Curso Primeros Pasos en R

Clase 6: Visualización de datos con `ggplot2`

Profesora: Ana María Alvarado

Pontificia Universidad Católica de Chile

Noviembre 2021

Clase 6: Visualización de datos con `ggplot2`

Introducción

Gráficos en R

R base tiene herramientas gráficas limitadas, tanto en la cantidad de opciones que se tiene, como su personalización. Es por eso, que se usarán los gráficos de `ggplot2`. Estas herramientas, permiten generar una gran cantidad de gráficos a partir de la misma base computacional.

Funciones R base y `ggplot2`

Gráfico	R Base	ggplot2
Puntos	plot()	geom_point()
Lineas	plot(..., type="l")	geom_line()
Histograma	hist()	geom_histogram()
Barras	barplot()	geom_bar()
Boxplot	boxplot()	geom_boxplot()

El paquete ggplot2

ggplot2 es un paquete de R para producir gráficos. A diferencia de la mayoría de los paquetes de gráficos, ggplot2 tiene una gramática subyacente, basada en la gramática de los gráficos, un sistema coherente para describir y construir gráficos, combinando componentes independientes.

Gráficos con `ggplot2`

Se basa en una **gramática de gráficos**, que permite describir los componentes de un gráfico como una combinación de capas:

.

1. Un gráfico se inicia con la función `ggplot()`, la que crea el sistema de coordenadas de nuestro gráfico.

2. Cada capa adicional se agrega con un símbolo `+`, acompañado de la función deseada.

Gráficos con `ggplot2`

Un gráfico básico de `ggplot` requiere definir al menos tres elementos:

1. Los **datos** a utilizar.

2. Los parámetros **estéticos** con que se registrarán las variables, es decir, cómo se asignarán las variables de nuestro conjunto de datos a ciertas propiedades visuales. Esto considera ejes de gráficos, colores, etc. La función para indicar esto es `aes()` (del inglés "aesthetics").

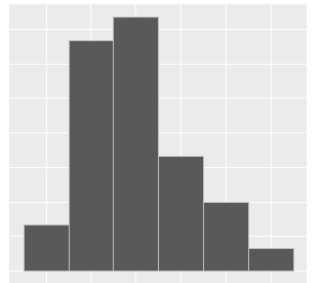
3. Una capa que indique la **forma** en que se representarán gráficamente los datos (con la función **geom_*()**).

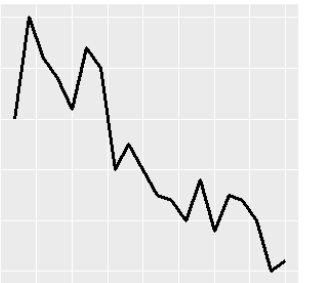
```
ggplot(data = <DATOS>, aes(<MAPEOS>)) +
       geom_<TIPO DE GRÁFICO>(...)
```

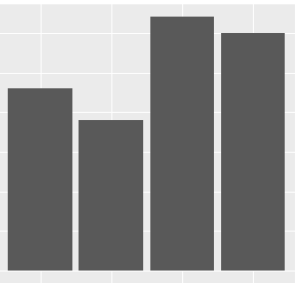
Algunos tipos de gráficos de `ggplot2`

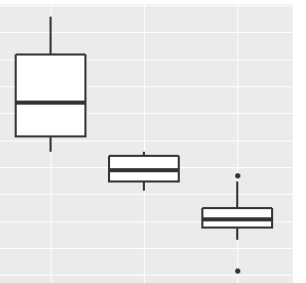
Las capas (geom)

Las capas sirven para proporcionar información sobre cómo queremos visualizar los datos. Esto se lleva a cabo a través de un **geom**.


`geom_histogram()`


`geom_line()`


`geom_col()`


`geom_boxplot()`

Otras geometrías disponibles:

- `geom_area()`
- `geom_quantile()`
- `geom_violin()`
- `geom_density()`
- `geom_errorbar()`
- `geom_smooth()`
- `geom_dotplot()`
- `geom_linerange()`
- `geom_text()`
- ¡Entre otras!

Cada uno de ellos contiene, distintos argumentos de personalización propios de cada tipo de gráfico.

Histograma (`geom_histogram()`)

```
ggplot(data = paises, mapping = aes(x = pib_per_capita)) +
  geom_histogram(color = "white",
                 fill ="#9292ff",
                 bins = 15)
```

Se usa para visualizar la distribución de los valores de una variable numérica.

- El argumento `color` (o `colour`) corresponde al color de la línea de las barras del histograma.
- El argumento `fill` corresponde al color de las barras del histograma.
- El argumento `bins` permite elegir el número de barras a graficar. También, se puede usar `bin-width` para definir el ancho de cada barra.

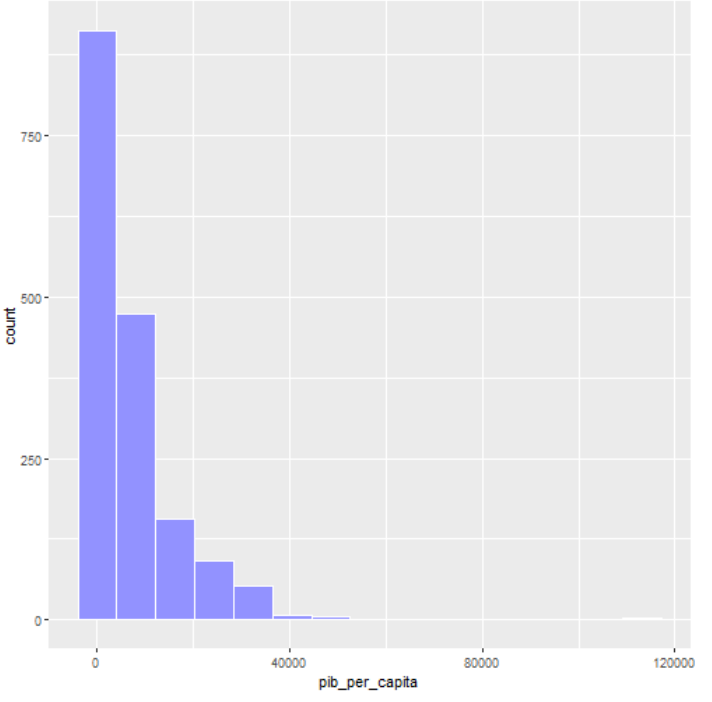
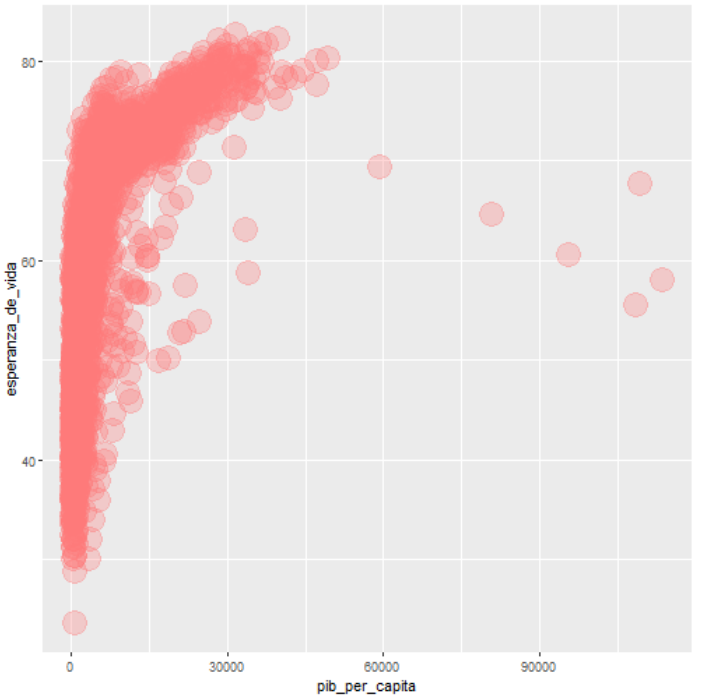


Gráfico de puntos (`geom_point()`)

```
ggplot(data = paises, mapping = aes(x = pib_per_capita,
                                    y = esperanza_de_vida)) +
  geom_point(color = "#ff7979", size = 8, alpha = 0.3)
```

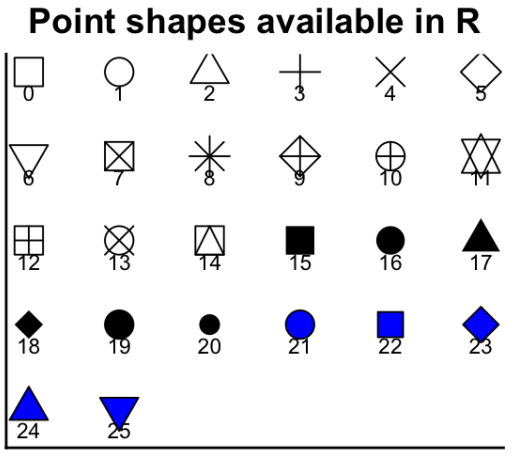
Se usa para encontrar relaciones o patrones entre dos variables (al menos una debe ser numérica).

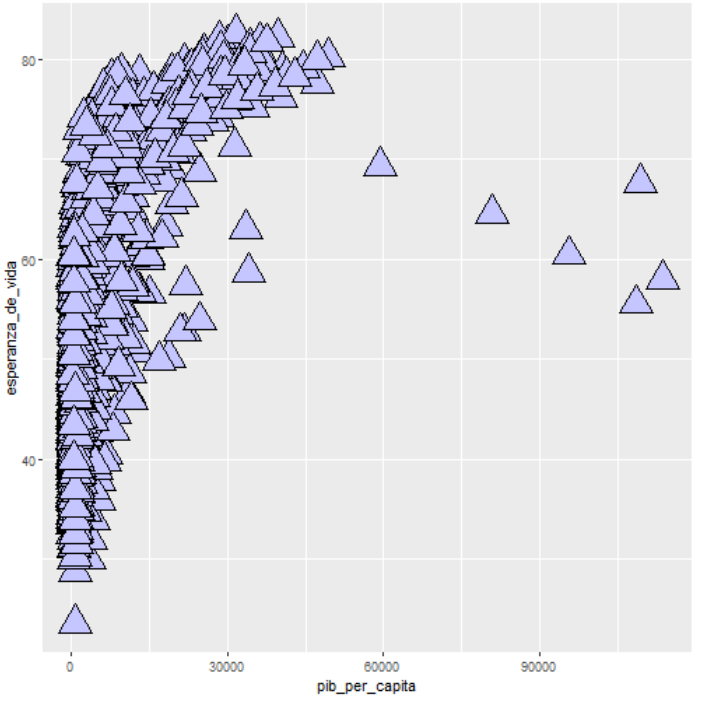
- El argumento `color` (o `colour`) permite elegir el color del punto.
- El argumento `size` permite elegir el tamaño de dibujo del punto. Se le ingresa un número natural.
- El argumento `alpha` permite elegir el nivel de transparencia de los puntos. Toma valores entre 0 y 1. Mientras más cercano a 0 los puntos son más transparentes.



```
ggplot(data = paises, mapping = aes(x = pib_per_capita,
                                    y = esperanza_de_vida)) +
  geom_point(shape = 24, size = 8,
             color = "black",  fill = "#c5c5ff")
```

El argumento shape permite elegir el tipo de punto a usar. En R hay 26 tipos de puntos:





Hay 5 tipos de puntos con relleno, desde el 21 al 25. Cuando se usa alguno de estos, el argumento `color` pasa a definir el color del borde, y el argumento `fill` pasa a ser el de relleno.

Box Plot (`geom_boxplot()`)

```
ggplot(data = paises, mapping = aes(x = continente, y = esperanza_de_vida) ) +
  geom_boxplot(color = "#157EA3",
               fill = "#EB0052")
```

Para observar la distribución de una variable numérica. Muy útil para comparar las distribuciones por grupo.

- Se define la variable numérica en el eje `y`. Se agrega una segunda variable en el eje `x`. Permite separar el boxplot en grupos.
- Se puede modificar la dirección de la caja desde `aes()`.
- El argumento `color` (o `colour`) permite elegir el color del borde.
- El argumento `fill` permite elegir el color de relleno de la caja.

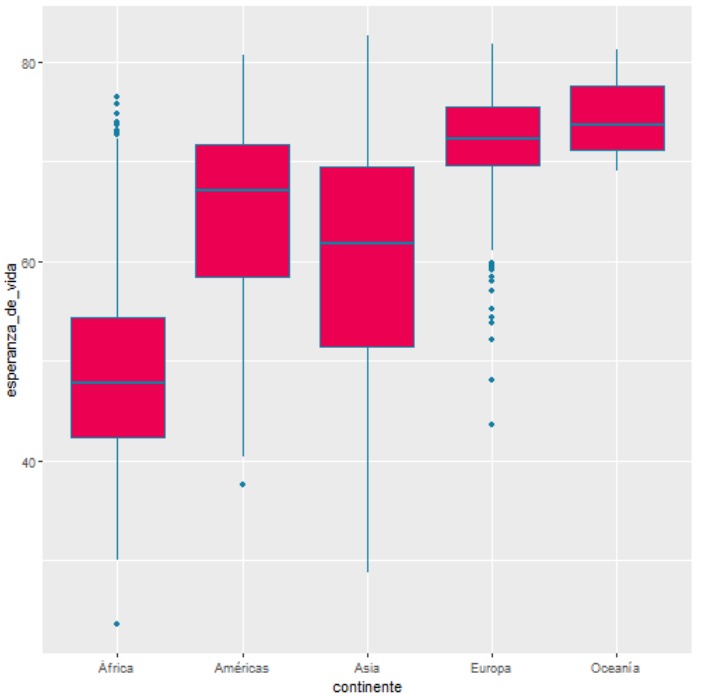
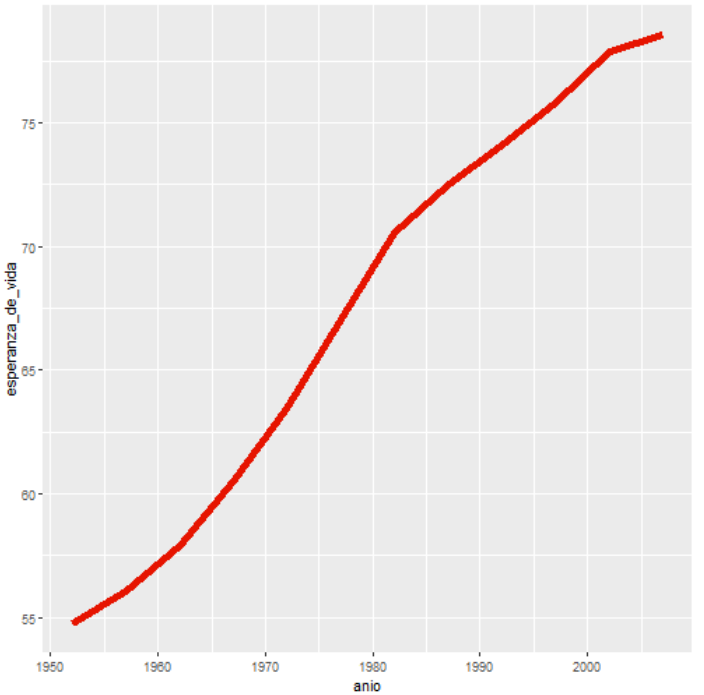


Gráfico de líneas (`geom_line()`)

```
ggplot(data = paises[paises$pais == "Chile",] ,
       aes(x = anio, y = esperanza_de_vida)) +
  geom_line( color = "#E61400", size = 2)
```

Se usa para analizar tendencias de una variable numérica en el tiempo. En el ejemplo, se presenta la evolución de la esperanza de vida en Chile desde 1960.

- El argumento `color` permite elegir el color de la línea.
- El argumento `size` permite elegir el grosor de la línea.



Customización de gráficos

Ejes y títulos

En `ggplot2` existen diferentes formas de definir títulos, subtítulos y nombres en los ejes. A continuación se presentaran dos formas de definir los mismos elementos de un gráfico:

Forma 1: Funciones `ggtitle`, `xlab`, `ylab`

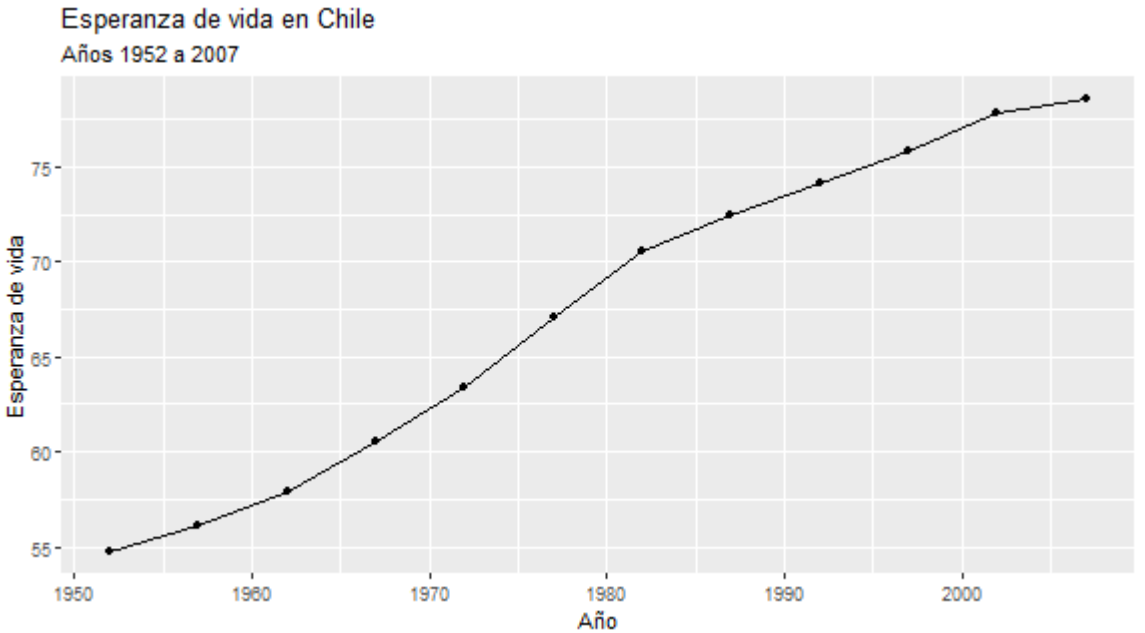
```
ggplot(...) +
 ggtitle(label = "Título", subtitle = "Sub título") +
 xlab(label = "Eje X") +
 ylab(label = "Eje Y")
```

Forma 2: Función `labs`

```
ggplot(...) +
 labs(title = "Título",
      subtitle = "Sub título",
      x = "Eje X",
      y = "Eje Y")
```

Ejes y títulos

```
ggplot(data = paises[paises$pais == "Chile", ] ,
       aes(x = anio, y = esperanza_de_vida)) +
  geom_line() +
  geom_point() +
  labs(title = "Esperanza de vida en Chile",
       subtitle = "Años 1952 a 2007",
       x = "Año",
       y = "Esperanza de vida")
```



Argumentos `color` y `size`

Dentro del comando `aes()` se puede indicar el argumento `color`. Este permite colorear los elementos del gráfico con respecto a una variable de la base de datos. Junto con esto, se puede incluir una leyenda que describirá qué significa cada color. La forma en que ggplot pinta los objetos visuales, depende del tipo de variable:

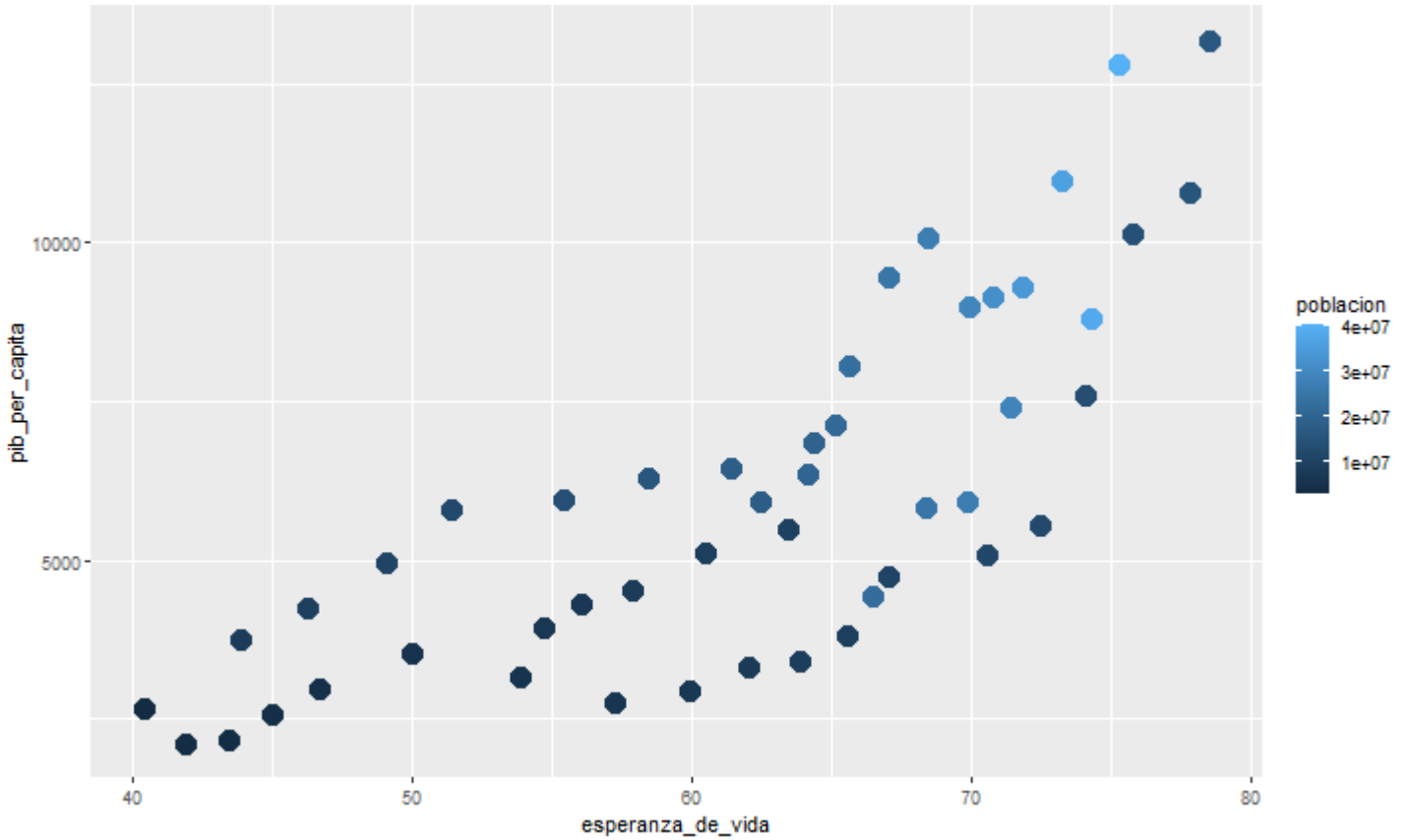
- **Variable numérica** : Elige un gradiente de colores, donde la intensidad del color depende de la variable numérica.
- **Variable categórica/factor** : Elige un color distinto por cada nivel del factor.

Otro argumento de `aes()` es `size`, el cual permite cambiar el tamaño de elementos visuales a través de una variable. Usualmente, se utiliza con variables numéricas continuas y es aplicado en gráficos de puntos.

Estos argumentos pueden ser definidos tanto dentro del comando `ggplot()`, como en los comandos de geometrías `geom_*()` respectivas.

```
ggplot(data = paises,
       aes(x = esperanza_de_vida, y = pib_per_capita)) +
  geom_point(aes( size = poblacion, color = continente), alpha = 0.7)
```

```r
ggplot(data = base_vecinos,
       aes(x = esperanza_de_vida, y = pib_per_capita)) +
  geom_point(aes(color = poblacion), size = 5)
```



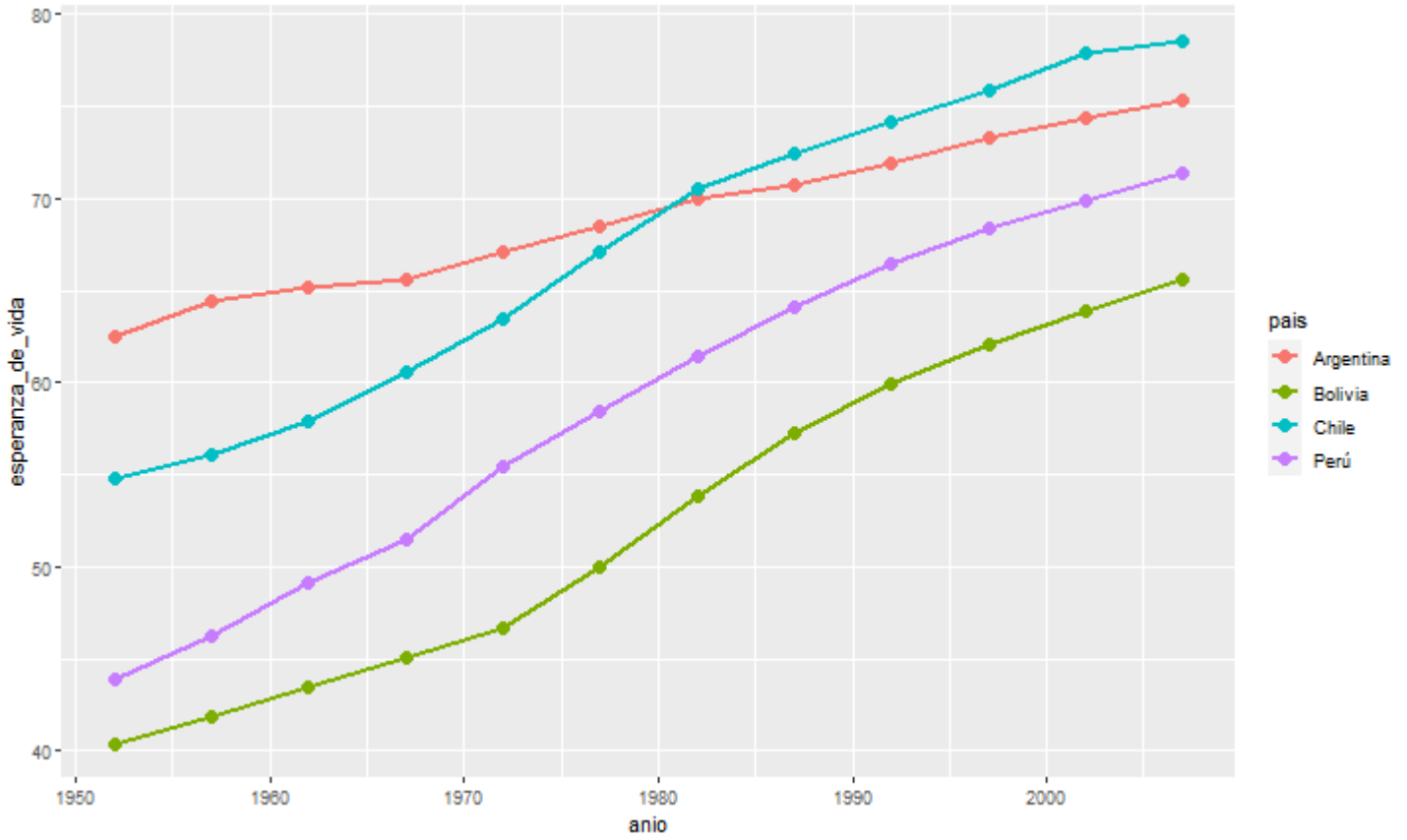
Personalización de la leyenda

La leyenda puede personalizarse mediante el argumento `theme()`, función que permite modificar la mayoría de los elementos estéticos de un ggplot. En el caso de la leyenda usaremos, se usarán los siguientes argumentos:

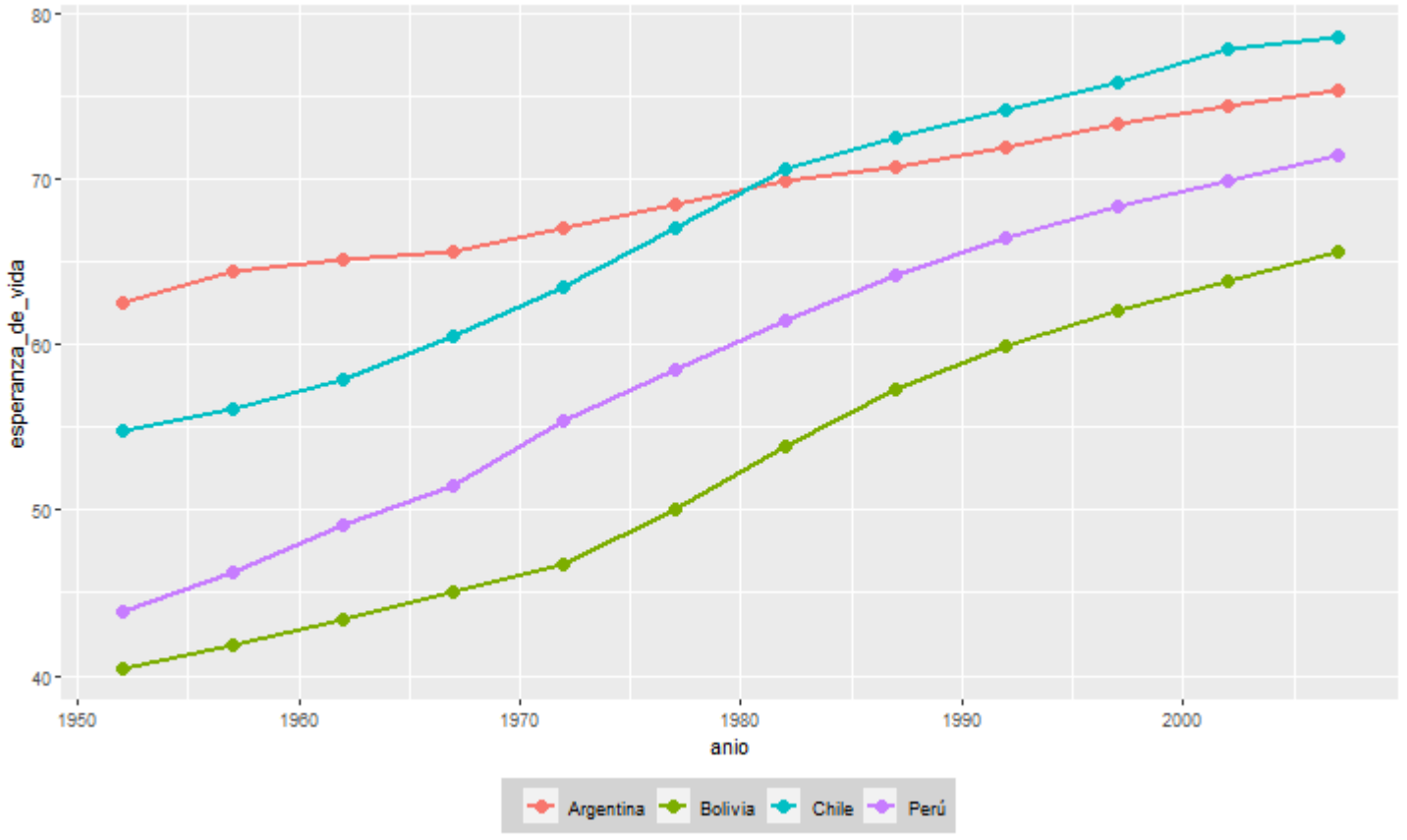
- **`legend.position`** - Permite modificar la posición de la leyenda. Recibe los argumentos `top`, `bottom`, `left`, `right`, `none`.
- **`legend.title`** - Permite modificar el título de la leyenda. Para quitar el nombre definir `element_blank()`.
- **`legend.background`** - Permite modificar el color del fondo del cuadro de la leyenda.
- etc.

A continuación, se visualizará un gráfico de lineas y puntos de la evolución de esperanza de vida de los paises vecinos a Chile a través de los años:

```
ggplot(data = base_vecinos,
       aes(x = anio, y = esperanza_de_vida, color = pais) ) +
  geom_point(size = 3) + geom_line(size = 1)
```



```
ggplot(data = base_vecinos,
       aes(x = anio, y = esperanza_de_vida, color = pais) ) +
  geom_point(size = 3) + geom_line(size = 1) +
  theme(legend.position = "bottom",
        legend.title = element_blank(),
        legend.background = element_rect(fill="lightgrey") )
```



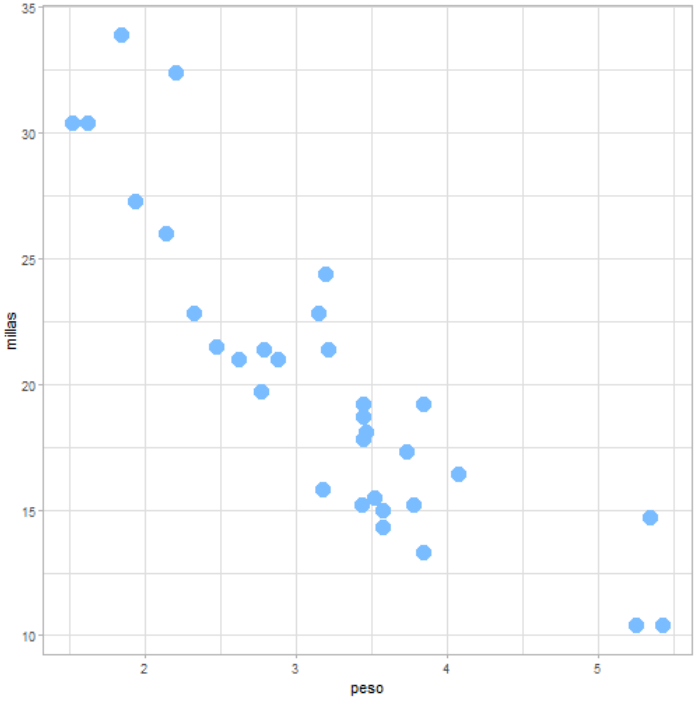
Temas

En la librería `ggplot` existe una variedad de temas precargados que permiten modificar de manera rápida cómo se ve el fondo de un gráfico. Estos son:

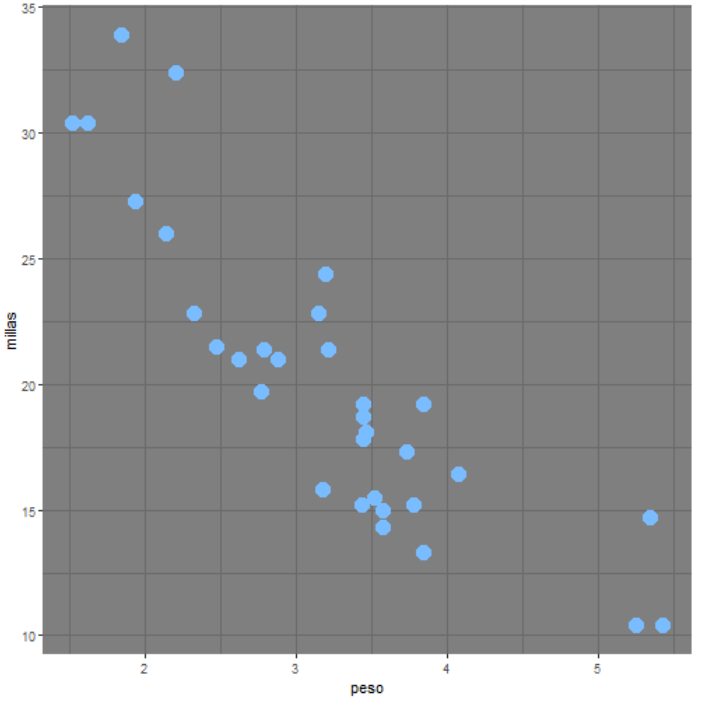
- `theme_grey()`/`theme_gray()` - Por defecto
- `theme_bw()`
- `theme_linedraw()`
- `theme_light()`
- `theme_dark()`
- `theme_minimal()`
- `theme_classic()`
- `theme_void()`
- `theme_test()`

A continuación, se usarán algunos de estos temas para evidenciar las diferencias entre ellos:

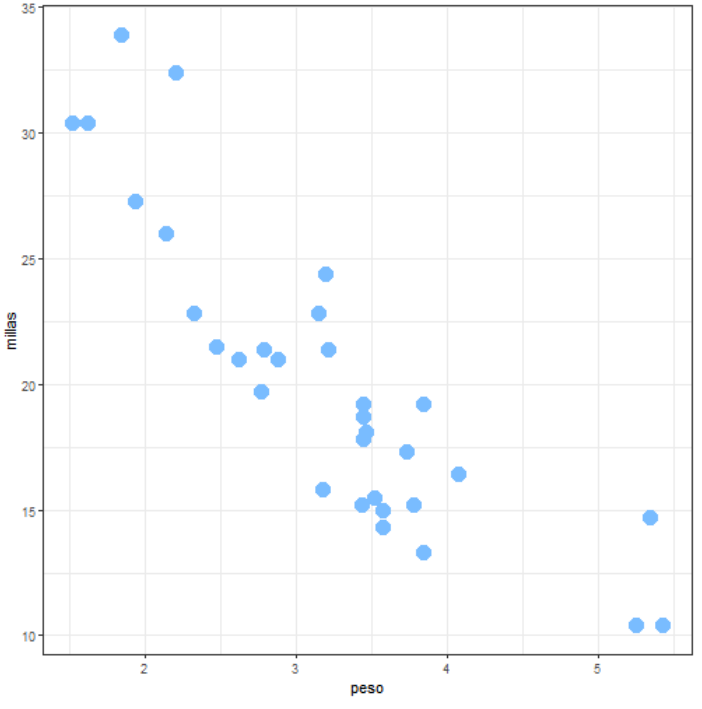
```r
ggplot(data = mtautos,
       aes(peso, millas)) +
  geom_point(color = "#79bcff",
             size = 5) +
  theme_light()
```



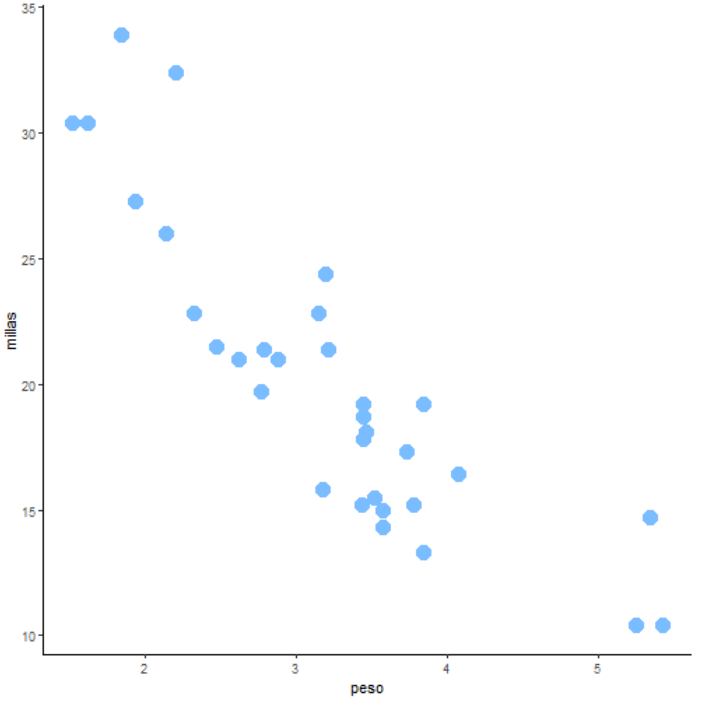

```r
ggplot(data = mtautos,
       aes(peso, millas)) +
  geom_point(color = "#79bcff",
             size = 5) +
  theme_dark()
```



```
ggplot(data = mtautos,
       aes(peso, millas)) +
  geom_point(color = "#79bcff",
             size = 5) +
  theme_bw()
```




```
ggplot(data = mtautos,
       aes(peso, millas)) +
  geom_point(color = "#79bcff",
             size = 5) +
  theme_classic()
```



Personalización de colores

Existen diferentes formas de personalizar los colores de un gráfico de `ggplot2`, esto depende si el gráfico está coloreado mediante una variable continua o discreta:

Variable Discreta

- **`scale_fill_manual()`**: Permite ingresar colores de manera manual a colores del argumento `fill` de un gráfico.
- **`scale_color_manual()`**: Permite ingresar colores de manera manual a colores del argumento `color` de un gráfico.

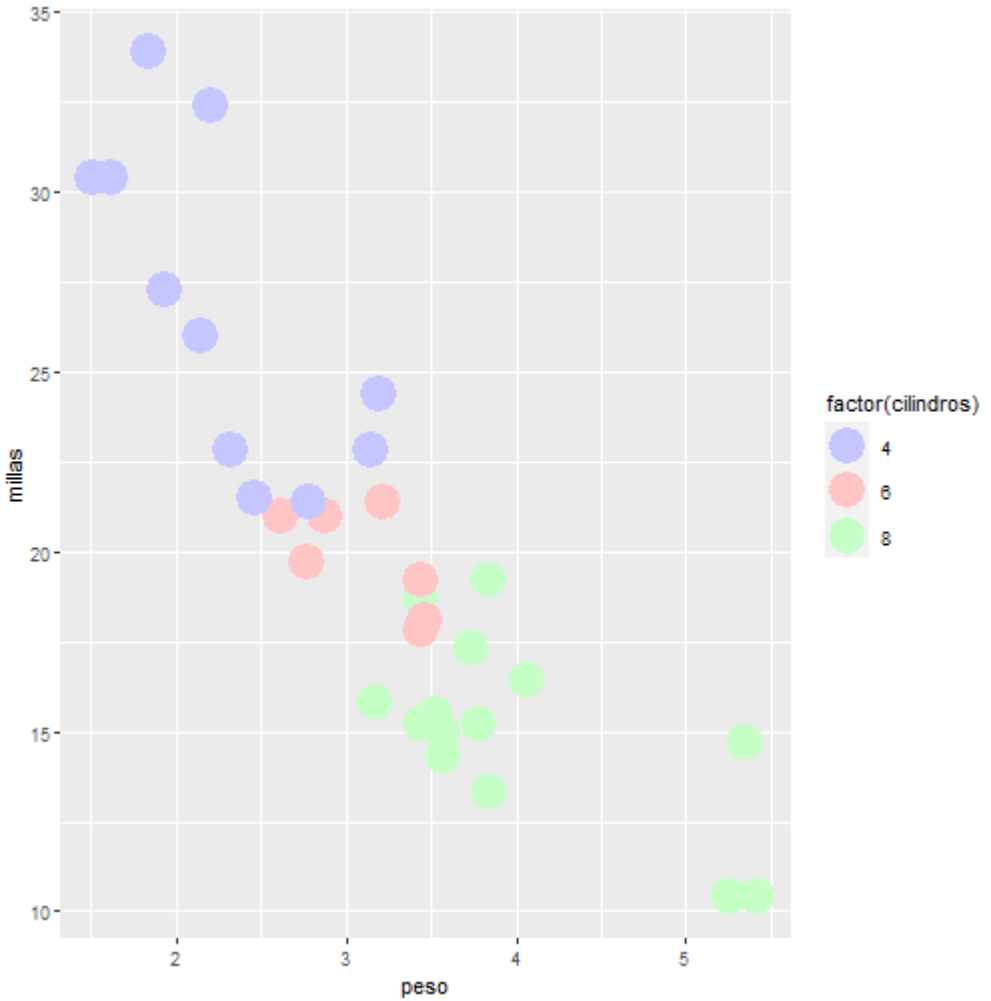
Variable Continua

- **`scale_<*>_gradient()`**: Gradiente secuencial de 2 colores.
- **`scale_<*>_gradientn()`**: Gradiente entre n colores.

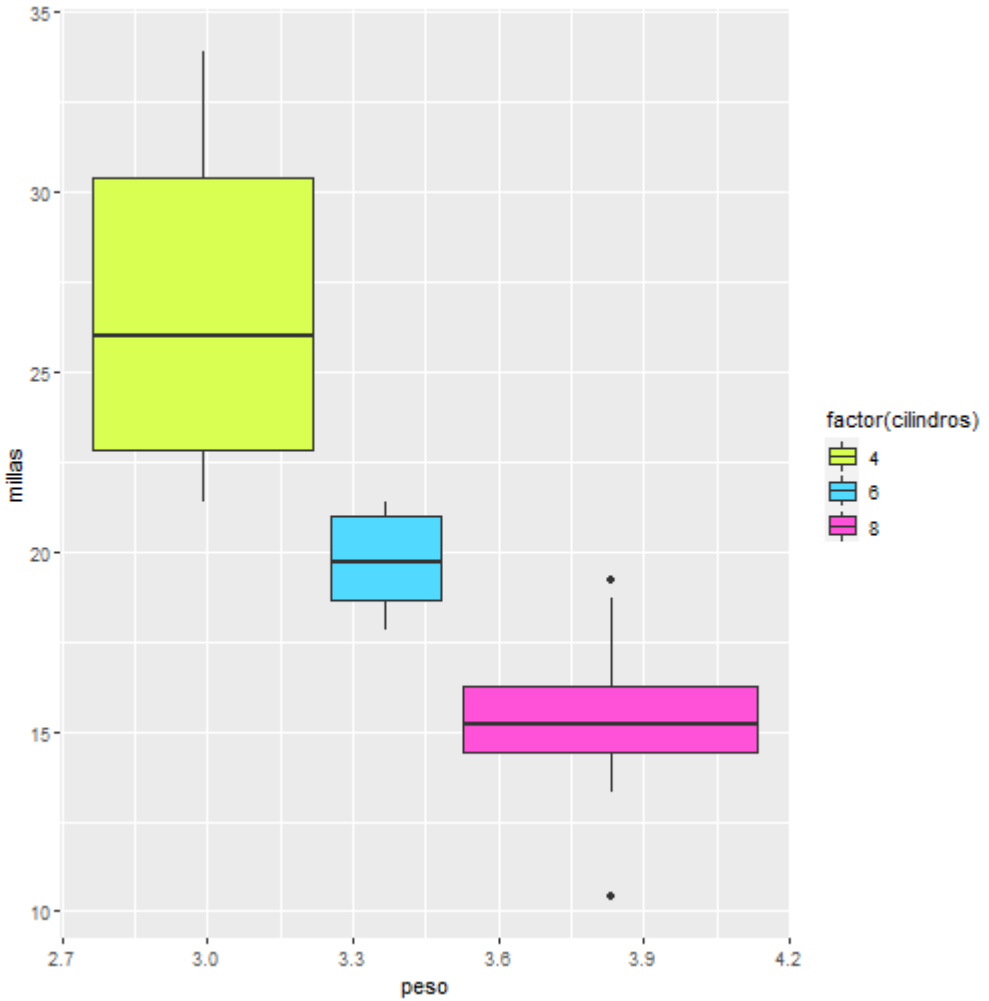
En este caso `<*>` puede ser tanto `color` como `fill`, dependiendo del tipo de elemento visual que se quiera personalizar.

A continuación se presentará un ejemplo de cada una de estas funciones:

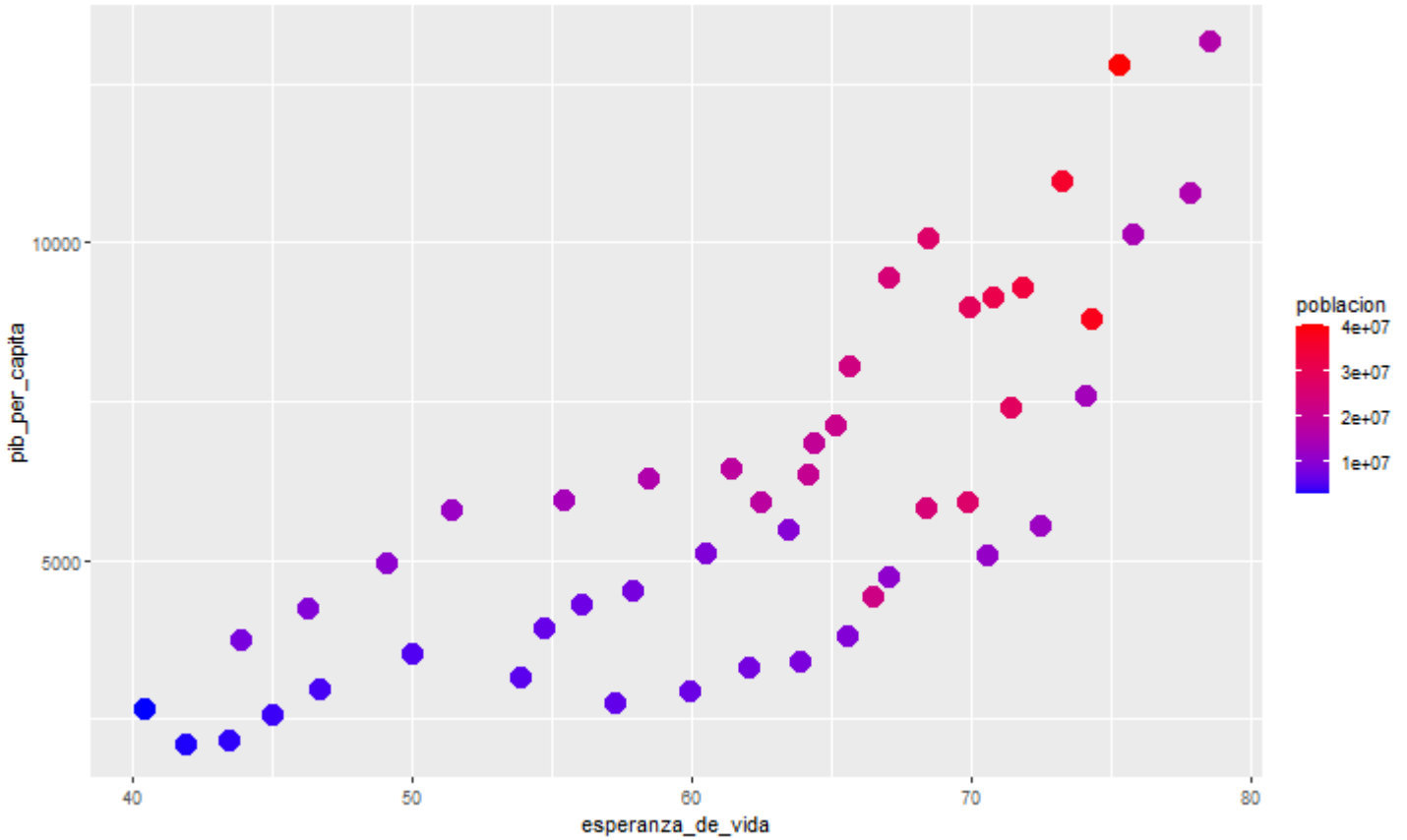
```
ggplot(data=mtautos, aes(x=peso, y=millas, col=factor(cilindros) )) +
  geom_point(size = 8) +
  scale_color_manual(values=c("#c5c5ff","#ffc5c5","#c5ffc5"))
```



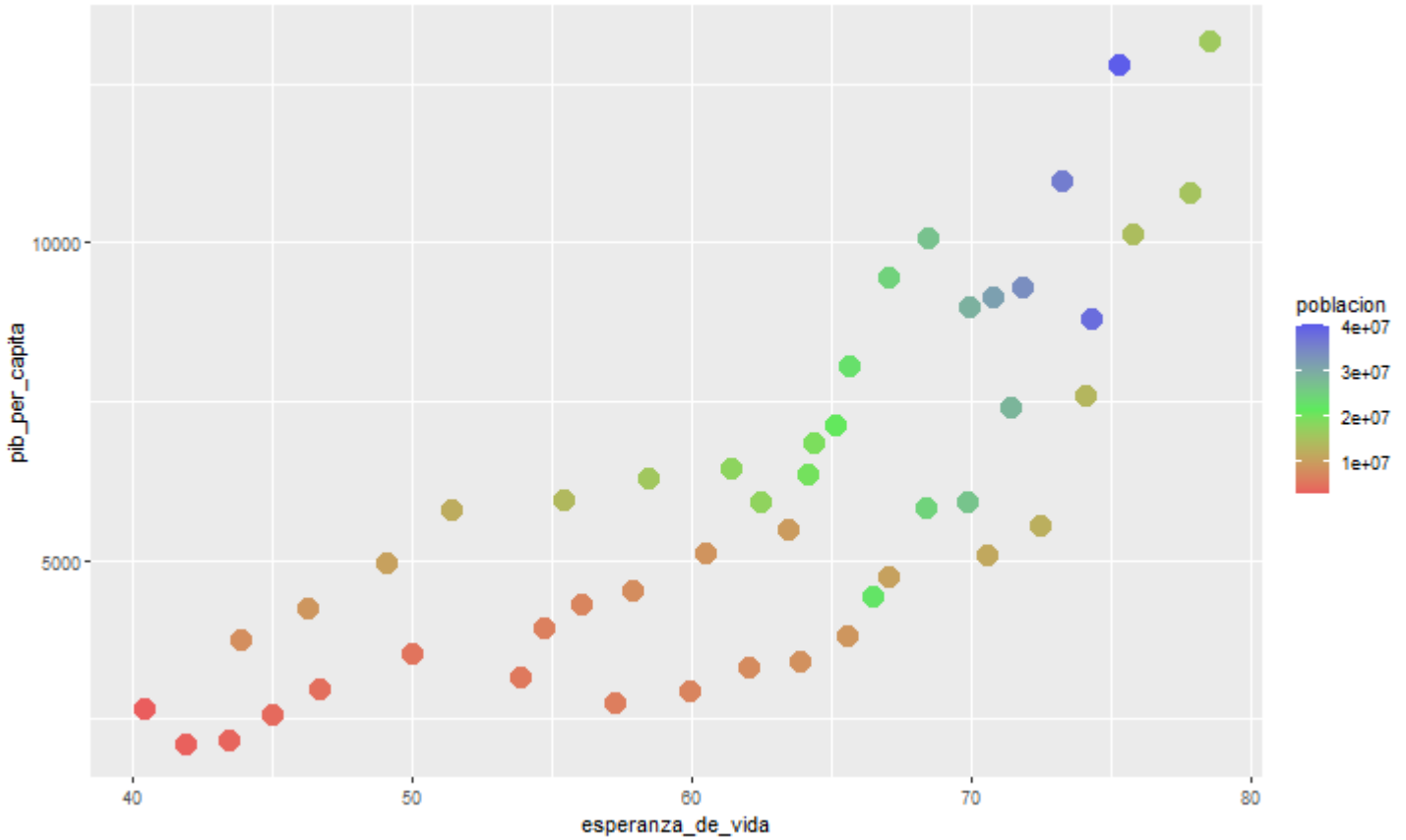
```
ggplot(data=mtautos, aes(x = peso, y = millas )) +
  geom_boxplot(aes(fill = factor(cilindros) )) +
  scale_fill_manual(values=c("#d9ff52","#52d9ff","#ff52d9"))
```



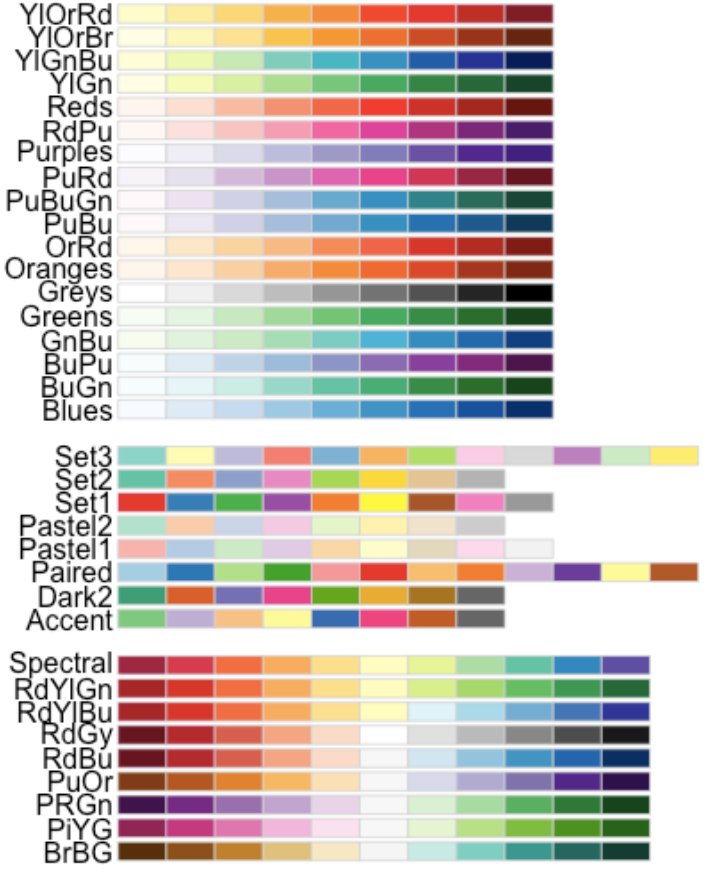
```r
ggplot(data = base_vecinos,
       aes(x = esperanza_de_vida, y = pib_per_capita)) +
  geom_point(aes(color = poblacion), size = 5) +
  scale_color_gradient(low = "blue", high = "red")
```



```
ggplot(data = base_vecinos,
       aes(x = esperanza_de_vida, y = pib_per_capita)) +
  geom_point(aes(color = poblacion), size = 5) +
  scale_color_gradientn(colors = c("#ea5d5d","#5dea5d","#5d5dea"))
```



Librería `RColorBrewer`

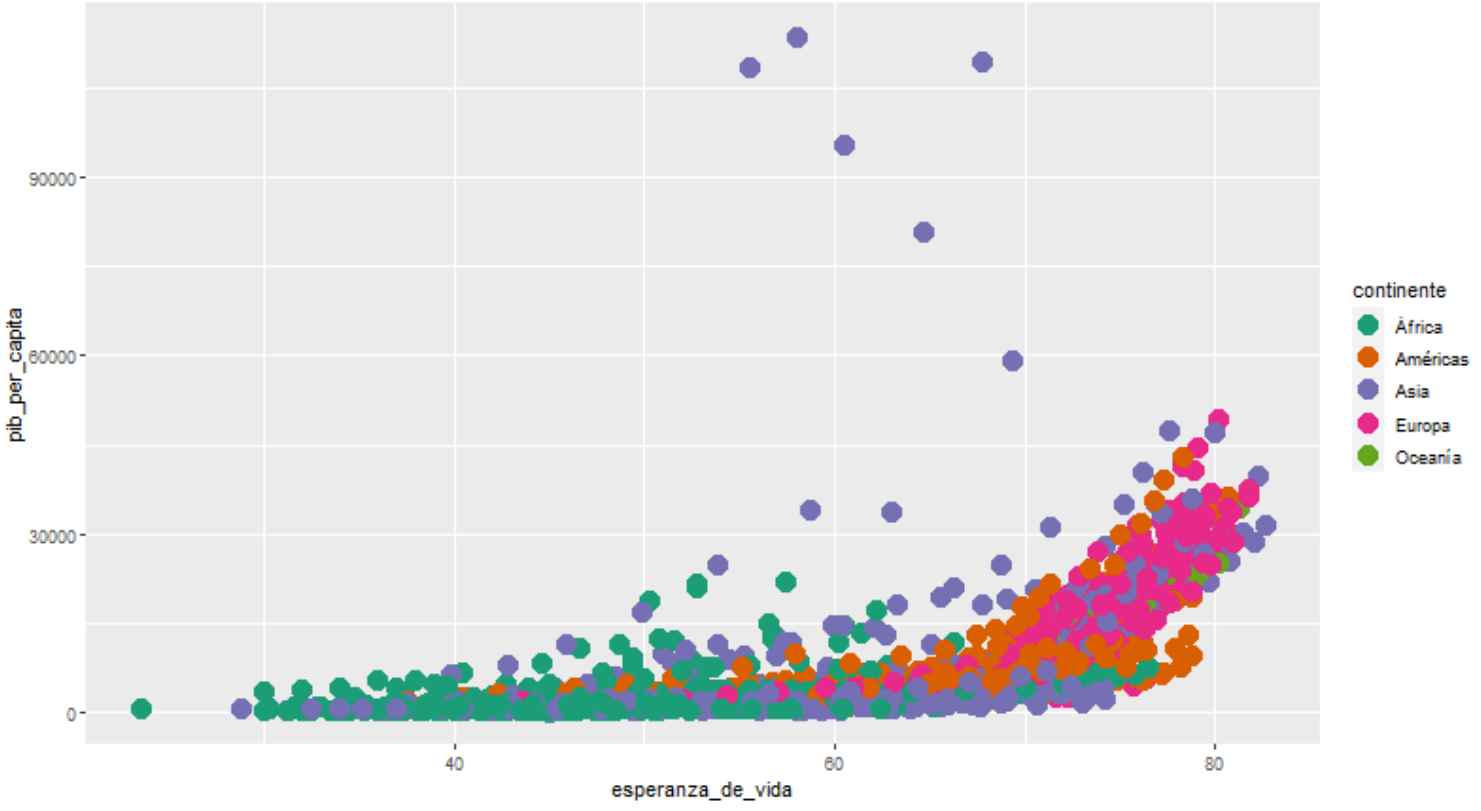


La librería `RColorBrewer` contiene un gran número de paletas de colores para los gráficos de R y `ggplot2`. Las paletas de colores se definen con el argumento `palette` dentro de los comandos `scale_color_brewer()` o `scale_fill_brewer()`.

```
install.packages("RColorBrewer")
library(RColorBrewer)
```

En las siguientes slides se ejemplificará el uso de estas paletas de colores.

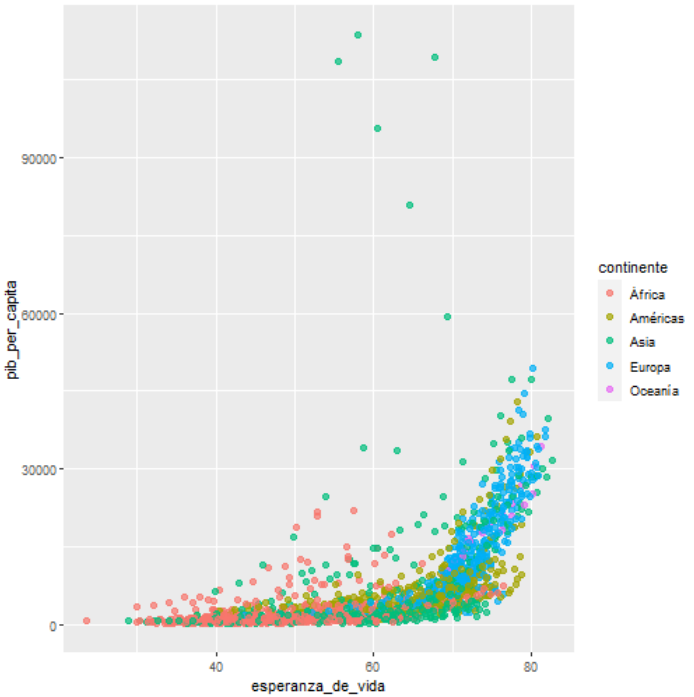
```
ggplot(data=paises,
       aes(x = esperanza_de_vida,
           y = pib_per_capita, color = continente))+
  geom_point(size = 5) +
  scale_color_brewer(palette="Dark2")
```

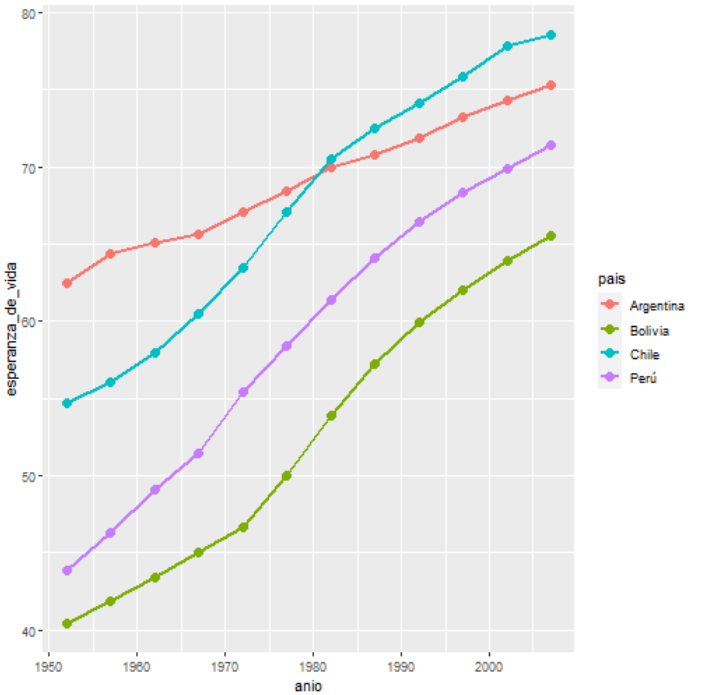


Paneles gráficos

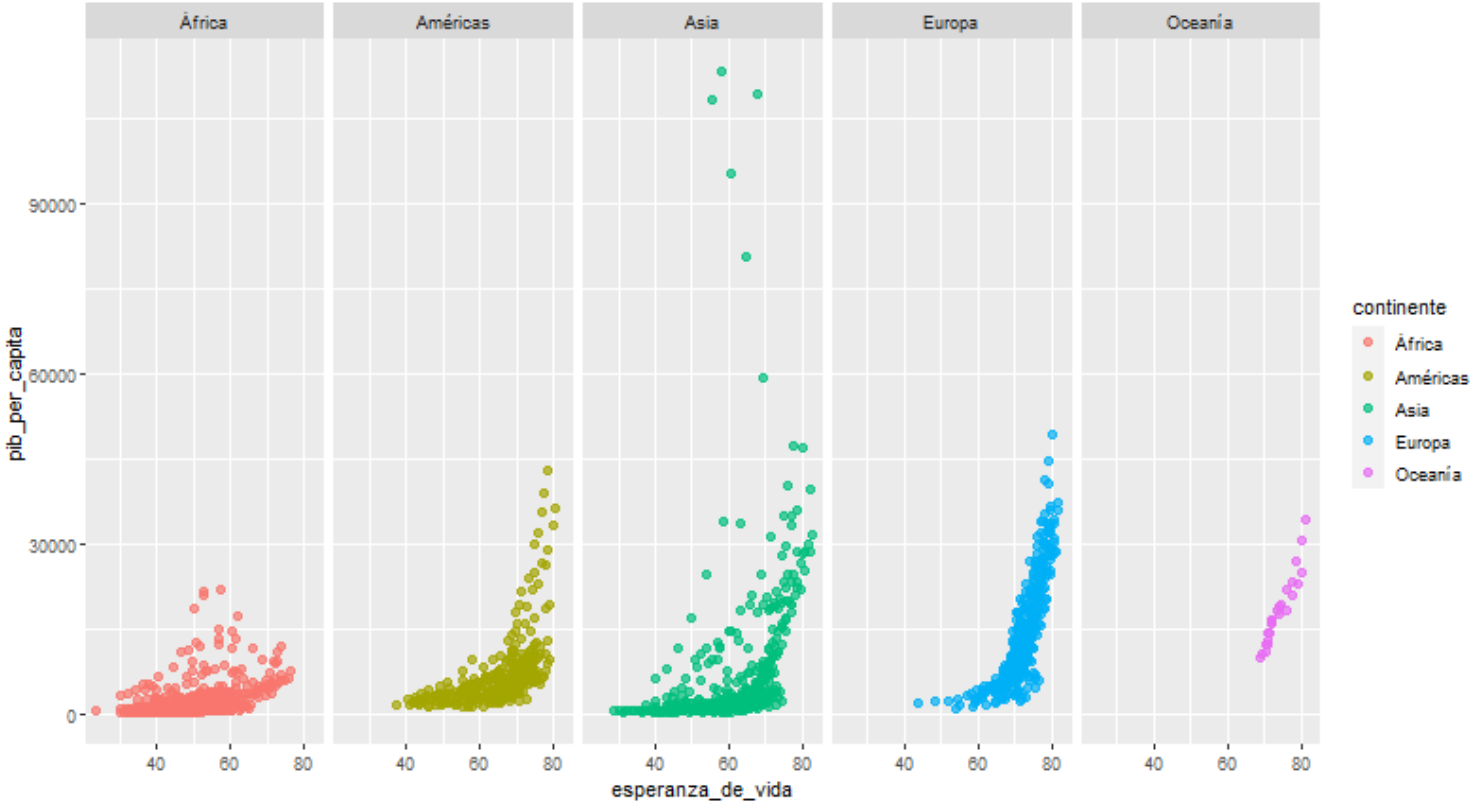
Comandos `facet_wrap()` y `facet_grid()`

Los comandos `facet_wrap()` y `facet_grid()` permiten separar un gráfico en gráficos relacionados en un mismo panel. Esto ayuda de sobre manera cuando se tiene una agrupación en un gráfico y se desea analizar cada grupo por separado. En las siguientes slidan se usarán los siguientes gráficos para ejemplificar estas funciones:

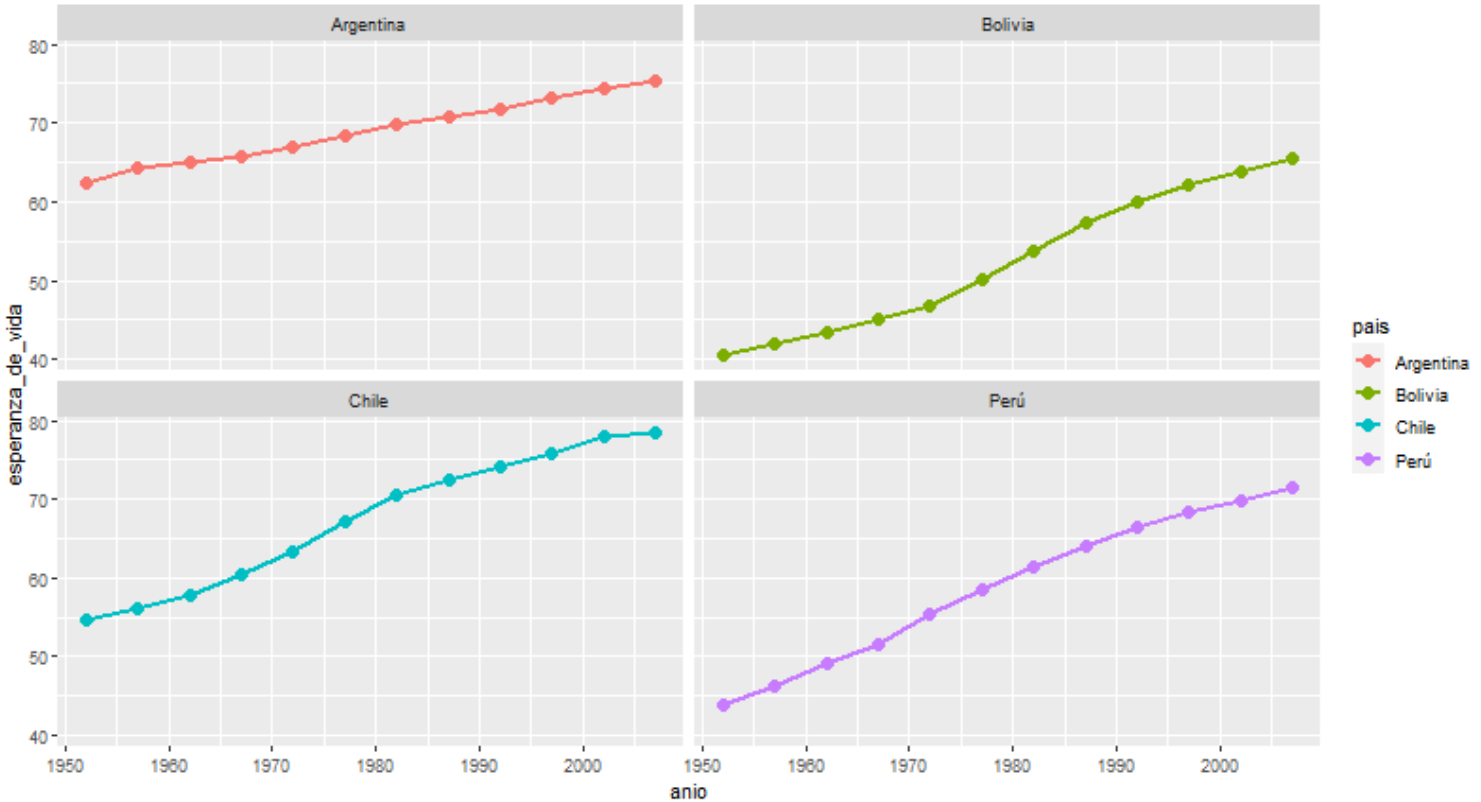




```r
ggplot(data=paises,
       aes(x = esperanza_de_vida,
           y = pib_per_capita, color = continente))+
  geom_point(size = 2, alpha = 0.7) +
  facet_grid(~continente)
```



```
ggplot(data = base_vecinos, aes(x = anio,y = esperanza_de_vida,
                                color = pais) ) +
  geom_point(size = 3) + geom_line(size = 1) +
  facet_wrap(~ pais)
```



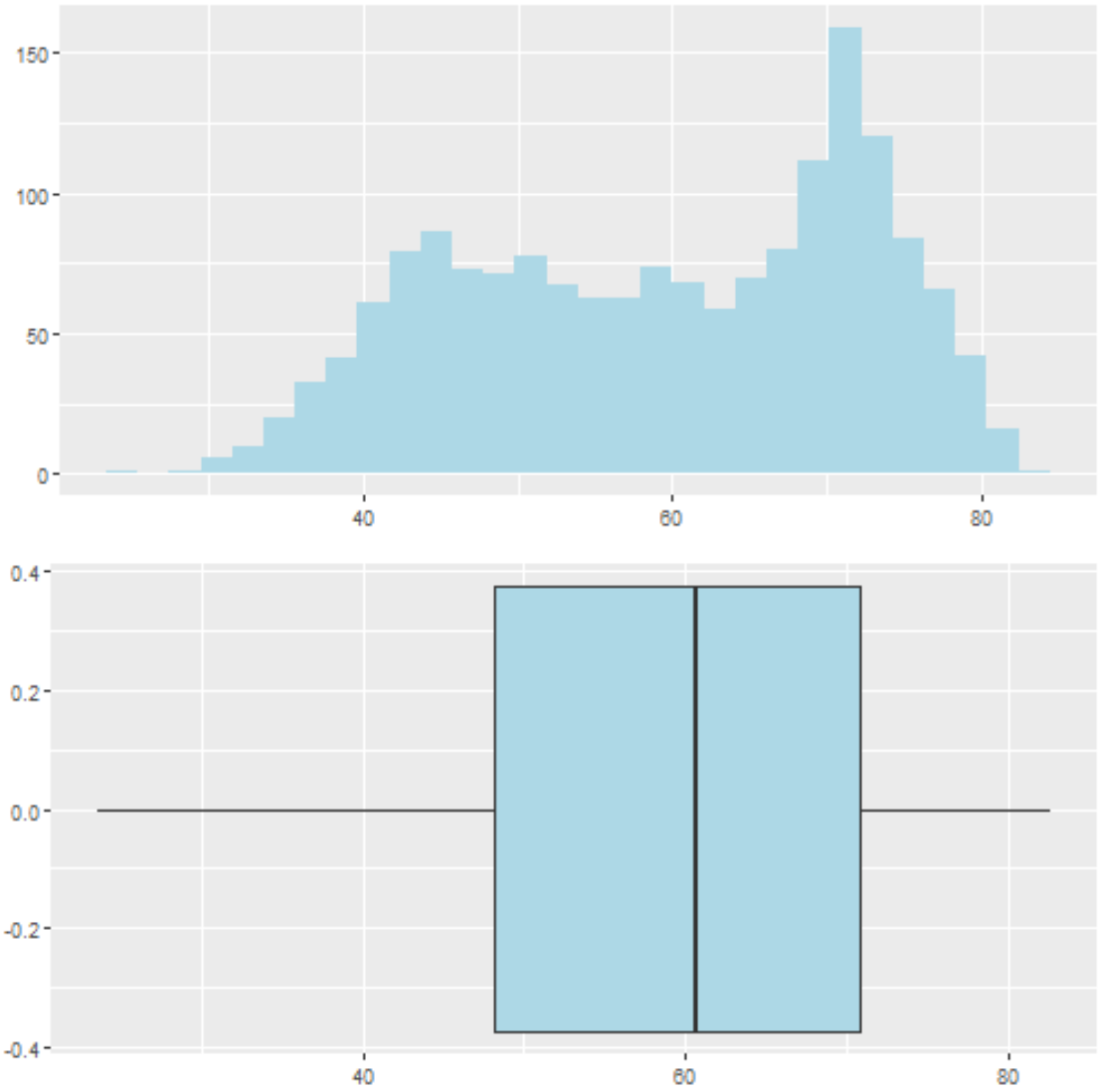
Matriz de gráficos

```
g1 = ggplot(data = paises, aes(x = esperanza_de_vida)) +
  geom_histogram(fill = "lightblue") +
  labs(x = element_blank(), y = element_blank())

g2 = ggplot(data = paises, aes(x = esperanza_de_vida)) +
  geom_boxplot(fill = "lightblue") +
  labs(x=element_blank())

gridExtra::grid.arrange(g1,g2, nrow=2)
```

El comando **`grid.arrange()`** de la librería `gridExtra` permite colocar en la misma ventana múltiples gráficos de `ggplot2`, en una especie de matriz.

```
grid.extra(plot1, plot2,...)
```

Esta función recibe un listado de elementos visuales seguidos del argumento `ncol` o `nrow`, los cuales nos permiten indicar cuál será el número de columnas o filas, respectivamente, de la matriz de gráficos.

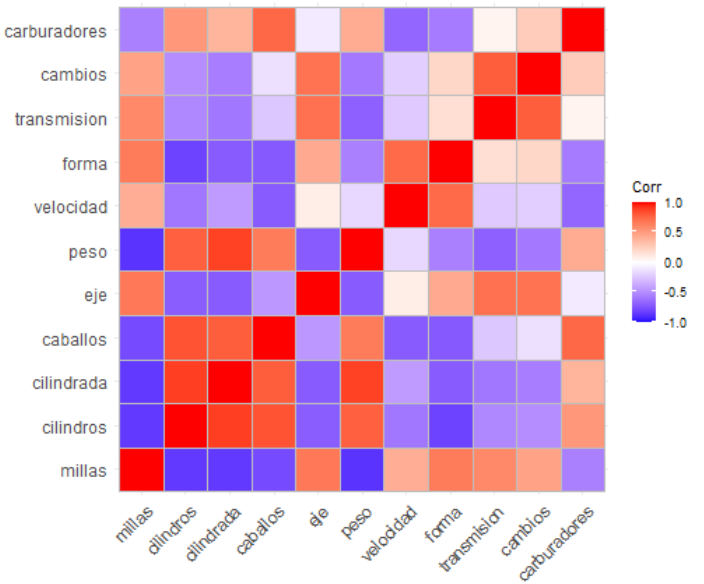

150
100
50
0
40
60
80
0.4
0.2
0.0
-0.2
-0.4
40
60
80

Otros gráficos relacionados

Gráficos de correlación

La función `ggcorrplot()` de la librería `ggcorrplot` grafica un correlograma a partir de una matriz de correlación de un set de datos. Los correlogramas son la mejor manera de analizar una matriz de correlación, dada su fácil interpretabilidad.

```
library(ggcorrplot)
cor_autos = cor(mtautos)
ggcorrplot(cor_autos)
```

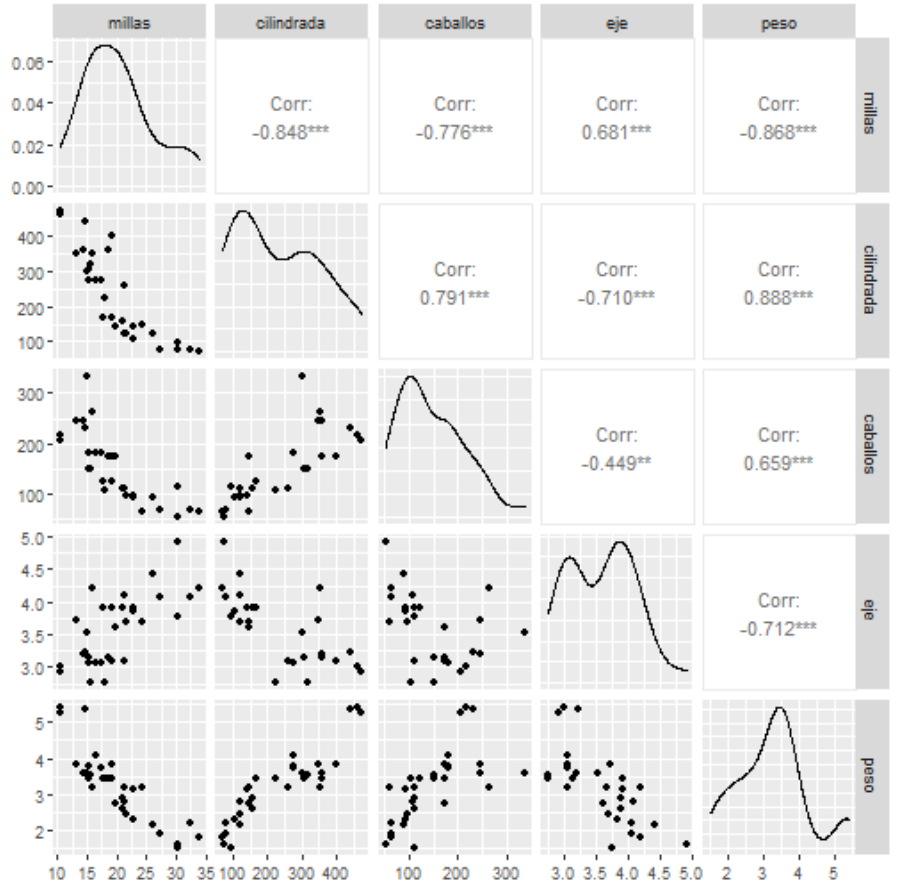


Algunos argumentos de la función:

- **`method`**: figura en cada cuadro (`square` o `circle`).
- **`type`**: tipo de pintado (`full`, `lower` o `upper`).
- **`colors`**: vector de 3 colores para representar la correlación.
- entre otros.

Matriz de dispersión

La función `ggpairs()` de la librería `GGally` realiza una matriz de gráficos de un conjunto de datos dados.



Taller práctico

Taller práctico 1

Retomando las datos de anuncios de viviendas de la Región Metropolitana, **viviendasRM.csv**, construya los siguientes gráficos:

1. Un histograma que permita visualizar la distribución del valor de las viviendas en UF.

2. Un boxplot para visualizar la distribución del valor de las viviendas en UF según la cantidad de habitaciones.

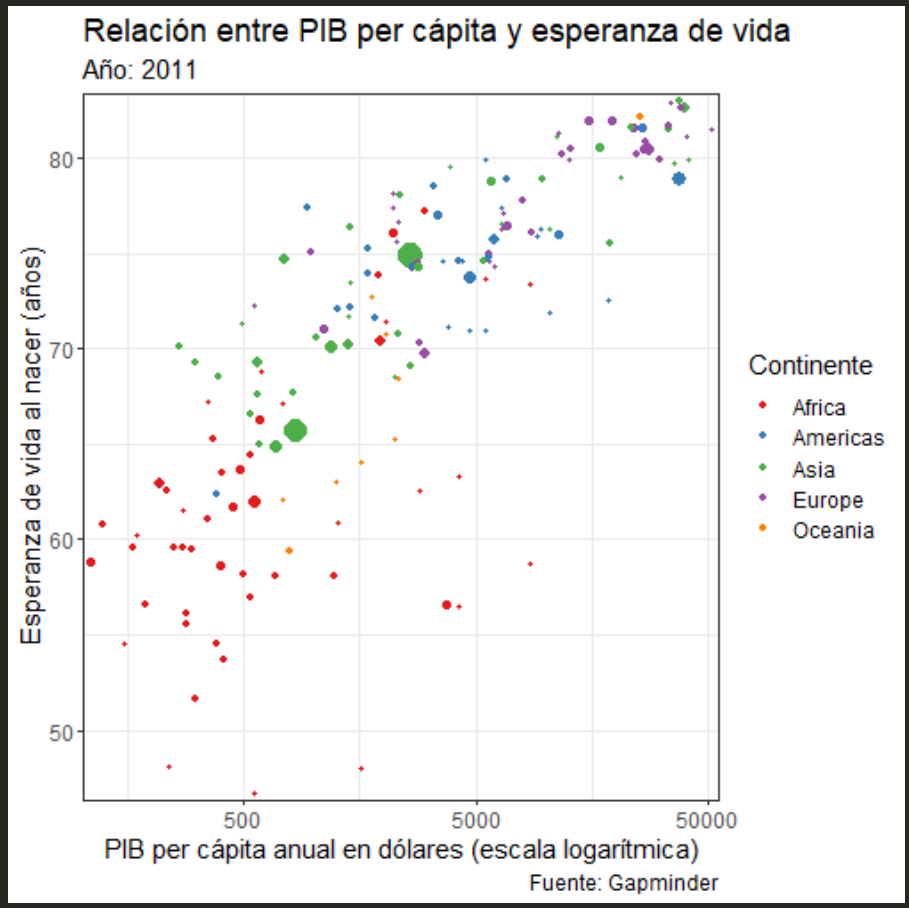
3. Un gráfico de dispersión (o puntos) para observar la relación entre el valor de las viviendas en UF y su superficie construida en m^2. Ajuste a escala logarítmica si es necesario o elimine los valores atípicos.

4. Realice un análisis de correlación entre todas las variables numéricas disponibles.

5. Realice un análisis de dispersión entre todas las variables numéricas disponibles.

Taller práctico 2

Usando los datos de países del package **datos** replique el siguiente gráfico.



Referencias y material complementario

Referencias y material complementario

Link: Funciones `facet_wrap()` y `facet_grid()`

Colores y personalización

- **Link:** Funciones base `ggplot2`
- **Link:** Paletas de colores `jcolors`
- **Link:** Funciones para distintas escalas de colores

Añadir lineas, segmentos, distribuciones

- **Link:** Líneas rectas, horizontales, verticales.
- **Link:** Líneas distribuciones.

¡Gracias!

Ana María Alvarado Celis
amalvara@uc.cl

Maite Vergara - Esteban Rucán
maite.vergara@uc.cl - errucan@uc.cl