Gráficos con ggplot2

Antonio Miñarro
aminarro@ub.edu

UNIVERSITAT DE BARCELONA

Departament de Genètica, Microbiologia i Estadística

15/06/2021

Esquema del tema

Esquema del tema

Esquema del tema

Esquema del tema

Esquema del tema

Esquema del tema

El paquete **ggplot2**

- ggplot2 es un package de R dedicado a la realización de gràficos de una forma muy versátil y elegante.
- Implementa la denominada **gramática de los gráficos**, un sistema para describir y construir gráficos.
- Esta gramática trata los gráficos como un conjunto de elementos independientes que pueden combinarse para formar el gráfico final.
- La filosofía es trabajar a base de diferentes capas, comenzando por una capa inicial que fija los datos a representar y añadiendo posteriormente capas gráficas y de anotaciones.

Podéis encontrar más información en:

1. H.Wickham, ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2009
2. A Layered Grammar of Graphics

Características de **ggplot2**

- Permite representar datos univariantes, multivariantes o categóricos.
- No forma parte de la distribución base de R por lo tanto hay que instalarlo y cargarlo posteriormente
- Forma parte de la colecció de packages para análisis de datos **tidyverse** que puede ser instalada en su totalidad con la instrucción

```
> install.packages("tidyverse")
```

- También puede instalarse de forma independiente

```
> install.packages("ggplot2")
> library(ggplot2)
```

Características de **ggplot2**

- Permite representar datos univariantes, multivariantes o categóricos.
- No forma parte de la distribución base de R por lo tanto hay que instalarlo y cargarlo posteriormente
- Forma parte de la colecció de packages para análisis de datos **tidyverse** que puede ser instalada en su totalidad con la instrucción

```
> install.packages("tidyverse")
```

- También puede instalarse de forma independiente

```
> install.packages("ggplot2")
> library(ggplot2)
```

Características de **ggplot2**

- Permite representar datos univariantes, multivariantes o categóricos.
- No forma parte de la distribución base de R por lo tanto hay que instalarlo y cargarlo posteriormente
- Forma parte de la colecció de packages para análisis de datos **tidyverse** que puede ser instalada en su totalidad con la instrucción

```
> install.packages("tidyverse")
```

- También puede instalarse de forma independiente

```
> install.packages("ggplot2")
> library(ggplot2)
```

Características de **ggplot2**

- Permite representar datos univariantes, multivariantes o categóricos.
- No forma parte de la distribución base de R por lo tanto hay que instalarlo y cargarlo posteriormente
- Forma parte de la colecció de packages para análisis de datos **tidyverse** que puede ser instalada en su totalidad con la instrucción

```
> install.packages("tidyverse")
```

- También puede instalarse de forma independiente

```
> install.packages("ggplot2")
> library(ggplot2)
```

Características de **ggplot2**

- Permite representar datos univariantes, multivariantes o categóricos.
- No forma parte de la distribución base de R por lo tanto hay que instalarlo y cargarlo posteriormente
- Forma parte de la colecció de packages para análisis de datos **tidyverse** que puede ser instalada en su totalidad con la instrucción

```
> install.packages("tidyverse")
```

- También puede instalarse de forma independiente

```
> install.packages("ggplot2")
> library(ggplot2)
```

Características de **ggplot2**

- Permite representar datos univariantes, multivariantes o categóricos.
- No forma parte de la distribución base de R por lo tanto hay que instalarlo y cargarlo posteriormente
- Forma parte de la colecció de packages para análisis de datos **tidyverse** que puede ser instalada en su totalidad con la instrucción

```
> install.packages("tidyverse")
```

- También puede instalarse de forma independiente

```
> install.packages("ggplot2")
> library(ggplot2)
```

Elementos de un ggplot

Como idea básica en un gráfico ggplot se combinan diferentes elementos que pueden dar lugar a diferentes capas (layers) en un mismo gráfico

Elementos de un ggplot

- Les datos que se quieren representar almacenados en un **data frame**.
- Objectos geométricos (**geoms**) que definen el aspecto global de la capa (barras, puntos, líneas, ...)
- Atributos estéticos (**aesthetics**) que son propiedades visuales de los geoms como la posición, el color de línea, formas de los puntos, etc.
- Un resumen estadístico de los datos (**stats**)(contaje, suavizado, ...). Normalmente está asociado al tipo de geom con que trabajamos y normalmente las opciones por defecto acostumbran a acertar el más idóeno, pero lo podemos modificar seggún nuestros intereses..
- **facets** i **scales** permiten visualizar diferentes subconjuntos de los datos y controlar la representación en el espacio.
- Diferentes elementos se pueden incluir en el gráfico con el operador +

Elementos de un ggplot

Como idea básica en un gráfico ggplot se combinan diferentes elementos que pueden dar lugar a diferentes capas (layers) en un mismo gráfico

Elementos de un ggplot

- Les datos que se quieren representar almacenados en un **data frame**.
- Objectos geométricos (**geoms**) que definen el aspecto global de la capa (barras, puntos, líneas, ...)
- Atributos estéticos (**aesthetics**) que son propiedades visuales de los geoms como la posición, el color de línea, formas de los puntos, etc.
- Un resumen estadístico de los datos (**stats**)(contaje, suavizado, ...). Normalmente está asociado al tipo de geom con que trabajamos y normalmente las opciones por defecto acostumbran a acertar el más idóeno, pero lo podemos modificar seggún nuestros intereses..
- **facets** i **scales** permiten visualizar diferentes subconjuntos de los datos y controlar la representación en el espacio.
- Diferentes elementos se pueden incluir en el gráfico con el operador +

Elementos de un ggplot

Como idea básica en un gráfico ggplot se combinan diferentes elementos que pueden dar lugar a diferentes capas (layers) en un mismo gráfico

Elementos de un ggplot

- Les datos que se quieren representar almacenados en un **data frame**.
- Objectos geométricos (**geoms**) que definen el aspecto global de la capa (barras, puntos, líneas, ...)
- Atributos estéticos (**aesthetics**) que son propiedades visuales de los geoms como la posición, el color de línea, formas de los puntos, etc.
- Un resumen estadístico de los datos (**stats**)(contaje, suavizado, ...). Normalmente está asociado al tipo de geom con que trabajamos y normalmente las opciones por defecto acostumbran a acertar el más idóeno, pero lo podemos modificar seggún nuestros intereses..
- **facets** i **scales** permiten visualizar diferentes subconjuntos de los datos y controlar la representación en el espacio.
- Diferentes elementos se pueden incluir en el gráfico con el operador +

Elementos de un ggplot

Como idea básica en un gráfico ggplot se combinan diferentes elementos que pueden dar lugar a diferentes capas (layers) en un mismo gráfico

Elementos de un ggplot

- Les datos que se quieren representar almacenados en un **data frame**.
- Objectos geométricos (**geoms**) que definen el aspecto global de la capa (barras, puntos, líneas, ...)
- Atributos estéticos (**aesthetics**) que son propiedades visuales de los geoms como la posición, el color de línea, formas de los puntos, etc.
- Un resumen estadístico de los datos (**stats**)(contaje, suavizado, ...). Normalmente está asociado al tipo de geom con que trabajamos y normalmente las opciones por defecto acostumbran a acertar el más idóeno, pero lo podemos modificar seggún nuestros intereses..
- **facets** i **scales** permiten visualizar diferentes subconjuntos de los datos y controlar la representación en el espacio.
- Diferentes elementos se pueden incluir en el gráfico con el operador +

Elementos de un ggplot

Como idea básica en un gráfico ggplot se combinan diferentes elementos que pueden dar lugar a diferentes capas (layers) en un mismo gráfico

Elementos de un ggplot

- Les datos que se quieren representar almacenados en un **data frame**.
- Objectos geométricos (**geoms**) que definen el aspecto global de la capa (barras, puntos, líneas, ...)
- Atributos estéticos (**aesthetics**) que son propiedades visuales de los geoms como la posición, el color de línea, formas de los puntos, etc.
- Un resumen estadístico de los datos (**stats**)(contaje, suavizado, ...). Normalmente está asociado al tipo de geom con que trabajamos y normalmente las opciones por defecto acostumbran a acertar el más idóeno, pero lo podemos modificar seggún nuestros intereses..
- **facets** i **scales** permiten visualizar diferentes subconjuntos de los datos y controlar la representación en el espacio.
- Diferentes elementos se pueden incluir en el gráfico con el operador +

Elementos de un ggplot

Como idea básica en un gráfico ggplot se combinan diferentes elementos que pueden dar lugar a diferentes capas (layers) en un mismo gráfico

Elementos de un ggplot

- Les datos que se quieren representar almacenados en un **data frame**.
- Objectos geométricos (**geoms**) que definen el aspecto global de la capa (barras, puntos, líneas, ...)
- Atributos estéticos (**aesthetics**) que son propiedades visuales de los geoms como la posición, el color de línea, formas de los puntos, etc.
- Un resumen estadístico de los datos (**stats**)(contaje, suavizado, ...). Normalmente está asociado al tipo de geom con que trabajamos y normalmente las opciones por defecto acostumbran a acertar el más idóeno, pero lo podemos modificar seggún nuestros intereses..
- **facets** i **scales** permiten visualizar diferentes subconjuntos de los datos y controlar la representación en el espacio.
- Diferentes elementos se pueden incluir en el gráfico con el operador +

Proceso para crear un gráfico

El proceso para la creación de un gráfico con ggplot2 difiere del proceso típico al que estamos acostumbrados.
Los pasos básicos son:

1. Crear un objeto de la clase *ggplot*, típicamente especificando los datos y algun **aesthetics**.
2. Añadir **geoms** y otros elementos para crear y personalizar el gráfico a través del operador +.

Conjuntos de datos utilizados

Entre otros utilizaremos los conjuntos de datos:

- El conjunto de datos sobre automóviles: **mtcars**

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.00	6.00	160.00	110.00	3.90	2.62	16.46	0.00	1.00	4.00	4.00
Mazda RX4 Wag	21.00	6.00	160.00	110.00	3.90	2.88	17.02	0.00	1.00	4.00	4.00
Datsun 710	22.80	4.00	108.00	93.00	3.85	2.32	18.61	1.00	1.00	4.00	1.00
Hornet 4 Drive	21.40	6.00	258.00	110.00	3.08	3.21	19.44	1.00	0.00	3.00	1.00
Hornet Sportabout	18.70	8.00	360.00	175.00	3.15	3.44	17.02	0.00	0.00	3.00	2.00
Valiant	18.10	6.00	225.00	105.00	2.76	3.46	20.22	1.00	0.00	3.00	1.00

- El conjunto sobre las flores de género *Iris*: **iris**

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.10	3.50	1.40	0.20	setosa
2	4.90	3.00	1.40	0.20	setosa
3	4.70	3.20	1.30	0.20	setosa
4	4.60	3.10	1.50	0.20	setosa
5	5.00	3.60	1.40	0.20	setosa
6	5.40	3.90	1.70	0.40	setosa

Conjuntos de datos utilizados (2)

- El conjunto de datos sobre calidad del aire: **airquality**

```
> data(airquality)
> attach(airquality)
> airquality$Obsdata<-paste('2016',Month,Day,sep='-')
> detach(airquality)
> print(xtable(head(airquality)),scalebox=0.75)
```

	Ozone	Solar.R	Wind	Temp	Month	Day	Obsdata
1	41	190	7.40	67	5	1	2016-5-1
2	36	118	8.00	72	5	2	2016-5-2
3	12	149	12.60	74	5	3	2016-5-3
4	18	313	11.50	62	5	4	2016-5-4
5			14.30	56	5	5	2016-5-5
6	28		14.90	66	5	6	2016-5-6

Conjuntos de datos utilizados (3)

Número de teléfonos por continente 1951-1961

```
> aux<-as.data.frame(t(WorldPhones))
> aux$cont<-rownames(aux)
> telef<-reshape(aux,direction='long',idvar='cont',
+         varying = list(1:7),times=colnames(aux)[1:7]
+         ,timevar='year',v.names='telef')
> head(telef)
```

```
                  cont year telef
N.Amer.1951    N.Amer 1951 45939
Europe.1951    Europe 1951 21574
Asia.1951        Asia 1951  2876
S.Amer.1951    S.Amer 1951  1815
Oceania.1951 Oceania 1951  1646
Africa.1951    Africa 1951    89
```

Inicializando un objeto **ggplot**

Utilizamos la función ggplot para inicializar un gráfico.

```
> objeto<-ggplot(dataframe)
```

o si no queremos guardar el objeto

```
> ggplot(dataframe)
```

¿Qué sucede si ejecutamos la siguiente instrucción?

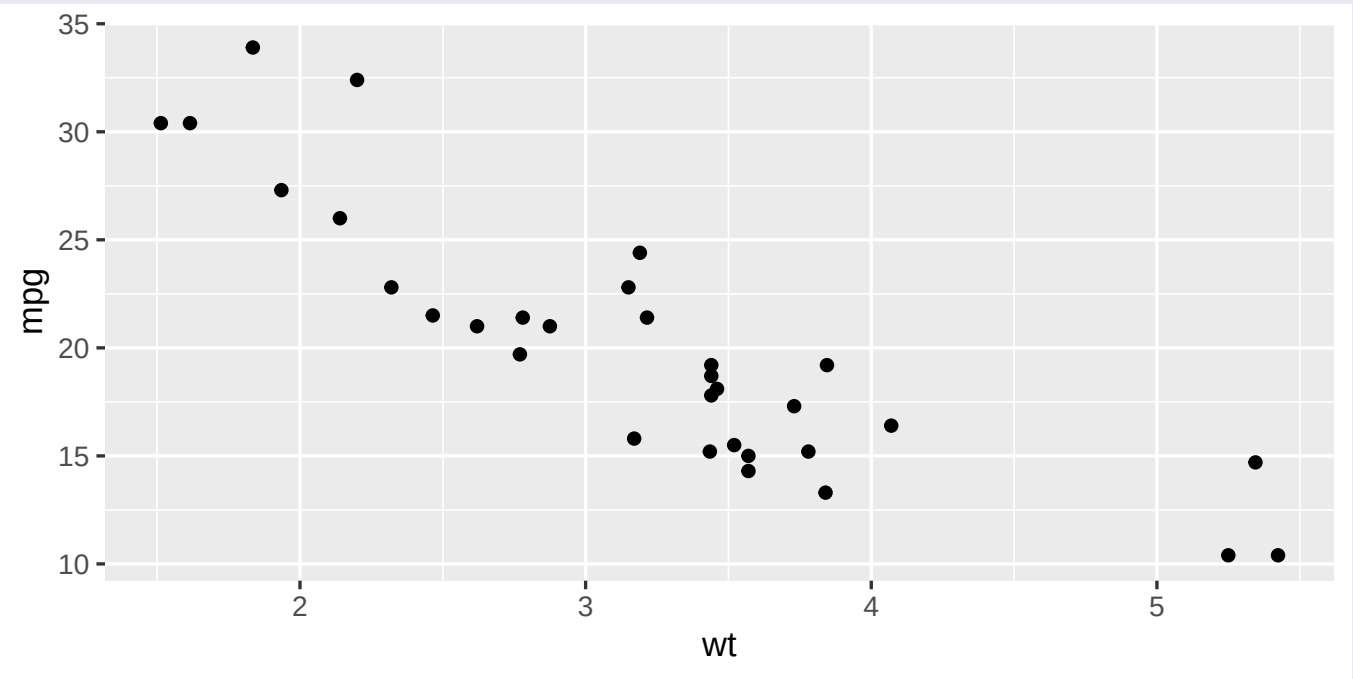
¿Qué sucede si ejecutamos la siguiente instrucción?

```
> library(ggplot2)
> ggplot(mtcars)
```

Un primer ejemplo de gráfico

El siguiente paso es añadir alguna capa sobre el gráfico. Por ejemplo la función **geom_point()** añade una capa de puntos.

```
> library(ggplot2)
> ggplot(data=mtcars)+geom_point(aes(x=wt,y=mpg))
```



Variaciones sobre un mismo tema

Existe una gran flexibilidad en la forma cómo se crea el gráfico. También funcionan las instrucciones:

```
> ggplot(data=mtcars,aes(x=wt,y=mpg))+geom_point()
```

o

```
> ggplot()+geom_point(data=mtcars,aes(x=wt,y=mpg))
```

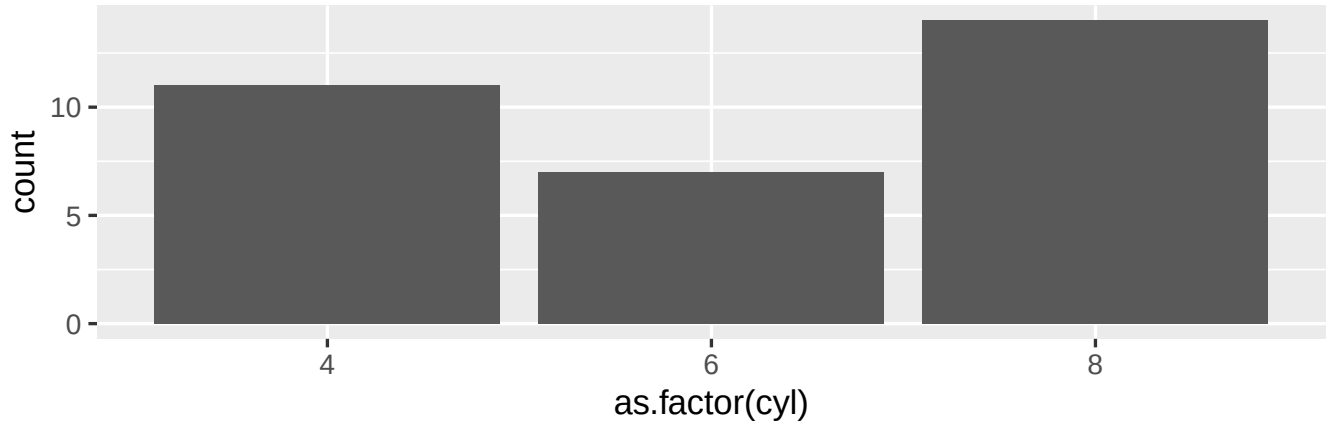
Incluir el **aes** en la llamada a *ggplot* permite que sea válida para diferentes **geom**s que podemos añadir posteriormente.

Aesthetic

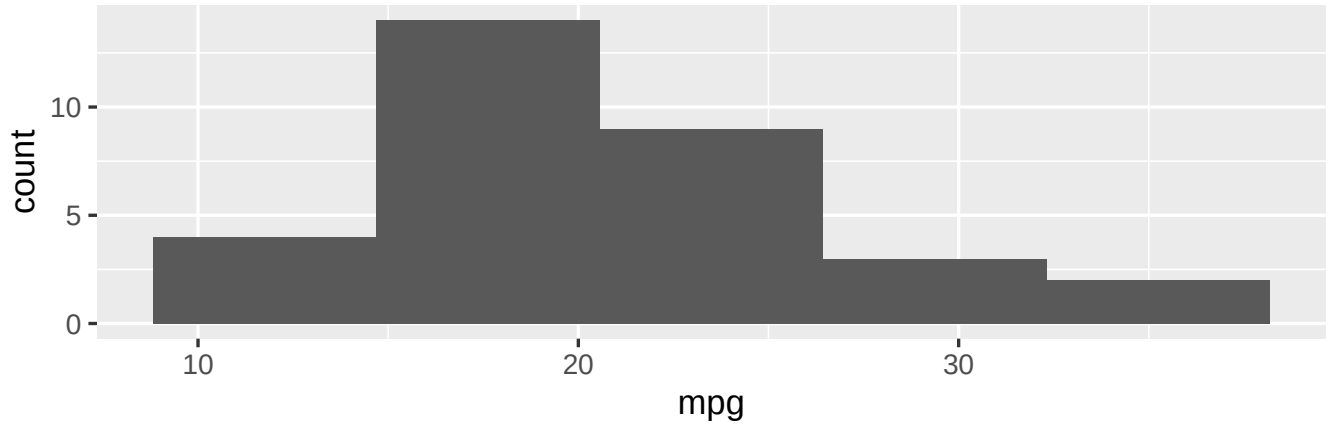
- En un ggplot *aesthetic* **aes()** se refiere a "aquello que podemos ver". Es una propiedad visual de un objeto. Incluye:
  - x,y: qué va en los ejes
  - color: color del exterior
  - fill: color del interior
  - shape: forma de los puntos
  - linetype: tipo de línea
  - size: tamaño
  - alpha: transparencia (1:opaco; 0:transparente)
- Cada tipo de geometría acepta un subconjunto de las posibles opciones
- Una de las funciones más utilizadas es la de definir grupos a través de diversas variables aesthetics o directamente con la opción *group*

Más ejemplos

```
> ggplot(mtcars)+geom_bar(aes(x=as.factor(cyl)))
```

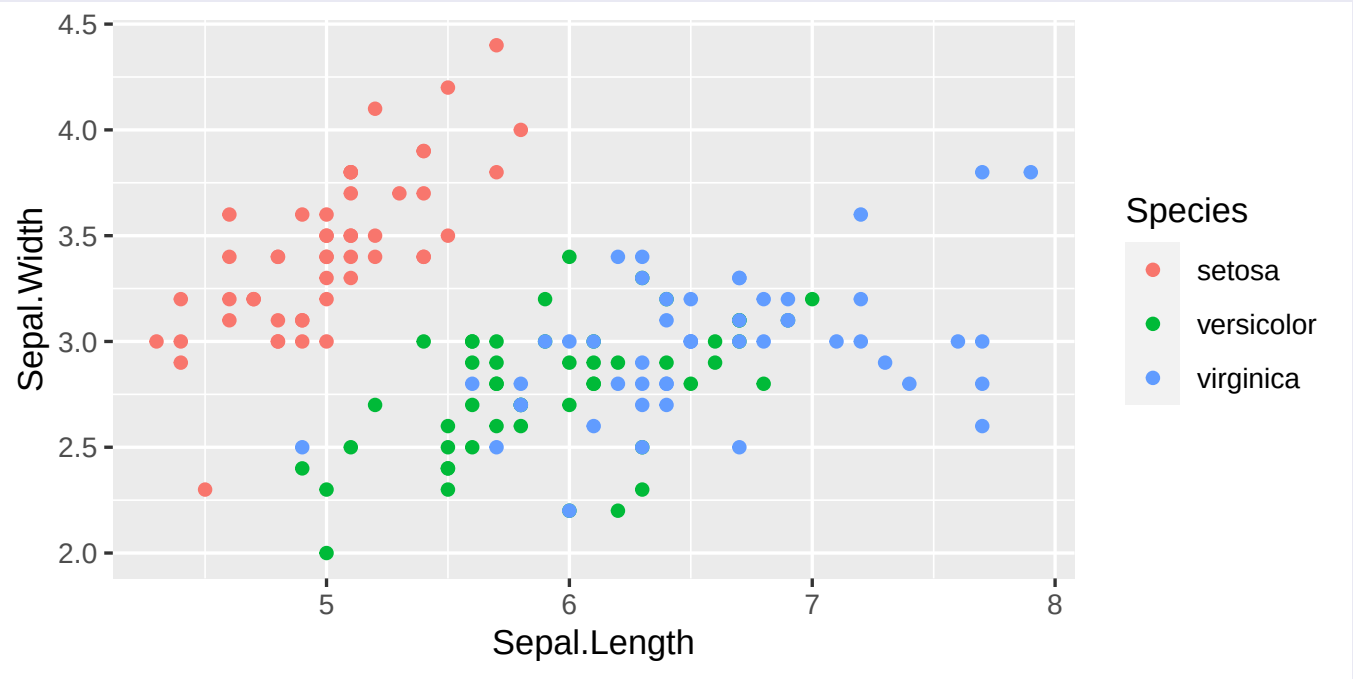



```
> ggplot(mtcars)+geom_histogram(aes(x=mpg),bins=5)
```



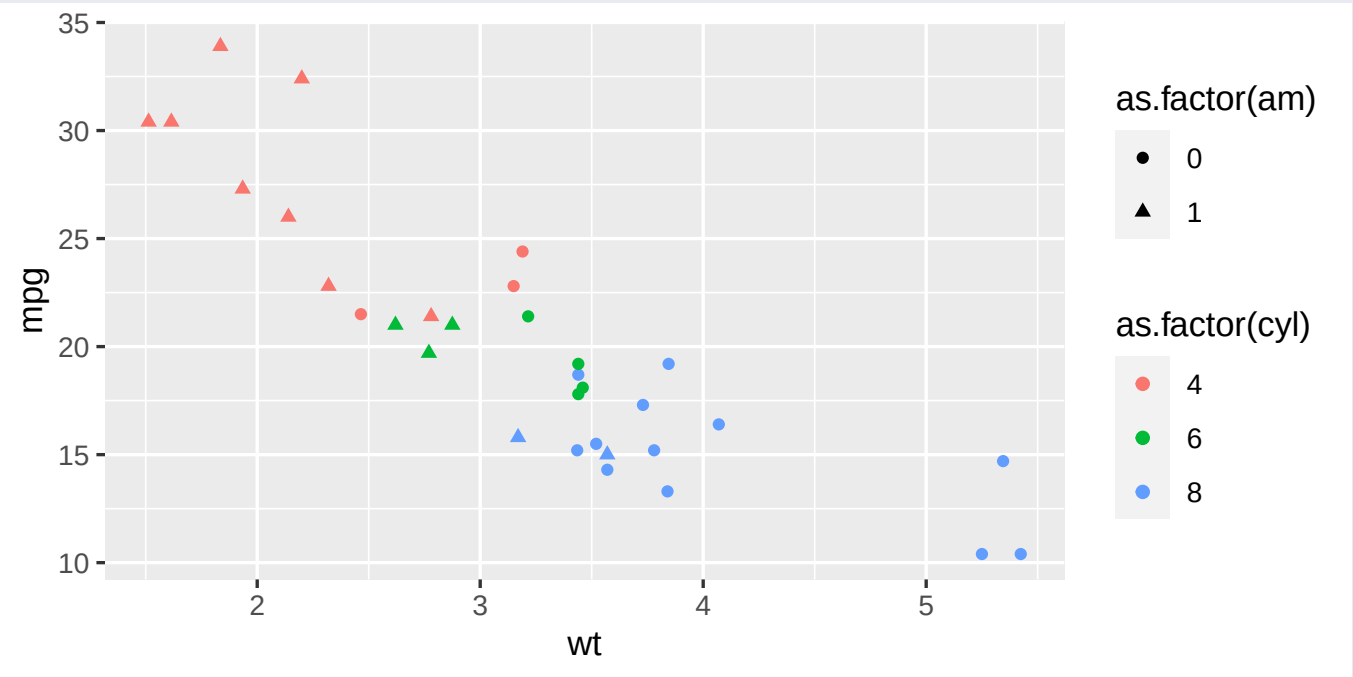
Definición de grupos a través de **aes**

```
> ggplot(data=iris) +
+  geom_point(aes(x=Sepal.Length, y=Sepal.Width,col=Species))
```



Definición de grupos a través de **aes** (2)

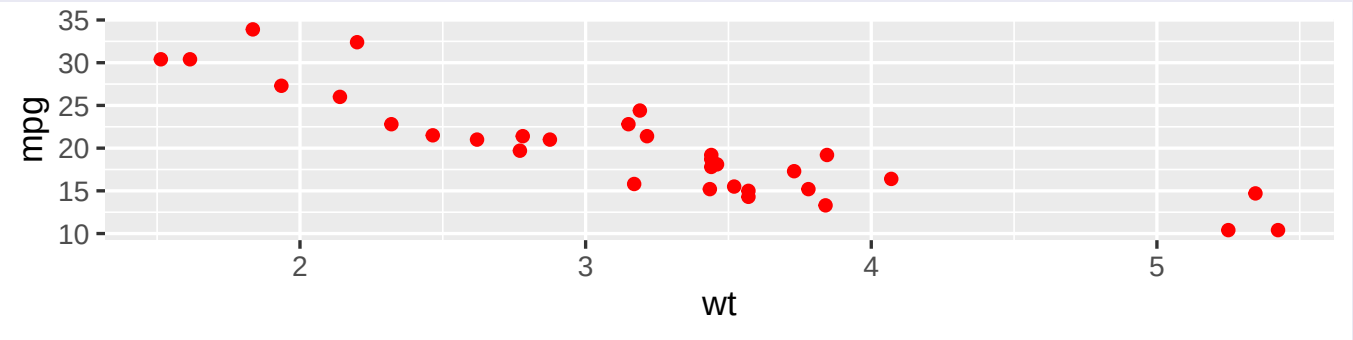
```
> ggplot(data=mtcars) +
+  geom_point(aes(x=wt, y=mpg,col=as.factor(cyl),shape=as.factor(am)))
```



Propiedades **aes** independientes de variables (constantes)

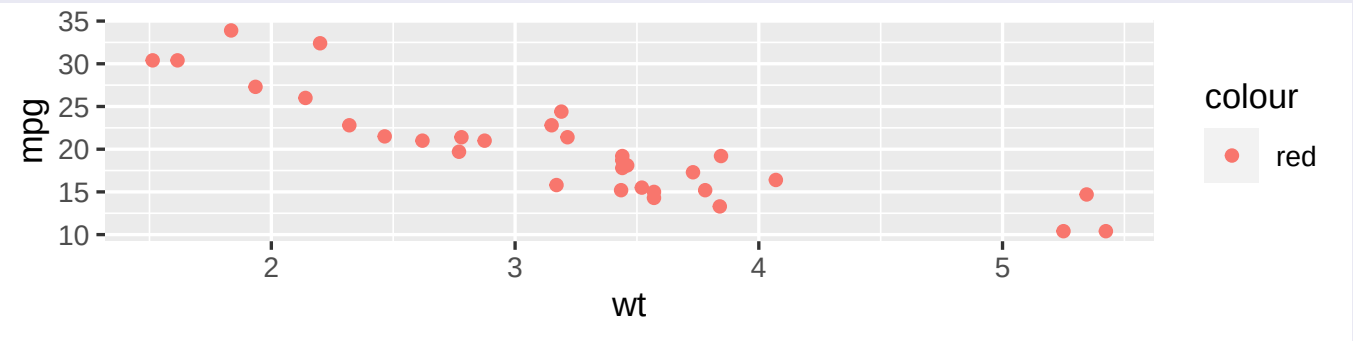
Veamos el resultado de la siguiente instrucción:

```
> ggplot(data=mtcars) + geom_point(aes(x=wt, y=mpg),col='red')
```



y compararlo con:

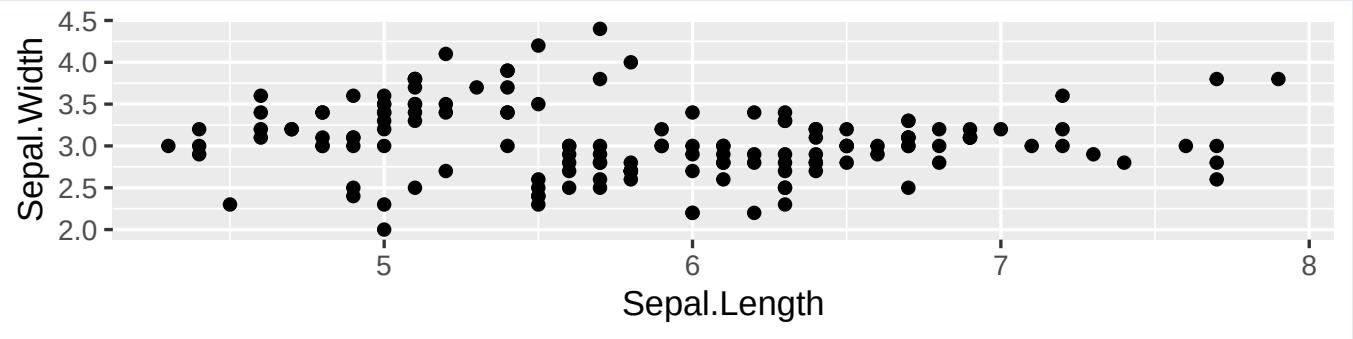
```
> ggplot(data=mtcars) + geom_point(aes(x=wt, y=mpg,col='red'))
```



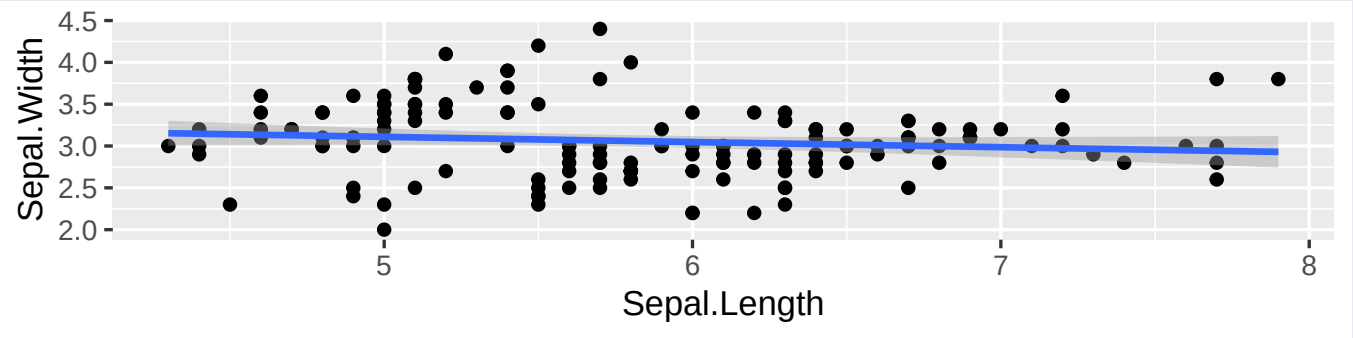
Esquema del tema

Modificación del gráfico añadiendo **geoms**

```
> p<-ggplot(data=iris,aes(x=Sepal.Length, y=Sepal.Width))
> p+geom_point()
```




```
> p+geom_point()+geom_smooth(method='lm')
```



Ejercicios

Ejercicios

1. Explicar el resultado que se obtiene

```
> p<-ggplot(data=iris)
> p+geom_point(aes(x=Sepal.Length, y=Sepal.Width))
> p+geom_point()+geom_smooth(method='lm')
```

2. Interpretar el resultado

```
> ggplot(data=mtcars) + geom_point(aes(x=wt,
+                                      y=mpg,col=as.factor(cyl)))
> ggplot(data=mtcars) + geom_point(aes(x=wt, y=mpg,col=cyl))
```

3. ¿Qué se obtiene si una misma variable se mapea en múltiples aesthetics?
4. ¿Qué sucede si un aesthetic se asigna al resultado de una comparación lógica?

Objetos geométricos (geom)

- Los objetos geométricos son las diferentes representaciones que realmente salen en la gráfica
- Una gráfica debe tener al menos una geometria pero no hay límite superior
- Las diferentes geometrías se añaden con el operador +

Algunas Geometrías

- puntos (geom_point)
- lineas (geom_line)
- lineas conectadas en orden (geom_path)
- box-plots (geom_boxplot)
- lineas de suavizado (geom_smooth)
- densidades (geom_density)
- textos (geom_text)
- recta (geom_abline)
- recta hor. (geom_hline)
- recta ver. (geom_vline)
- histograma (geom_histogram)
- polígonos (geom_polygon)
- etiquetas (geom_label)
- ...

Opciones de las geoms

Por supuesto cada geom tiene su conjunto de opciones

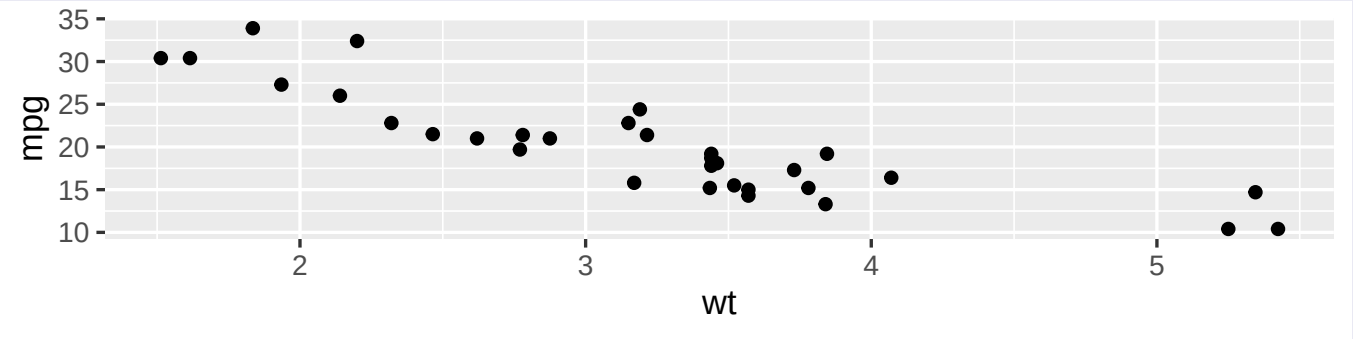
```
geom_point(mapping = NULL, data = NULL, stat = "identity",
  position = "identity", ..., na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE)
geom_line(mapping = NULL, data = NULL, stat = "identity",
  position = "identity", na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE, ...)
geom_bar(mapping = NULL, data = NULL, stat = "count",
  position = "stack", ..., width = NULL, binwidth = NULL,
  na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)
  ......
```

La opción **data** puede sobreescribir el conjunto de datos definido en la llamada a *ggplot*().

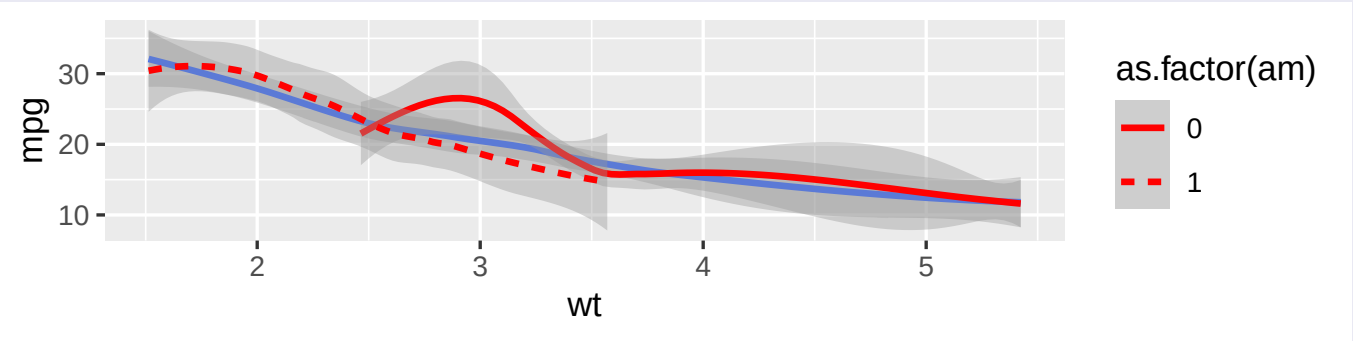
Aplicación de diferentes geometrías

Veamos el resultado de aplicar diferentes geometrías

```
> p<-ggplot(mtcars,aes(x=wt,y=mpg))
> p+geom_point()
```

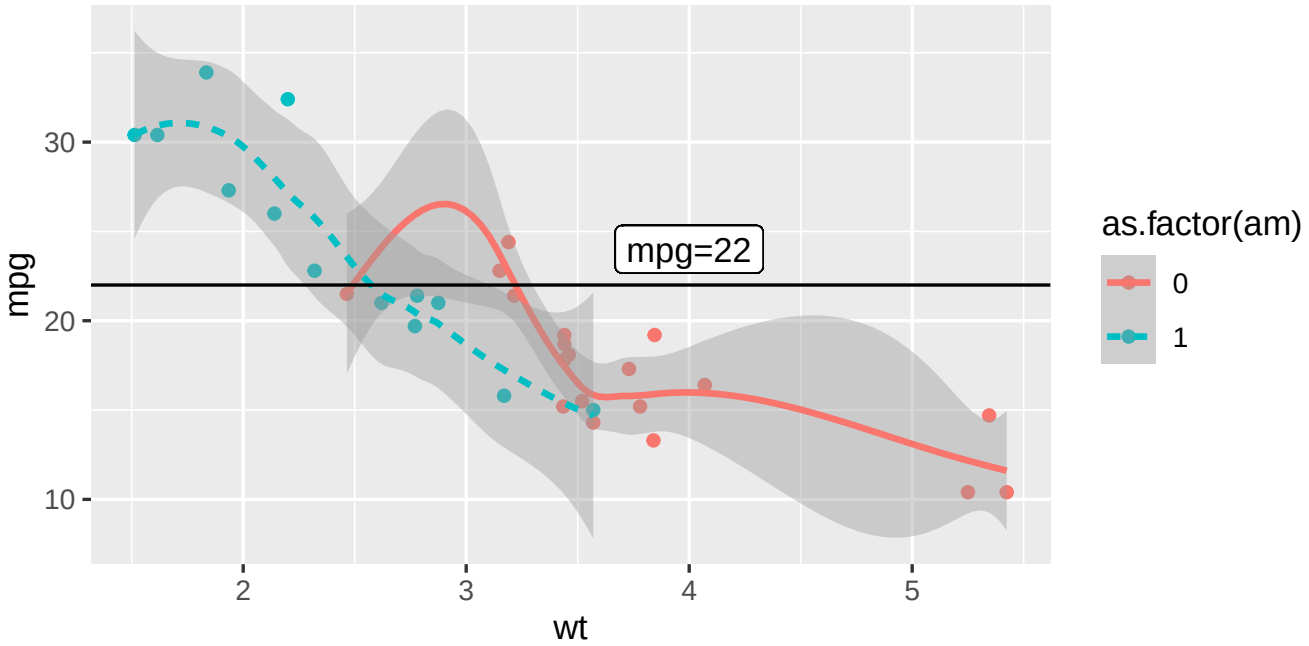



```
> p+geom_smooth()+geom_smooth(aes(linetype=as.factor(am)),col='red')
```



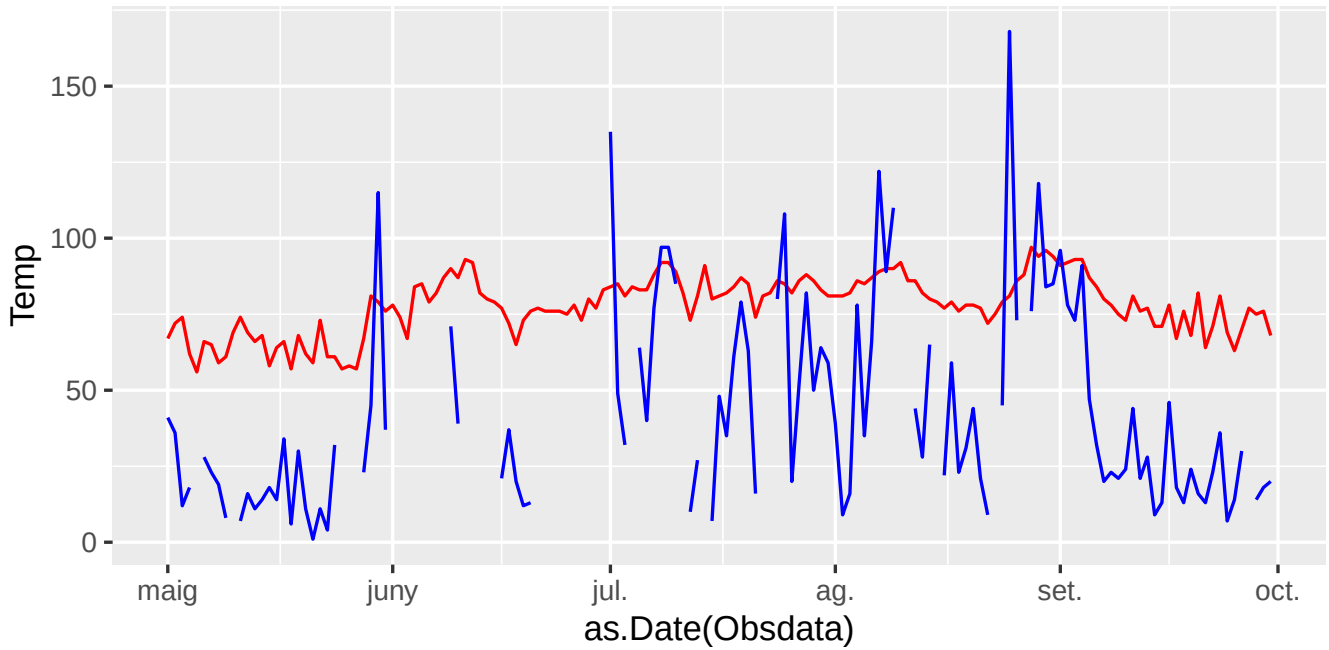
Combinando geometrías

```
> p+geom_point(aes(color=as.factor(am)))+
+  geom_smooth(aes(linetype=as.factor(am),col=as.factor(am)))+
+  geom_hline(yintercept=22)+
+  geom_label(aes(x=4,y=24,label='mpg=22'))
```



Combinando geometrías (2)

```
> ggplot(airquality,aes(x=as.Date(Obsdata)))+
+  geom_line(aes(y=Temp),col='red')+
+  geom_line(aes(y=Ozone),color='blue')
```



Ejercicios

Definimos

```
> p1<-ggplot(data=iris,aes(x=Sepal.Length,y=Sepal.Width))
```

Imaginar los resultados y probar estas modificaciones

1. `p1+geom_point()+ geom_smooth(aes(fill=Species))`
2. `p1+geom_point(aes(colour=Species))+ geom_smooth()`
3. `p2<-p1+aes(colour=Species)`
4. `p2+geom_point()+geom_smooth(method='lm')`

Para más información sobre otras geoms ver ggplot2 cheatsheets

Ejercicios

Definimos

```
> p1<-ggplot(data=iris,aes(x=Sepal.Length,y=Sepal.Width))
```

Imaginar los resultados y probar estas modificaciones

1. `p1+geom_point()+ geom_smooth(aes(fill=Species))`
2. `p1+geom_point(aes(colour=Species))+ geom_smooth()`
3. `p2<-p1+aes(colour=Species)`
4. `p2+geom_point()+geom_smooth(method='lm')`

Para más información sobre otras geoms ver ggplot2 cheatsheets

Ejercicios

Definimos

```
> p1<-ggplot(data=iris,aes(x=Sepal.Length,y=Sepal.Width))
```

Imaginar los resultados y probar estas modificaciones

1. `p1+geom_point()+ geom_smooth(aes(fill=Species))`
2. `p1+geom_point(aes(colour=Species))+ geom_smooth()`
3. `p2<-p1+aes(colour=Species)`
4. `p2+geom_point()+geom_smooth(method='lm')`

Para más información sobre otras geoms ver ggplot2 cheatsheets

Ejercicios

Definimos

```
> p1<-ggplot(data=iris,aes(x=Sepal.Length,y=Sepal.Width))
```

Imaginar los resultados y probar estas modificaciones

1. `p1+geom_point()+ geom_smooth(aes(fill=Species))`
2. `p1+geom_point(aes(colour=Species))+ geom_smooth()`
3. `p2<-p1+aes(colour=Species)`
4. `p2+geom_point()+geom_smooth(method='lm')`

Para más información sobre otras geoms ver ggplot2 cheatsheets

Ejercicios

Definimos

```
> p1<-ggplot(data=iris,aes(x=Sepal.Length,y=Sepal.Width))
```

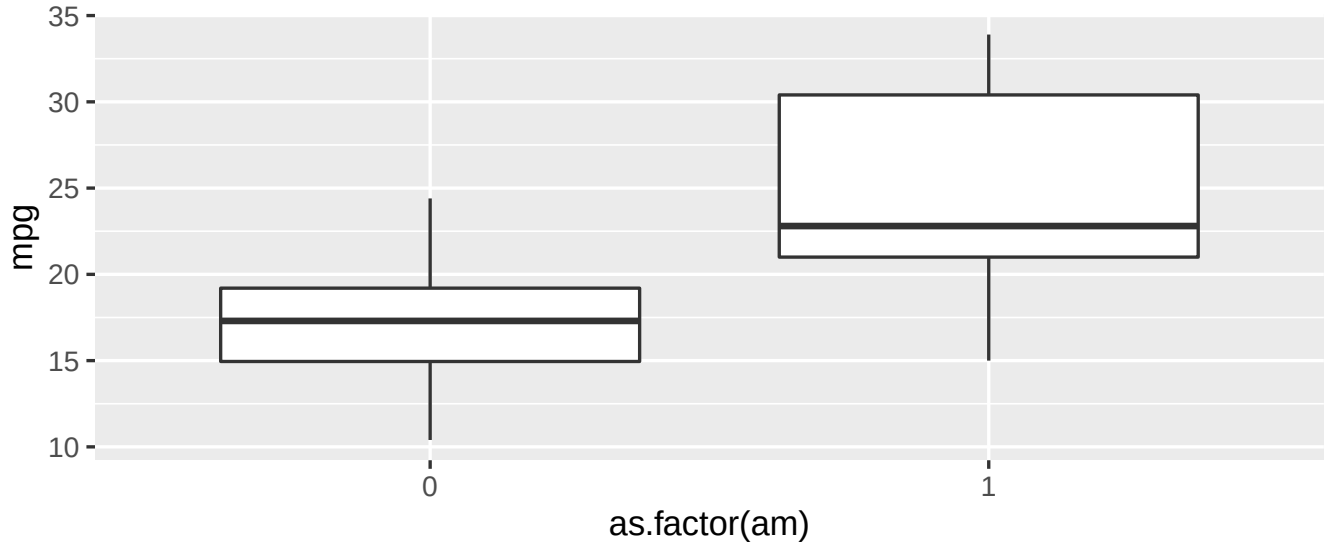
Imaginar los resultados y probar estas modificaciones

1. `p1+geom_point()+`
   `geom_smooth(aes(fill=Species))`
2. `p1+geom_point(aes(colour=Species))+`
   `geom_smooth()`
3. `p2<-p1+aes(colour=Species)`
4. `p2+geom_point()+geom_smooth(method='lm')`

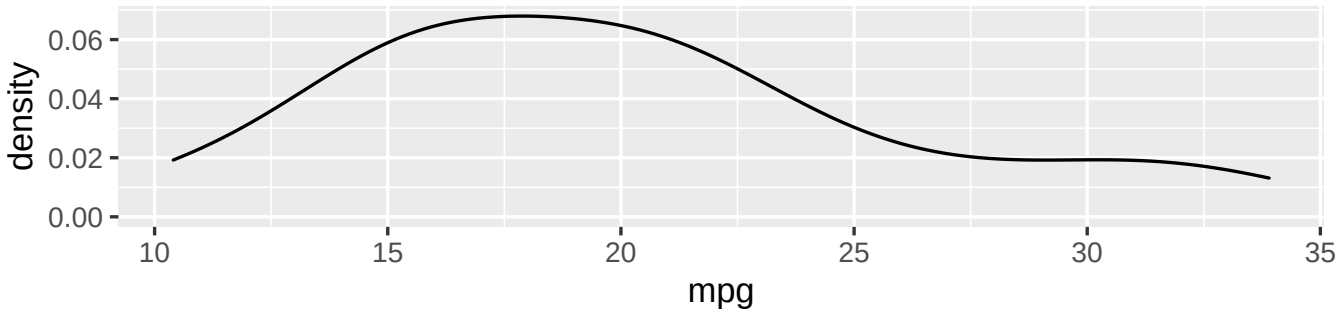
Para más información sobre otras geoms ver ggplot2 cheatsheets

Otras geoms

```
> ggplot(mtcars)+geom_boxplot(aes(x=as.factor(am),y=mpg))
```

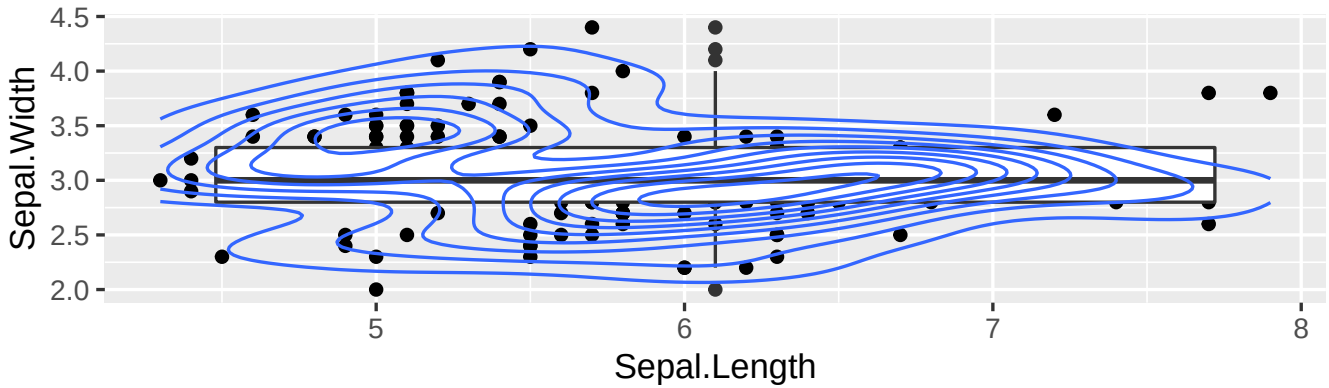



```
> ggplot(mtcars)+geom_density(aes(mpg))
```

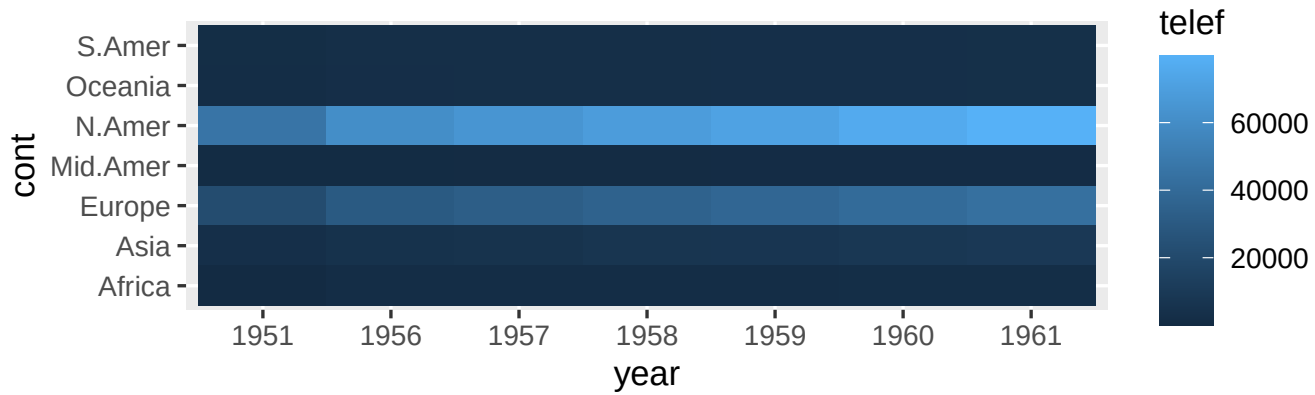


Algunas geoms 2D

```
> p1<-ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width))
> p1+geom_point()+geom_boxplot()+geom_density2d()
```




```
> p2<-ggplot(telef,aes(x=year,y=cont))
> p2+geom_tile(aes(fill=telef))
```



Esquema del tema

Facets

Otra forma de representar diferentes variables categóricas es separar los gráficos en **facets** (facetas), subgráficos para cada valor de la variable.

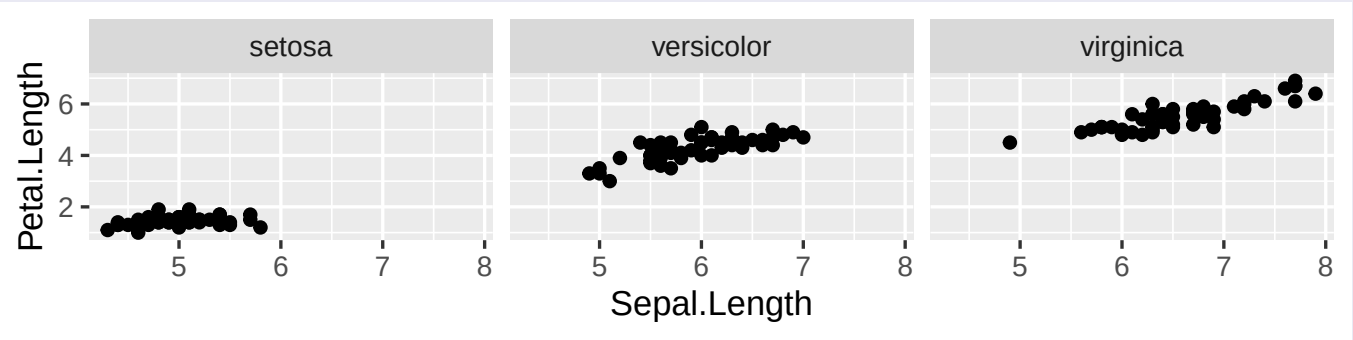
Separar por una variable categórica única

```
facet_wrap(~variable,nrow,ncol)
```

Separar por la combinación de dos variables

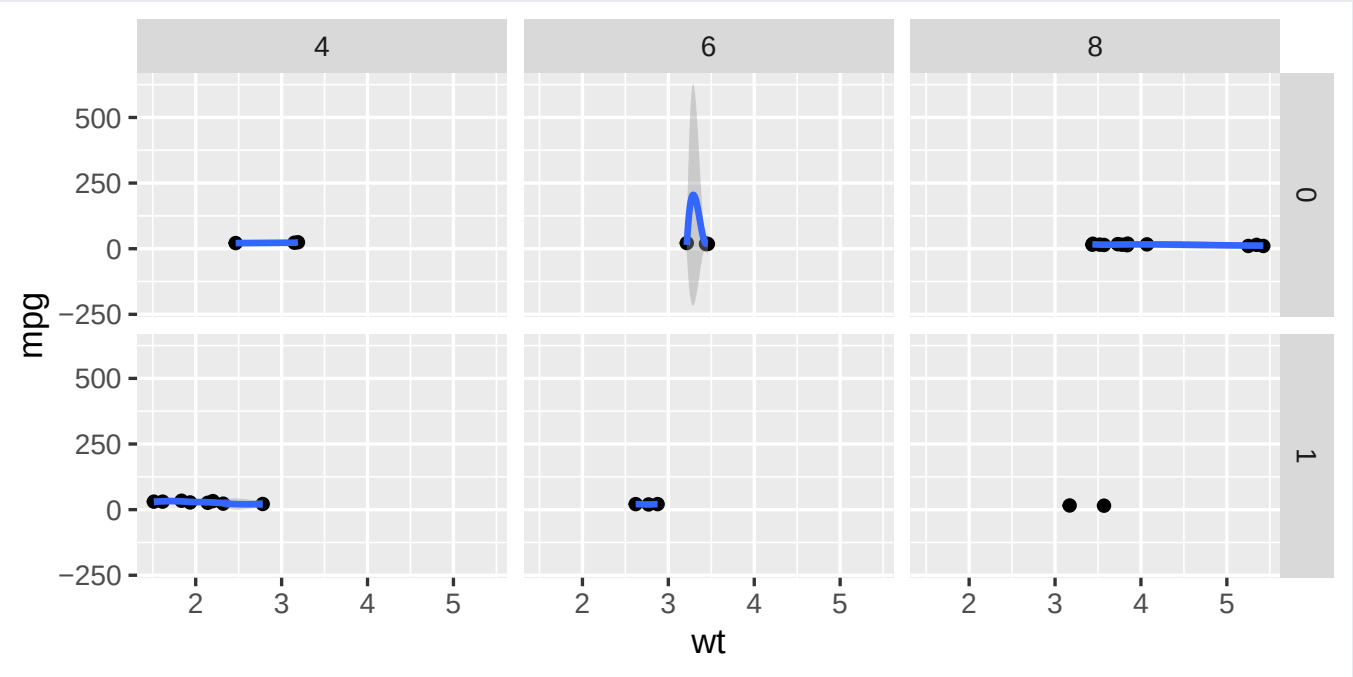
```
facet_grid(variable 1~variable 2)
```

```
> ggplot(iris)+geom_point(aes(x=Sepal.Length,
+    y=Petal.Length))+ facet_wrap(~Species)
```



Facets (2)

```
> ggplot(mtcars,aes(x=wt,y=mpg))+geom_point()+
+  geom_smooth()+
+  facet_grid(as.factor(am)~as.factor(cyl))
```



Ejercicios

1. Interpretar el resultado de la siguiente instrucción

```
> ggplot(mtcars,aes(x=wt,y=mpg))+geom_point()+
+  geom_smooth()+
+  facet_grid(as.factor(am)~as.factor(gear))
```

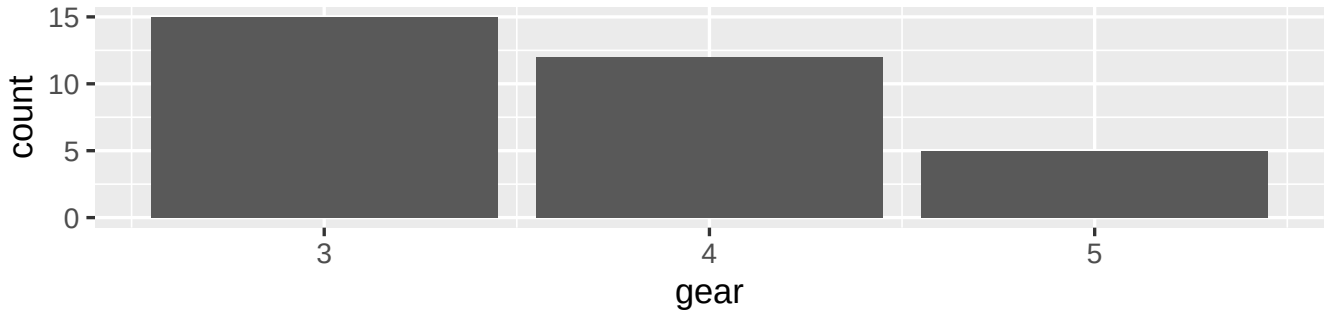
2. ¿Qué ocurre si intentamos separar por una variable continua? ¿Cómo podemos solucionarlo?

Esquema del tema

Transformaciones estadísticas

Si aplicamos la geom_bar a unos datos, por defecto efectúa el contaje de las categorías de la varible y es ese contaje el que representa

```
> ggplot(mtcars)+geom_bar(aes(x=gear))
```



En realidad lo que hace es calcular

	V1
3	15
4	12
5	5

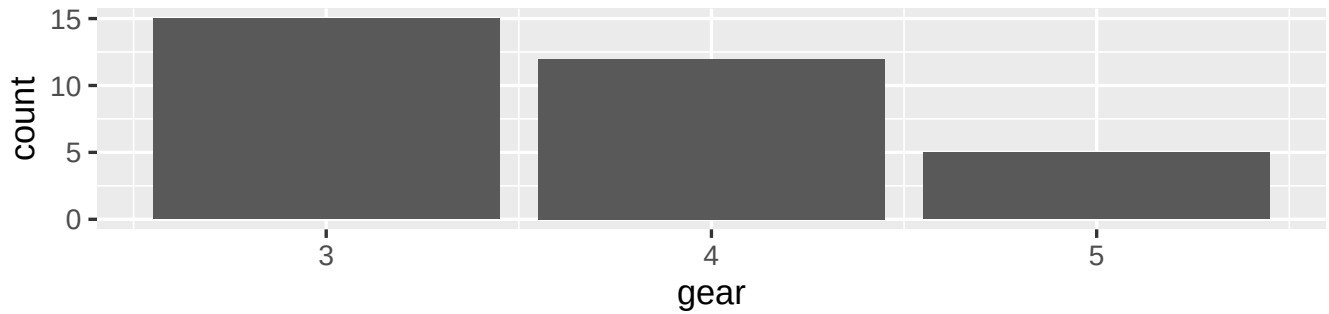
Propiedad **stat**

La propiedad **stat** determina la operación estadística que se realiza sobre los datos. Cada **geom** tiene asociada una propiedad **stat** por defecto.

geom	stat
geom_bar()	stat_count()
geom_col()	stat_identity()
geom_point()	stat_identity()
geom_smooth()	stat_smooth()

Las llamadas son intercambiables, por ejemplo

```
> ggplot(mtcars)+stat_count(aes(x=gear))
```



Opciones de las **stats**

Como siempre cada stat tiene su conjunto de opciones

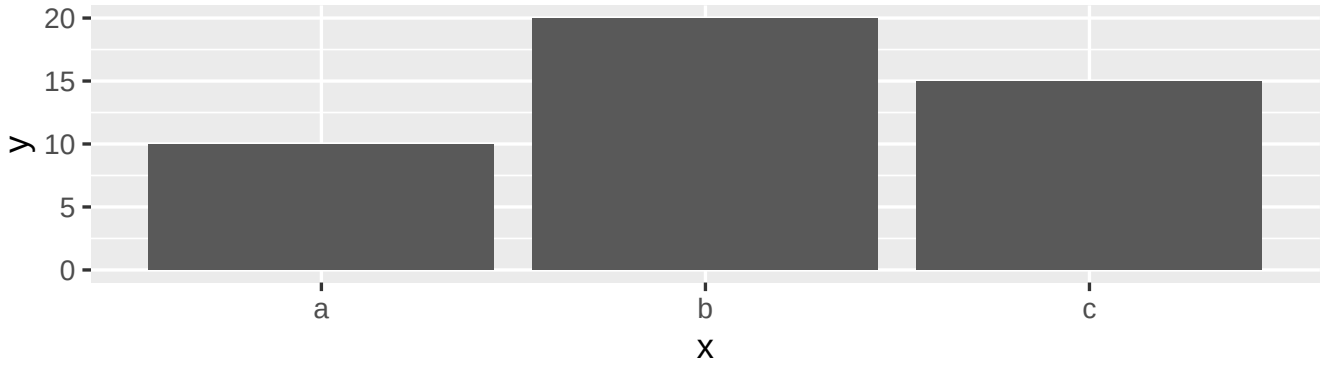
```
stat_count(mapping = NULL, data = NULL, geom = "bar",
  position = "stack", ..., width = NULL,
  na.rm = FALSE,show.legend = NA, inherit.aes = TRUE)
stat_smooth(mapping = NULL, data = NULL, geom = "smooth",
  position = "identity", ..., method = "auto", formula = y ~ x,
  se = TRUE, n = 80, span = 0.75, fullrange = FALSE,
  level = 0.95,  method.args = list(), na.rm = FALSE,
  show.legend = NA,  inherit.aes = TRUE)
  ......
```

Compárese por ejemplo con

```
geom_bar(mapping = NULL, data = NULL, stat = "count",
  position = "stack", ..., width = NULL, binwidth = NULL,
  na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)
geom_smooth(mapping = NULL, data = NULL, stat = "smooth",
  position = "identity", ..., method = "auto", formula = y ~ x,
  se = TRUE, na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)
```

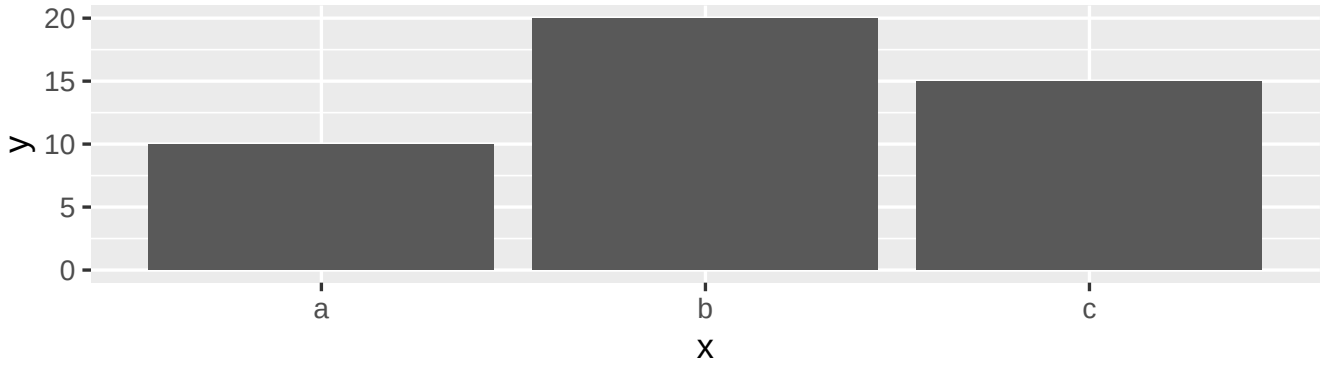
Modificación de la **stat**

```
> aux=data.frame(x=c('a','b','c'),y=c(10,20,15))
> ggplot(aux)+geom_bar(aes(x,y))
> ggplot(aux)+geom_bar(aes(x,y),stat='identity')
```



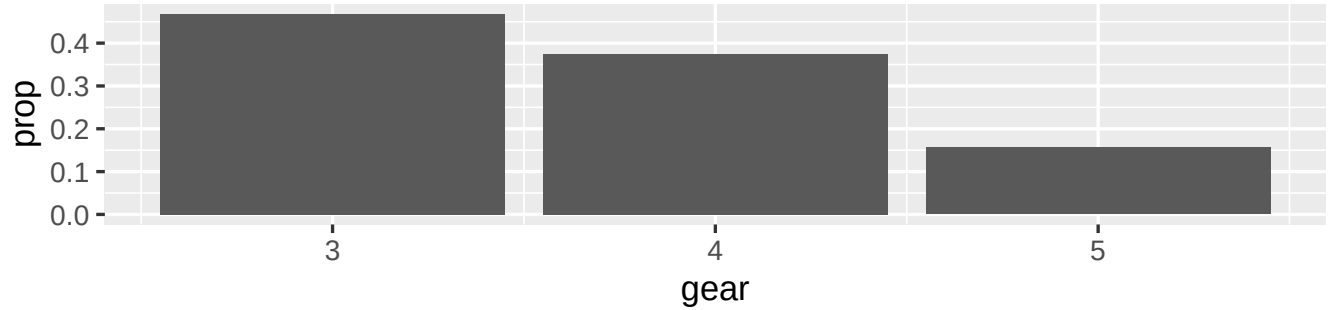
Se habría obtenido el mismo resultado con

```
> ggplot(aux)+geom_col(aes(x,y))
```

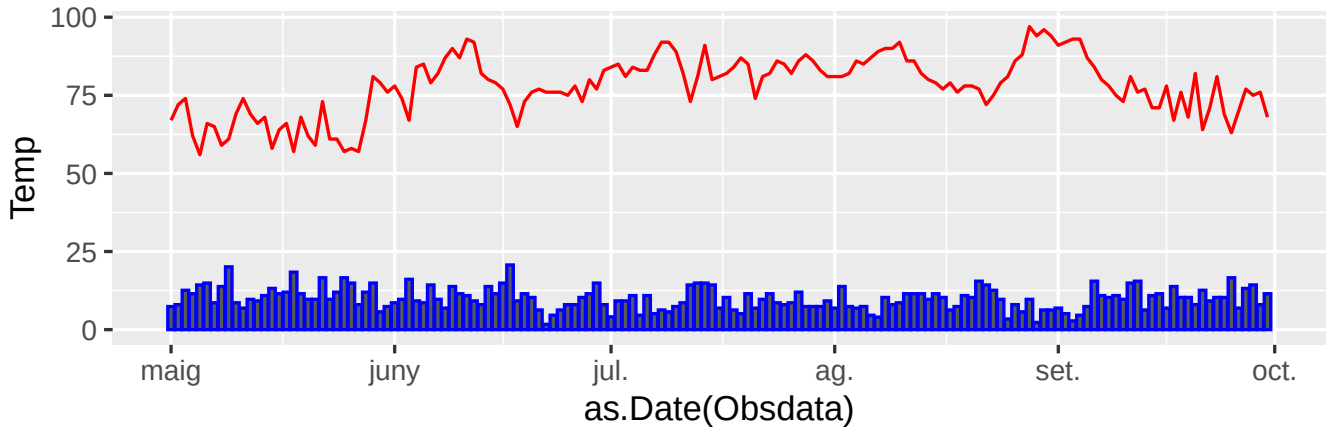


Modificación de la **stat** (2)

```
> ggplot(mtcars)+geom_bar(aes(gear,y=..prop..))
```

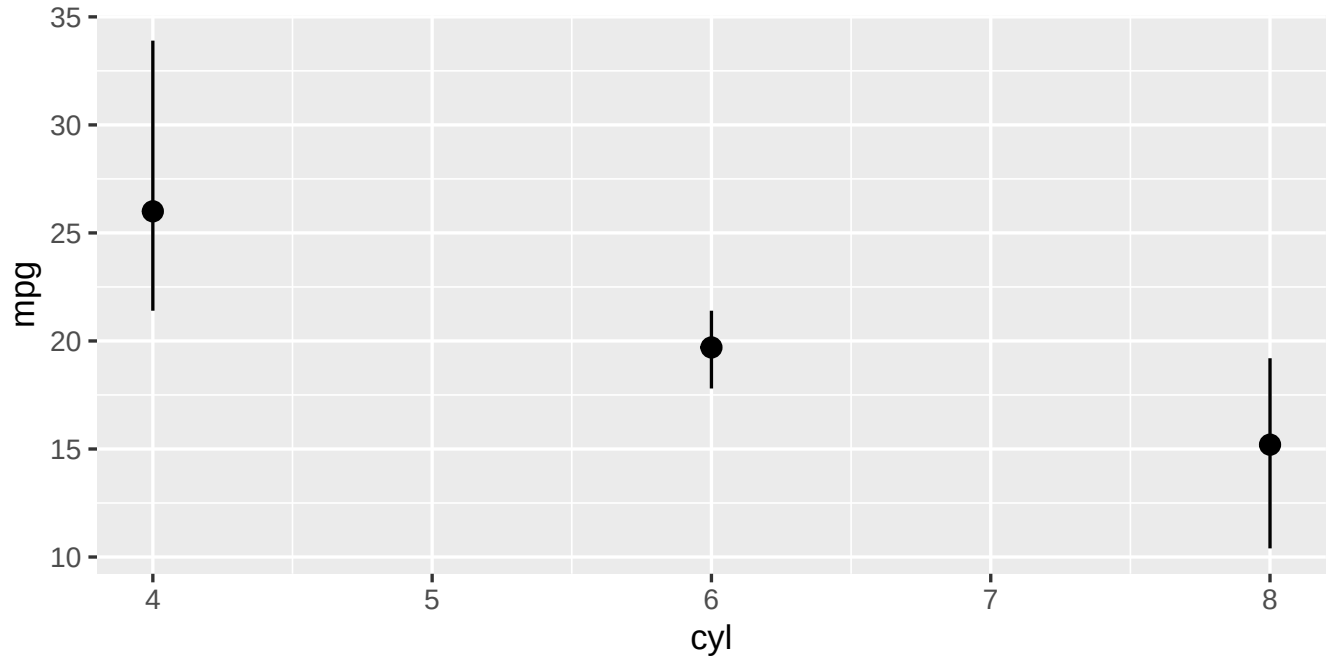



```
> ggplot(airquality,aes(x=as.Date(Obsdata)))+
+  geom_line(aes(y=Temp),col='red')+
+  geom_bar(aes(y=Wind),color='blue',stat='identity')
```



stat_summary()

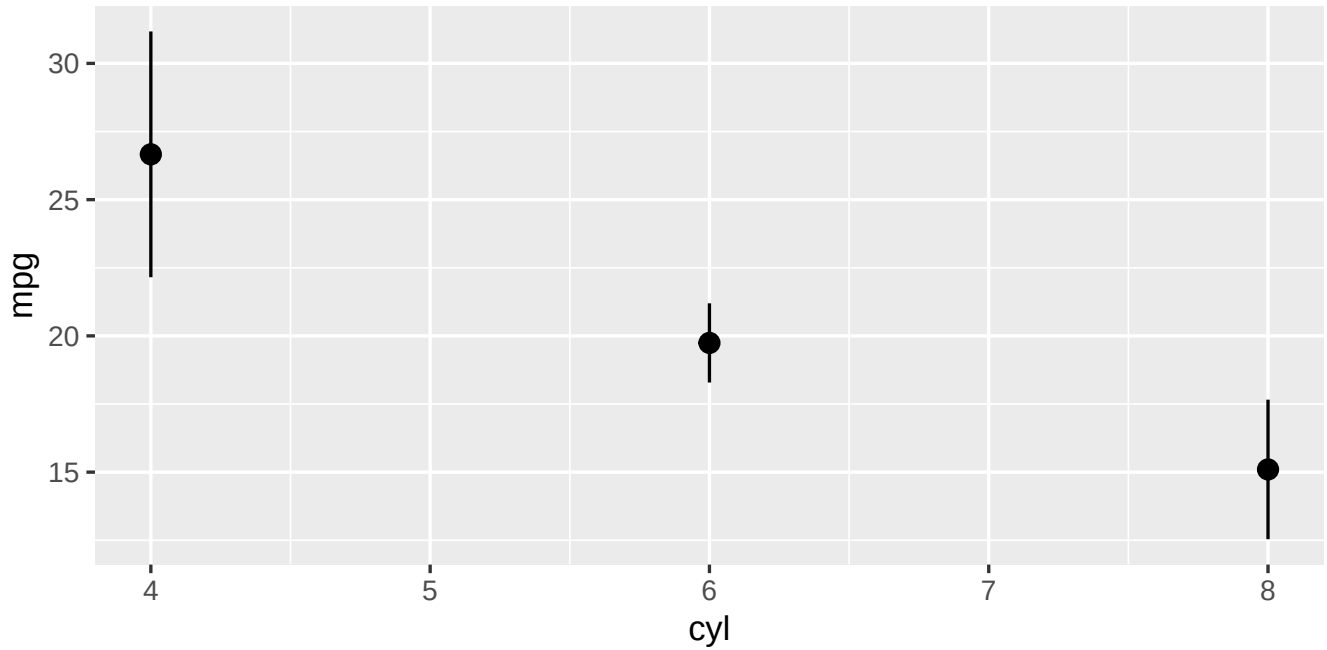
```
> ggplot(mtcars)+
+  stat_summary(aes(x=cyl,y=mpg),fun.ymin=min,
+               fun.ymax=max,fun.y=median)
```



stat_summary() (2)

Las funciones pueden personalizarse

```
> ggplot(mtcars)+
+   stat_summary(aes(x=cyl,y=mpg),
+   fun.ymin=function(x) mean(x)-sd(x),
+   fun.ymax=function(x) mean(x)+sd(x),
+   fun.y=mean)
```



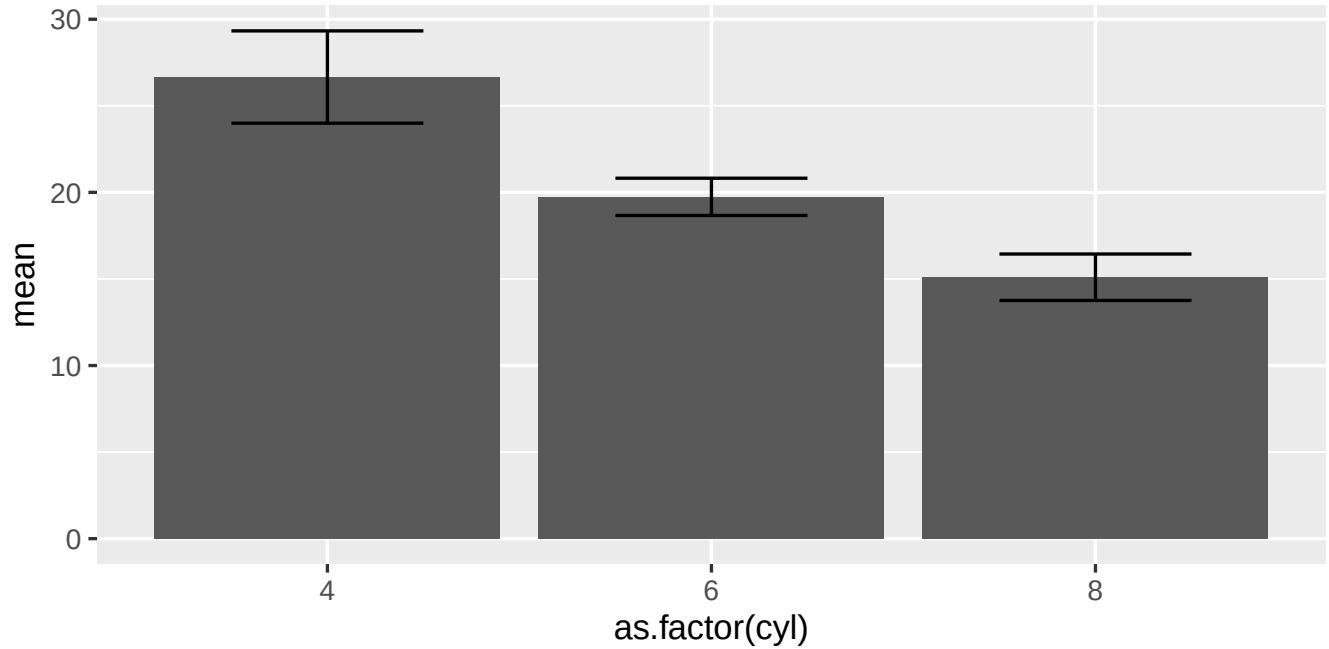
stat_summary() (3)

```
> summaryAmp<-function(x,colvar,colfac){
+  fact<-aggregate(x[,colvar],list(x[,colfac]),mean)$Group.1
+  means<-aggregate(x[,colvar],list(x[,colfac]),mean)$x
+  sds<-aggregate(x[,colvar],list(x[,colfac]),sd)$x
+  len<-aggregate(x[,colvar],list(x[,colfac]),length)$x
+  ses<-sds/sqrt(len)
+  cis<-ses*qnorm(0.975)
+  df<-data.frame(factor=fact,mean=means,sd=sds,se=ses,ci=cis)
+  colnames(df)[1]<-colfac
+  return(df)
+ }
> car.new<-summaryAmp(mtcars,'mpg','cyl')
> print(xtable(car.new))
```

	cyl	mean	sd	se	ci
1	4.00	26.66	4.51	1.36	2.67
2	6.00	19.74	1.45	0.55	1.08
3	8.00	15.10	2.56	0.68	1.34

stat_summary() (4)

```
> p1<-ggplot(car.new,aes(x=as.factor(cyl),y=mean))
> p2<-p1+geom_bar(stat='identity')
> p3<-p2+geom_errorbar(aes(ymin=mean-ci,ymax=mean+ci),width=0.5)
> p3
```



Grupos

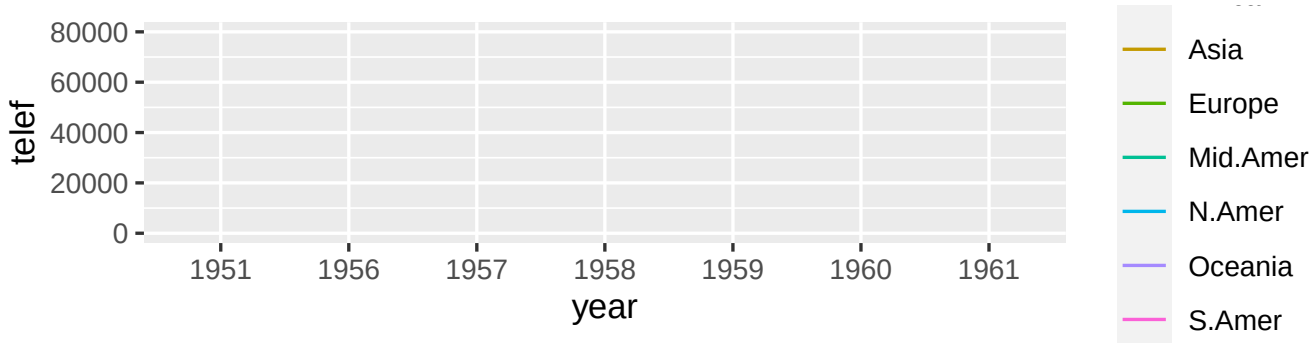
Es frecuente disponer de datos agrupados. Por ejemplo en los datos sobre el número de teléfonos

```
                cont year telef
N.Amer.1951   N.Amer 1951 45939
Europe.1951   Europe 1951 21574
Asia.1951       Asia 1951  2876
S.Amer.1951   S.Amer 1951  1815
Oceania.1951 Oceania 1951  1646
Africa.1951   Africa 1951    89
```

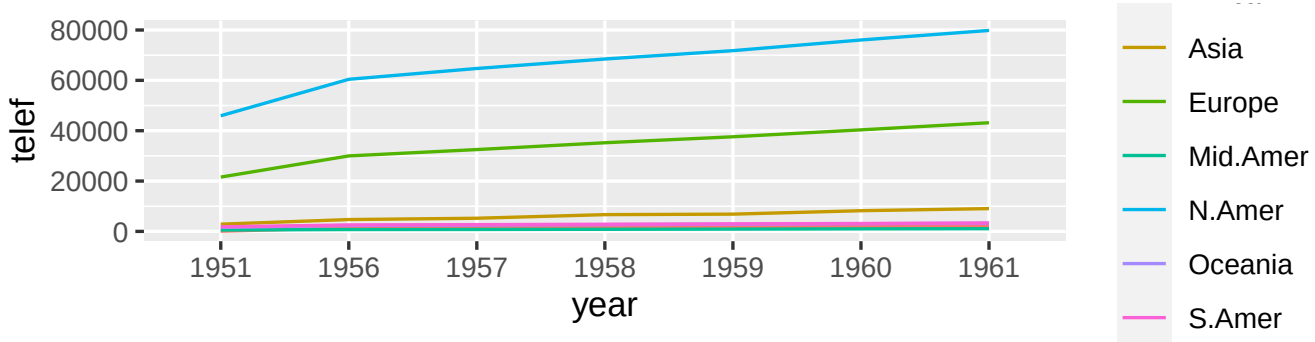
Si queremos graficar la evolución temporal del número de teléfonos por continentes hemos de utilizar la opción **group**.

Grupos (2)

```
> ggplot(telef,aes(x=year,y=telef,col=cont))+geom_line()
```




```
> ggplot(telef,aes(x=year,y=telef,col=cont))+geom_line(aes(group=cont))
```

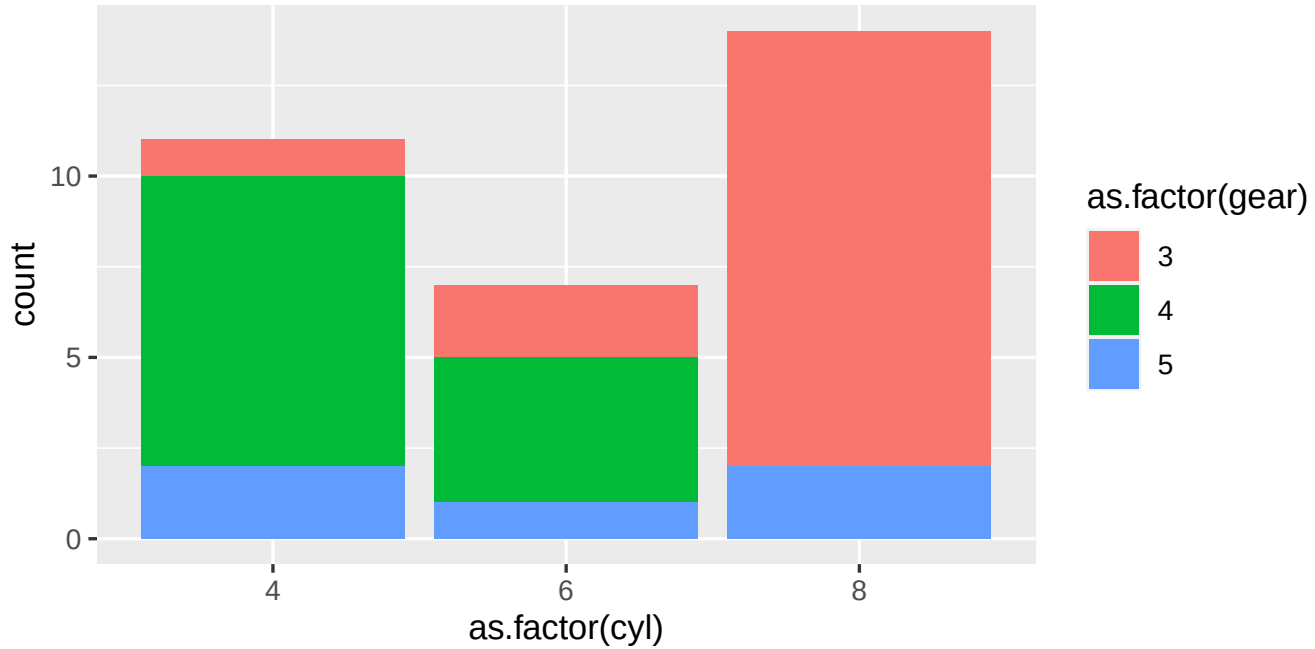


Esquema del tema

Ajustes de posición

Supongamos que utilizamos dos variables para definir el eje X y una propiedad **aes**. ¿Cuál es el resultado?

```
> ggplot(mtcars)+geom_bar(aes(x=as.factor(cyl),
+   fill=as.factor(gear)))
```

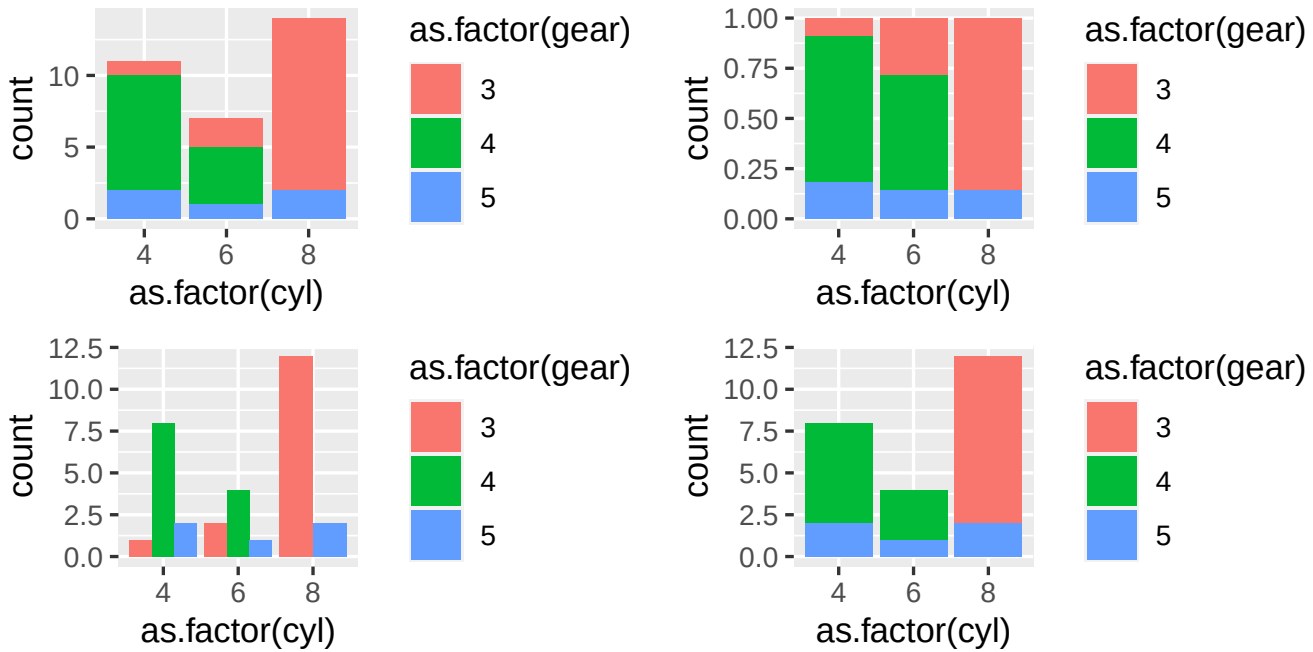


El resultado son unas barras apiladas, opcion por defecto.

Los ajustes de posición permiten modificar ciertas opciones.

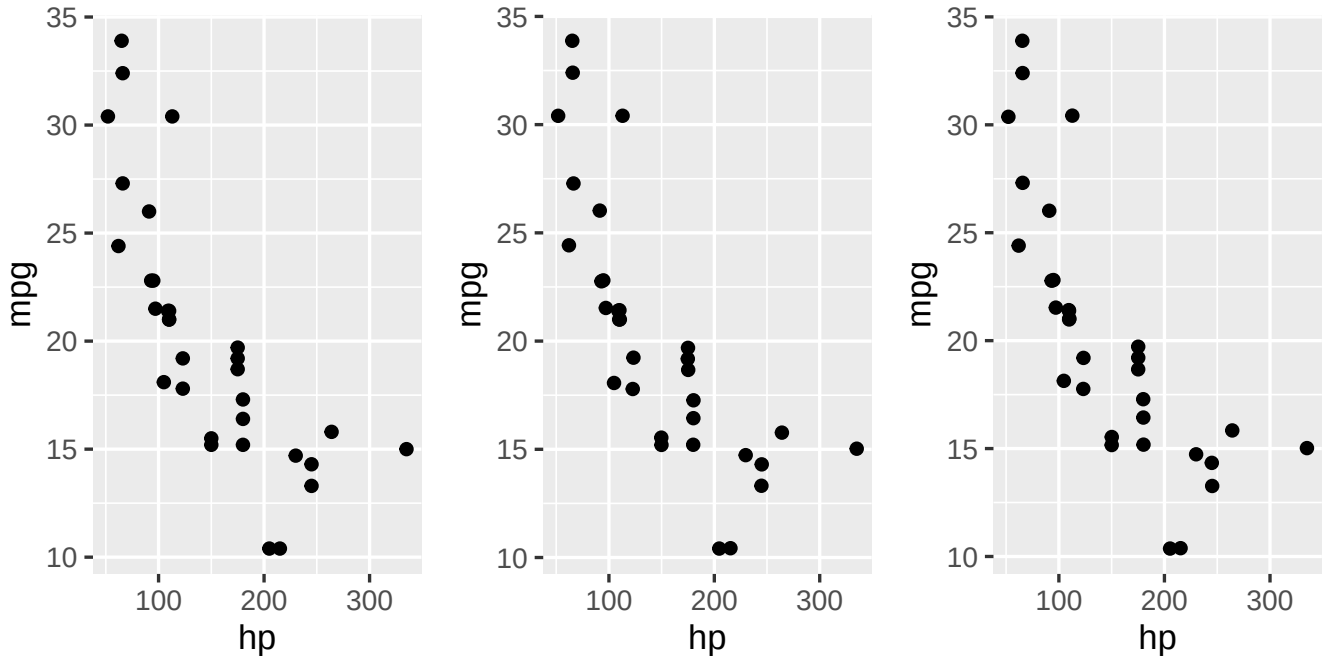
Modificación de la posición (geom_bar())

```
> library(gridExtra)
> p1<-ggplot(mtcars,aes(x=as.factor(cyl),fill=as.factor(gear)))
> grid.arrange(p1+geom_bar(position='stack'),
+ p1+geom_bar(position='fill'),
+ p1+geom_bar(position='dodge'),
+ p1+geom_bar(position='identity'),ncol=2)
```



Modificación de la posición (jitter)

```
> p1<-ggplot(mtcars,aes(x=hp,y=mpg))
> grid.arrange(p1+geom_point(),
+ p1+geom_point(position='jitter'),
+ p1+geom_jitter(),ncol=3)
```

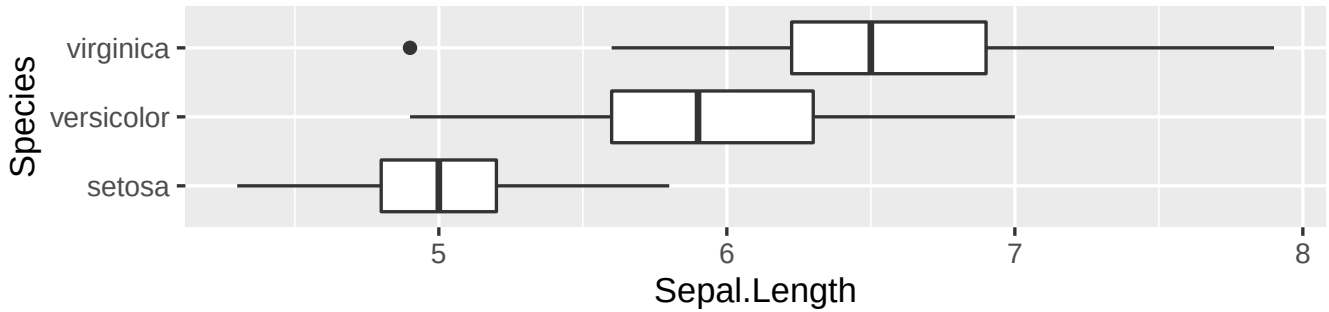


Sistemas de coordenadas

ggplot presenta diferentes sistemas de coordenadas además de las tradicionales coordenadas cartesianas.

Coordenadas	Efecto
coord_flip()	Intercambia ejes X e Y
coord_quickmap()	Ajusta el aspecto para mapas
coord_polar()	Coordenadas polares

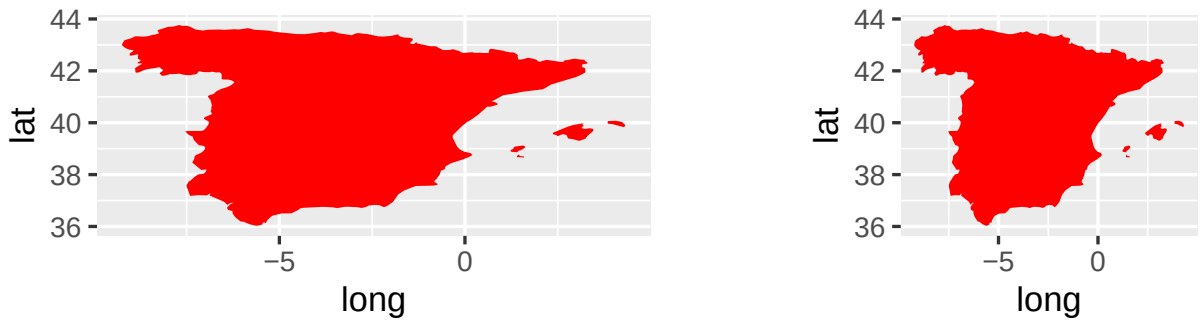
```
> p<-ggplot(iris,aes(x=Species,y=Sepal.Length))+geom_boxplot()
> p+coord_flip()
```



Modificación de las coordenadas

```
> p<-ggplot(mtcars,aes(x=as.factor(carb),fill=as.factor(carb)))
> grid.arrange(p+geom_bar(),
+ p+geom_bar()+coord_polar(),ncol=2)
```


```
> sp<-map_data('world','Spain')
> p<-ggplot(sp,aes(long,lat,group=group))
> grid.arrange(p+geom_polygon(fill='red'),
+ p+geom_polygon(fill='red')+coord_quickmap(),ncol=2)
```



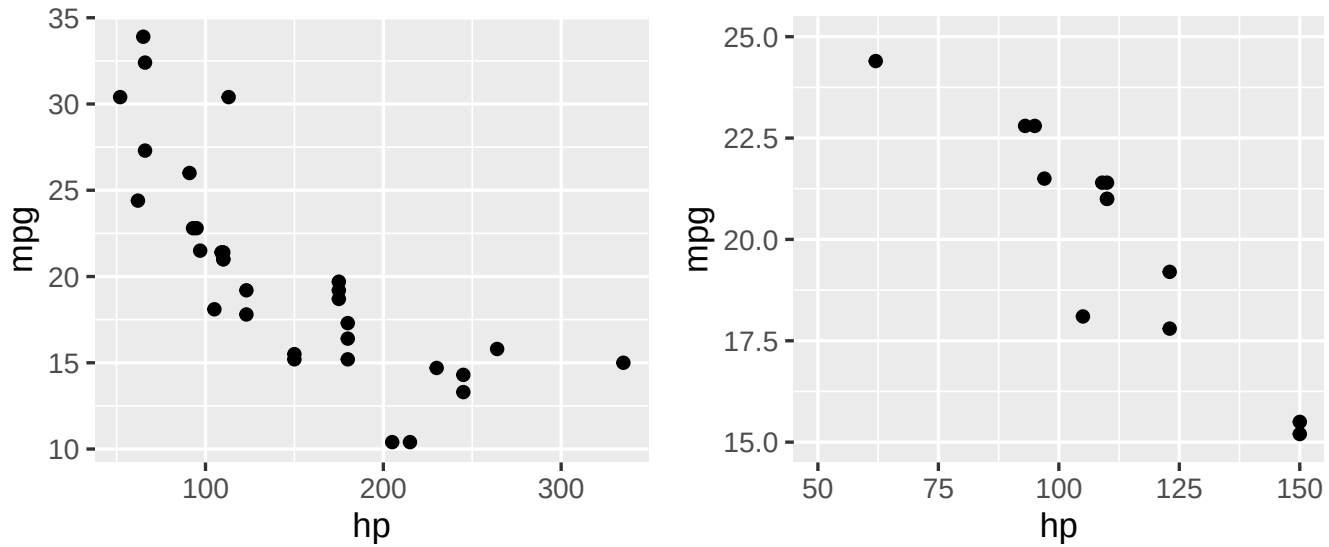
Modificación de las coordenadas (2)

Cada función que define las coordenadas tiene sus propias opciones.

```
coord_cartesian(xlim = NULL, ylim = NULL, expand = TRUE)
coord_polar(theta = "x", start = 0, direction = 1)
```

Por ejemplo para hacer un zoom

```
> p1<-ggplot(mtcars,aes(x=hp,y=mpg))
> grid.arrange(p1+geom_point(),p1+geom_point()+
+ coord_cartesian(xlim=c(50,150),ylim=c(15,25)),ncol=2)
```



Resumen

Hasta el momento hemos visto como construir un gráfico ggplot a partir de la adición de sucesivas capas

Capas en la gramática de los gráficos

```
ggplot(data = <DATA>) +
 <GEOM_FUNCTION>(
  mapping = aes(<MAPPINGS>),
  stat = <STAT>,
  position = <POSITION>
 ) +
 <COORDINATE_FUNCTION> +
 <FACET_FUNCTION>
```