

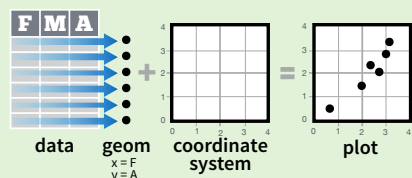
Visualización de Datos usando ggplot2

Guía Rápida

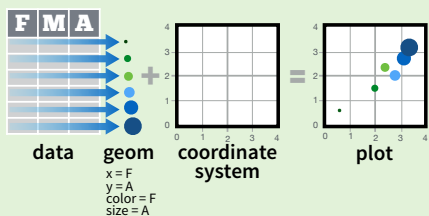


Conceptos Básicos

ggplot2 se basa en la idea que cualquier gráfica se puede construir usando estos tres componentes: **datos**, **coordenadas** y **objetos geométricos (geoms)**. Este concepto se llama: **gramática de las gráficas**.



Para visualizar resultados, asigne variables a las propiedades visuales, o **estéticas**, como **tamaño**, **color** y **posición x o y**.



Para construir una gráfica complete este patrón

```
ggplot(data = <DATOS>) +
  <FUNCION_GEOM> (
    mapping =aes( <ESTETICAS> ),
    stat = <STAT> ,
    position = <POSICION>
  ) +
  <FUNCION_COORDINADAS> +
  <FUNCION_FACETA> +
  <FUNCION_ESCALA> +
  <FUNCION_TEMA>
```

Requerido

No Requerido, se proveen valores iniciales

```
ggplot(data = mpg, aes(x = cty, y = hwy))
```

Este comando comienza una gráfica, complétela mediante agregando capas, un **geom** por capa.



```
qplot(x = cty, y = hwy, data = mpg, geom = "point")
```

Este comando es una gráfica completa, tiene datos, las estéticas están asignadas y por lo menos un geom.

```
last_plot()
```

Devuelve la última gráfica

```
ggsave("plot.png", width = 5, height = 5)
```

La última gráfica es grabada como una imagen de 5 por 5 pulgs., usa el mismo tipo de archivo que la extensión

Geoms - Funciones geom se utilizan para visualizar resultados. Asigne variables a las propiedades estéticas del geom. Cada geom forma una capa.

Geométricas Elementales

- `a <- ggplot(economics, aes(date, unemploy))`
- `b <- ggplot(seals, aes(x = long, y = lat))`
- `a + geom_blank()`
(Bueno para expandir límites)
- `b + geom_curve(aes(yend = lat + 1, xend=long+1,curvature=z))` - x, xend, y, yend, alpha, angle, color, curvature, linetype, size
- `a + geom_path(lineend="butt", linejoin="round", linemitre=1)`
x, y, alpha, color, group, linetype, size
- `a + geom_polygon(aes(group = group))`
x, y, alpha, color, fill, group, linetype, size
- `b + geom_rect(aes(xmin = long, ymin=lat, xmax= long + 1, ymax = lat + 1))` - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size
- `a + geom_ribbon(aes(ymin=unemploy - 900, ymax=unemploy + 900))` - x, ymax, ymin, alpha, color, fill, group, linetype, size

Segmentos Lineares

propiedades básicas: x, y, alpha, color, linetype, size

- `b + geom_abline(aes(intercept=0, slope=1))`
- `b + geom_hline(aes(yintercept = lat))`
- `b + geom_vline(aes(xintercept = long))`
- `b + geom_segment(aes(yend=lat+1, xend=long+1))`
- `b + geom_spoke(aes(angle = 1:1155, radius = 1))`

Una Variable

Continua

- `c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)`
- `c + geom_area(stat = "bin")`
x, y, alpha, color, fill, linetype, size
- `c + geom_density(kernel = "gaussian")`
x, y, alpha, color, fill, group, linetype, size, weight
- `c + geom_dotplot()`
x, y, alpha, color, fill
- `c + geom_freqpoly()`
x, y, alpha, color, group, linetype, size
- `c + geom_histogram(binwidth = 5)`
x, y, alpha, color, fill, linetype, size, weight
- `c2 + geom_qq(aes(sample = hwy))`
x, y, alpha, color, fill, linetype, size, weight

Discreta

- `d <- ggplot(mpg, aes(fl))`
- `d + geom_bar()`
x, alpha, color, fill, linetype, size, weight

Dos Variables

X Continua, Y Continua

- `e <- ggplot(mpg, aes(cty, hwy))`
- `e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE)`
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust
- `e + geom_jitter(height = 2, width = 2)`
x, y, alpha, color, fill, shape, size
- `e + geom_point()`
x, y, alpha, color, fill, shape, size, stroke
- `e + geom_quantile()`
x, y, alpha, color, group, linetype, size, weight
- `e + geom_rug(sides = "bl")`
x, y, alpha, color, linetype, size
- `e + geom_smooth(method = lm)`
x, y, alpha, color, fill, group, linetype, size, weight
- `e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE)`
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

X Discreta, Y Continua

- `f <- ggplot(mpg, aes(class, hwy))`
- `f + geom_col()`
x, y, alpha, color, fill, group, linetype, size
- `f + geom_boxplot()`
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight
- `f + geom_dotplot(binaxis = "y", stackdir = "center")`
x, y, alpha, color, fill, group
- `f + geom_violin(scale = "area")`
x, y, alpha, color, fill, group, linetype, size, weight

X Discreta, Y Discreta

- `g <- ggplot(diamonds, aes(cut, color))`
- `g + geom_count()`
x, y, alpha, color, fill, shape, size, stroke

Tres Variables

- `seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))`
- `l <- ggplot(seals, aes(long, lat))`
- `l + geom_raster(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE)`
x, y, alpha, fill
- `l + geom_tile(aes(fill = z))`
x, y, alpha, color, fill, linetype, size, width

Distribución Bivariada Continua

- `h <- ggplot(diamonds, aes(carat, price))`
- `h + geom_bin2d(binwidth = c(0.25, 500))`
x, y, alpha, color, fill, linetype, size, weight
- `h + geom_density2d()`
x, y, alpha, colour, group, linetype, size
- `h + geom_hex()`
x, y, alpha, colour, fill, size

Función Continua

- `i <- ggplot(economics, aes(date, unemploy))`
- `i + geom_area()`
x, y, alpha, color, fill, linetype, size
- `i + geom_line()`
x, y, alpha, color, group, linetype, size
- `i + geom_step(direction = "hv")`
x, y, alpha, color, group, linetype, size

Visualizando el Error

- `df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)`
- `j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))`
- `j + geom_crossbar(fatten = 2)`
x, y, ymax, ymin, alpha, color, fill, group, linetype, size
- `j + geom_errorbar()`
x, ymax, ymin, alpha, color, group, linetype, size, width (also `geom_errorbarh()`)
- `j + geom_linerange()`
x, ymin, ymax, alpha, color, group, linetype, size
- `j + geom_pointrange()`
x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

Mapas

- `data <- data.frame(murder = USArrests$Murder, state = tolower(rownames(USArrests)))`
- `map <- map_data("state")`
- `k <- ggplot(data, aes(fill = murder))`
- `k + geom_map(aes(map_id = state), map = map) + expand_limits(x = map$long, y = map$lat)`
map_id, alpha, color, fill, linetype, size

Argumentos

Traducción de argumentos comunes
label = etiqueta, angle=ángulo
size=tamaño, weight=peso
alpha=transparencia
fontface=tipo de letra
hjust=ajuste horizontal
lineheight=grosor de línea

