

Sección 2

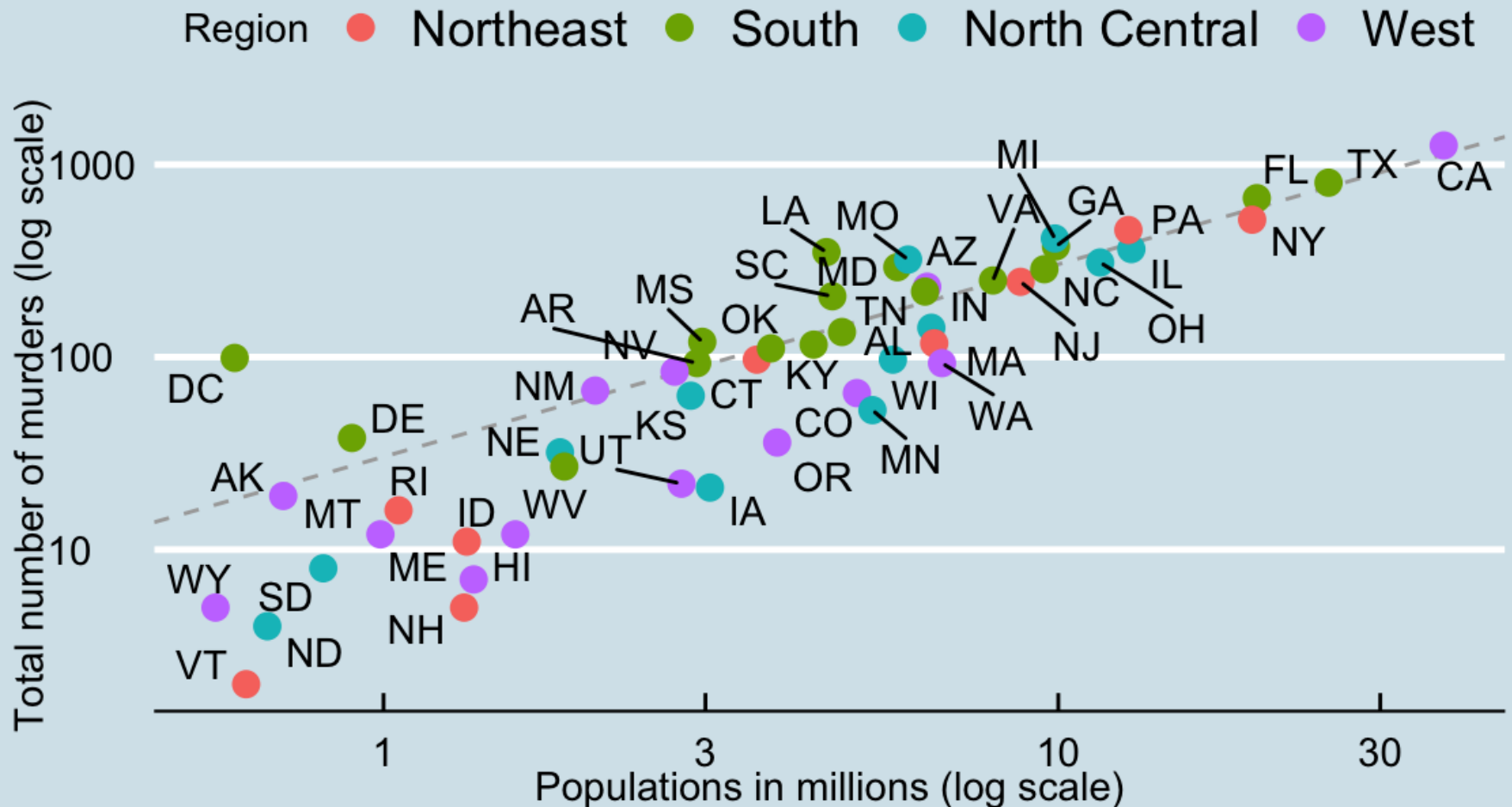
[Paquete ggplot2](#)

ggplot2

- Utilizar ggplot2 para realizar visualizaciones de datos en R
- Poder explicar cada componente gráfico
- Identificar cual es el componente estético del gráfico más apropiado
- Entender y aplicar correctamente el componente escala.

Primer utilización ggplot2

US Gun Murders in 2010



Componentes gráficos

Existen 3 componentes principales

1. Datos: Dataset que queremos visualizar
2. Geometría: Tipo de gráfico (scatterplot, boxplot, barplot, histogram, qqplot, smooth density, etc.)
3. Mapeo estético: Variables utilizadas para mapear el gráfico (eje x, el el eje y y los colores)

Componentes adicionales

1. Escala
2. Títulos, textos internos, referencias, estilo del gráfico
3. Estilo/ formato de texto y gráfico

Capas de gráficos -Layers-

- Para crear gráficos agregamos capas '+'

```
DATA %>% ggplot() + CAPA1 + CAPA2 + ... + CAPAN
```

- Las capas definen componentes
 - Posición de los ejes
 - Color
 - Tamaño
- Con **aes()** se conectan - mapean los datos a sus características en los gráficos
- La función Aesthetic mappings no requiere de nombres de variables:
 - Ej. utilizar 'total' en lugar de 'murders\$total'

Definimos objeto gráfico

Para un gráfico de cantidad de asesinatos por ej:

```
library(tidyverse)
```

```
library(dslabs)
```

```
data(murders)
```

```
p <- murders %>% ggplot(aes(population/10^6, total,  
label = abb))
```

- **Datos:** Podemos asociar un dataset a un objeto ggplot de 3 maneras:

```
ggplot(data = x)
```

```
ggplot(x)
```

```
x %>% ggplot()
```

Imprimimos un objeto gráfico

```
library(tidyverse)
```

```
library(dslabs)
```

```
data(murders)
```

```
ggplot(data = murders)
```

```
murders %>% ggplot()
```

```
p <- ggplot(data = murders)
```

```
class(p)
```

```
print(p)      # equivalente a solo escribir p
```

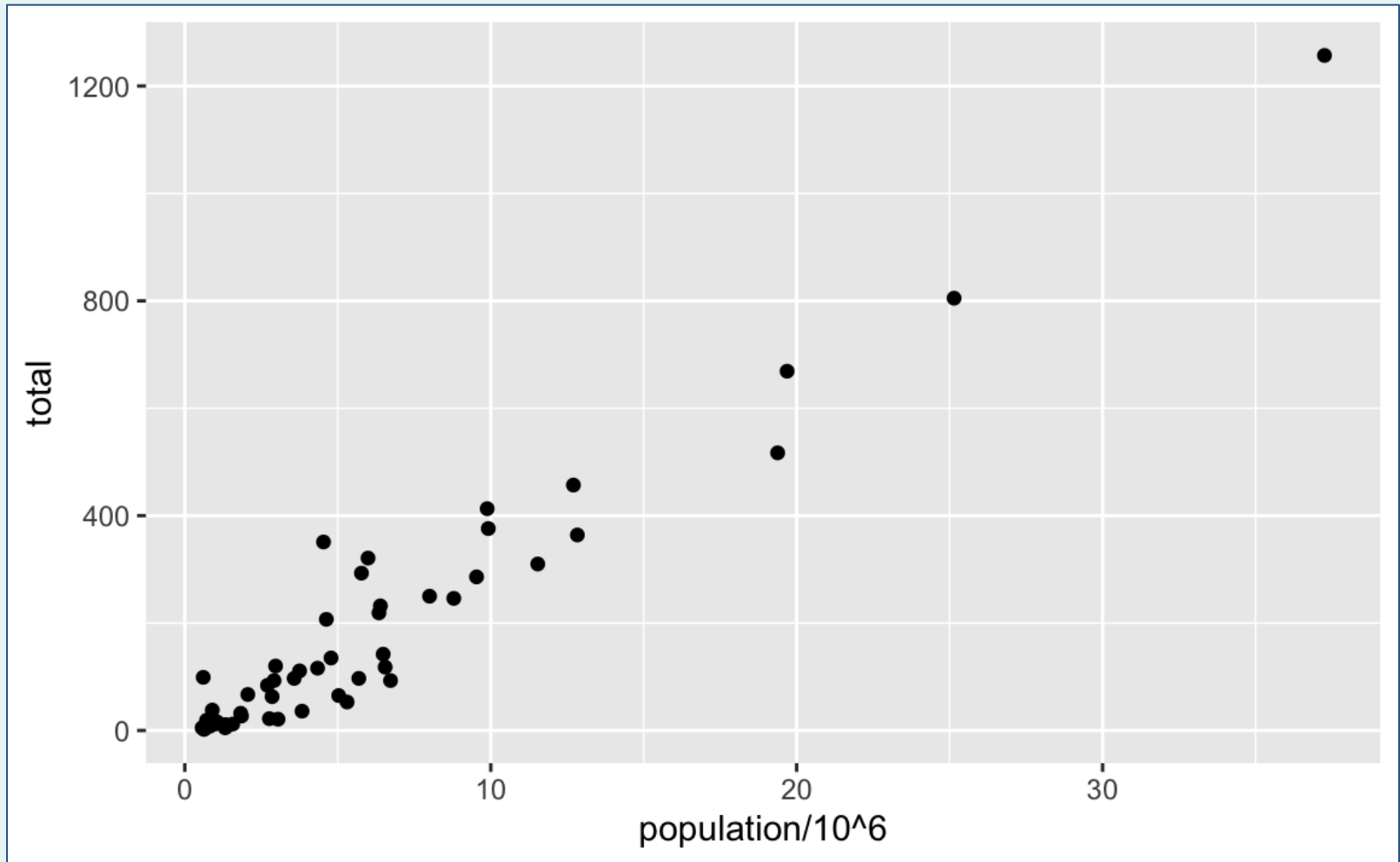
```
p
```

Definimos componentes -Geometría-

- El componente geometría se define en la capa como el tipo de gráfico a utilizar.
- Utilizamos la función aes para conectar los datos a lo que vemos en el gráfico
 - Ej. Para un gráfico de población vs cantidad de asesinatos por estado:

```
murders %>% ggplot() +  
  geom_point(aes(x = population/10^6, y = total))  
  
p <- ggplot(data = murders) #De otra forma  
p + geom_point(aes(population/10^6, total))
```


Resultado de ejecución

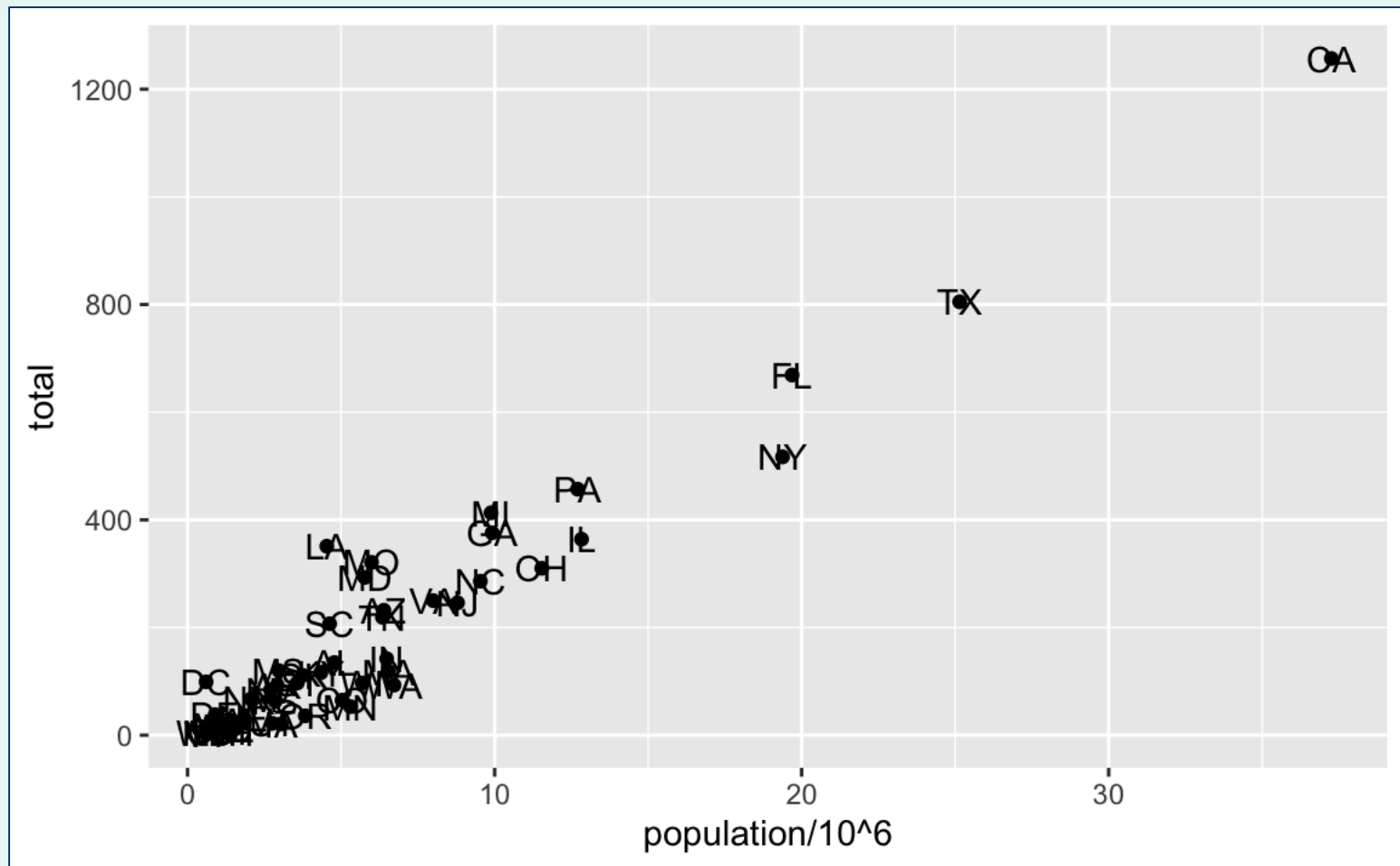


Segunda capa - textos

- Para agregar textos al gráfico de asesinatos utilizamos:
 - `geomlabel()`: Agrega el texto con rectángulos pequeños
 - `geomtext()`: agrega el texto

```
p + geom_point(aes(population/10^6, total)) +  
geom_text(aes(population/10^6, total, label = abb))
```

Resultado de ejecución



Comportamiento aes

- Notemos la diferencia entre los comandos:

```
p_test <- p + geom_text(aes(population/10^6, total,  
label = abb))
```

```
> p_test <- p + geom_text(aes(population/10^6, total), label  
= abb)  
Error in layer(data = data, mapping = mapping, stat = stat,  
geom = GeomText, :  
objeto 'abb' no encontrado
```

- Definimos un mapeo global para todos los gráficos

```
p <- murders %>% ggplot(aes(population/10^6, total,  
label = abb))
```

Parámetros componente geometría

- Argumentos de entrada:

- Tamaño de los puntos

```
p + geom_point(aes(population/10^6, total), size = 3) +  
geom_text(aes(population/10^6, total, label = abb))
```

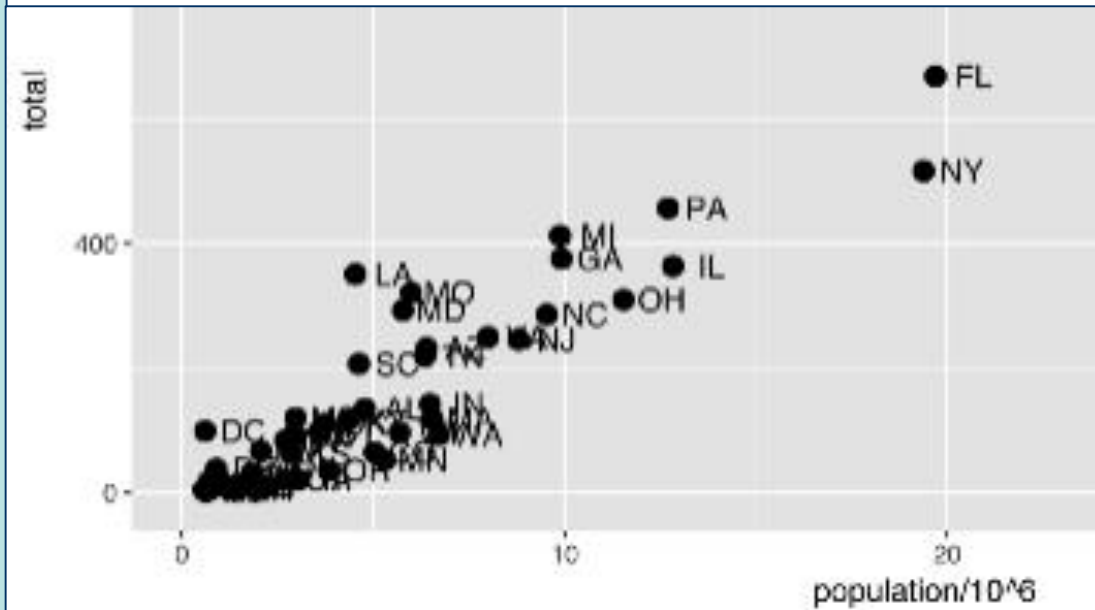
- Aumento de espaciado entre puntos - textos

```
p + geom_point(aes(population/10^6, total), size = 3) +  
geom_text(aes(population/10^6, total, label = abb),  
nudge_x = 1)
```

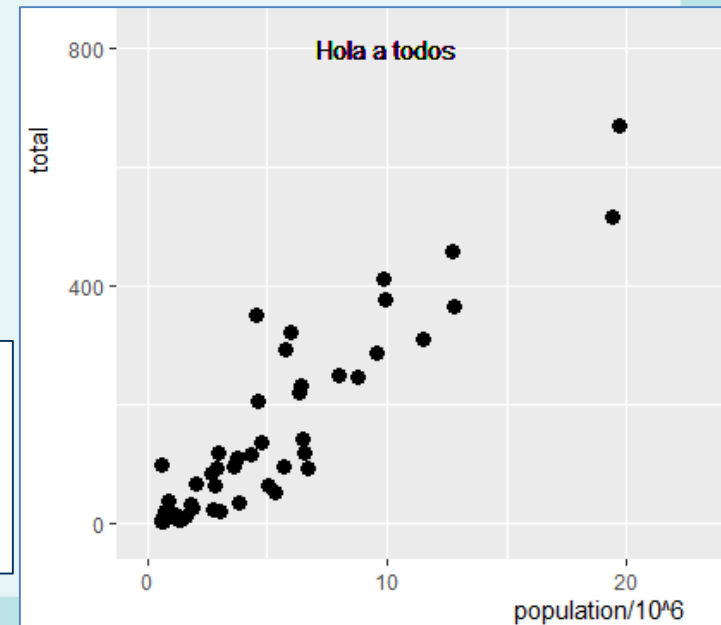
Código anterior con aes globales

```
p <- murders %>% ggplot(aes(population/10^6, total, label = abb))
```

```
p + geom_point(size = 3) + geom_text(nudge_x = 1.5)
```



```
p +  
geom_text(aes(x=10, y=800, label="Hola  
a todos"))
```

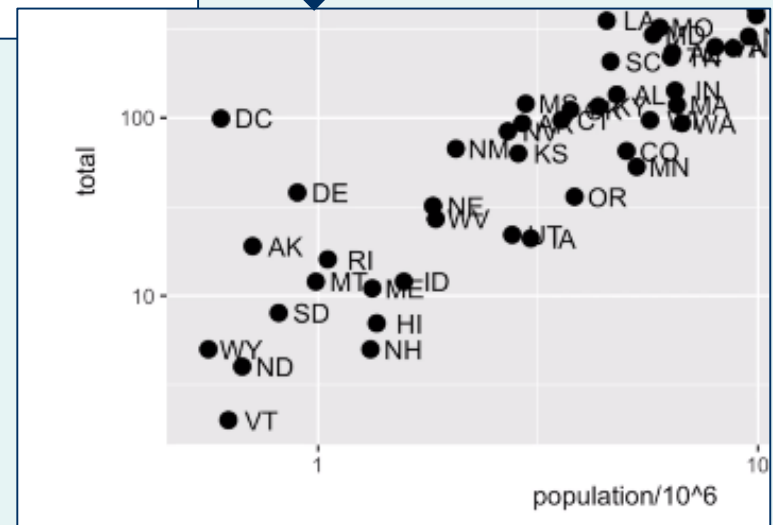


Customización de gráficos

- Escala logarítmica

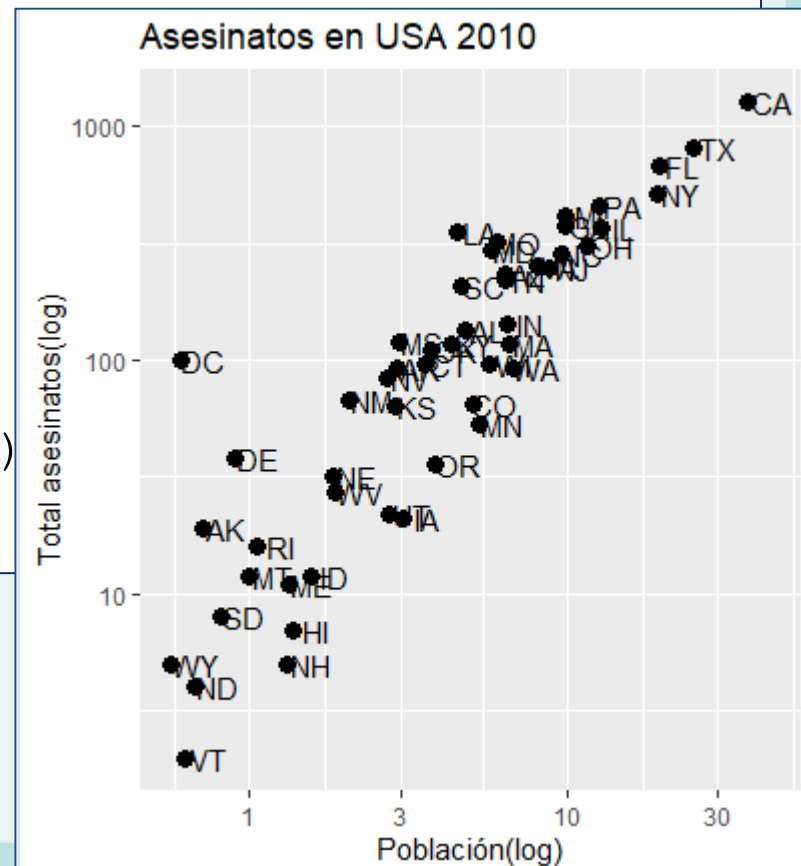
```
p + geom_point(size = 3) +  
geom_text(nudge_x = 0.05) +  
scale_x_continuous(trans = "log10") +  
scale_y_continuous(trans = "log10")
```

```
p + geom_point(size = 3) +  
geom_text(nudge_x = 0.075) +  
scale_x_log10() +  
scale_y_log10()
```



Agregamos textos y títulos

```
p + geom_point(size = 3) +  
  geom_text(nudge_x = 0.075) +  
  scale_x_log10() +  
  scale_y_log10() +  
  xlab("Población(log)") +  
  ylab("Total asesinatos(log)") +  
  ggtitle("Asesinatos en USA 2010")
```



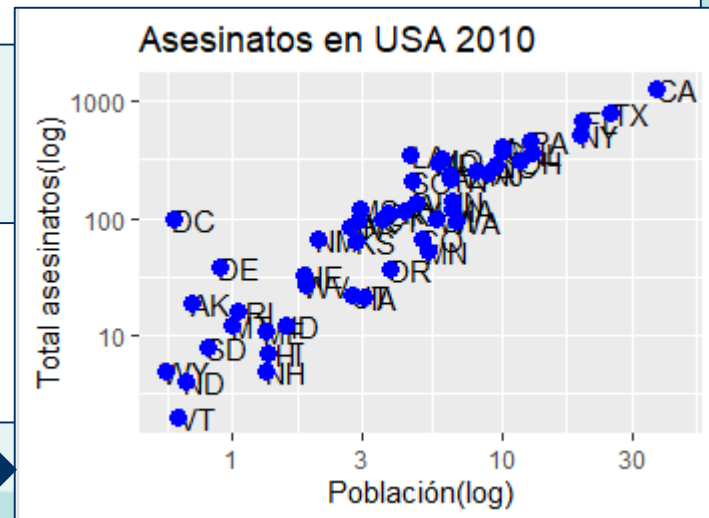
Agregamos color

- Re definimos el objeto p sin capa de puntos:

```
p <- murders %>% ggplot(aes(population/10^6, total, label =  
abb)) + geom_text(nudge_x = 0.075) + scale_x_log10() +  
scale_y_log10() + xlab("Población(log)") + ylab("Total  
asesinatos(log)") + ggtitle("Asesinatos en USA 2010")
```

- Cambiamos el color a azul

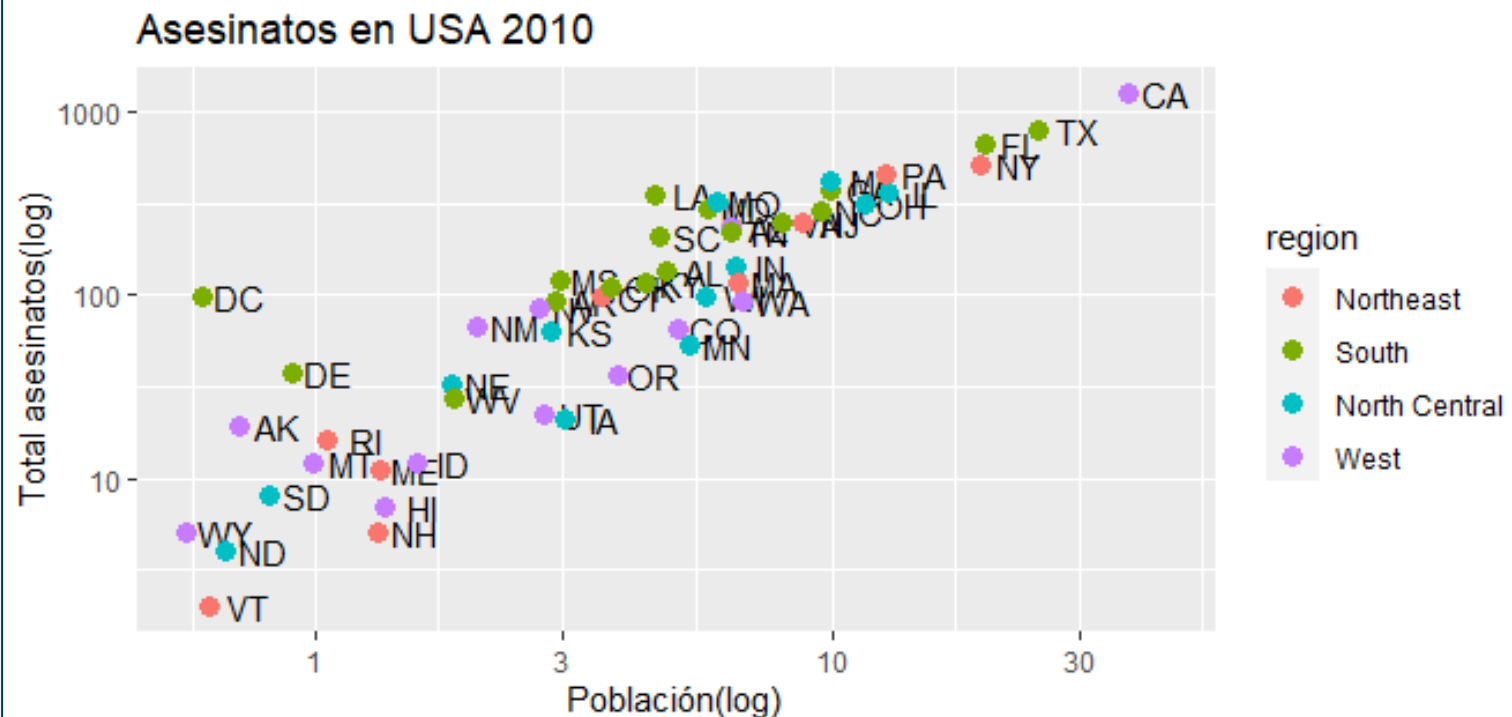
```
p + geom_point(size = 3, color = "blue")
```



Mapeamos el parámetro color

- Asignamos a cada estado un color de acuerdo a su región

```
p + geom_point(aes(col = region), size = 3)
```



Geometrías de líneas

- Verticales: `geom_vline(xintercept)`
 - `xintercept`, punto de corte con el eje x
- Horizontales: `geom_hline(yintercept)`
 - `yintercept`: punto de corte con el eje y
- Diagonales: `geom_abline(intercept , slope)`
 - `intercept`, punto de corte con el eje y
 - `slope`, pendiente por defecto = 1
- En el ejemplo, dibujamos una línea para identificar la frecuencia o índice de asesinatos:

```
r <- sum(murders$total) / sum(murders$population) * 10^6  
p <- p + geom_point(aes(col = region), size = 3) +  
  geom_abline(intercept = log10(r))
```

Ajustes en estilo

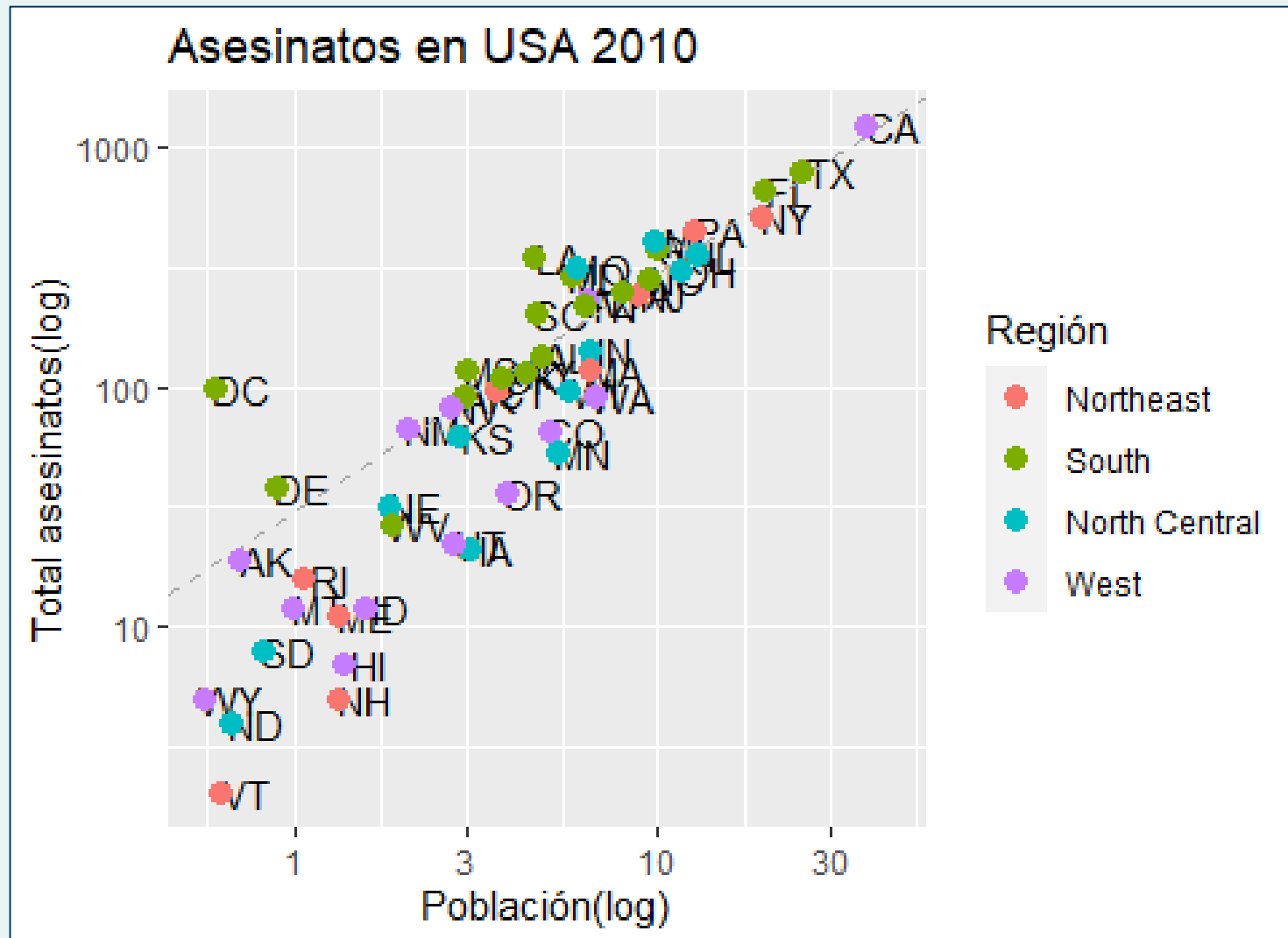
- Modificamos el estilo de la línea de frec. de asesinatos

```
p + geom_abline(intercept = log10(r), lty = 2, color =  
"darkgrey") +  
  geom_point(aes(col = region), size = 3)
```

- Cambiamos el título de las referencias

```
p <- p + scale_color_discrete(name = "Región")
```

Obtenemos el gráfico



Paquetes AddOn -ggthemes-

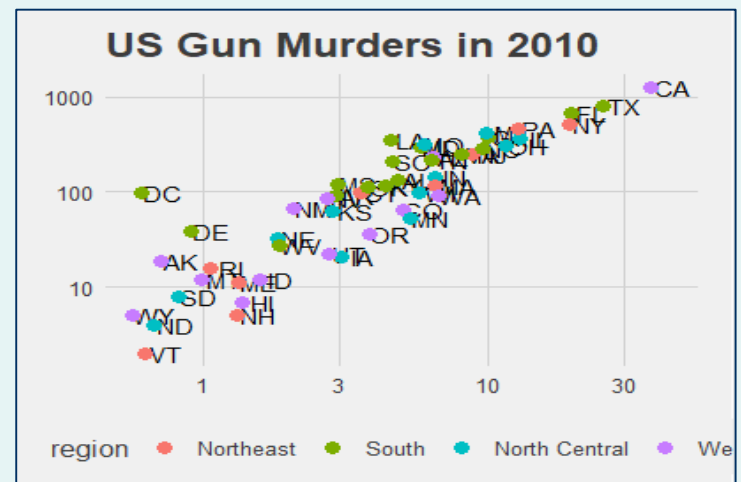
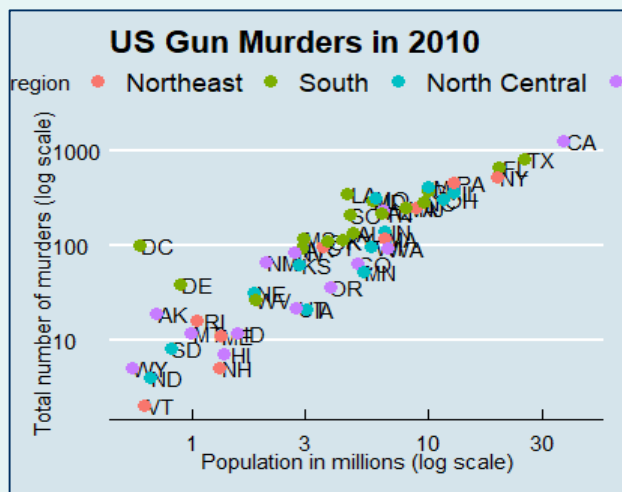
Este paquete contiene estilos de gráficos, muchos estilos predeterminados incluidos, ej:

[theme_few\(\)](#), [theme_solarized\(\)](#),
[theme_fivethirtyeight\(\)](#)

```
library(ggthemes)
```

```
p <- p + theme_economist()
```

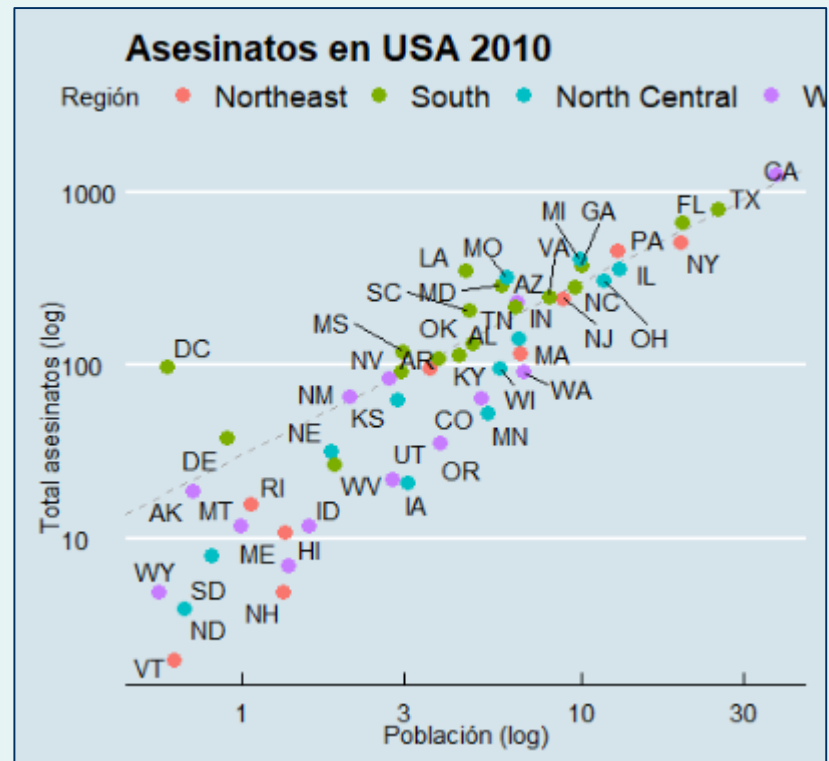
```
p <- p + theme_fivethirtyeight()
```



Paquetes AddOn -ggrepel-

Asegura que las capas no se superpongan entre sí.
Por ejemplo los puntos del texto en nuestro gráfico de asesinatos.

```
library(ggrepel)  
p <- p + geom_text_repel()
```



Código completo para el gráfico anterior

```
# Cargamos las librerías
library(tidyverse)
library(dslabs)
library(ggrepel)
library(ggthemes)
#Creamos dataset
data(murders)

# Definimos la recta a graficar con la función intercept
r <- murders %>%
  summarize(rate = sum(total) / sum(population) * 10^6) %>%
  .$rate

# Realizamos el gráfico combinando todos los elementos y capas
murders %>%
  ggplot(aes(population/10^6, total, label = abb)) +
  geom_abline(intercept = log10(r), lty = 2, color = "darkgrey") +
  geom_point(aes(col = region), size = 3) +
  geom_text_repel() +
  scale_x_log10() +
  scale_y_log10() +
  xlab("Población (log)") +
  ylab("Total asesinatos (log)") +
  ggtitle("Asesinatos en USA 2010") +
  scale_color_discrete(name = "Región") +
  theme_economist()
```


Histograma de alturas de hombres

```
# Cargamos los datos de alturas
```

```
library(tidyverse)
```

```
library(dslabs)
```

```
data(heights)
```

```
# Definimos el objeto gráfico p
```

```
p <- heights %>%
```

```
  filter(sex == "Male") %>%
```

```
  ggplot(aes(x = height))
```

```
# histogramas básicos
```

```
p + geom_histogram()
```

```
#Histograma con tamaño de barras=1, relleno =azul, borde= negro
```

```
p + geom_histogram(binwidth = 1, fill = "blue", col = "black") +
```

```
  xlab("Altura de hombres en inches") +
```

```
  ggtitle("Histograma")
```

```
#Xlab setea texto de eje x y ggtitle título de gráfico
```

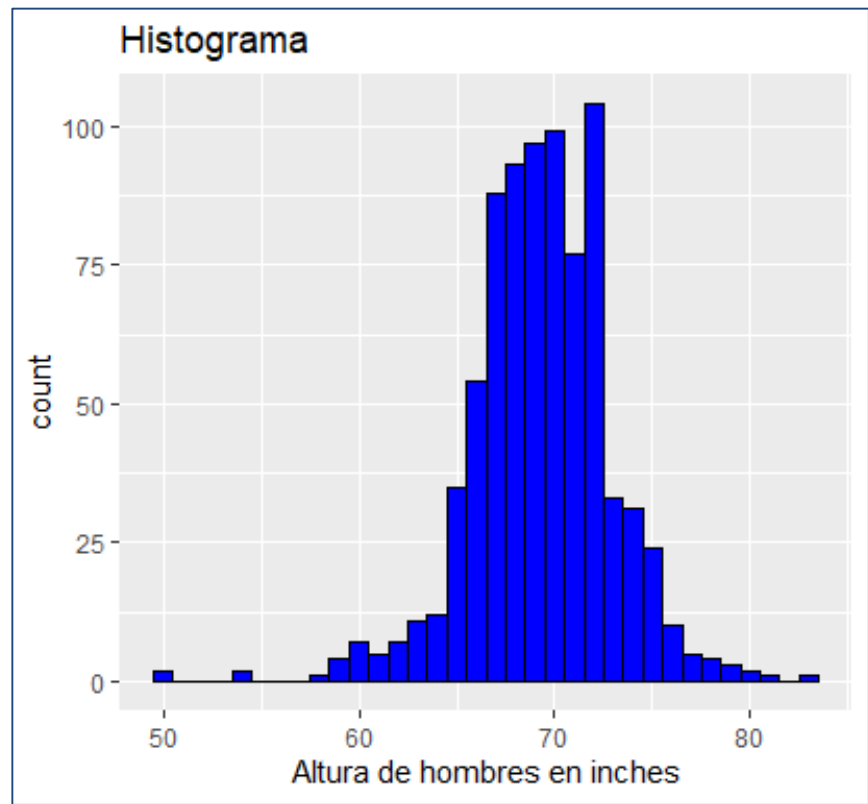


Gráfico de densidad de alturas

```
# Cargamos los datos de alturas
```

```
library(tidyverse)
```

```
library(dslabs)
```

```
data(heights)
```

```
# Definimos el objeto gráfico p
```

```
p <- heights %>%
```

```
  filter(sex == "Male") %>%
```

```
  ggplot(aes(x = height))
```

```
# histogramas básicos
```

```
p + geom_density()
```

```
p + geom_density(fill = "blue") +
```

```
xlab("Altura de hombres en inches") +
```

```
ggtitle("Grafico de densidad")
```

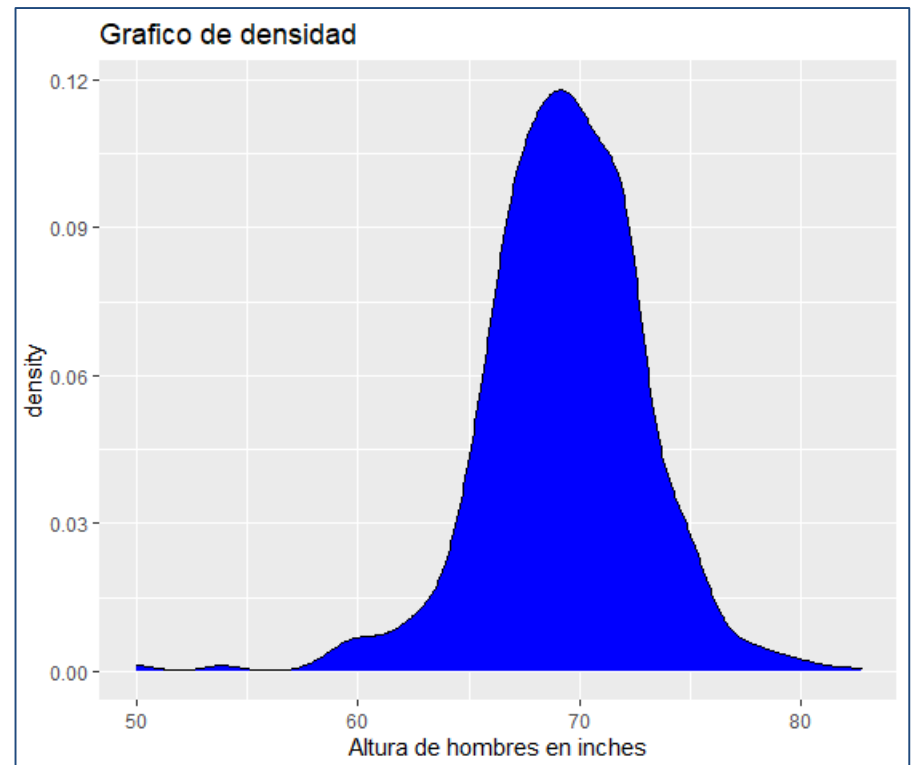


Gráfico Q-Q

```
# Cargamos los datos de alturas
```

```
library(tidyverse)
```

```
library(dslabs)
```

```
data(heights)
```

```
# QQ-plot con promedio= 0 sd=1
```

```
p <- heights %>% filter(sex == "Male") %>%
```

```
  ggplot(aes(sample = height))
```

```
p + geom_qq()
```

```
# QQ-plot con promedio y sd de datos
```

```
params <- heights %>%
```

```
  filter(sex == "Male") %>%
```

```
  summarize(mean = mean(height), sd = sd(height))
```

```
p + geom_qq(dparams = params) +
```

```
  geom_abline() # Distribución normal
```

```
# QQ-plot vs distribución normal con promedio y sd de datos en escala
```

```
heights %>%
```

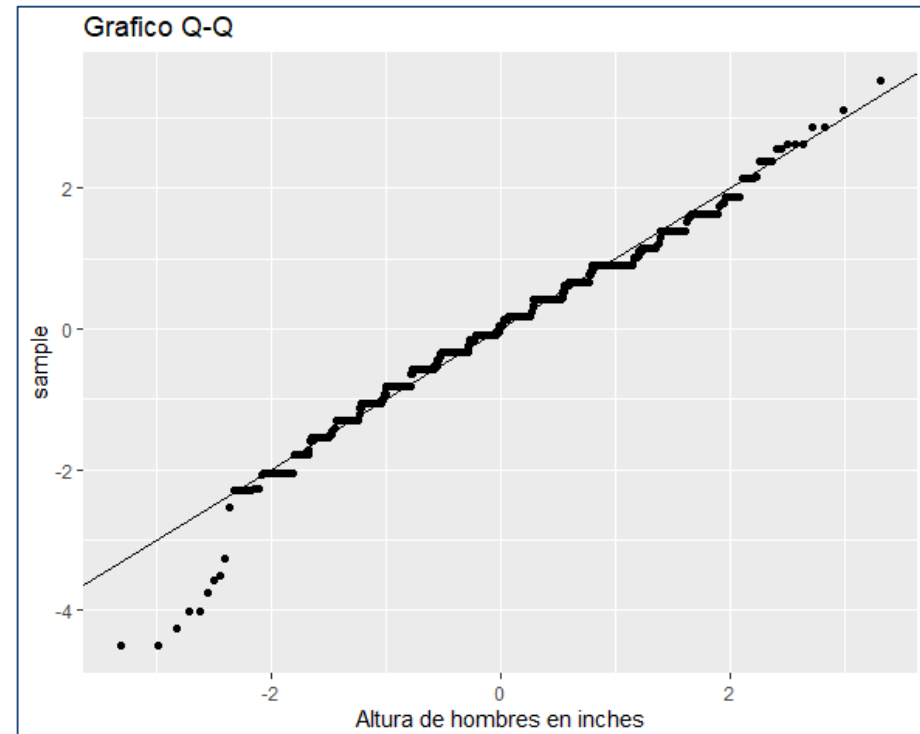
```
  ggplot(aes(sample = scale(height))) +
```

```
  geom_qq() +
```

```
  geom_abline() +
```

```
  xlab("Altura de hombres en inches") +
```

```
  ggtitle("Grafico Q-Q")
```



3 gráficos simultáneos

Definimos 3 gráficos p1, p2, p3

```
p <- heights %>% filter(sex == "Male") %>% ggplot(aes(x = height))  
p1 <- p + geom_histogram(binwidth = 1, fill = "blue", col = "black")  
p2 <- p + geom_histogram(binwidth = 2, fill = "blue", col = "black")  
p3 <- p + geom_histogram(binwidth = 3, fill = "blue", col = "black")
```

Colocamos los 3 gráficos en una grilla de 1 fila, 3 columnas

```
library(gridExtra)
```

```
grid.arrange(p1, p2, p3, ncol = 3)
```

