**Guía Completa de Gráficos**

**Estadísticos en R**

Esta es una guía donde puedes encontrar una gran variedad de gráficos, en el código existen aproximadamente de 3 a 5 tipos de gráficos, puedes escoger con el código de prueba cual prefieres, (dependiendo del tipo puede ser: vertical, horizontal, con más diseño, con líneas extra, hoja cuadricula, etc.) pon el código de ejemplo a prueba en Rstudio y escoge el que tu prefieras.

En los ejemplos de los gráficos encontraras solo una imagen de guía, pero en el código existen diferentes. Pon a prueba el que gustes. Te aconsejo mucho que escojas el que más te parezca y te guíes con una IA para que te ayude a adaptar esa gráfica con los datos que estás trabajando, también de esta forma puede que encuentres más opciones para personalizar el gráfico a tu gusto o agregarle más cosas de las que las que aparecen en los ejemplos.

Estas son las librerías que te recomiendo que instales para que puedas utilizar las gráficas, ya que, en los ejemplos solo llaman a las librerías y se usan. Así que utiliza este código, solo debes ejecutarlo una vez:

### Librerías Recomendadas:

install.packages(c("ggplot2", "gridExtra", "vioplot", "car", "moments", "tidyr"))

**Índice**

Tabla de contenido

[**Guía Completa de Gráficos** 1](#_Toc205214804)

[**Estadísticos en R** 1](#_Toc205214805)

[**Índice** 2](#_Toc205214806)

[Histograma 4](#_Toc205214807)

[**¿Qué es?** 4](#_Toc205214808)

[**¿Para qué sirve?** 4](#_Toc205214809)

[**¿Cómo se usa?** 4](#_Toc205214810)

[**Implementación en R** 4](#_Toc205214811)

[Gráfico de Polígonos de Frecuencia 5](#_Toc205214812)

[**¿Qué es?** 6](#_Toc205214813)

[**¿Para qué sirve?** 6](#_Toc205214814)

[**¿Cómo se usa?** 6](#_Toc205214815)

[**Implementación en R** 6](#_Toc205214816)

[Gráfico Circular (Pie Chart) 7](#_Toc205214817)

[**¿Qué es?** 8](#_Toc205214818)

[**¿Para qué sirve?** 8](#_Toc205214819)

[**¿Cómo se usa?** 8](#_Toc205214820)

[**Implementación en R** 8](#_Toc205214821)

[Gráfico de Cajas y Bigotes (Boxplot) 9](#_Toc205214822)

[**¿Qué es?** 10](#_Toc205214823)

[**¿Para qué sirve?** 10](#_Toc205214824)

[**¿Cómo se usa?** 10](#_Toc205214825)

[**Implementación en R** 10](#_Toc205214826)

[Gráfico de Dispersión 13](#_Toc205214827)

[**¿Qué es?** 13](#_Toc205214828)

[**¿Para qué sirve?** 13](#_Toc205214829)

[**¿Cómo se usa?** 13](#_Toc205214830)

[**Implementación en R** 13](#_Toc205214831)

[Gráfico de Barras 16](#_Toc205214832)

[**¿Qué es?** 16](#_Toc205214833)

[**¿Para qué sirve?** 16](#_Toc205214834)

[**¿Cómo se usa?** 16](#_Toc205214835)

[**Implementación en R** 16](#_Toc205214836)

[Gráfico de Líneas 18](#_Toc205214837)

[**¿Qué es?** 19](#_Toc205214838)

[**¿Para qué sirve?** 19](#_Toc205214839)

[**¿Cómo se usa?** 19](#_Toc205214840)

[**Implementación en R** 19](#_Toc205214841)

[Gráfico de Densidad 21](#_Toc205214842)

[**¿Qué es?** 21](#_Toc205214843)

[**¿Para qué sirve?** 21](#_Toc205214844)

[**¿Cómo se usa?** 21](#_Toc205214845)

[**Implementación en R** 21](#_Toc205214846)

[Gráfico Q-Q 24](#_Toc205214847)

[**¿Qué es?** 24](#_Toc205214848)

[**¿Para qué sirve?** 24](#_Toc205214849)

[**¿Cómo se usa?** 24](#_Toc205214850)

[**Implementación en R** 24](#_Toc205214851)

[Gráfico de Violín 26](#_Toc205214852)

[**¿Qué es?** 27](#_Toc205214853)

[**¿Para qué sirve?** 27](#_Toc205214854)

[**¿Cómo se usa?** 27](#_Toc205214855)

[**Implementación en R** 27](#_Toc205214856)

[Conclusión 31](#_Toc205214857)

[Consejos Generales: 31](#_Toc205214858)

[Librerías Recomendadas: 1](#_Toc205214859)

## 

## Histograma

**¿Qué es?**

Un histograma es una representación gráfica de la distribución de frecuencias de una variable cuantitativa continua. Consiste en barras rectangulares cuya altura representa la frecuencia de cada intervalo de valores.

**¿Para qué sirve?**

* Visualizar la forma de la distribución de los datos
* Identificar la tendencia central, dispersión y simetría
* Detectar valores atípicos o outliers
* Comparar distribuciones entre diferentes grupos

**¿Cómo se usa?**

Es ideal para analizar variables continuas como edad, peso, altura, ingresos, temperaturas, etc. Especialmente útil en análisis exploratorio de datos.

**Implementación en R**

# Cargar librerías necesarias

library(ggplot2)

# Crear datos de ejemplo

set.seed(123)

datos <- rnorm(1000, mean = 50, sd = 10)

# Método 1: Con R base

hist(datos,

main = "Histograma de Datos Normales",

xlab = "Valores",

ylab = "Frecuencia",

col = "lightblue",

border = "black",

breaks = 20)

# Método 2: Con ggplot2 (más versátil)

df <- data.frame(valores = datos)

ggplot(df, aes(x = valores)) +

geom\_histogram(bins = 20, fill = "lightblue", color = "black", alpha = 0.7) +

labs(title = "Histograma de Datos Normales",

x = "Valores",

y = "Frecuencia") +

theme\_minimal()

# Histograma con curva de densidad superpuesta

ggplot(df, aes(x = valores)) +

geom\_histogram(aes(y = ..density..), bins = 20,

fill = "lightblue", color = "black", alpha = 0.7) +

geom\_density(color = "red", size = 1) +

labs(title = "Histograma con Curva de Densidad",

x = "Valores",

y = "Densidad") +

theme\_minimal()

**Instrucciones paso a paso:**

1. Instalar ggplot2 si no lo tienes: install.packages("ggplot2")
2. Cargar la librería con library(ggplot2)
3. Crear o cargar tus datos
4. Usar hist() para gráficos básicos o ggplot() para mayor personalización
5. Ajustar el número de bins/breaks según tus datos

## 

## Gráfico de Polígonos de Frecuencia

**¿Qué es?**

Un polígono de frecuencia es un gráfico de líneas que conecta los puntos medios de las barras de un histograma. Muestra la distribución de frecuencias como una línea continua.

**¿Para qué sirve?**

* Comparar múltiples distribuciones en el mismo gráfico
* Visualizar tendencias en la distribución
* Mostrar distribuciones de forma más suave que los histogramas
* Ideal para comparar dos o más grupos

**¿Cómo se usa?**

Útil cuando necesitas comparar distribuciones de diferentes grupos o cuando quieres una representación más suave de la distribución.

**Implementación en R**

# Crear datos de ejemplo para dos grupos

set.seed(123)

grupo1 <- rnorm(500, mean = 45, sd = 8)

grupo2 <- rnorm(500, mean = 55, sd = 10)

# Método 1: Con R base

# Crear histogramas para obtener frecuencias

h1 <- hist(grupo1, breaks = 20, plot = FALSE)

h2 <- hist(grupo2, breaks = 20, plot = FALSE)

# Crear el gráfico

plot(h1$mids, h1$counts, type = "l", col = "blue", lwd = 2,

main = "Polígonos de Frecuencia",

xlab = "Valores", ylab = "Frecuencia")

lines(h2$mids, h2$counts, col = "red", lwd = 2)

legend("topright", legend = c("Grupo 1", "Grupo 2"),

col = c("blue", "red"), lwd = 2)

# Método 2: Con ggplot2

df\_combinado <- data.frame(

valores = c(grupo1, grupo2),

grupo = factor(rep(c("Grupo 1", "Grupo 2"), each = 500))

)

ggplot(df\_combinado, aes(x = valores, color = grupo)) +

geom\_freqpoly(bins = 20, size = 1.2) +

labs(title = "Polígonos de Frecuencia por Grupo",

x = "Valores",

y = "Frecuencia",

color = "Grupo") +

theme\_minimal() +

scale\_color\_manual(values = c("blue", "red"))

# Polígono de frecuencia con densidad

ggplot(df\_combinado, aes(x = valores, color = grupo)) +

geom\_density(size = 1.2) +

labs(title = "Polígonos de Densidad por Grupo",

x = "Valores",

y = "Densidad",

color = "Grupo") +

theme\_minimal() +

scale\_color\_manual(values = c("blue", "red"))

**Instrucciones paso a paso:**

1. Crear datos para cada grupo que quieras comparar
2. Usar geom\_freqpoly() en ggplot2 para crear polígonos de frecuencia
3. Usar geom\_density() para polígonos de densidad más suaves
4. Asignar diferentes colores a cada grupo
5. Añadir leyenda para identificar los grupos

## 

## Gráfico Circular (Pie Chart)

**¿Qué es?**

Un gráfico circular es una representación donde los datos se muestran como sectores de un círculo. Cada sector representa una categoría y su tamaño es proporcional a la frecuencia o porcentaje de esa categoría.

**¿Para qué sirve?**

* Mostrar la composición de un total
* Visualizar proporciones y porcentajes
* Comparar partes de un conjunto
* Mostrar distribución de variables categóricas

**¿Cómo se usa?**

Ideal para variables categóricas con pocas categorías (máximo 5-7). No es recomendable para comparar valores muy similares o cuando hay muchas categorías.

**Implementación en R**

# Crear datos de ejemplo

ventas\_por\_region <- c(Norte = 25, Sur = 30, Este = 20, Oeste = 25)

# Método 1: Con R base

pie(ventas\_por\_region,

main = "Ventas por Región",

col = rainbow(length(ventas\_por\_region)),

labels = paste(names(ventas\_por\_region), "\n",

round(ventas\_por\_region/sum(ventas\_por\_region)\*100, 1), "%"))

# Método 2: Con ggplot2 (más personalizable)

library(ggplot2)

df\_ventas <- data.frame(

region = names(ventas\_por\_region),

ventas = as.numeric(ventas\_por\_region)

)

# Calcular porcentajes

df\_ventas$porcentaje <- round(df\_ventas$ventas/sum(df\_ventas$ventas)\*100, 1)

ggplot(df\_ventas, aes(x = "", y = ventas, fill = region)) +

geom\_bar(stat = "identity", width = 1) +

coord\_polar("y", start = 0) +

labs(title = "Ventas por Región",

fill = "Región") +

theme\_void() +

geom\_text(aes(label = paste0(porcentaje, "%")),

position = position\_stack(vjust = 0.5)) +

scale\_fill\_brewer(palette = "Set3")

# Método 3: Con librería plotly (interactivo)

library(plotly)

plot\_ly(df\_ventas, labels = ~region, values = ~ventas, type = 'pie',

textposition = 'inside',

textinfo = 'label+percent',

hoverinfo = 'text',

text = ~paste('Región:', region, '\nVentas:', ventas),

marker = list(colors = rainbow(nrow(df\_ventas)),

line = list(color = '#FFFFFF', width = 1))) %>%

layout(title = "Ventas por Región - Gráfico Interactivo",

showlegend = FALSE)

**Instrucciones paso a paso:**

1. Organizar datos en un vector nombrado o data frame
2. Para R base: usar pie() directamente
3. Para ggplot2: usar geom\_bar() + coord\_polar()
4. Calcular porcentajes para las etiquetas
5. Personalizar colores con scale\_fill\_brewer() o rainbow()
6. Para gráficos interactivos, instalar plotly: install.packages("plotly")

## 

## Gráfico de Cajas y Bigotes (Boxplot)

**¿Qué es?**

Un boxplot es un gráfico que muestra la distribución de los datos a través de cinco estadísticos: mínimo, primer cuartil (Q1), mediana (Q2), tercer cuartil (Q3) y máximo. También muestra valores atípicos.

**¿Para qué sirve?**

* Identificar la mediana, cuartiles y rango intercuartílico
* Detectar valores atípicos (outliers)
* Comparar distribuciones entre diferentes grupos
* Evaluar la simetría de la distribución
* Identificar la dispersión de los datos

**¿Cómo se usa?**

Excelente para comparar distribuciones entre grupos, identificar outliers y obtener una vista rápida de los estadísticos descriptivos principales.

**Implementación en R**

# Crear datos de ejemplo

set.seed(123)

datos\_grupo\_A <- rnorm(100, mean = 50, sd = 10)

datos\_grupo\_B <- rnorm(100, mean = 60, sd = 15)

datos\_grupo\_C <- rnorm(100, mean = 45, sd = 8)

# Método 1: Boxplot simple con R base

boxplot(datos\_grupo\_A,

main = "Boxplot Simple",

ylab = "Valores",

col = "lightblue")

# Método 2: Boxplot múltiple con R base

datos\_combinados <- list(

"Grupo A" = datos\_grupo\_A,

"Grupo B" = datos\_grupo\_B,

"Grupo C" = datos\_grupo\_C

)

boxplot(datos\_combinados,

main = "Comparación de Grupos",

ylab = "Valores",

col = c("lightblue", "lightgreen", "pink"),

names = c("Grupo A", "Grupo B", "Grupo C"))

# Añadir puntos para mostrar outliers claramente

stripchart(datos\_combinados,

vertical = TRUE,

method = "jitter",

add = TRUE,

pch = 20,

col = 'red')

# Método 3: Con ggplot2 (más personalizable)

library(ggplot2)

df\_completo <- data.frame(

valores = c(datos\_grupo\_A, datos\_grupo\_B, datos\_grupo\_C),

grupo = factor(rep(c("Grupo A", "Grupo B", "Grupo C"), each = 100))

)

ggplot(df\_completo, aes(x = grupo, y = valores, fill = grupo)) +

geom\_boxplot(alpha = 0.7, outlier.color = "red", outlier.size = 2) +

labs(title = "Comparación de Distribuciones por Grupo",

x = "Grupo",

y = "Valores") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE) # Remover leyenda redundante

# Boxplot con puntos individuales

ggplot(df\_completo, aes(x = grupo, y = valores, fill = grupo)) +

geom\_boxplot(alpha = 0.7, outlier.shape = NA) +

geom\_jitter(width = 0.2, alpha = 0.5, size = 0.8) +

labs(title = "Boxplot con Puntos Individuales",

x = "Grupo",

y = "Valores") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE)

# Boxplot horizontal

ggplot(df\_completo, aes(x = grupo, y = valores, fill = grupo)) +

geom\_boxplot(alpha = 0.7) +

coord\_flip() +

labs(title = "Boxplot Horizontal",

x = "Grupo",

y = "Valores") +

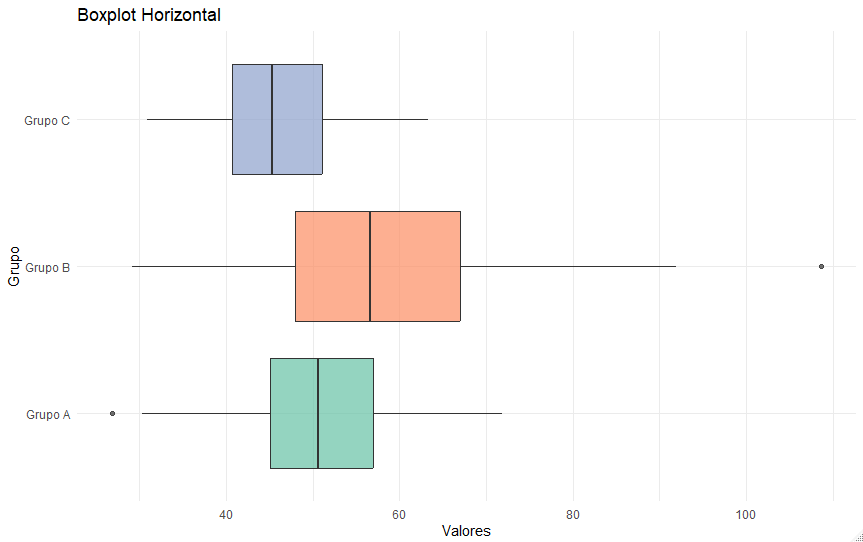
theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE)

**Instrucciones paso a paso:**

1. Organizar datos por grupos si es necesario
2. Para un solo grupo: boxplot(datos)
3. Para múltiples grupos: crear lista o data frame
4. En ggplot2: usar geom\_boxplot()
5. Personalizar colores con scale\_fill\_brewer()
6. Usar geom\_jitter() para mostrar puntos individuales
7. coord\_flip() para boxplots horizontales



## Gráfico de Dispersión

**¿Qué es?**

Un gráfico de dispersión muestra la relación entre dos variables cuantitativas mediante puntos en un plano cartesiano. Cada punto representa una observación con sus valores en ambas variables.

**¿Para qué sirve?**

* Identificar correlaciones entre variables
* Detectar patrones y tendencias
* Identificar valores atípicos
* Evaluar la fuerza y dirección de relaciones lineales
* Visualizar agrupaciones naturales en los datos

**¿Cómo se usa?**

Fundamental en análisis de correlación, regresión, y para explorar relaciones entre variables continuas como altura-peso, experiencia-salario, etc.

**Implementación en R**

# Crear datos de ejemplo

set.seed(123)

n <- 100

altura <- rnorm(n, mean = 170, sd = 10)

peso <- 2.3 \* altura + rnorm(n, mean = -220, sd = 15)

# Método 1: Con R base

plot(altura, peso,

main = "Relación entre Altura y Peso",

xlab = "Altura (cm)",

ylab = "Peso (kg)",

pch = 19, # Tipo de punto

col = "blue",

cex = 0.8) # Tamaño del punto

# Añadir línea de regresión

abline(lm(peso ~ altura), col = "red", lwd = 2)

# Añadir coeficiente de correlación

correlacion <- cor(altura, peso)

text(x = min(altura) + 5, y = max(peso) - 5,

labels = paste("r =", round(correlacion, 3)),

cex = 1.2)

# Método 2: Con ggplot2

library(ggplot2)

df\_datos <- data.frame(altura = altura, peso = peso)

ggplot(df\_datos, aes(x = altura, y = peso)) +

geom\_point(color = "blue", alpha = 0.6, size = 2) +

geom\_smooth(method = "lm", color = "red", se = TRUE) +

labs(title = "Relación entre Altura y Peso",

x = "Altura (cm)",

y = "Peso (kg)",

caption = paste("Correlación r =", round(correlacion, 3))) +

theme\_minimal()

# Gráfico de dispersión con grupos

# Crear factor de grupos (ejemplo: género)

genero <- sample(c("Masculino", "Femenino"), n, replace = TRUE)

df\_datos$genero <- genero

ggplot(df\_datos, aes(x = altura, y = peso, color = genero)) +

geom\_point(alpha = 0.7, size = 2) +

geom\_smooth(method = "lm", se = TRUE) +

labs(title = "Relación entre Altura y Peso por Género",

x = "Altura (cm)",

y = "Peso (kg)",

color = "Género") +

theme\_minimal() +

scale\_color\_manual(values = c("blue", "red"))

# Gráfico de dispersión con tamaño variable (bubble chart)

edad <- sample(18:65, n, replace = TRUE)

df\_datos$edad <- edad

ggplot(df\_datos, aes(x = altura, y = peso, size = edad, color = genero)) +

geom\_point(alpha = 0.6) +

labs(title = "Relación Altura-Peso con Edad y Género",

x = "Altura (cm)",

y = "Peso (kg)",

size = "Edad",

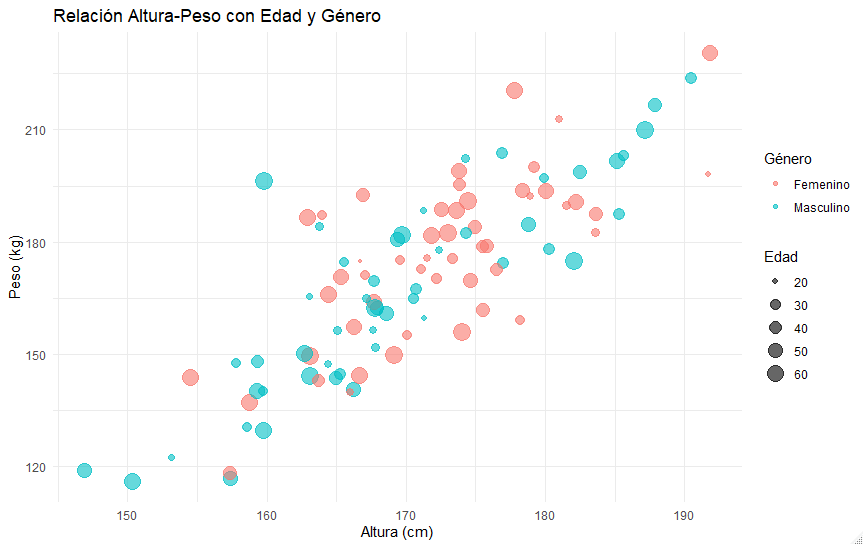
color = "Género") +

theme\_minimal() +

scale\_size\_continuous(range = c(1, 6))

**Instrucciones paso a paso:**

1. Preparar dos variables numéricas
2. Para R base: usar plot(x, y)
3. Añadir línea de tendencia con abline(lm(y ~ x))
4. En ggplot2: usar geom\_point() para los puntos
5. Usar geom\_smooth(method = "lm") para línea de regresión
6. Para grupos: mapear color o forma a una variable categórica
7. Para bubble charts: mapear tamaño a una tercera variable



## 

## Gráfico de Barras

**¿Qué es?**

Un gráfico de barras representa datos categóricos mediante barras rectangulares cuya longitud es proporcional a los valores que representan. Puede ser vertical (columnas) u horizontal.

**¿Para qué sirve?**

* Comparar cantidades entre categorías
* Mostrar distribuciones de variables categóricas
* Visualizar frecuencias o totales por grupo
* Comparar rendimiento entre diferentes entidades

**¿Cómo se usa?**

Ideal para variables categóricas como ventas por producto, frecuencias por categoría, comparaciones entre grupos, rankings, etc.

**Implementación en R**

# Crear datos de ejemplo

productos <- c("Producto A", "Producto B", "Producto C", "Producto D", "Producto E")

ventas <- c(120, 95, 180, 75, 140)

# Método 1: Con R base (vertical)

barplot(ventas,

names.arg = productos,

main = "Ventas por Producto",

ylab = "Ventas ($000)",

col = rainbow(length(productos)),

las = 2) # Rotar etiquetas del eje x

# Método 2: Con R base (horizontal)

barplot(ventas,

names.arg = productos,

main = "Ventas por Producto",

xlab = "Ventas ($000)",

col = rainbow(length(productos)),

horiz = TRUE)

# Método 3: Con ggplot2

library(ggplot2)

df\_ventas <- data.frame(

producto = productos,

ventas = ventas

)

# Gráfico de barras vertical

ggplot(df\_ventas, aes(x = producto, y = ventas, fill = producto)) +

geom\_bar(stat = "identity", alpha = 0.8) +

labs(title = "Ventas por Producto",

x = "Producto",

y = "Ventas ($000)") +

theme\_minimal() +

theme(axis.text.x = element\_text(angle = 45, hjust = 1)) +

guides(fill = FALSE) + # Remover leyenda redundante

scale\_fill\_brewer(palette = "Set3")

# Gráfico de barras horizontal

ggplot(df\_ventas, aes(x = reorder(producto, ventas), y = ventas, fill = producto)) +

geom\_bar(stat = "identity", alpha = 0.8) +

coord\_flip() +

labs(title = "Ventas por Producto (Ordenado)",

x = "Producto",

y = "Ventas ($000)") +

theme\_minimal() +

guides(fill = FALSE) +

scale\_fill\_brewer(palette = "Set3")

# Gráfico de barras agrupadas

# Crear datos para múltiples categorías

trimestre <- rep(c("Q1", "Q2", "Q3", "Q4"), each = 3)

producto\_trim <- rep(c("Producto A", "Producto B", "Producto C"), 4)

ventas\_trim <- c(120, 95, 180, 135, 110, 195, 140, 120, 200, 155, 125, 185)

df\_trimestral <- data.frame(

trimestre = trimestre,

producto = producto\_trim,

ventas = ventas\_trim

)

ggplot(df\_trimestral, aes(x = trimestre, y = ventas, fill = producto)) +

geom\_bar(stat = "identity", position = "dodge", alpha = 0.8) +

labs(title = "Ventas Trimestrales por Producto",

x = "Trimestre",

y = "Ventas ($000)",

fill = "Producto") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2")

# Gráfico de barras apiladas

ggplot(df\_trimestral, aes(x = trimestre, y = ventas, fill = producto)) +

geom\_bar(stat = "identity", position = "stack", alpha = 0.8) +

labs(title = "Ventas Trimestrales Acumuladas",

x = "Trimestre",

y = "Ventas Totales ($000)",

fill = "Producto") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2")

**Instrucciones paso a paso:**

1. Organizar datos en vectores o data frame
2. Para R base: usar barplot()
3. Para horizontal: añadir horiz = TRUE
4. En ggplot2: usar geom\_bar(stat = "identity")
5. Para agrupar: usar position = "dodge"
6. Para apilar: usar position = "stack"
7. reorder() para ordenar barras por valor
8. coord\_flip() para convertir a horizontal

## 

## 

## Gráfico de Líneas

**¿Qué es?**

Un gráfico de líneas conecta puntos de datos mediante líneas, mostrando la evolución de una variable a lo largo del tiempo o de otra variable continua.

**¿Para qué sirve?**

* Mostrar tendencias temporales
* Visualizar series de tiempo
* Comparar evolución de múltiples variables
* Identificar patrones estacionales o cíclicos
* Mostrar relaciones funcionales entre variables

**¿Cómo se usa?**

Fundamental para análisis de series temporales, datos financieros, evolución de ventas, temperaturas a lo largo del tiempo, etc.

**Implementación en R**

# Crear datos de ejemplo (serie temporal)

fechas <- seq(as.Date("2023-01-01"), as.Date("2023-12-31"), by = "month")

ventas\_2023 <- c(450, 520, 480, 600, 750, 820, 900, 880, 700, 650, 580, 950)

ventas\_2022 <- c(420, 480, 450, 550, 680, 750, 820, 800, 620, 580, 520, 850)

# Método 1: Con R base

plot(fechas, ventas\_2023,

type = "l", # tipo línea

col = "blue",

lwd = 2,

main = "Evolución de Ventas 2023",

xlab = "Mes",

ylab = "Ventas ($000)",

ylim = c(400, 1000))

# Añadir puntos

points(fechas, ventas\_2023, col = "blue", pch = 19)

# Método 2: Múltiples líneas con R base

plot(fechas, ventas\_2023,

type = "l",

col = "blue",

lwd = 2,

main = "Comparación de Ventas 2022 vs 2023",

xlab = "Mes",

ylab = "Ventas ($000)",

ylim = c(400, 1000))

lines(fechas, ventas\_2022, col = "red", lwd = 2)

points(fechas, ventas\_2023, col = "blue", pch = 19)

points(fechas, ventas\_2022, col = "red", pch = 17)

# Añadir leyenda

legend("topleft",

legend = c("2023", "2022"),

col = c("blue", "red"),

lwd = 2,

pch = c(19, 17))

# Método 3: Con ggplot2

library(ggplot2)

library(dplyr)

# Preparar datos en formato largo

df\_ventas\_tiempo <- data.frame(

fecha = rep(fechas, 2),

ventas = c(ventas\_2023, ventas\_2022),

año = factor(rep(c("2023", "2022"), each = length(fechas)))

)

ggplot(df\_ventas\_tiempo, aes(x = fecha, y = ventas, color = año)) +

geom\_line(size = 1.2) +

geom\_point(size = 2) +

labs(title = "Evolución de Ventas por Año",

x = "Fecha",

y = "Ventas ($000)",

color = "Año") +

theme\_minimal() +

scale\_color\_manual(values = c("2022" = "red", "2023" = "blue")) +

scale\_x\_date(date\_labels = "%b", date\_breaks = "1 month") +

theme(axis.text.x = element\_text(angle = 45, hjust = 1))

# Gráfico de líneas con área sombreada

ggplot(df\_ventas\_tiempo, aes(x = fecha, y = ventas, fill = año, color = año)) +

geom\_area(alpha = 0.3, position = "identity") +

geom\_line(size = 1.2) +

labs(title = "Evolución de Ventas con Área Sombreada",

x = "Fecha",

y = "Ventas ($000)") +

theme\_minimal() +

scale\_color\_manual(values = c("2022" = "red", "2023" = "blue")) +

scale\_fill\_manual(values = c("2022" = "red", "2023" = "blue"))

**Instrucciones paso a paso:**

1. Organizar datos con variable temporal o secuencial
2. Para R base: usar plot() con type = "l"
3. Añadir múltiples líneas con lines()
4. En ggplot2: usar geom\_line() y geom\_point()
5. Para datos temporales: usar scale\_x\_date()
6. Mapear color a grupos para múltiples series
7. geom\_area() para gráficos de área

## Gráfico de Densidad

**¿Qué es?**

Un gráfico de densidad es una versión suavizada del histograma que muestra la distribución de probabilidad de una variable continua mediante una curva suave.

**¿Para qué sirve?**

* Visualizar la forma de la distribución de manera suave
* Comparar distribuciones de múltiples grupos
* Identificar modas (picos) en la distribución
* Evaluar normalidad de los datos
* Mostrar densidades de probabilidad

**¿Cómo se usa?**

Ideal para análisis exploratorio de datos, comparación de distribuciones entre grupos, y cuando se necesita una representación más suave que los histogramas.

**Implementación en R**

# Crear datos de ejemplo

set.seed(123)

grupo\_A <- rnorm(1000, mean = 50, sd = 10)

grupo\_B <- rnorm(1000, mean = 60, sd = 8)

grupo\_C <- rnorm(1000, mean = 45, sd = 12)

# Método 1: Con R base

# Densidad simple

density\_A <- density(grupo\_A)

plot(density\_A,

main = "Gráfico de Densidad - Grupo A",

xlab = "Valores",

ylab = "Densidad",

col = "blue",

lwd = 2)

# Añadir área bajo la curva

polygon(density\_A, col = rgb(0, 0, 1, 0.3), border = "blue")

# Múltiples densidades

plot(density(grupo\_A),

main = "Comparación de Densidades",

xlab = "Valores",

ylab = "Densidad",

col = "blue",

lwd = 2,

xlim = c(10, 90),

ylim = c(0, 0.06))

lines(density(grupo\_B), col = "red", lwd = 2)

lines(density(grupo\_C), col = "green", lwd = 2)

legend("topright",

legend = c("Grupo A", "Grupo B", "Grupo C"),

col = c("blue", "red", "green"),

lwd = 2)

# Método 2: Con ggplot2

library(ggplot2)

# Preparar datos

df\_grupos <- data.frame(

valores = c(grupo\_A, grupo\_B, grupo\_C),

grupo = factor(rep(c("Grupo A", "Grupo B", "Grupo C"), each = 1000))

)

# Gráfico de densidad básico

ggplot(df\_grupos, aes(x = valores)) +

geom\_density(fill = "lightblue", alpha = 0.7, color = "black") +

labs(title = "Gráfico de Densidad",

x = "Valores",

y = "Densidad") +

theme\_minimal()

# Múltiples densidades por grupo

ggplot(df\_grupos, aes(x = valores, fill = grupo, color = grupo)) +

geom\_density(alpha = 0.6) +

labs(title = "Comparación de Densidades por Grupo",

x = "Valores",

y = "Densidad",

fill = "Grupo",

color = "Grupo") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

scale\_color\_brewer(palette = "Set2")

# Densidades separadas (facetas)

ggplot(df\_grupos, aes(x = valores, fill = grupo)) +

geom\_density(alpha = 0.7, color = "black") +

facet\_wrap(~grupo, ncol = 1) +

labs(title = "Densidades por Grupo (Separadas)",

x = "Valores",

y = "Densidad") +

theme\_minimal() +

guides(fill = FALSE) +

scale\_fill\_brewer(palette = "Set2")

# Densidad con histograma superpuesto

ggplot(df\_grupos[df\_grupos$grupo == "Grupo A", ], aes(x = valores)) +

geom\_histogram(aes(y = ..density..), bins = 30,

fill = "lightblue", alpha = 0.7, color = "black") +

geom\_density(color = "red", size = 1.2) +

labs(title = "Histograma con Curva de Densidad",

x = "Valores",

y = "Densidad") +

theme\_minimal()

# Densidad con estadísticas superpuestas

media\_A <- mean(grupo\_A)

mediana\_A <- median(grupo\_A)

ggplot(data.frame(valores = grupo\_A), aes(x = valores)) +

geom\_density(fill = "lightblue", alpha = 0.7, color = "black") +

geom\_vline(xintercept = media\_A, color = "red", linetype = "dashed", size = 1) +

geom\_vline(xintercept = mediana\_A, color = "blue", linetype = "dashed", size = 1) +

labs(title = "Densidad con Media y Mediana",

x = "Valores",

y = "Densidad",

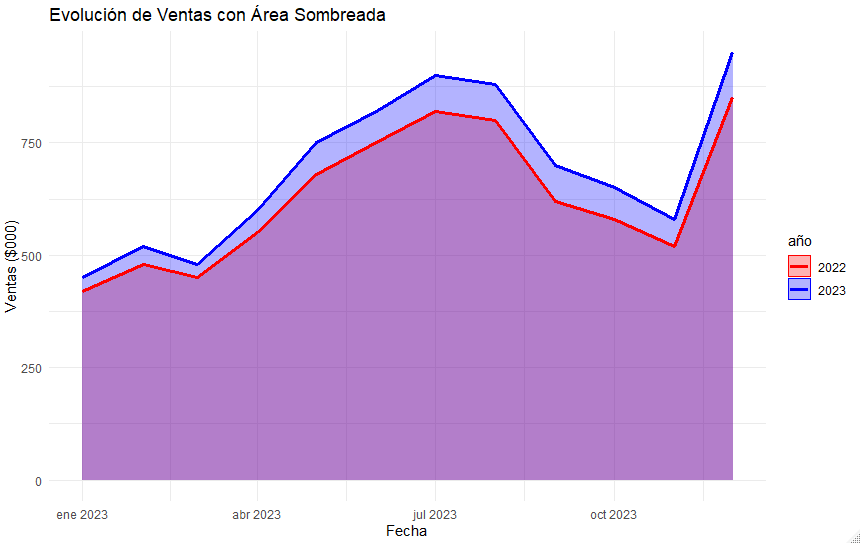
caption = paste("Media (rojo):", round(media\_A, 2),

"| Mediana (azul):", round(mediana\_A, 2))) +

theme\_minimal()

**Instrucciones paso a paso:**

1. Para R base: usar density() para calcular densidad y plot() para graficar
2. polygon() para llenar área bajo la curva
3. En ggplot2: usar geom\_density()
4. Mapear fill y color para múltiples grupos
5. facet\_wrap() para separar por grupos
6. Combinar con geom\_histogram() para comparar con histograma
7. geom\_vline() para añadir líneas de referencia



## Gráfico Q-Q

**¿Qué es?**

Un gráfico Q-Q (Quantile-Quantile) compara los cuantiles de dos distribuciones. Comúnmente se usa para evaluar si los datos siguen una distribución específica (como la normal).

**¿Para qué sirve?**

* Evaluar normalidad de los datos
* Comparar distribuciones empíricas con teóricas
* Identificar desviaciones de la normalidad
* Detectar colas pesadas o asimétricas
* Validar supuestos para análisis estadísticos

**¿Cómo se usa?**

Fundamental antes de aplicar pruebas paramétricas, análisis de regresión, ANOVA, etc. Si los puntos siguen aproximadamente una línea recta, los datos son aproximadamente normales.

**Implementación en R**

# Crear datos de ejemplo

set.seed(123)

datos\_normales <- rnorm(200, mean = 50, sd = 10)

datos\_no\_normales <- rexp(200, rate = 0.1) # Distribución exponencial

# Método 1: Con R base

# Q-Q plot contra distribución normal

qqnorm(datos\_normales,

main = "Q-Q Plot - Datos Normales",

xlab = "Cuantiles Teóricos",

ylab = "Cuantiles de la Muestra")

qqline(datos\_normales, col = "red", lwd = 2)

# Q-Q plot para datos no normales

qqnorm(datos\_no\_normales,

main = "Q-Q Plot - Datos No Normales",

xlab = "Cuantiles Teóricos",

ylab = "Cuantiles de la Muestra")

qqline(datos\_no\_normales, col = "red", lwd = 2)

# Método 2: Con ggplot2

library(ggplot2)

# Función auxiliar para crear Q-Q plot

crear\_qq\_plot <- function(datos, titulo) {

n <- length(datos)

datos\_ordenados <- sort(datos)

cuantiles\_teoricos <- qnorm(ppoints(n))

df\_qq <- data.frame(

teoricos = cuantiles\_teoricos,

muestra = datos\_ordenados

)

ggplot(df\_qq, aes(x = teoricos, y = muestra)) +

geom\_point(alpha = 0.6, color = "blue") +

geom\_smooth(method = "lm", se = FALSE, color = "red") +

labs(title = titulo,

x = "Cuantiles Teóricos",

y = "Cuantiles de la Muestra") +

theme\_minimal()

}

# Crear gráficos

crear\_qq\_plot(datos\_normales, "Q-Q Plot - Datos Normales (ggplot2)")

crear\_qq\_plot(datos\_no\_normales, "Q-Q Plot - Datos No Normales (ggplot2)")

# Método 3: Con ggplot2 y geom\_qq

df\_datos <- data.frame(

normales = datos\_normales,

no\_normales = datos\_no\_normales

)

# Reshape para formato largo

library(tidyr)

df\_largo <- df\_datos %>%

pivot\_longer(cols = everything(), names\_to = "tipo", values\_to = "valores")

ggplot(df\_largo, aes(sample = valores)) +

geom\_qq() +

geom\_qq\_line(color = "red") +

facet\_wrap(~tipo, scales = "free") +

labs(title = "Comparación Q-Q Plots",

x = "Cuantiles Teóricos",

y = "Cuantiles de la Muestra") +

theme\_minimal()

# Q-Q plot con bandas de confianza

library(car) # Requiere instalar: install.packages("car")

# Para datos normales

qqPlot(datos\_normales,

main = "Q-Q Plot con Bandas de Confianza",

xlab = "Cuantiles Teóricos",

ylab = "Cuantiles de la Muestra")

# Interpretación automatizada

interpretar\_qq <- function(datos, nombre = "datos") {

# Prueba de Shapiro-Wilk para normalidad

if(length(datos) <= 5000) {

shapiro\_test <- shapiro.test(datos)

cat("Prueba de Shapiro-Wilk para", nombre, ":\n")

cat("p-valor:", shapiro\_test$p.value, "\n")

if(shapiro\_test$p.value > 0.05) {

cat("Los datos parecen seguir una distribución normal (p > 0.05)\n\n")

} else {

cat("Los datos NO siguen una distribución normal (p <= 0.05)\n\n")

}

}

# Crear el Q-Q plot

qqnorm(datos, main = paste("Q-Q Plot -", nombre))

qqline(datos, col = "red", lwd = 2)

}

# Aplicar interpretación

par(mfrow = c(1, 2))

interpretar\_qq(datos\_normales, "Datos Normales")

interpretar\_qq(datos\_no\_normales, "Datos No Normales")

par(mfrow = c(1, 1))

**Instrucciones paso a paso:**

1. Para R base: usar qqnorm() y qqline()
2. En ggplot2: usar geom\_qq() y geom\_qq\_line()
3. Instalar librería car para bandas de confianza: install.packages("car")
4. Interpretar: puntos en línea recta = normalidad
5. Complementar con prueba de Shapiro-Wilk
6. facet\_wrap() para comparar múltiples distribuciones
7. Desviaciones sistemáticas indican no normalidad

## 

## Gráfico de Violín

**¿Qué es?**

Un gráfico de violín combina un boxplot con un gráfico de densidad. Muestra la distribución de los datos mediante la anchura del "violín" que representa la densidad en cada valor.

**¿Para qué sirve?**

* Mostrar distribución completa de los datos
* Comparar formas de distribución entre grupos
* Identificar multimodalidad (múltiples picos)
* Combinar información de boxplot y densidad
* Visualizar asimetría y dispersión

**¿Cómo se usa?**

Excelente para comparar distribuciones entre grupos cuando se necesita más detalle que un boxplot simple, especialmente útil para identificar distribuciones bimodales o asimétricas.

**Implementación en R**

# Crear datos de ejemplo con diferentes distribuciones

set.seed(123)

grupo\_normal <- rnorm(200, mean = 50, sd = 10)

grupo\_bimodal <- c(rnorm(100, mean = 40, sd = 5), rnorm(100, mean = 70, sd = 5))

grupo\_asimetrico <- rgamma(200, shape = 2, scale = 10)

# Preparar datos

df\_violin <- data.frame(

valores = c(grupo\_normal, grupo\_bimodal, grupo\_asimetrico),

grupo = factor(rep(c("Normal", "Bimodal", "Asimétrico"), each = 200))

)

# Método 1: Con ggplot2 (principal método)

library(ggplot2)

# Gráfico de violín básico

ggplot(df\_violin, aes(x = grupo, y = valores, fill = grupo)) +

geom\_violin(alpha = 0.7) +

labs(title = "Gráficos de Violín por Grupo",

x = "Grupo",

y = "Valores") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE)

# Gráfico de violín con boxplot superpuesto

ggplot(df\_violin, aes(x = grupo, y = valores, fill = grupo)) +

geom\_violin(alpha = 0.7) +

geom\_boxplot(width = 0.2, fill = "white", alpha = 0.8) +

labs(title = "Gráficos de Violín con Boxplot",

x = "Grupo",

y = "Valores") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE)

# Gráfico de violín con puntos individuales

ggplot(df\_violin, aes(x = grupo, y = valores, fill = grupo)) +

geom\_violin(alpha = 0.7) +

geom\_jitter(width = 0.2, alpha = 0.4, size = 0.8) +

stat\_summary(fun = median, geom = "point", size = 3, color = "black") +

labs(title = "Gráficos de Violín con Puntos y Mediana",

x = "Grupo",

y = "Valores") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE)

# Gráfico de violín horizontal

ggplot(df\_violin, aes(x = grupo, y = valores, fill = grupo)) +

geom\_violin(alpha = 0.7) +

geom\_boxplot(width = 0.1, fill = "white", alpha = 0.8) +

coord\_flip() +

labs(title = "Gráficos de Violín Horizontales",

x = "Grupo",

y = "Valores") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE)

# Comparación: Boxplot vs Violin plot

library(gridExtra)

p1 <- ggplot(df\_violin, aes(x = grupo, y = valores, fill = grupo)) +

geom\_boxplot(alpha = 0.7) +

labs(title = "Boxplot",

x = "Grupo", y = "Valores") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE)

p2 <- ggplot(df\_violin, aes(x = grupo, y = valores, fill = grupo)) +

geom\_violin(alpha = 0.7) +

labs(title = "Violin Plot",

x = "Grupo", y = "Valores") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set2") +

guides(fill = FALSE)

grid.arrange(p1, p2, ncol = 2, top = "Comparación: Boxplot vs Violin Plot")

# Método 2: Con vioplot (librería especializada)

# install.packages("vioplot")

library(vioplot)

# Crear gráfico con vioplot

vioplot(grupo\_normal, grupo\_bimodal, grupo\_asimetrico,

names = c("Normal", "Bimodal", "Asimétrico"),

col = c("lightblue", "lightgreen", "pink"),

main = "Gráficos de Violín con vioplot")

# Añadir boxplots

boxplot(grupo\_normal, grupo\_bimodal, grupo\_asimetrico,

names = c("Normal", "Bimodal", "Asimétrico"),

add = TRUE,

col = rgb(1, 1, 1, 0.8),

outline = FALSE)

# Análisis de la distribución por grupo

analizar\_distribucion <- function(datos, nombre) {

cat("Análisis para grupo:", nombre, "\n")

cat("Media:", round(mean(datos), 2), "\n")

cat("Mediana:", round(median(datos), 2), "\n")

cat("Desviación estándar:", round(sd(datos), 2), "\n")

cat("Asimetría:", round(moments::skewness(datos), 2), "\n")

cat("Curtosis:", round(moments::kurtosis(datos), 2), "\n")

cat("------------------------\n")

}

# Aplicar análisis (requiere: install.packages("moments"))

library(moments)

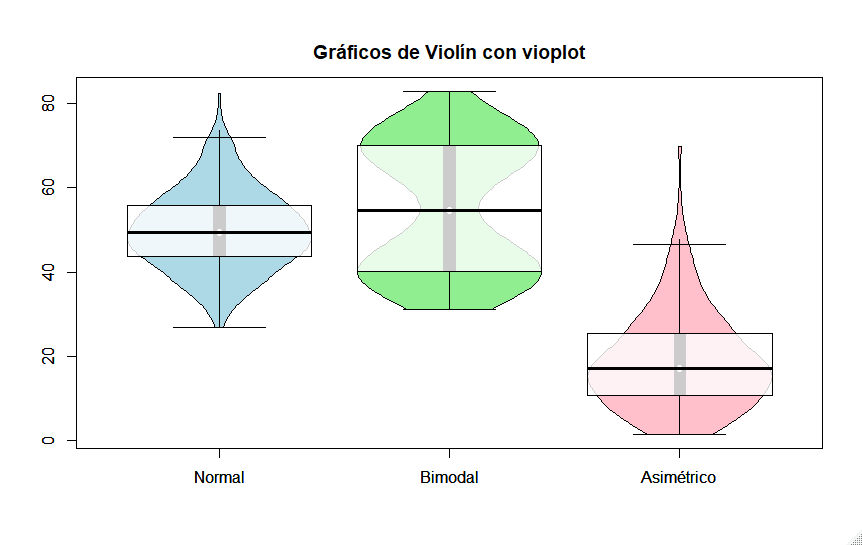
analizar\_distribucion(grupo\_normal, "Normal")

analizar\_distribucion(grupo\_bimodal, "Bimodal")

analizar\_distribucion(grupo\_asimetrico, "Asimétrico")

**Instrucciones paso a paso:**

1. Instalar ggplot2 para método principal
2. Usar geom\_violin() para crear el gráfico básico
3. Combinar con geom\_boxplot() para más información
4. geom\_jitter() para mostrar puntos individuales
5. stat\_summary() para añadir estadísticas (mediana, media)
6. Para vioplot: install.packages("vioplot")
7. Interpretar: anchura = densidad, forma = distribución



## 

## Conclusión

Esta guía cubre los gráficos estadísticos más importantes en R, desde visualizaciones básicas hasta análisis avanzados de distribuciones. Cada tipo de gráfico tiene su propósito específico:

* **Histogramas y densidad**: Para entender distribuciones univariadas
* **Boxplots y violin plots**: Para comparar grupos y detectar outliers
* **Scatter plots**: Para relaciones entre variables
* **Gráficos de líneas**: Para series temporales y tendencias
* **Gráficos de barras**: Para variables categóricas
* **Q-Q plots**: Para evaluar normalidad

### Consejos Generales:

1. Siempre ajusta los colores y títulos para mayor claridad
2. ggplot2 ofrece mayor flexibilidad que R base
3. Combina diferentes tipos de gráficos según sea necesario
4. Considera tu audiencia al elegir el tipo de visualización
5. Incluye estadísticas descriptivas cuando sea relevante