

# **ggplot2 (continuación)**

**Entornos de Análisis de Datos: R**

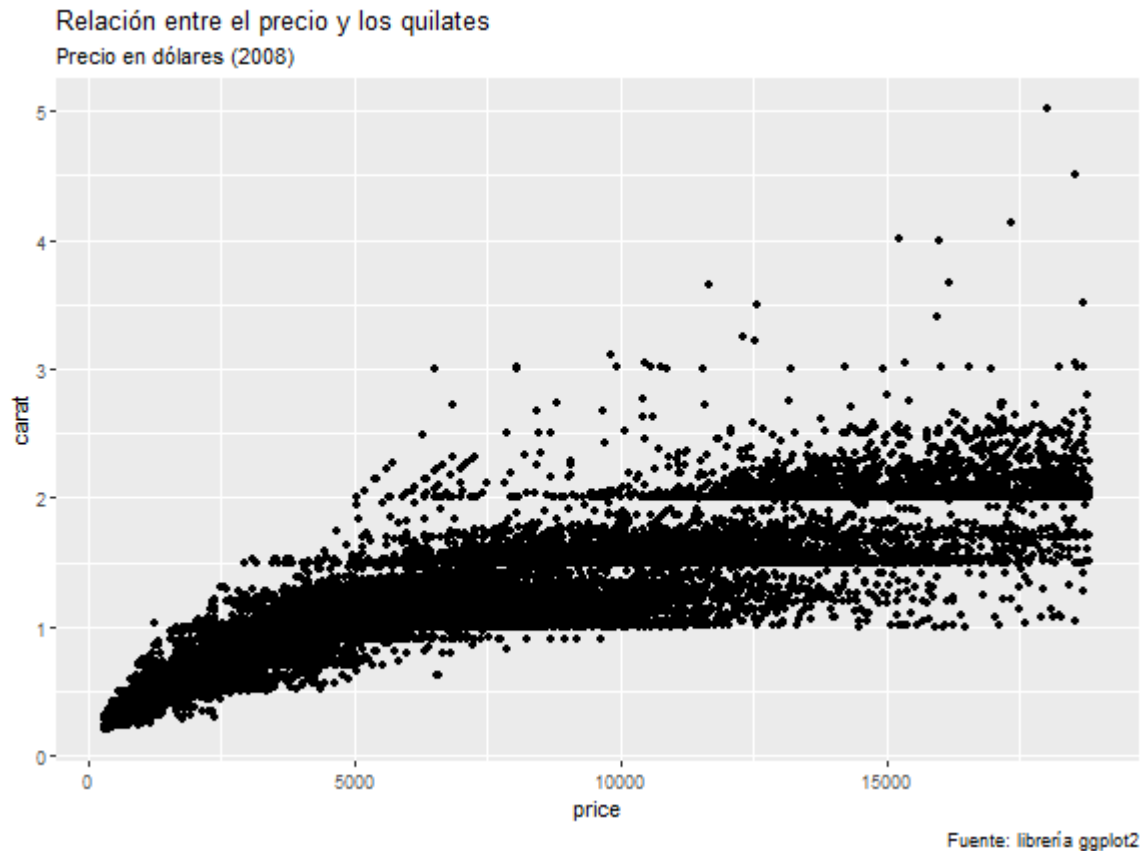
**Alberto Torres**

**2020-02-02**

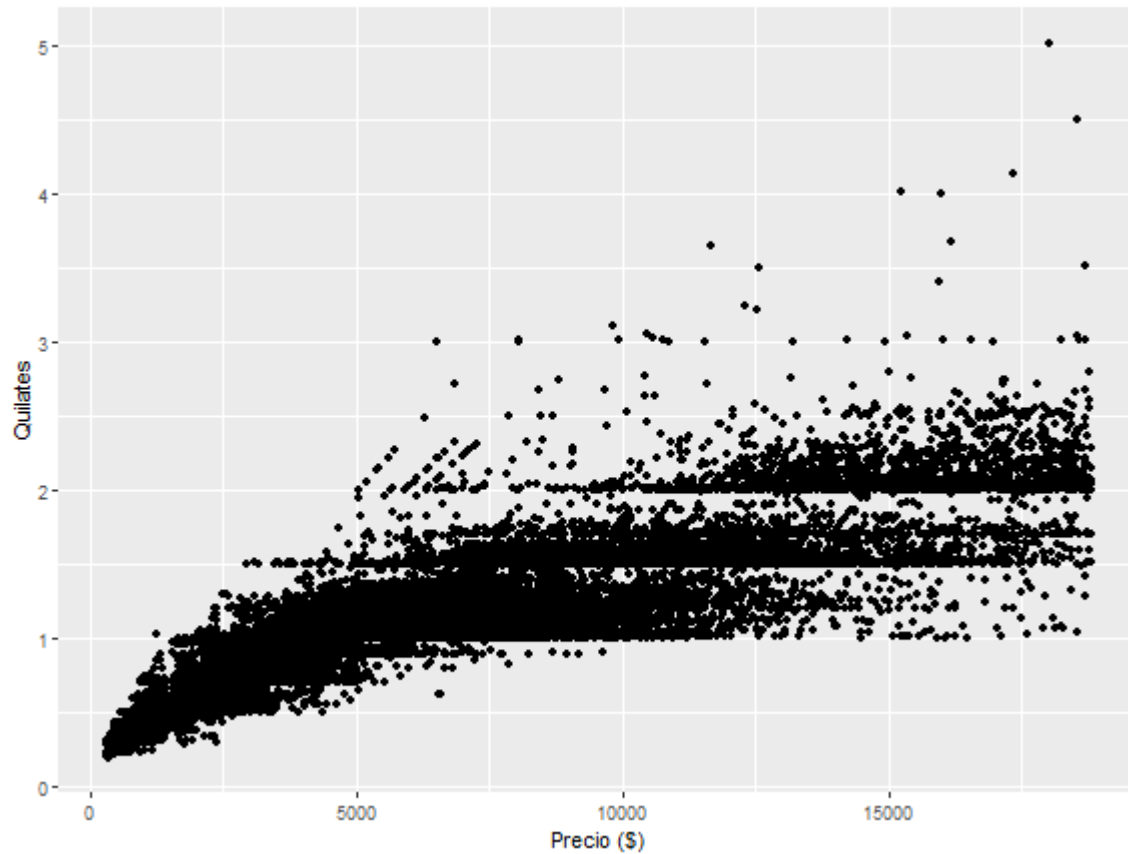
# Etiquetas

- se añaden con la función `labs()` :
  - `title` : título, en la parte superior
  - `subtitle` : debajo del título
  - `caption` : debajo del gráfico, a la derecha
  - `x` : eje x
  - `y` : eje y
- también existen las funciones `ggtitle()`, `xlab()` e `ylab()`

```
ggplot(diamonds, aes(x = price, y = carat)) +
  geom_point() +
  labs(
    title = "Relación entre el precio y los quilates",
    subtitle = "Precio en dólares (2008)",
    caption = "Fuente: librería ggplot2"
  )
```



```
ggplot(diamonds, aes(x = price, y = carat)) +  
  geom_point() +  
  xlab("Precio ($)") +  
  ylab("Quilates")
```



# Anotaciones

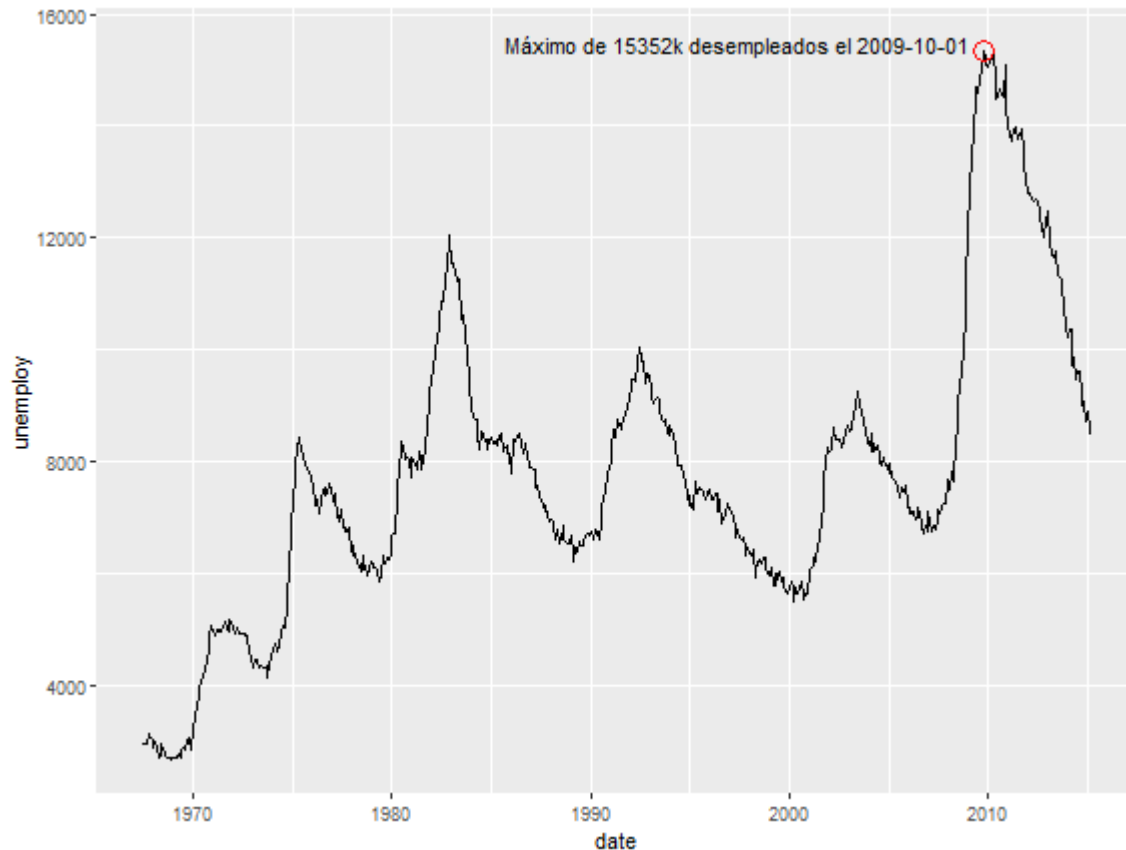
- Funciones `geom_text()` y `annotate()`
  - `geom_text()` : capa de texto (datos en un data frame)
  - `annotate()` : podemos pasar los datos como vectores
- Otras funciones:
  - `geom_hline()` , `geom_vline()` y `geom_abline()` : líneas rectas, verticales u horizontales
  - `geom_rect()` : rectángulo
  - `geom_segment()` : flecha

```

max_idx <- which.max(economics$unemploy)
max_data <- slice(economics, max_idx)

ggplot(economics, aes(x = date, y = unemploy)) +
  geom_line() +
  geom_text(data = max_data,
            label = str_glue_data(max_data, "Máximo de {unemploy}k desempleados el",
                                  nudge_y = 100, nudge_x = -250, hjust = "right") +
            annotate(geom = "point", x = max_data$date, y = max_data$unemploy,
                    color = "red", shape = 21, fill = "transparent", size = 5)

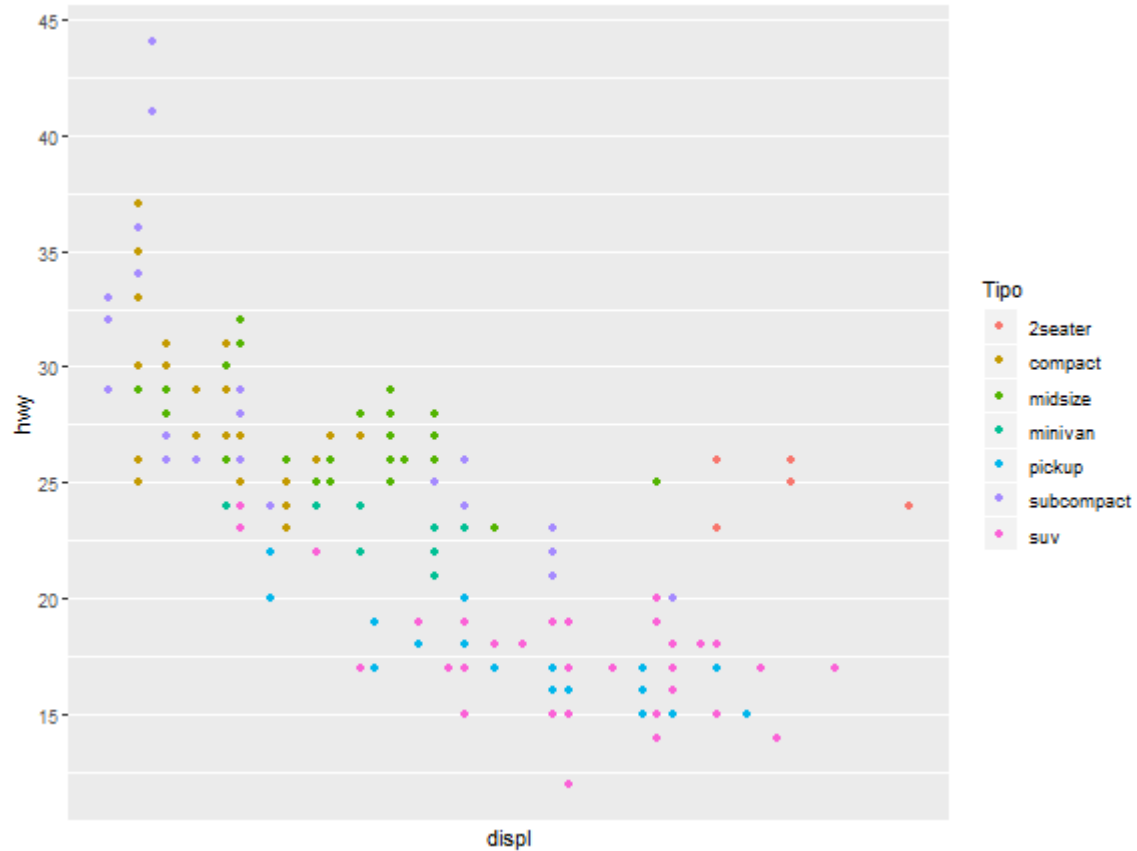
```



# Escalas

- Siempre se añaden escalas por defecto
- Se pueden cambiar con las funciones:
  - `xlim()` , `ylim()` : reducir o ampliar las escalas de los ejes x e y
  - Familia `scale_<AES>_<TIPO>()`
- Algunos ejemplos:
  - `scale_x_log10()` , `scale_y_log10()`
  - `scale_x_continuous()` , `scale_y_continuous()`
  - `scale_x_discrete()` , `scale_y_discrete()`
  - `scale_x_date()` , `scale_y_date()`
  - `scale_color_discrete()` , `scale_color_continuous()`

```
ggplot(mpg, aes(x = displ, y = hwy, color = class)) +
  geom_point() +
  scale_y_continuous(breaks = seq(10, 50, by = 5)) +
  scale_x_continuous(labels = NULL, breaks = NULL) +
  scale_color_discrete(name = "Tipo")
```



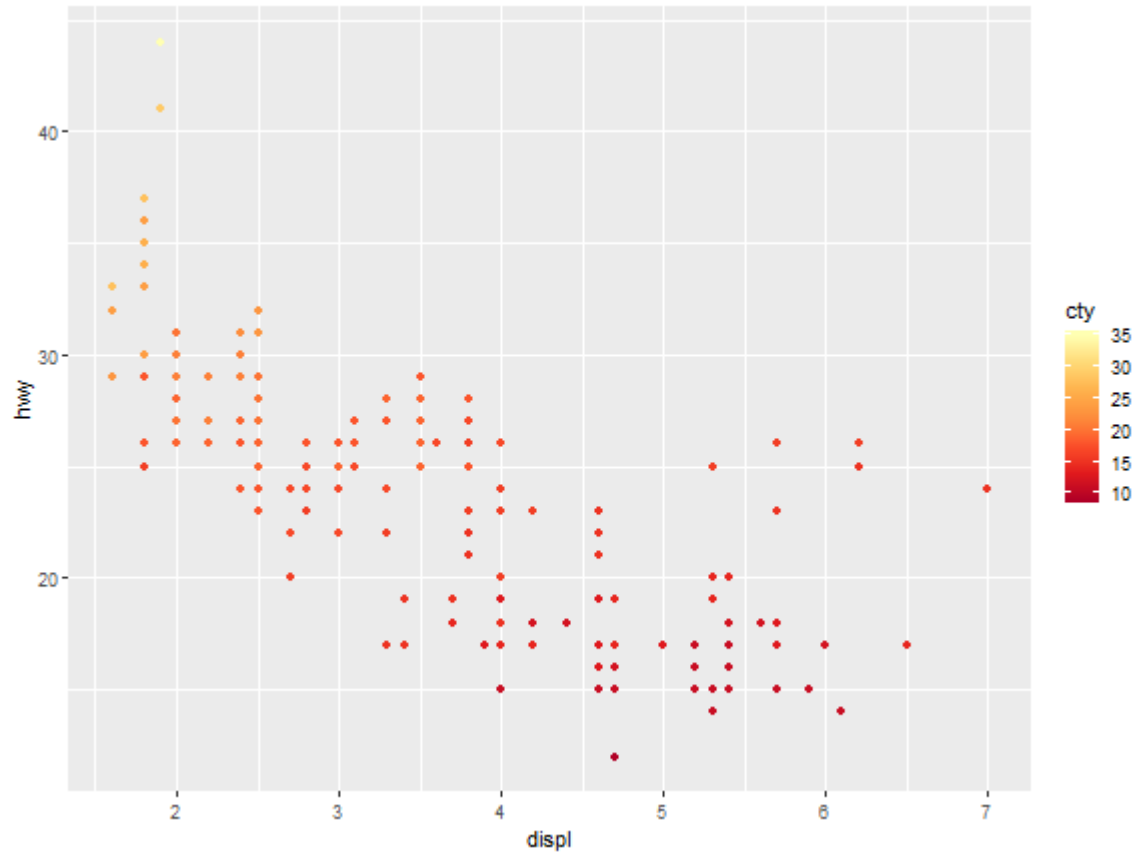


# Colores

- Para personalizar los colores del gráfico se puede usar una de las múltiples escalas predefinidas en <http://colorbrewer2.org/>
  - `scale_color_brewer()` para escalas discretas
  - `scale_color_distiller()` para escalas continuas
- También se puede establecer la escala de forma manual
  - `scale_color_manual()` para escalas discretas
  - `scale_color_gradient()` o `scale_color_gradient2()` para escalas continuas

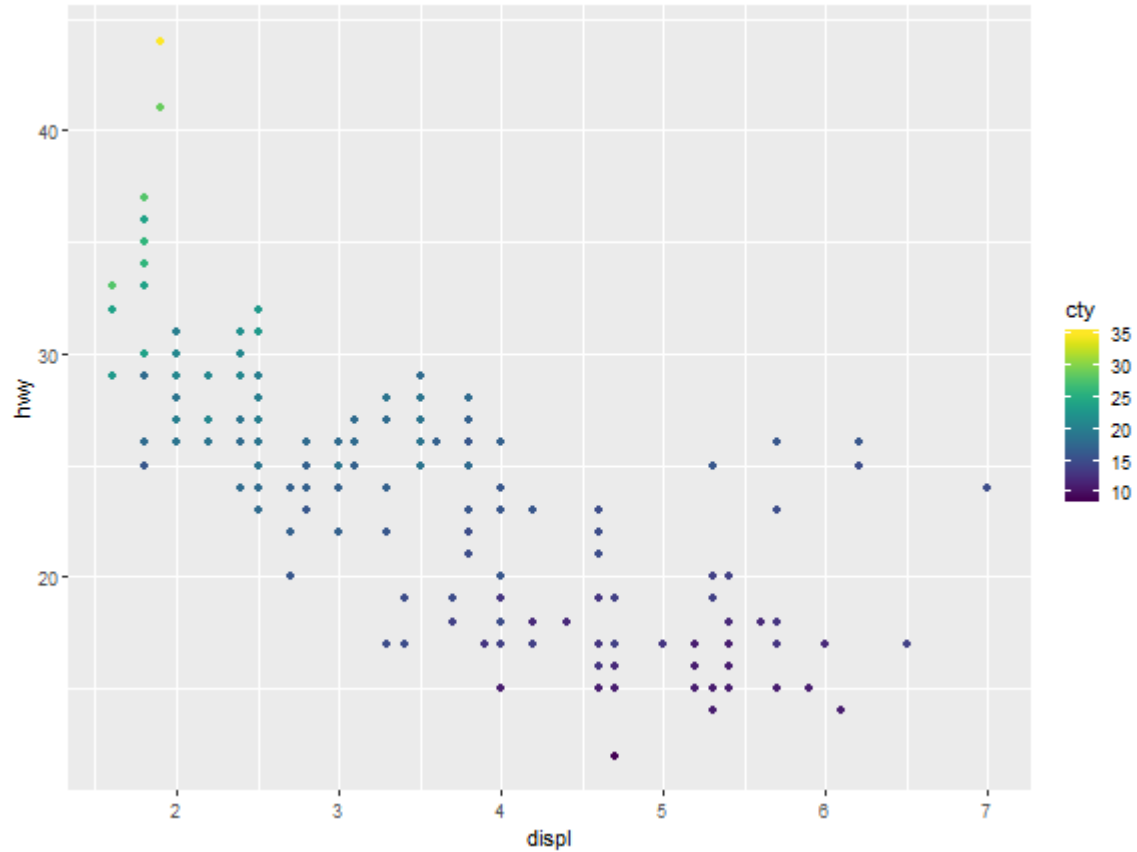
```
ggplot(mpg, aes(x = displ, y = hwy, color = drv)) +  
  geom_point() +  
  scale_color_brewer(palette = "Pastel1")
```

```
ggplot(mpg, aes(x = displ, y = hwy, color = cty)) +  
  geom_point() +  
  scale_color_distiller(palette = "YlOrRd")
```



```
library(viridis)

ggplot(mpg, aes(x = displ, y = hwy, color = cty)) +
  geom_point() +
  scale_color_viridis()
```

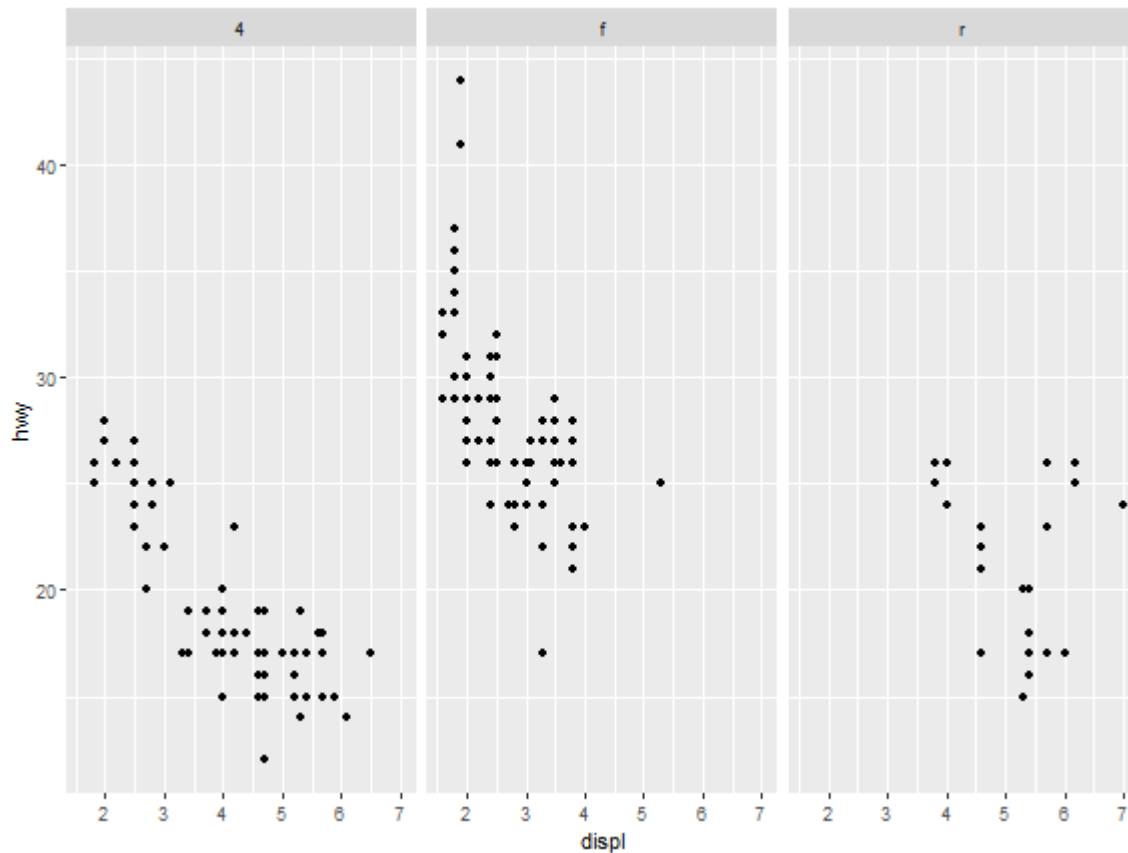


# Facetas

- Otra opción para representar variables adicionales son las facetas
- Cada faceta es un subgráfico realizado con un subconjunto de los datos

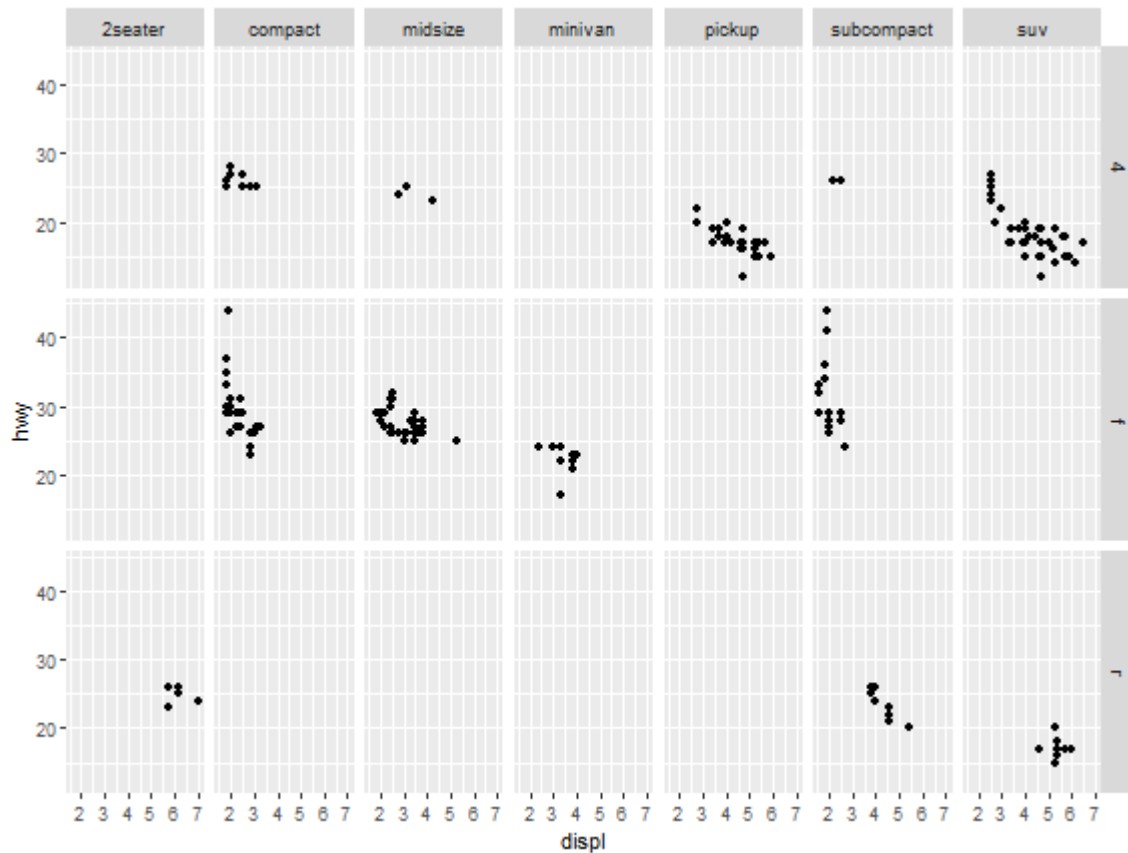
# facet\_wrap

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  facet_wrap(~drv)
```



# facet\_grid

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  facet_grid(drv ~ class)
```



# Temas

- Función `theme()`
- Personalizar los componentes de un gráfico: títulos, etiquetas, fuentes, fondos, leyendas, etc...
- Ver [documentación](#)
- Ejemplo: cambiar posición de la leyenda
  - `+ theme(legend.position = "left")`
  - `+ theme(legend.position = "right")`
  - `+ theme(legend.position = "bottom")`



# Temas por defecto y exportar gráficos

- Existen varios temas por defecto. Algunos ejemplos (**lista completa**):
  - `theme_bw()`
  - `theme_dark()`
  - `theme_minimal()`
  - `theme_classic()`
- Para guardar un gráfico, se usa la función `ggsave()`
  - `ggsave("grafico.pdf", fig.width = 8, fig.height = 6)`
- **Resumen** de todas las funciones de `ggplot2`