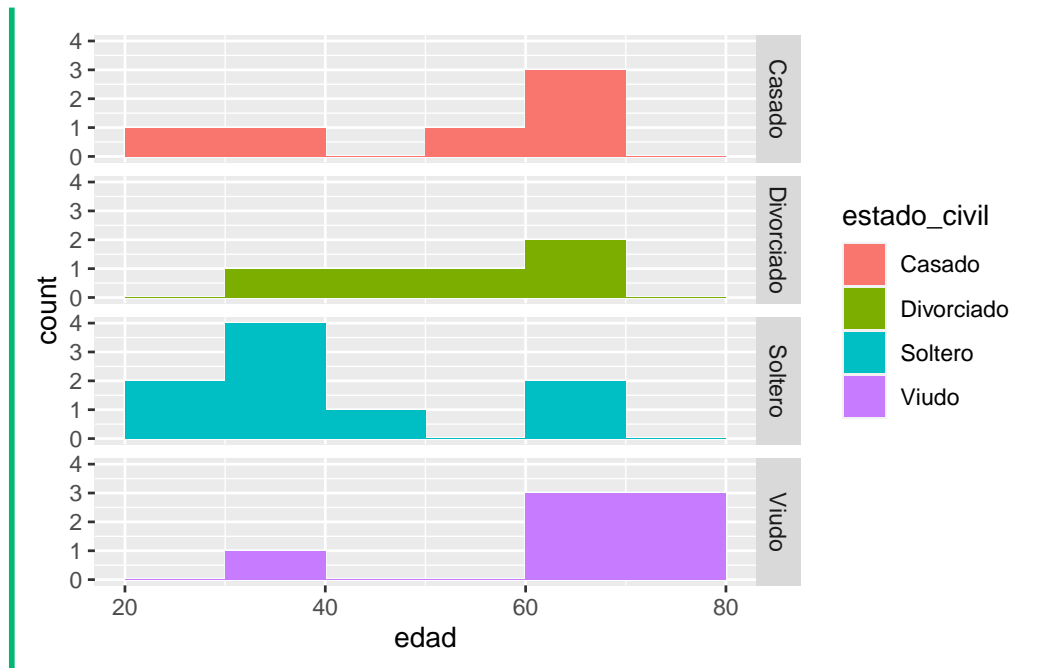
💡 Solución

```
ggplot(df, aes(x = edad, fill = estado_civil)) +  
  geom_histogram(breaks = seq(20, 80, 10), position = "identity", alpha=0.4)
```



Para dibujar cada histograma por separado se puede usar la función `facet_wrap` o `facet_grid` del paquete `ggplot2`.

```
ggplot(df, aes(x = edad, fill = estado_civil)) +  
  geom_histogram(breaks = seq(20, 80, 10)) +  
  # Añadir la faceta del estado civil  
  facet_grid(rows = vars(estado_civil))
```



3.2 Ejercicios propuestos

Ejercicio 3.5. El conjunto de datos [neonatos](#) contiene información sobre una muestra de 320 recién nacidos en un hospital durante un año que cumplieron el tiempo normal de gestación.

- Construir la tabla de frecuencias de la puntuación Apgar al minuto de nacer. Si se considera que una puntuación Apgar de 3 o menos indica que el neonato está deprimido, ¿qué porcentaje de niños está deprimido en la muestra?
- Comparar las distribuciones de frecuencias de las puntuaciones Apgar al minuto de nacer según si la madre es mayor o menor de 20 años. ¿En qué grupo hay más neonatos deprimidos?
- Construir la tabla de frecuencias para el peso de los neonatos, agrupando en clases de amplitud 0.5 desde el 2 hasta el 4.5. ¿En qué intervalo de peso hay más neonatos?
- Comparar la distribución de frecuencias relativas del peso de los neonatos según si la madre fuma o no. Si se considera como peso bajo un peso menor de 2.5 kg, ¿En qué grupo hay un mayor porcentaje de niños con peso bajo?
- Construir el diagrama de barras de la puntuación Apgar al minuto. ¿Qué puntuación Apgar es la más frecuente?
- Construir el diagrama de frecuencias relativas acumuladas de la puntuación Apgar al minuto. ¿Por debajo de que puntuación estarán la mitad de los niños?

- g. Comparar mediante diagramas de barras de frecuencias relativas las distribuciones de las puntuaciones Apgar al minuto según si la madre ha fumado o no durante el embarazo. ¿Qué se puede concluir?
- h. Construir el histograma de pesos, agrupando en clases de amplitud 0.5 desde el 2 hasta el 4.5. ¿En qué intervalo de peso hay más niños?
- i. Comparar la distribución de frecuencias relativas del peso de los neonatos según si la madre fuma o no. ¿En qué grupo se aprecia menor peso de los niños de la muestra?
- j. Comparar la distribución de frecuencias relativas del peso de los neonatos según si la madre fumaba o no antes del embarazo. ¿Qué se puede concluir?
- k. Construir el diagrama de caja y bigotes del peso. ¿Entre qué valores se considera que el peso de un neonato es normal? ¿Existen datos atípicos?
- l. Comparar el diagrama de cajas y bigotes del peso, según si la madre fumó o no durante el embarazo y si era mayor o no de 20 años. ¿En qué grupo el peso tiene más dispersión central? ¿En qué grupo pesan menos los niños de la muestra?
- m. Comparar el diagrama de cajas de la puntuación Apgar al minuto y a los cinco minutos. ¿En qué variable hay más dispersión central?

4 Estadística Descriptiva

4.1 Ejercicios Resueltos

Para la realización de esta práctica se requieren los siguientes paquetes:

```
library(tidyverse)
# Incluye los siguientes paquetes:
# - readr: para la lectura de archivos csv.
# - dplyr: para el preprocesamiento y manipulación de datos.
library(vtable) # para resúmenes estadísticos.
library(skimr) # para resúmenes estadísticos.
library(summarytools) # para resúmenes estadísticos.
library(knitr) # para el formateo de tablas.
library(kableExtra) # para personalizar el formato de las tablas.
```

Ejercicio 4.1. En una encuesta a 25 matrimonios sobre el número de hijos que tenían se obtuvieron los siguientes datos:

1, 2, 4, 2, 2, 2, 3, 2, 1, 1, 0, 2, 2, 0, 2, 2, 1, 2, 2, 3, 1, 2, 2, 1, 2

- a. Crear un conjunto de datos con la variable `hijos`.

 Solución

```
df <- data.frame(hijos = c(1, 2, 4, 2, 2, 2, 3, 2, 1, 1, 0, 2, 2, 0, 2, 2, 1, 2, 2, 3, 1, 2, 2, 1, 2))
```

- b. Calcular el tamaño muestral.

 Solución

```
nrow(df)
```

```
[1] 25
```


c. Calcular la media.

💡 Solución

```
mean(df$hijos)

[1] 1.76
```

d. Calcular la mediana.

💡 Solución

```
median(df$hijos)

[1] 2
```

e. Calcular la moda.

💡 Solución

El paquete base de R no tiene implementada ninguna función para calcular la moda, así que definiremos nuestra propia función.

```
moda <- function(x) {
  u <- unique(x) # Vector con los valores de la muestra sin repetir (sin ordenar)
  tab <- tabulate(match(x, u)) # Frecuencias absolutas de los valores en u.
  u[tab == max(tab)] # Valor con la mayor frecuencia.
}

moda(df$hijos)

[1] 2
```

f. Calcular el mínimo.

💡 Solución

```
min(df$hijos)

[1] 0
```

g. Calcular el máximo.

💡 Solución

```
max(df$hijos)
```

```
[1] 4
```

- h. Calcular los cuartiles.

💡 Solución

```
quantile(df$hijos, prob=c(0.25, 0.5, 0.75))
```

```
25% 50% 75%  
1    2    2
```

- i. Calcular los percentiles 5 y 95.

💡 Solución

```
quantile(df$hijos, prob=c(0.05, 0.95))
```

```
5% 95%  
0.2 3.0
```

- j. Calcular el rango.

💡 Solución

```
max(df$hijos) - min(df$hijos)
```

```
[1] 4
```

- k. Calcular el rango intercuartílico.

💡 Solución

```
IQR(df$hijos)
```

```
[1] 1
```

- l. Calcular la varianza

💡 Solución

R dispone de la función `var` para calcular la *cuasivarianza* o *varianza corregida* $\sum \frac{(x_i - \bar{x})^2}{n-1}$, pero no dispone de una función para calcular la varianza, de manera que para calcularla hay que corregir la cuasivarianza.

```
n <- nrow(df)
# Cuasivarianza
print(paste("Cuasivarianza:", var(df$hijos)))

[1] "Cuasivarianza: 0.773333333333333"

# Varianza
print(paste("Varianza: ", var(df$hijos)*(n-1)/n))

[1] "Varianza: 0.7424"
```

m. Calcular la desviación típica.

💡 Solución

R dispone de la función `sd` para calcular la *cuasidesviación típica* o *desviación típica corregida* $\sqrt{\sum \frac{(x_i - \bar{x})^2}{n-1}}$, pero no dispone de una función para calcular la desviación típica, de manera que para calcularla hay que corregir la cuasidesviación típica.

```
n <- nrow(df)
# Cuasidesviación típica
print(paste("Cuasidesviación típica:", sd(df$hijos)))

[1] "Cuasidesviación típica: 0.879393730551528"

# Desviación típica
print(paste("Desviación típica: ", sd(df$hijos)*sqrt((n-1)/n)))

[1] "Desviación típica: 0.861626369141521"
```

n. Calcular el coeficiente de variación.

💡 Solución

```
sd(df$hijos) / abs(mean(df$hijos))
```

```
[1] 0.4996555
```

- o. Calcular el coeficiente de asimetría.

💡 Solución

Para calcular el coeficiente de asimetría se utiliza el paquete *moments*.

```
library(moments)
skewness(df$hijos)
```

```
[1] 0.1068549
```

Como g_1 está próxima a 0, la distribución es casi simétrica.

- p. Calcular el coeficiente de apuntamiento.

💡 Solución

Para calcular el coeficiente de apuntamiento se utiliza el paquete *moments*.

```
library(moments)
kurtosis(df$hijos)
```

```
[1] 3.71169
```

Como $g_2 > 0$, la distribución es más apuntada de lo normal (leptocúrtica). Como además $g_2 \notin (-2, 2)$ se puede concluir que la muestra es demasiado apuntada para provenir de una población normal.

Ejercicio 4.2. El fichero `colesterol.csv` contiene información de una muestra de pacientes donde se han medido la edad, el sexo, el peso, la altura y el nivel de colesterol, además de su nombre.

- a. Crear un data frame con los datos de todos los pacientes del estudio a partir del fichero `colesterol.csv`.

💡 Solución

```
df <- read.csv("https://raw.githubusercontent.com/asalber/estadistica-practicas/master/colesterol.csv")
df
```

	nombre	edad	sexo	peso	altura	colesterol
1	José Luis Martínez Izquierdo	18	H	85	1.79	182
2	Rosa Díaz Díaz	32	M	65	1.73	232

3	Javier García Sánchez	24	H	NA	1.81	191
4	Carmen López Pinzón	35	M	65	1.70	200
5	Marisa López Collado	46	M	51	1.58	148
6	Antonio Ruiz Cruz	68	H	66	1.74	249
7	Antonio Fernández Ocaña	51	H	62	1.72	276
8	Pilar Martín González	22	M	60	1.66	NA
9	Pedro Gálvez Tenorio	35	H	90	1.94	241
10	Santiago Reillo Manzano	46	H	75	1.85	280
11	Macarena Álvarez Luna	53	M	55	1.62	262
12	José María de la Guía Sanz	58	H	78	1.87	198
13	Miguel Angel Cuadrado Gutiérrez	27	H	109	1.98	210
14	Carolina Rubio Moreno	20	M	61	1.77	194

- b. Calcular el tamaño muestral según el sexo.

💡 Solución 1

```
table(df$sexo)
```

```
H M
8 6
```

💡 Solución 2

```
library(dplyr)
count(df, sexo)
```

```
  sexo n
1    H 8
2    M 6
```

- c. Calcular la media y la desviación típica del nivel de colesterol sin tener en cuenta los datos perdidos.

💡 Solución

```
print(paste("Media:", mean(df$colesterol, na.rm = TRUE)))
```

```
[1] "Media: 220.230769230769"
```