UNIVERSIDAD DE EL SALVADOR

FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE

DEPARTAMENTO DE MATEMÁTICA

Licenciatura en Estadística

Control Estadístico del Paquete R

INTRODUCCIÓN AL ANÁLISIS GRÁFICOS CON GGPLOT2

Alumna:
Erika Beatríz Guillén Pineda

Fecha de elaboración
Santa Ana - 28 de noviembre de 2015

1. Introducción

Los gráficos son una forma muy útil de estudiar los datos antes de analizarlos. Si no inspeccionamos los datos mediante un gráfico, realmente no sabremos cómo interpretar los resultados.Al realizar un gráfico en R éste es enviado a un dispositivo gráfico, que no es más que una ventana gráfica o un archivo.Por otro lado existen diferentes sistemas gráficos que podemos utilizar: **base, lattice o ggplot2** y generalmente no podremos mezclar.

2. Gráficos en R con ggplot2

2.1. El paquete ggplot2

Tiene las siguientes características

- Con el R es posible obtener el mismo resultado usando diferentes **caminos**
- El paquete ggplot es uno de los entornos gráficos del R
- Permite elaborar un gráfico a partir de un proceso de acumulación de **capas o layers**
- Tiene un cierto nivel de complejidad pero se obtienen resultados muy profesionales.

2.2. Capas o Layers en ggplot2

El esquema básico de la gramática de ggplot es: ggplot(data.frame, aes(x = variable)) + geom$_{f}orma()$
$Ungráficoenggplot2puedetenervariascapas, ensuconjunto, lascapasformanelgráficoalcombinar$:

Un data frame y las variables a ser graficadas

Una o varias capas indicando, entre otros:

El tipo de objeto a graficar o **geom** (barra, línea, punto, etc.)

Las transformaciones estadísticas a los datos

La posición de los objetos en el gráfico

Una escala para cada variable a ser graficada

Un sistema de coordenadas

La especificación de **facetas** del gráfico

2.3. Uso Básico

Para definir un gráfico es necesario siempre empezar por la instrucción ggplot. Por ejemplo podemos utilizar la base de datos cars y representar la distancia necesaria de frenado en función de la velocidad.

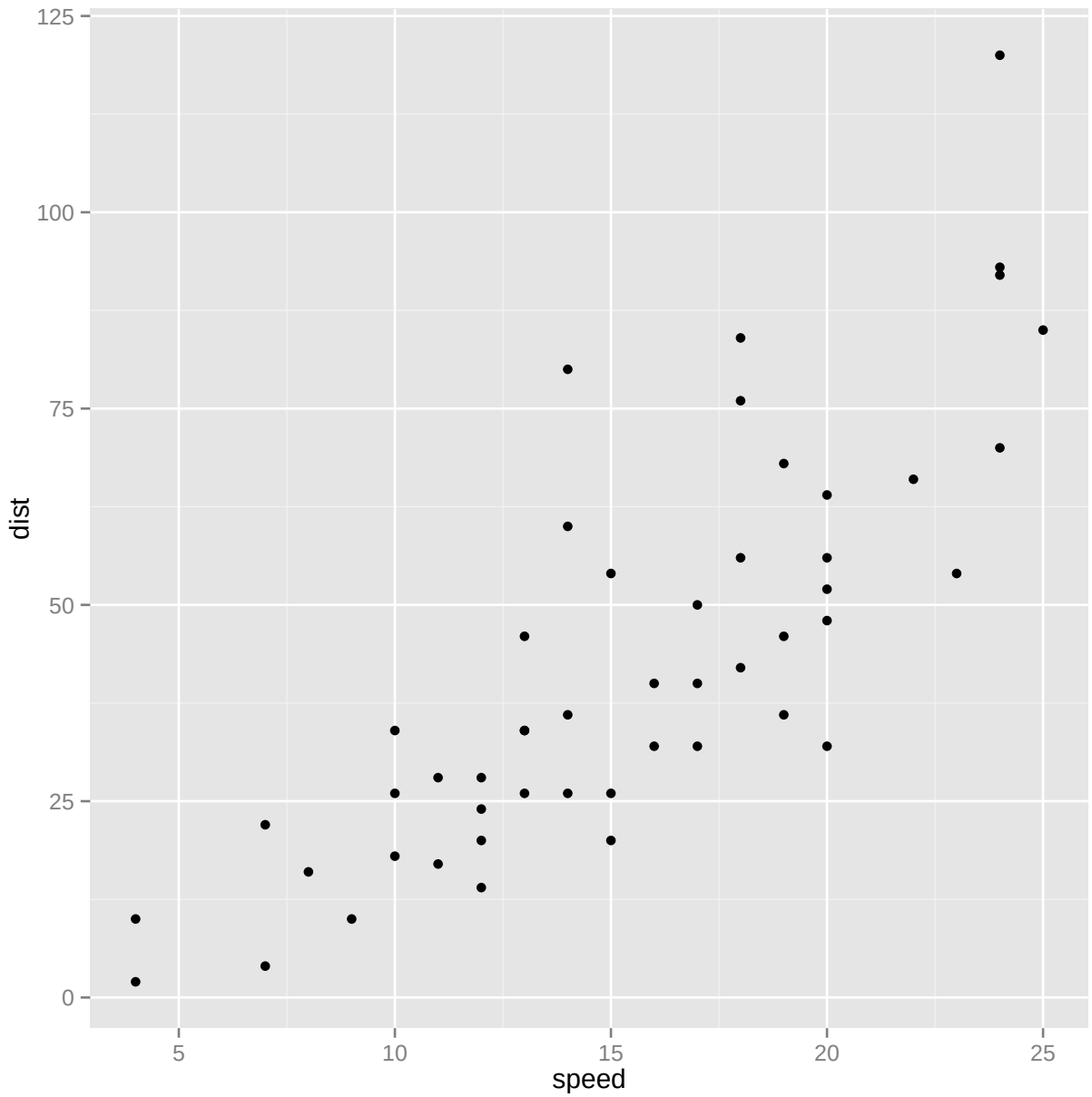
Ejenplo 1:

Activar el Directorio de Trabajo

```
getwd()

## [1] "C:/Users/User/Documents/GGPLOT2_TAREA"

setwd("C:/Users/User/Documents/GGPLOT2_TAREA")
```

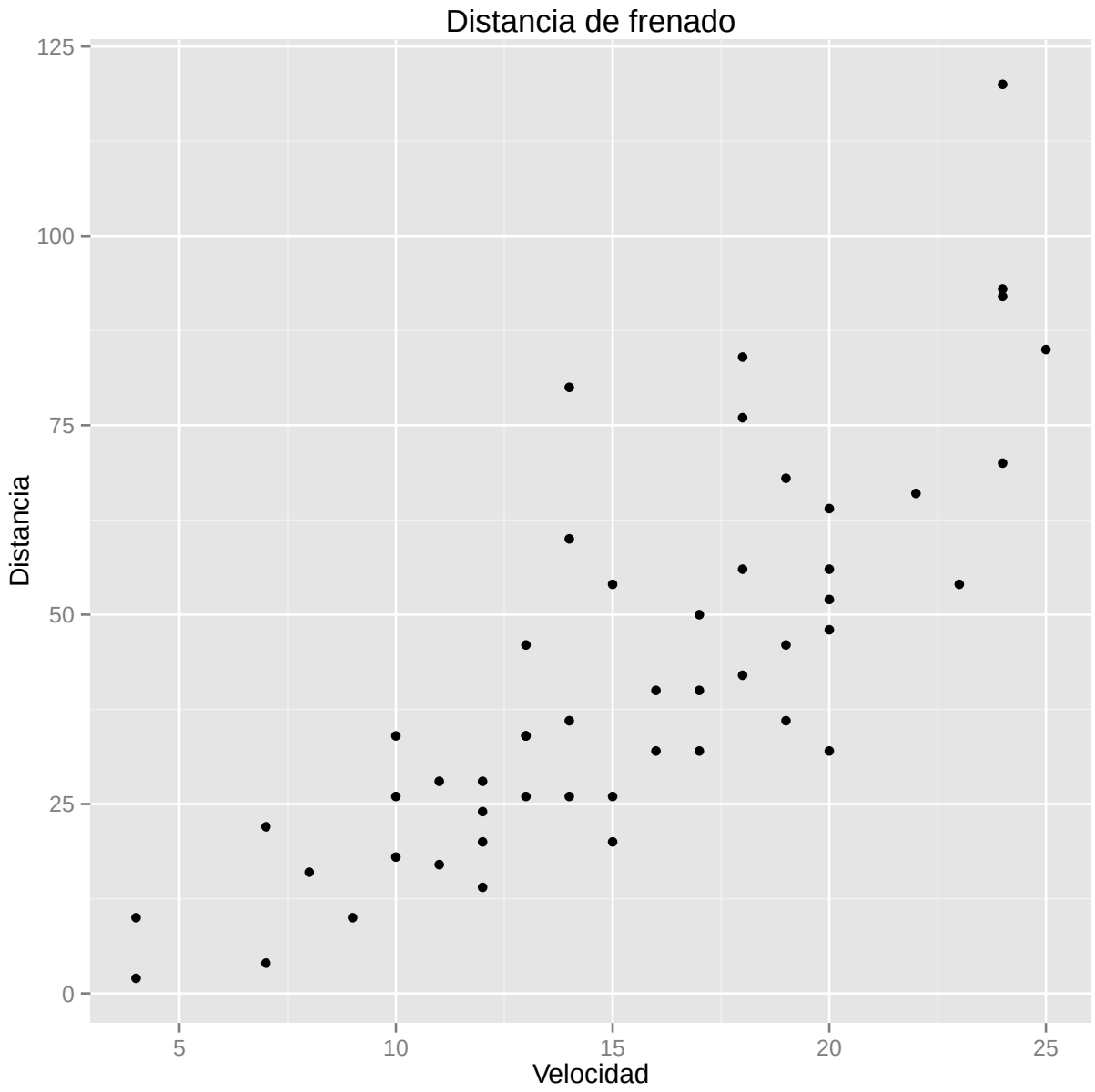
Activar la Librería

```
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.2.2

ggplot(cars,aes(x=speed,y=dist)) + geom_point()
```



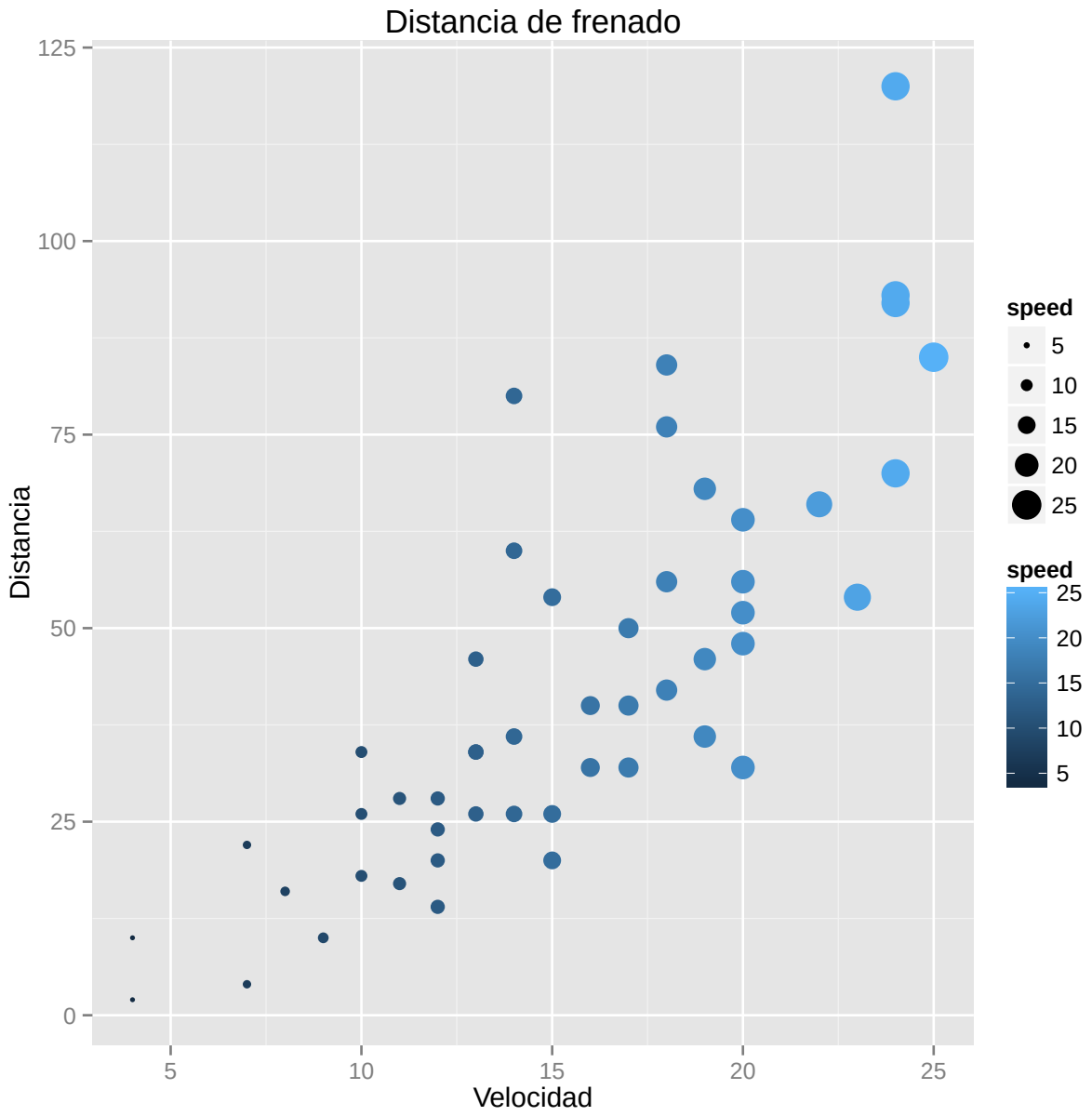
En esta instrucción, se define la base de datos que utilizamos y la 'estética' básica de la gráfica, es decir qué variables vamos a representar. Una vez definida, se añade una 'geometria', en este caso los puntos que indican la distancia necesaria para frenar en cada caso respecto a la velocidad del coche. De manera muy simple podemos añadir opciones a la gráfica. Por ejemplo, podemos etiquetar los ejes

```
# ETIQUETANDO LOS EJES DE LA GR\'AFICA
ggplot(cars,aes(x=speed,y=dist)) + geom_point() +
  labs(title="Distancia de frenado",
  x="Velocidad",  y="Distancia")
```



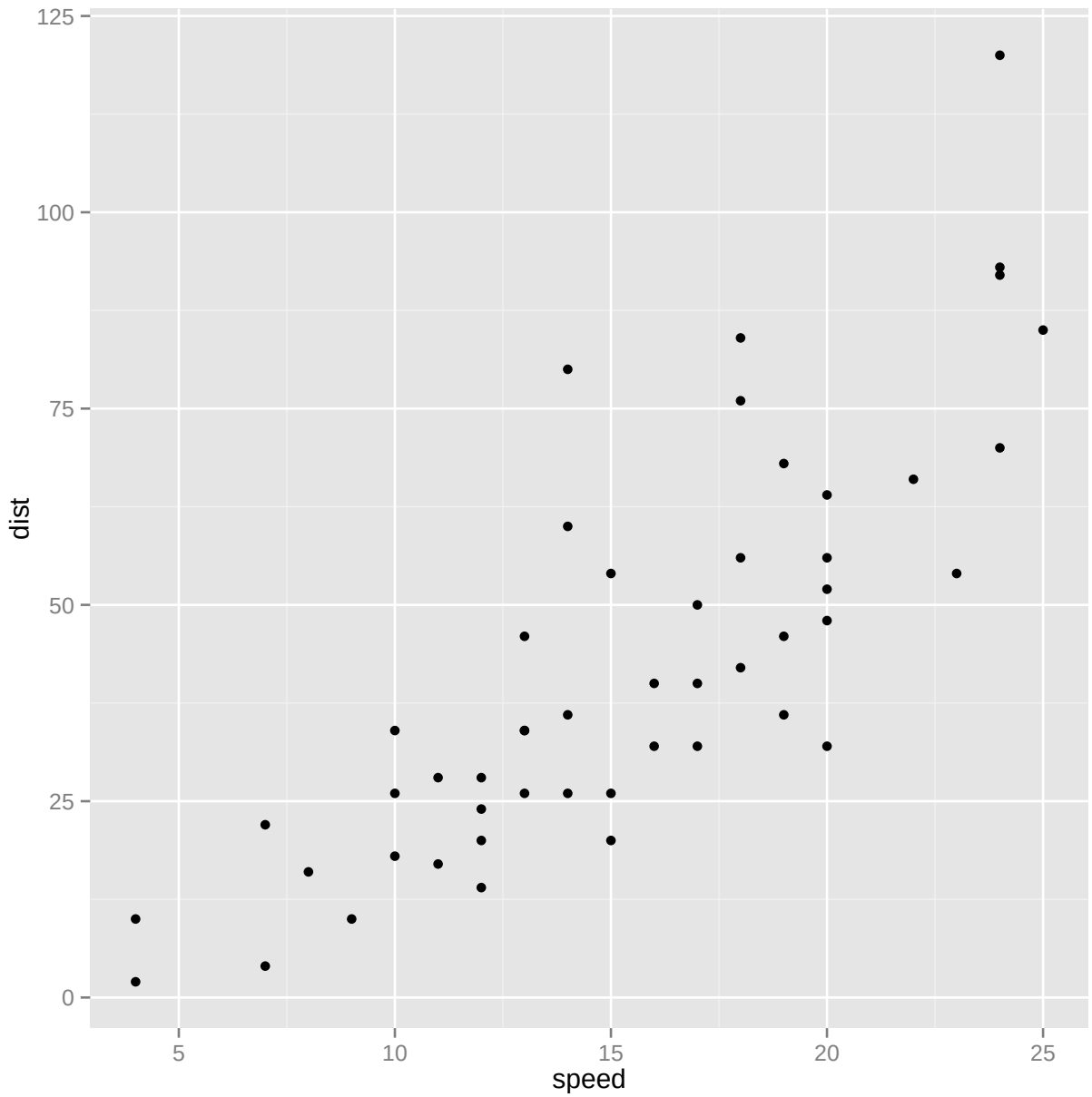
La geometrÍa admite su propia esTética. Por ejemplo, podemos hacer que los puntos aumenten de tamaño en función de la velocidad y a la vez que el color sea distinto

```
ggplot(cars,aes(x=speed,y=dist)) + geom_point(aes(size=speed,color=speed)) +
labs(title="Distancia de frenado",x="Velocidad", y="Distancia")
```



Los gráficos en ggplot2 pueden asignarse a objetos para ser modificados posteriormente. Por ejemplo

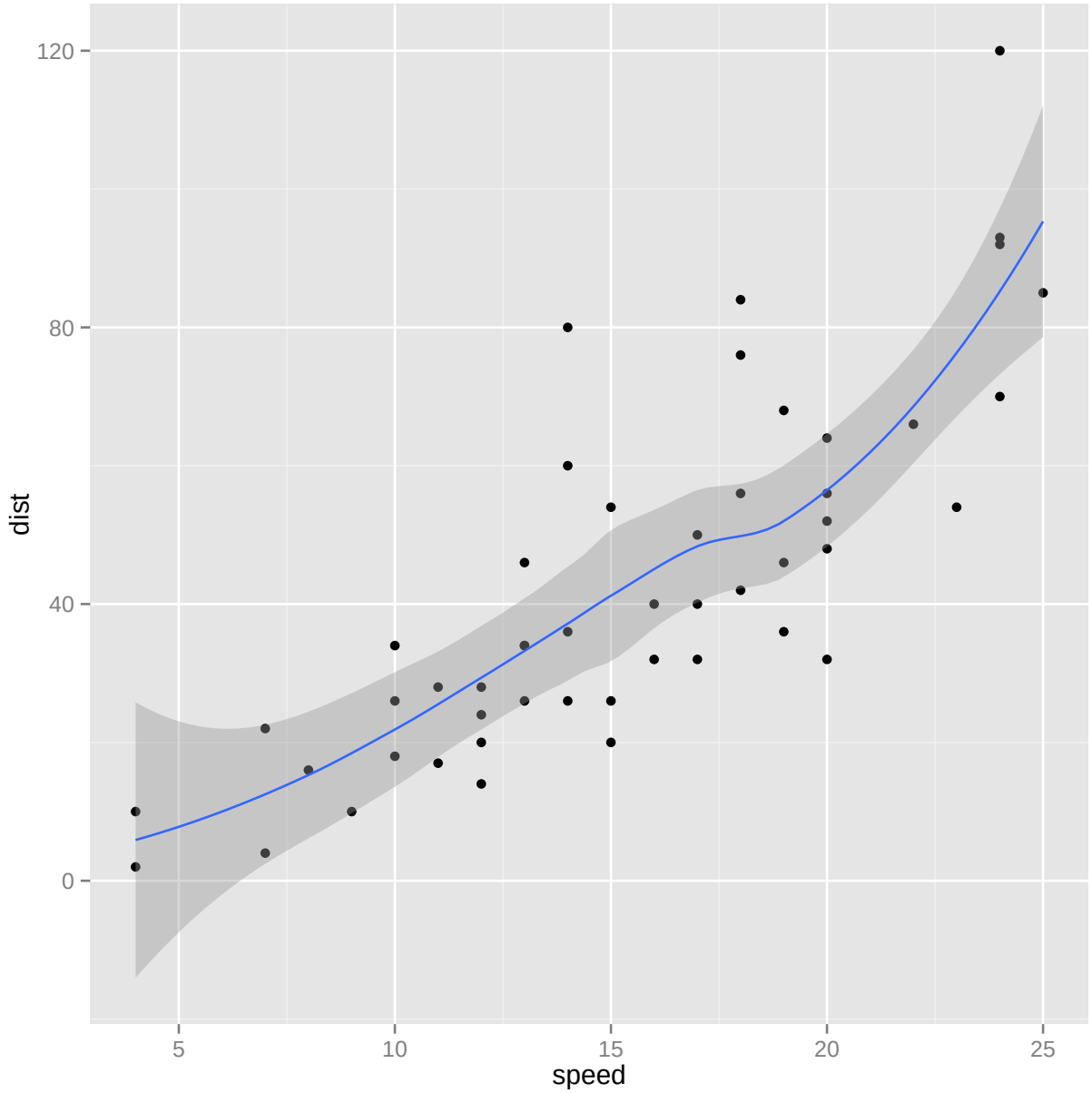
```
p1 <- ggplot(cars,aes(x=speed,y=dist))
p1 + geom_point()
```



Ahora podemos obtener una gráfica distinta aprovechando que tenemos el objeto p1

```
p1 + geom_point() + geom_smooth()
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using loess.
Use 'method = x' to change the smoothing method.
```

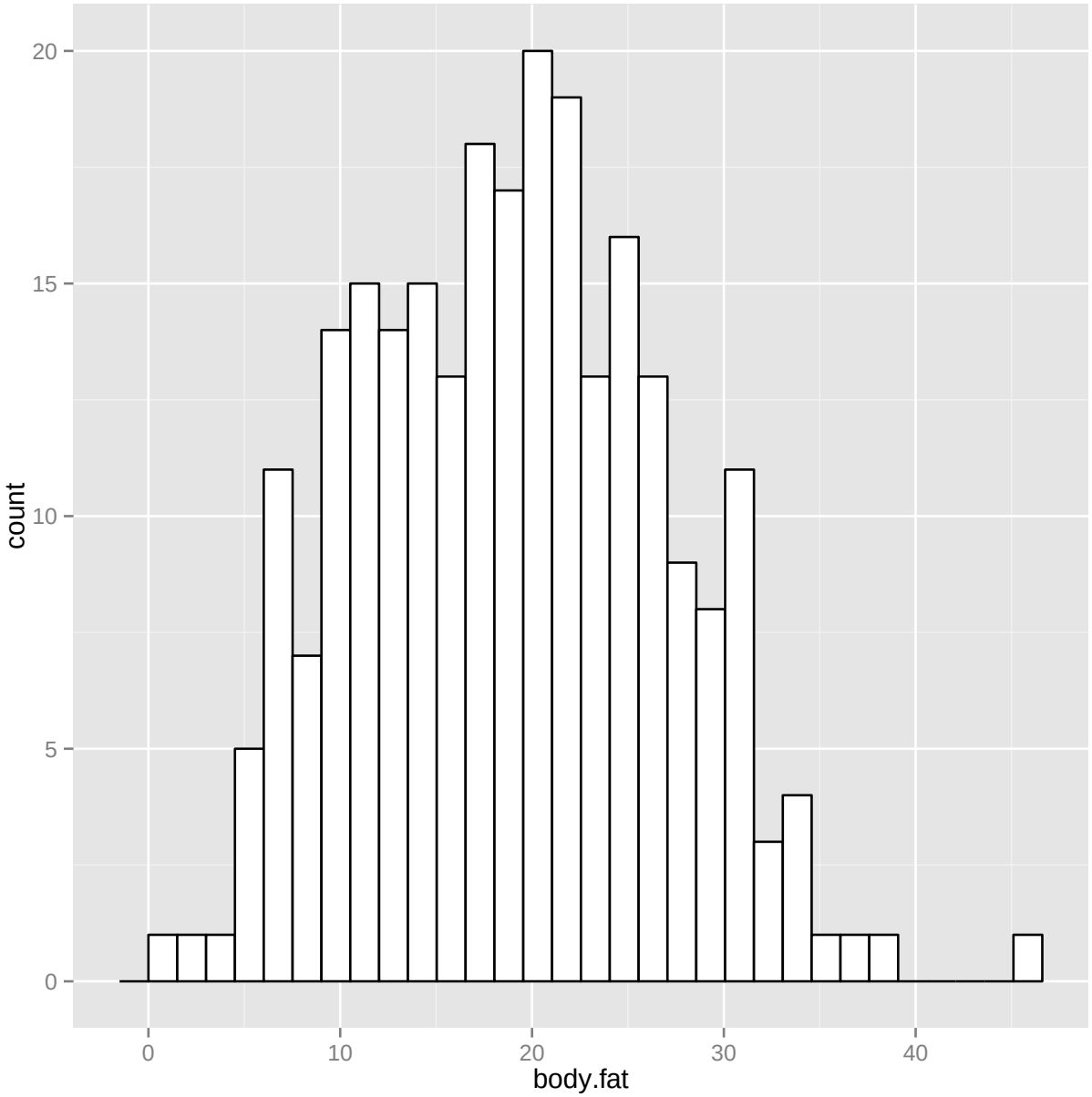


2.4. Histogramas

Un histograma permite representar la distribución de valores de una variable contúnua observados en una muestra. Por ejemplo, en los datos fat disponible en la libreria UsingR se recogen datos de distintas variables fisiológicas de una muestra de personas. Podemos obtener una gráfica de la distribución de grasa corporal mediante:

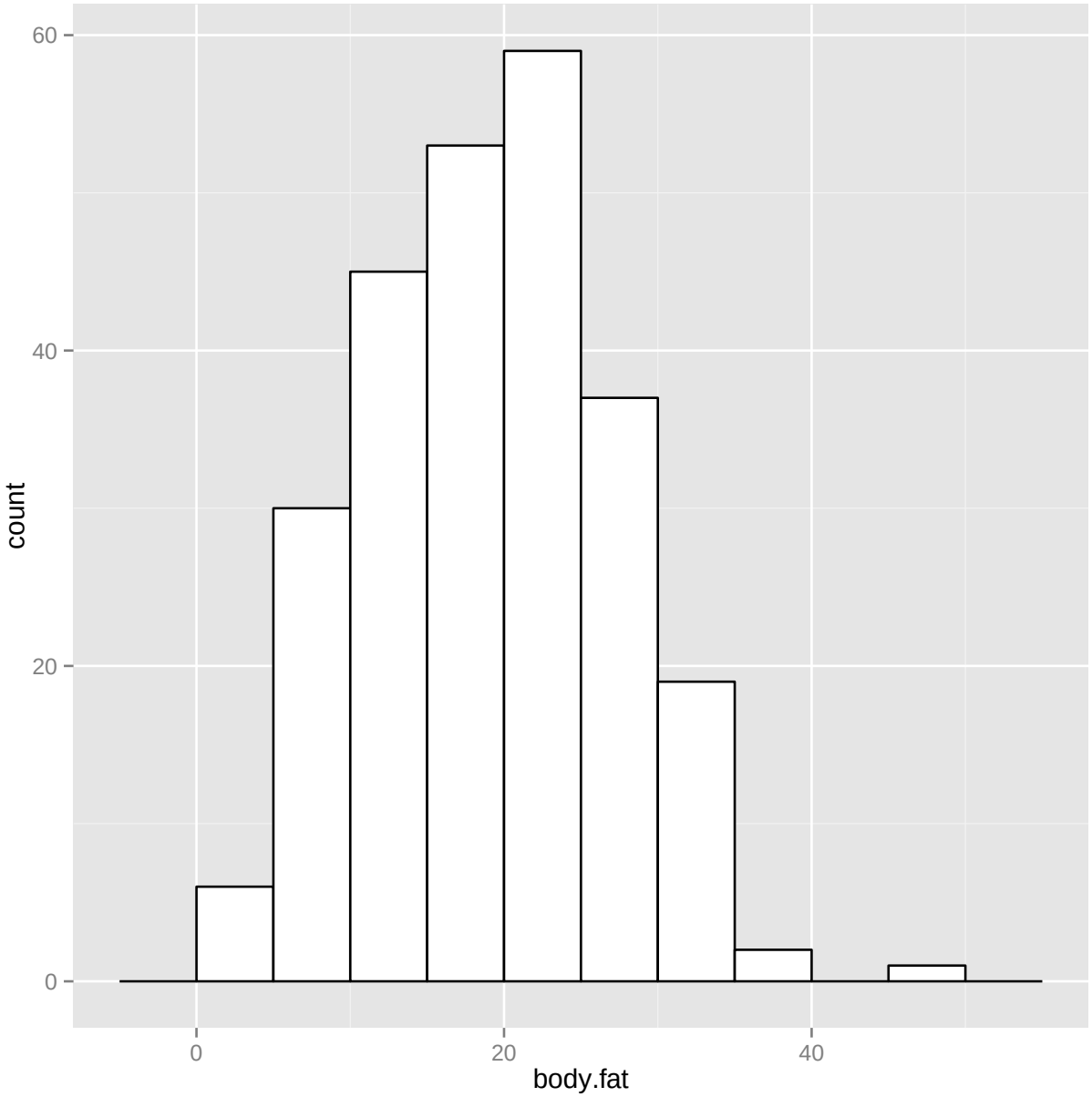
```
install.packages("UsingR")

## Installing package into 'C:/Users/User/Documents/R/win-library/3.2'
## (as 'lib' is unspecified)

## Error in contrib.url(repos, "source"): trying to use CRAN without setting a
mirror
```

```
library(UsingR)

## Warning: package 'UsingR' was built under R version 3.2.2

## Loading required package: MASS
## Loading required package: HistData

## Warning: package 'HistData' was built under R version 3.2.2

## Loading required package: Hmisc

## Warning: package 'Hmisc' was built under R version 3.2.2

## Loading required package: grid
## Loading required package: lattice

## Warning: package 'lattice' was built under R version 3.2.2

## Loading required package: survival
## Loading required package: Formula

## Warning: package 'Formula' was built under R version 3.2.2

##
## Attaching package: 'Hmisc'
##
## The following objects are masked from 'package:base':
##
##    format.pval, round.POSIXt, trunc.POSIXt, units
##
##
## Attaching package: 'UsingR'
##
## The following object is masked from 'package:survival':
##
##    cancer
##
## The following object is masked from 'package:ggplot2':
##
##    movies

ggplot(fat,aes(x=body.fat)) + geom_histogram(fill="white",color="black")

## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



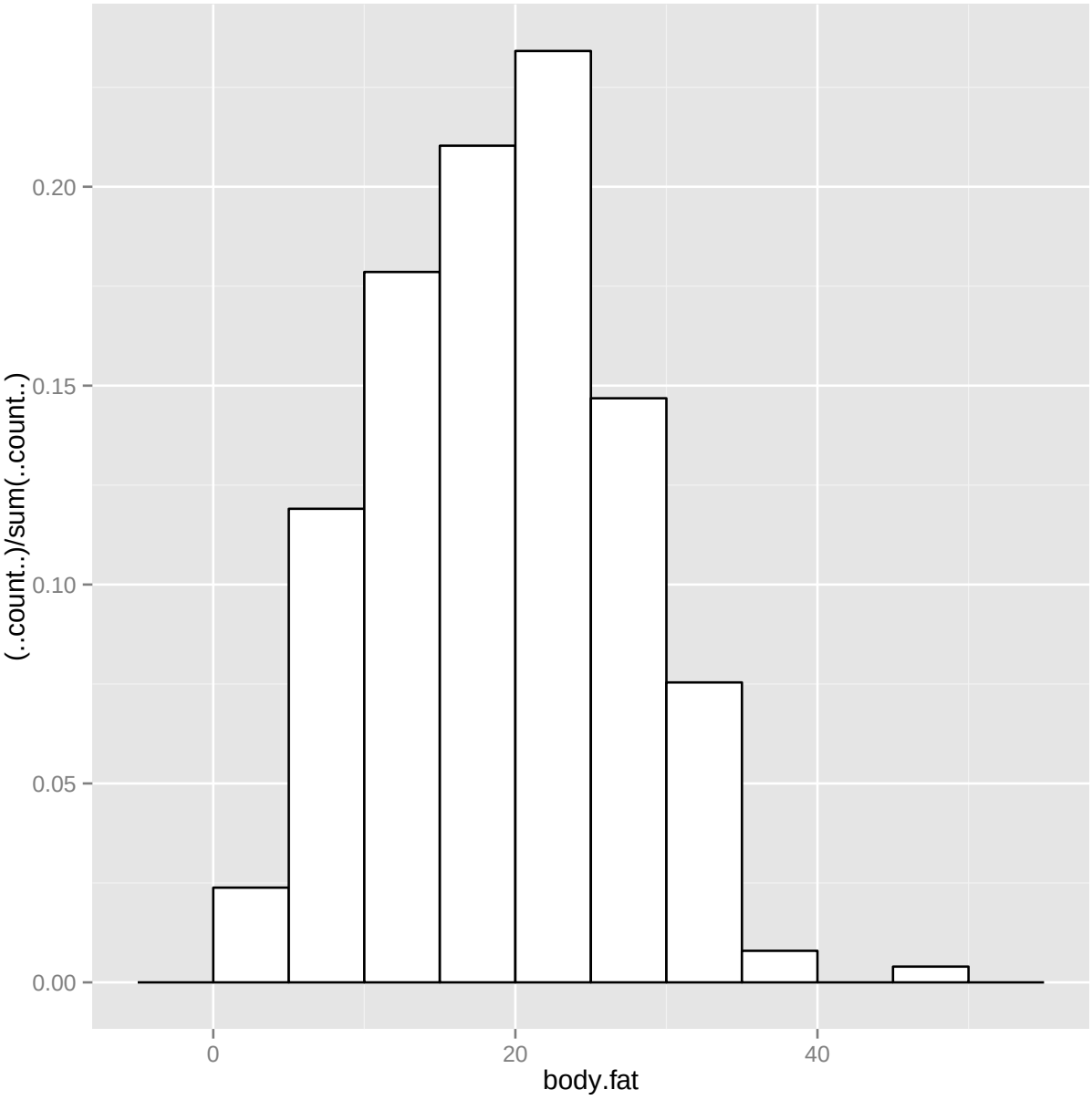
Podemos observar que el resultado es un tanto confuso. Podemos mejorar la gráfica cambiando la anchura de los intervalos

```
ggplot(fat,aes(x=body.fat)) + geom_histogram(fill="white",
                        color="black",binwidth=5)
```



Por defecto, el histograma representa la frecuencia absoluta. Para obtener el porcentage debemos especificarlo en la estética

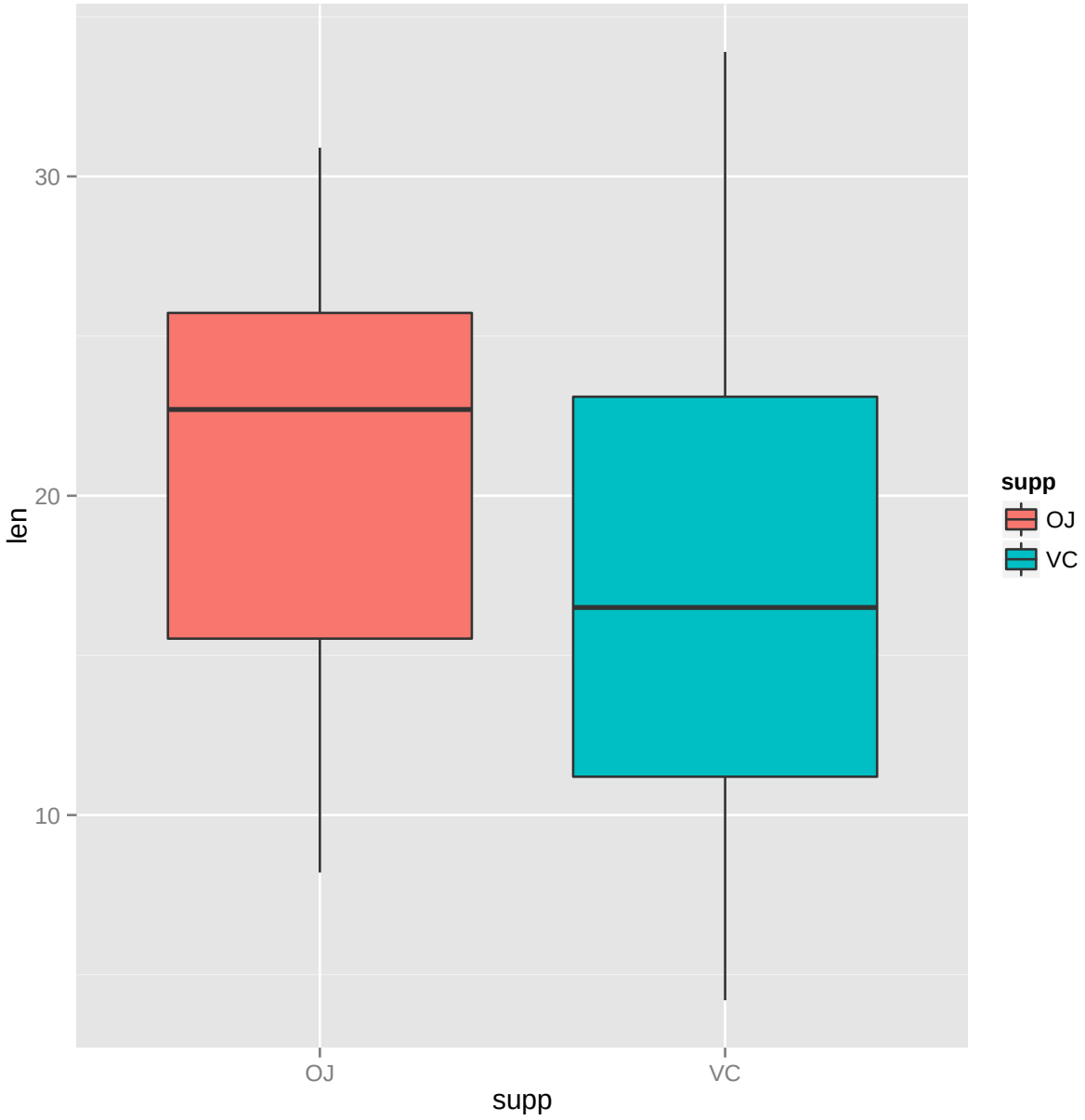
```
ggplot(fat,aes(x=body.fat)) + geom_histogram(aes(y = (..count..)/sum(..count..)),
                        fill="white",color="black",binwidth=5)
```



2.5. Boxplot

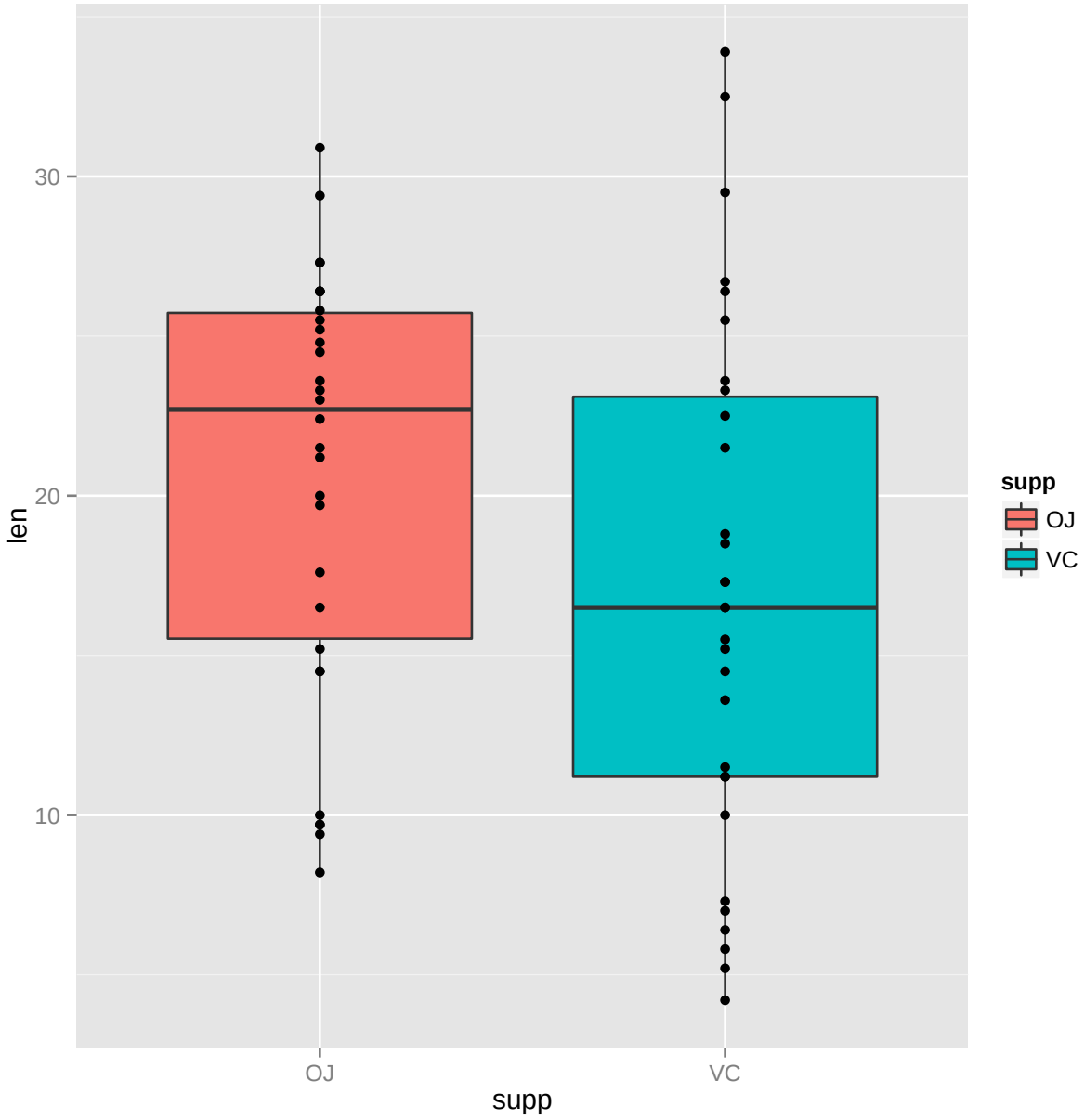
El gráfico boxplot permite resumir de manera eficiente distintos índices estadísticos. Por ejemplo:

```
p <- ggplot(ToothGrowth,aes(supp,len))
p + geom_boxplot(aes(fill=supp))
```



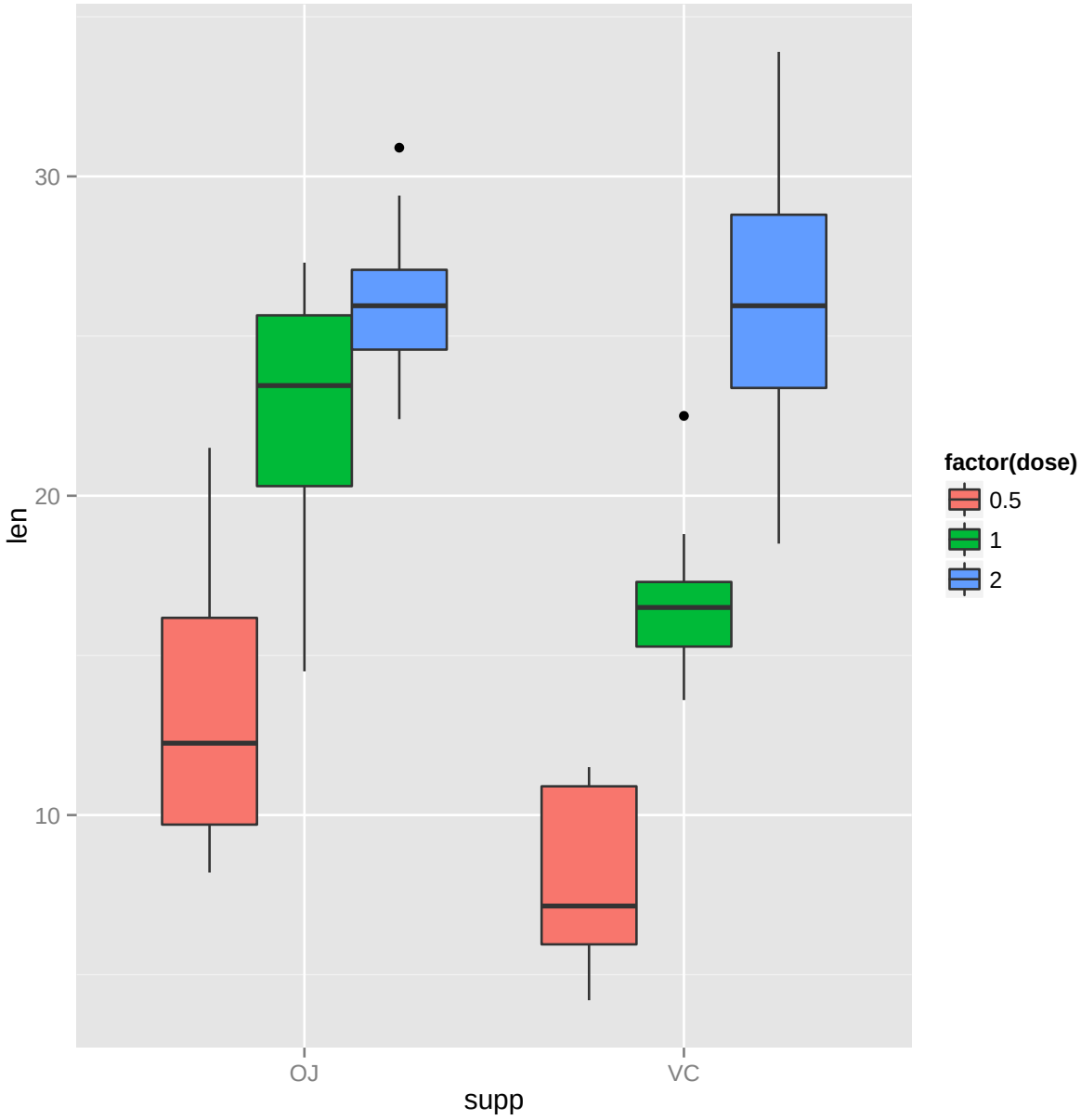
Los valores correspondientes al límite superior de la caja indican el percentil 75, el límite inferior corresponde al percentil 25 y la línia horizontoal a la mediana o percentil 50. Las líneas verticales indican el valor máximo y mínimo obervado. Podemos superponer los puntos correspondientes a las obervaciones añadiendo

```
p + geom_boxplot(aes(fill=supp))+ geom_point()
```



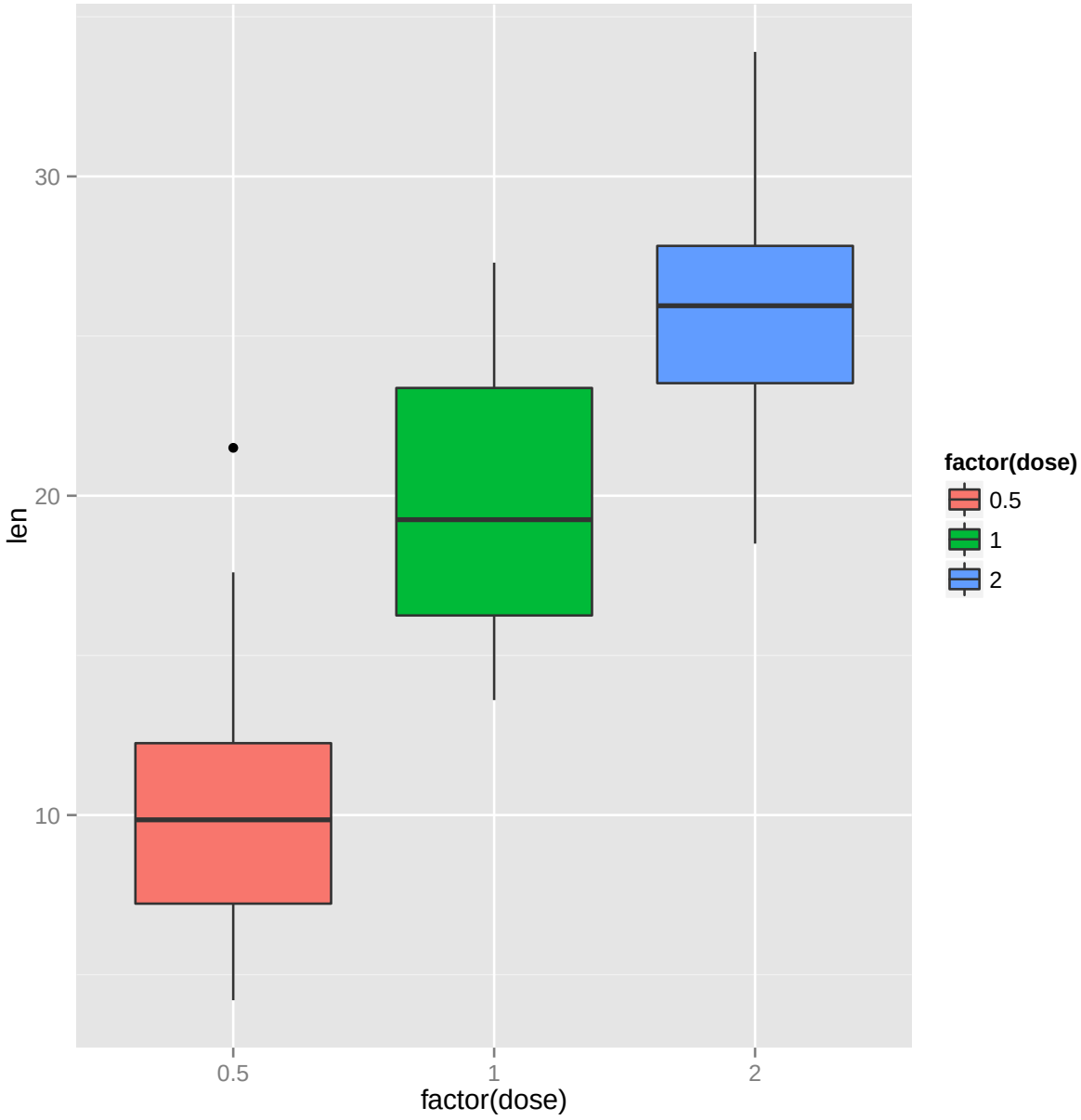
En muchos casos es interesante considerar distintos factores. En los datos de crecimeinto de los dientes podemos considerar el suplemento alimenticio y la dosis. Para ello, basta con a nadir una instrucción a la estética del boxplot

```
p + geom_boxplot(aes(fill=factor(dose)))
```



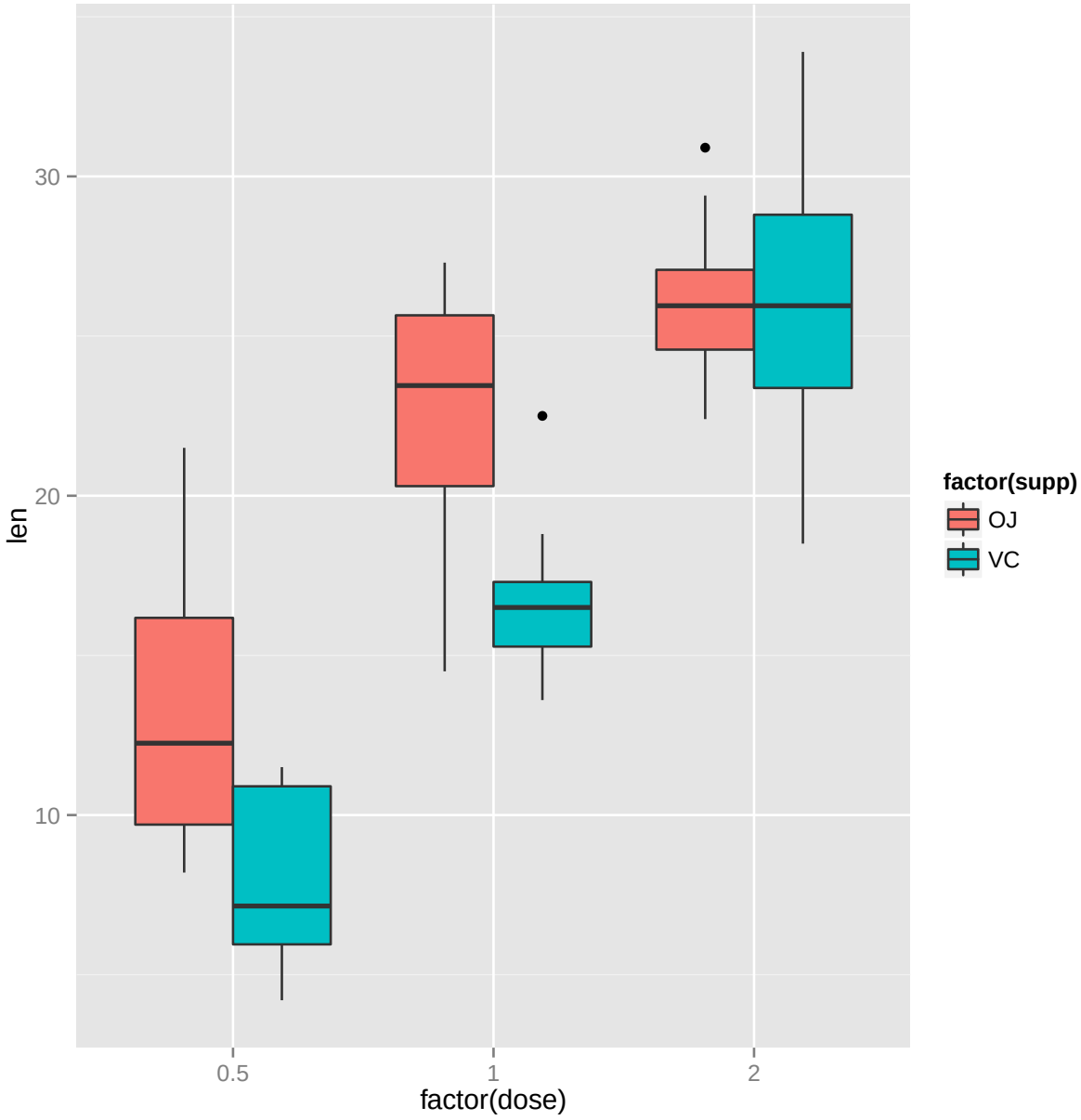
Evidentemente, podemos cambiar el orden de los factores si consideramos que es más interesante. Por ejemplo:

```
p <- ggplot(ToothGrowth,aes(factor(dose),len))
p + geom_boxplot(aes(fill=factor(dose)))
```



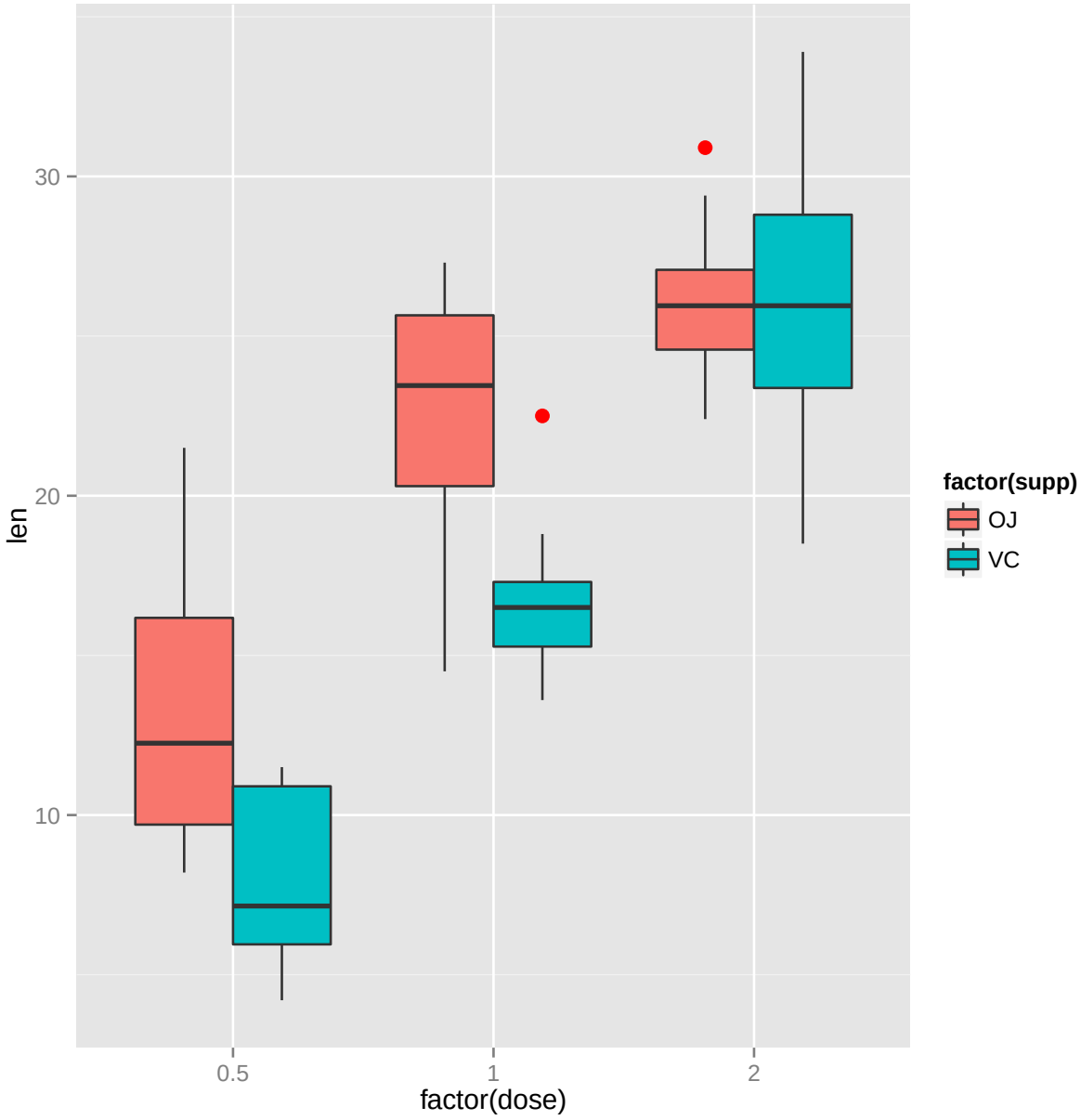
Ahora podemos subdividir por el suplemento alimenticio

```
p + geom_boxplot(aes(fill=factor(supp)))
```



Podemos resaltar los outliers o datos extremos

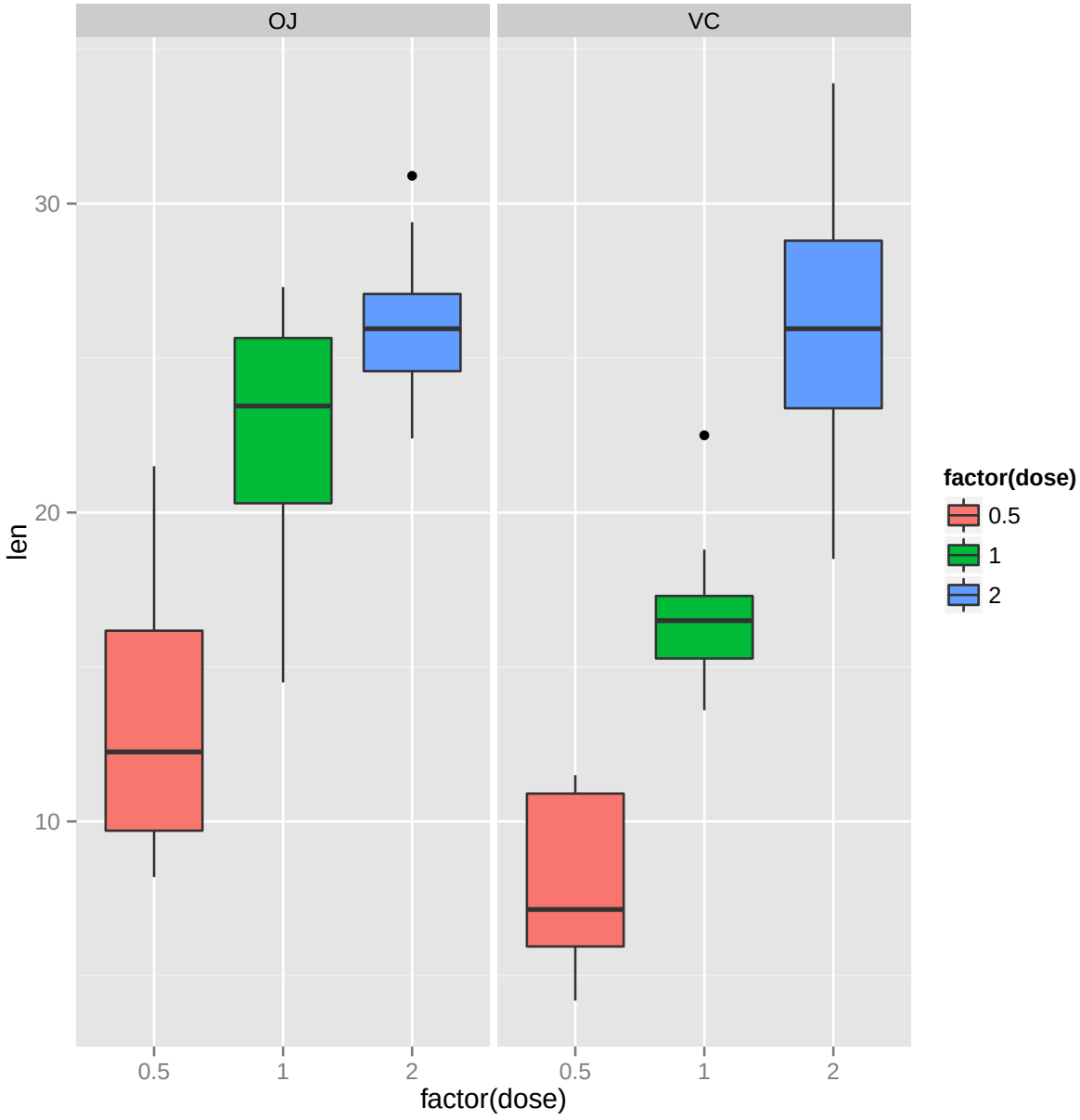
```
p + geom_boxplot(aes(fill=factor(supp)), outlier.colour = "red",
                 outlier.size = 3)
```



2.6. Gráficos por Subgrupos

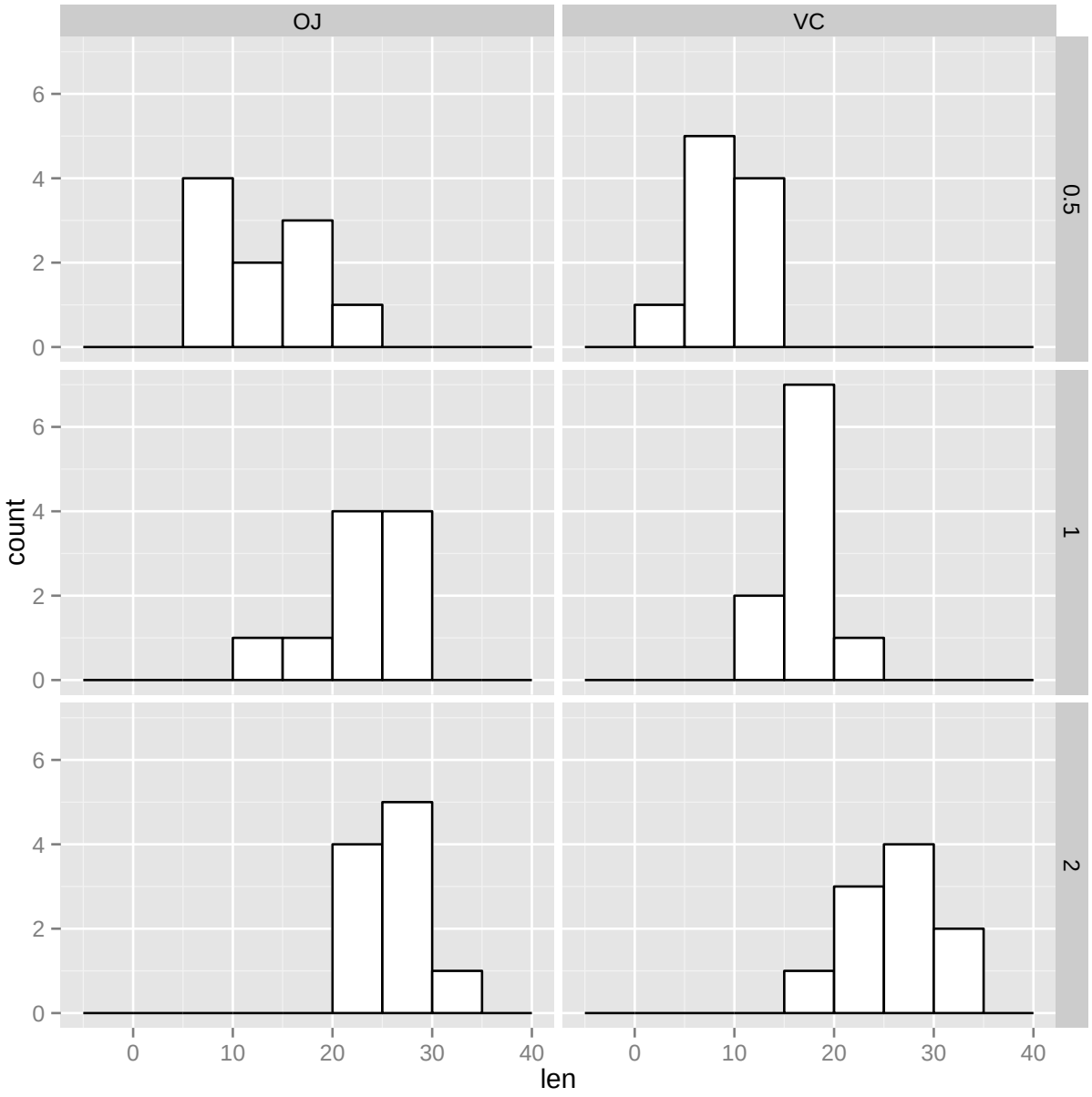
En muchos casos será necesario comparar resultados en distintos grupos. Algunos procedimientos, p.e. geom boxplot ya incluyen esta posibilidad para dos grupos, pero podría ser necesario considerar más variables y subdividir las gráficas. El paquete ggplot2 permite hacer esto mediante la instruccions facets grid y sus variantes. Veamos como funciona con unos ejemplos. Para empezar, representaremos un boxplot separando las gráficas por suplemento alimenticio. Para ello, facet grid especifica qué variable aparecerá en los subgrupos de filas(en este caso ninguna) y qué variable definira las columnas (en este caso el suplemento alimenticio) en una matriz de gráficos donde se mostrará el gráfico requerido para cada combinación de fila y columna

```
library(grid)
ggplot(ToothGrowth,aes(factor(dose),len)) + geom_boxplot(aes(fill=factor(dose))) +
facet_grid(.~supp)
```



Para apreciar las posibilidades de esta opción, vamos a representar los histogramas de distribución de la longitud de dientes por dosis y suplemento alimenticio

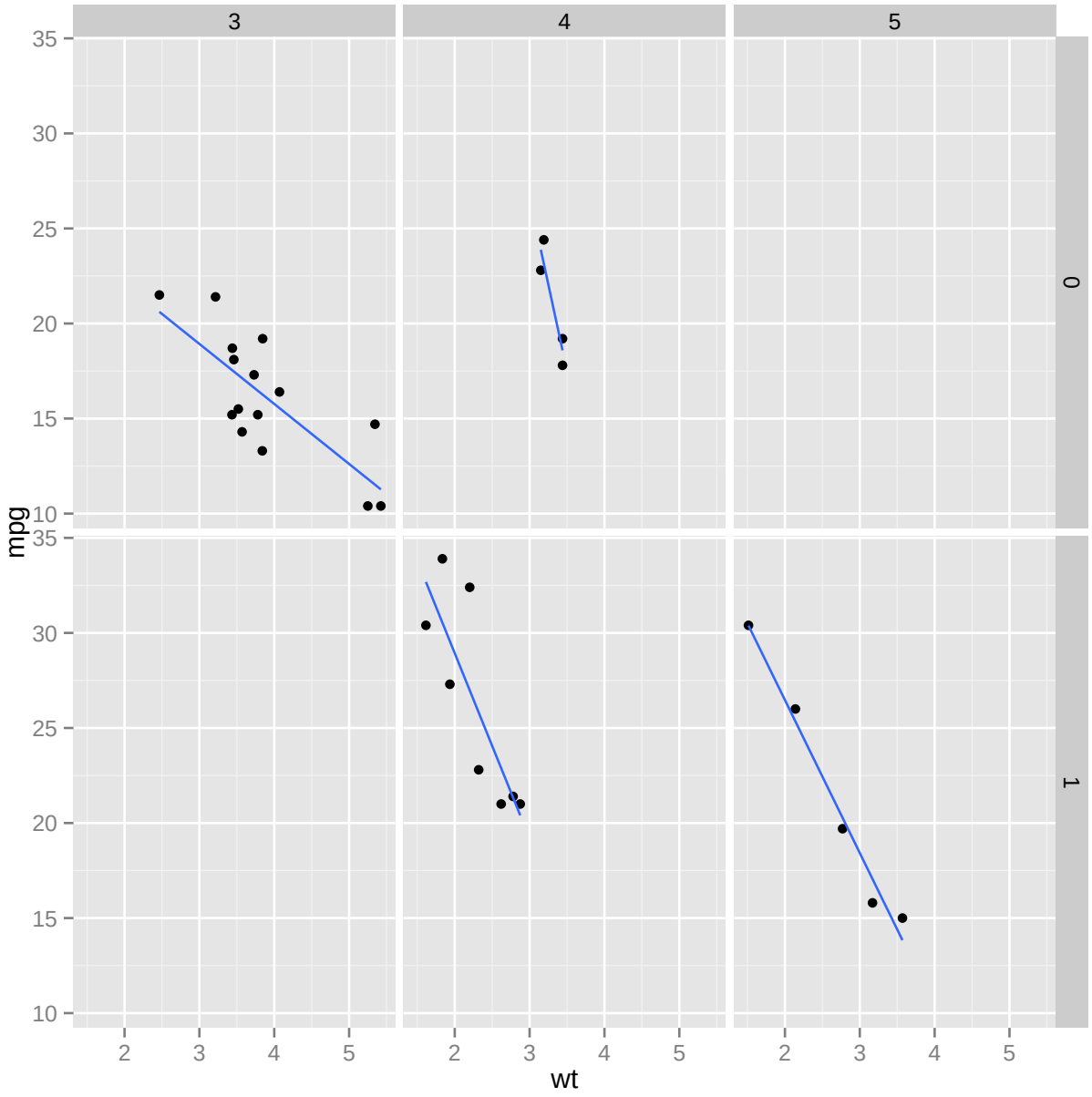
```
ggplot(ToothGrowth,aes(x=len)) + geom_histogram(fill="white",
      color="black",binwidth=5) + facet_grid(dose~supp)
```



La opción de facet grid puede utilizarse con cualquier tipo de gráficos

Evidentemente, podemos utilizar más opciones. Por ejemplo, podemos incluir la recta de regresión en las observaciones de cada subgrupo

```
p + geom_point() + facet_grid(am~gear) + geom_smooth(se=F,method="lm")
```



2.7. Transformación de las escalas de los ejes

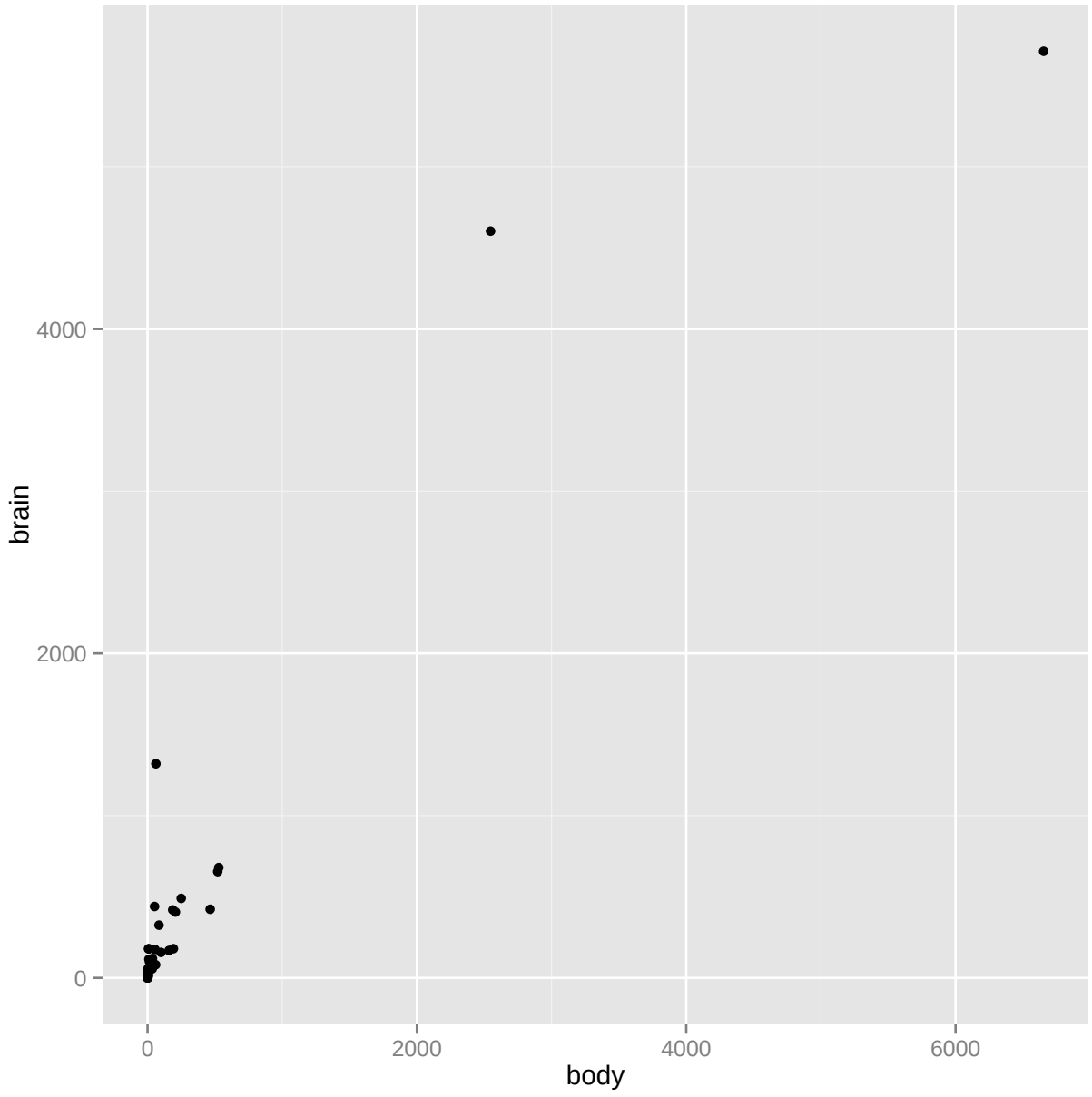
En algunos casos, será necesario transformas las escalas de los ejes para obtener unos resultados más claros. Veamos como puede hacerse. Consideremos los datos mammals que se encuentran en la libreria MASS:

```
library(MASS)
head(mammals)

##                    body brain
## Arctic fox        3.385  44.5
## Owl monkey        0.480  15.5
```

```
## Mountain beaver   1.350   8.1
## Cow             465.000 423.0
## Grey wolf        36.330 119.5
## Goat             27.660 115.0
```

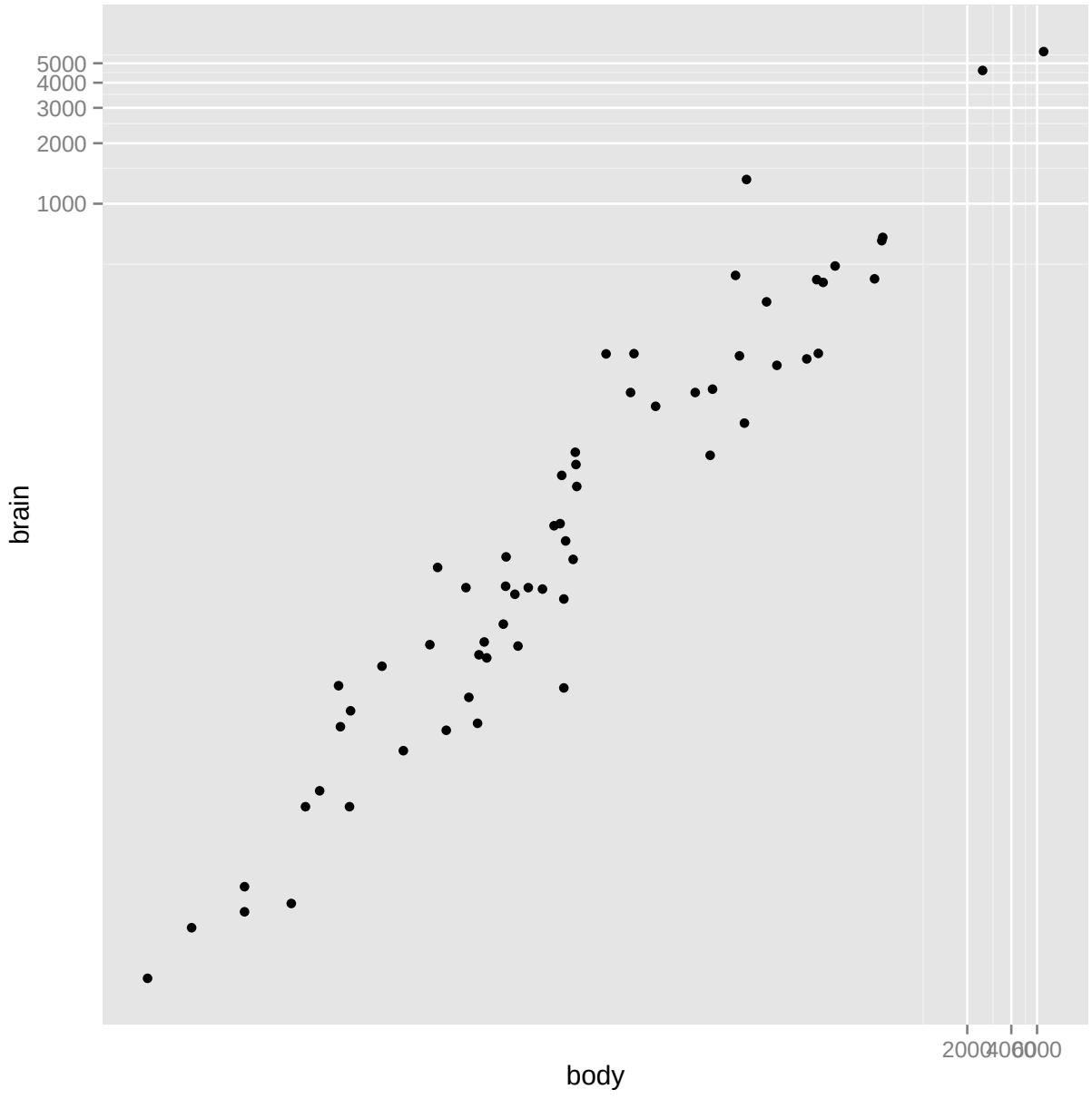
Si representamos el peso del cerebro respecto al peso del cuerpo, obtenemos

```
p <- ggplot(mammals,aes(x=body,y=brain))
p + geom_point()
```



Dado que existe una diferencia muy grande en el tama no de los animales, puede ser más interesante representar estos datos en escala logarítmica

```
p + geom_point()+coord_trans(x="log",y="log")
```



2.8. Gráfico de Barras

Los gráficos de barras son apropiados para representar las frecuencias absolutas de los valores de factores. Por ejemplo, la base de datos birthwt contiene información acerca del peso de recien nacidos en función de distintos características de la madre.

```
head(birthwt)

##    low age lwt race smoke ptl ht ui ftv  bwt
## 85   0  19 182    2     0   0  0  1   0 2523
```

```
## 86   0  33 155    3     0   0  0  0   3 2551
## 87   0  20 105    1     1   0  0  0   1 2557
## 88   0  21 108    1     1   0  0  1   2 2594
## 89   0  18 107    1     1   0  0  1   0 2600
## 91   0  21 124    3     0   0  0  0   0 2622
```

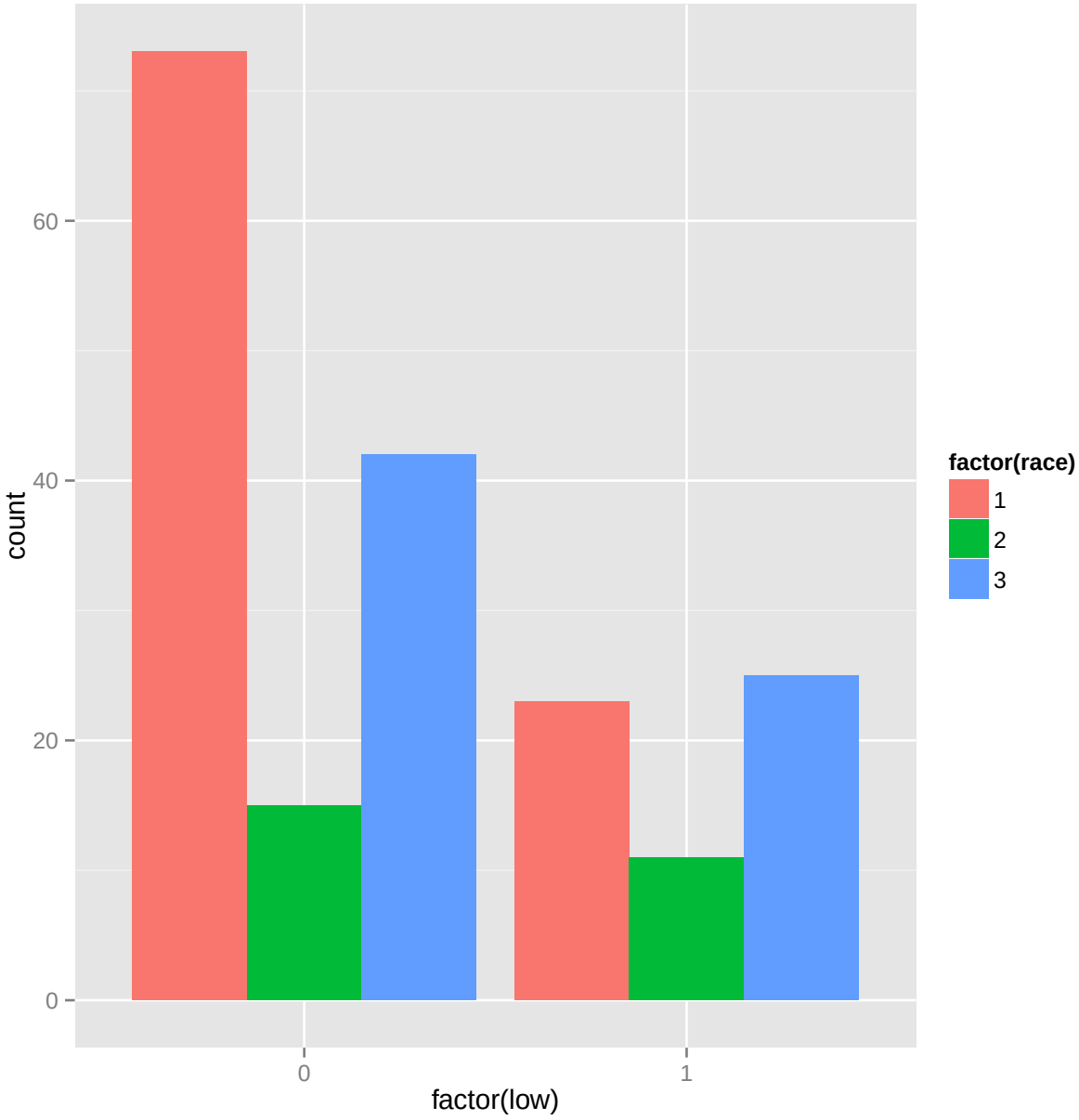
Podemos obtener una tabla de frecuencias que indique cuantos niños nacen con poco peso en relación, por ejemplo, a la raza de la madre:

```
with(birthwt, table(race,low))
```

```
##     low
## race  0  1
##    1 73 23
##    2 15 11
##    3 42 25
```

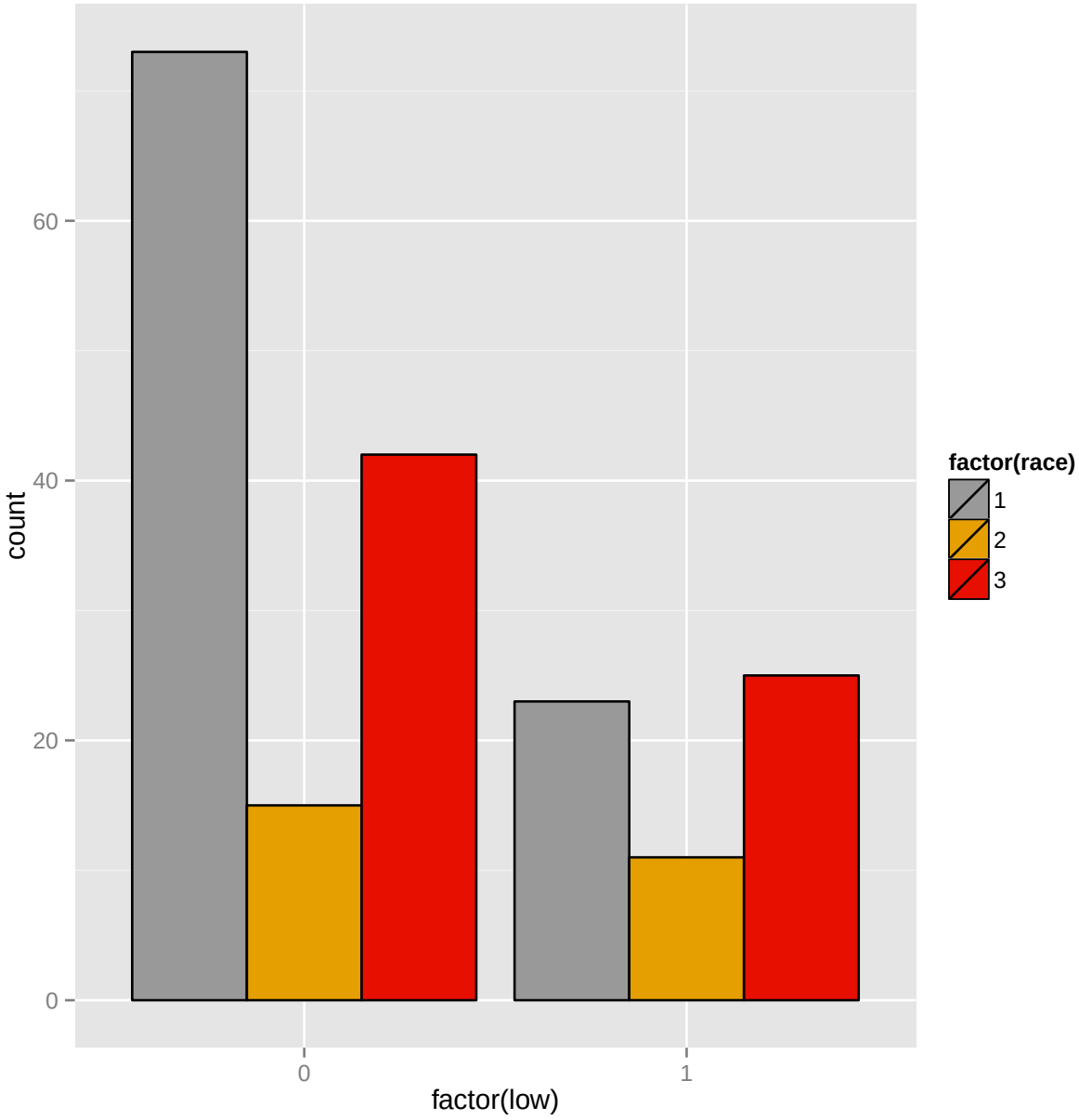
Podemos obtener la gráfica mediante el siguiente procedimiento

```
ggplot(birthwt,aes(x=factor(low),fill=factor(race)))+
  geom_bar(position=position_dodge())
```



Podemos cambiar el color de las barras haciendo

```
ggplot(birthwt,aes(x=factor(low),fill=factor(race)))+
geom_bar(position=position_dodge(),color="black") +
scale_fill_manual(values=c("#999999", "#E69F00", "#E70F00"))
```



En general, es más interesante representar los porcentajes. En las tablas podemos hacer, por ejemplo, el procentage de bajo peso por raza:

```
t <- with(birthwt, table(race,low))
tp<- round(prop.table(t,1),1)
tp
```

```
##      low
## race   0   1
##    1 0.8 0.2
##    2 0.6 0.4
##    3 0.6 0.4
```

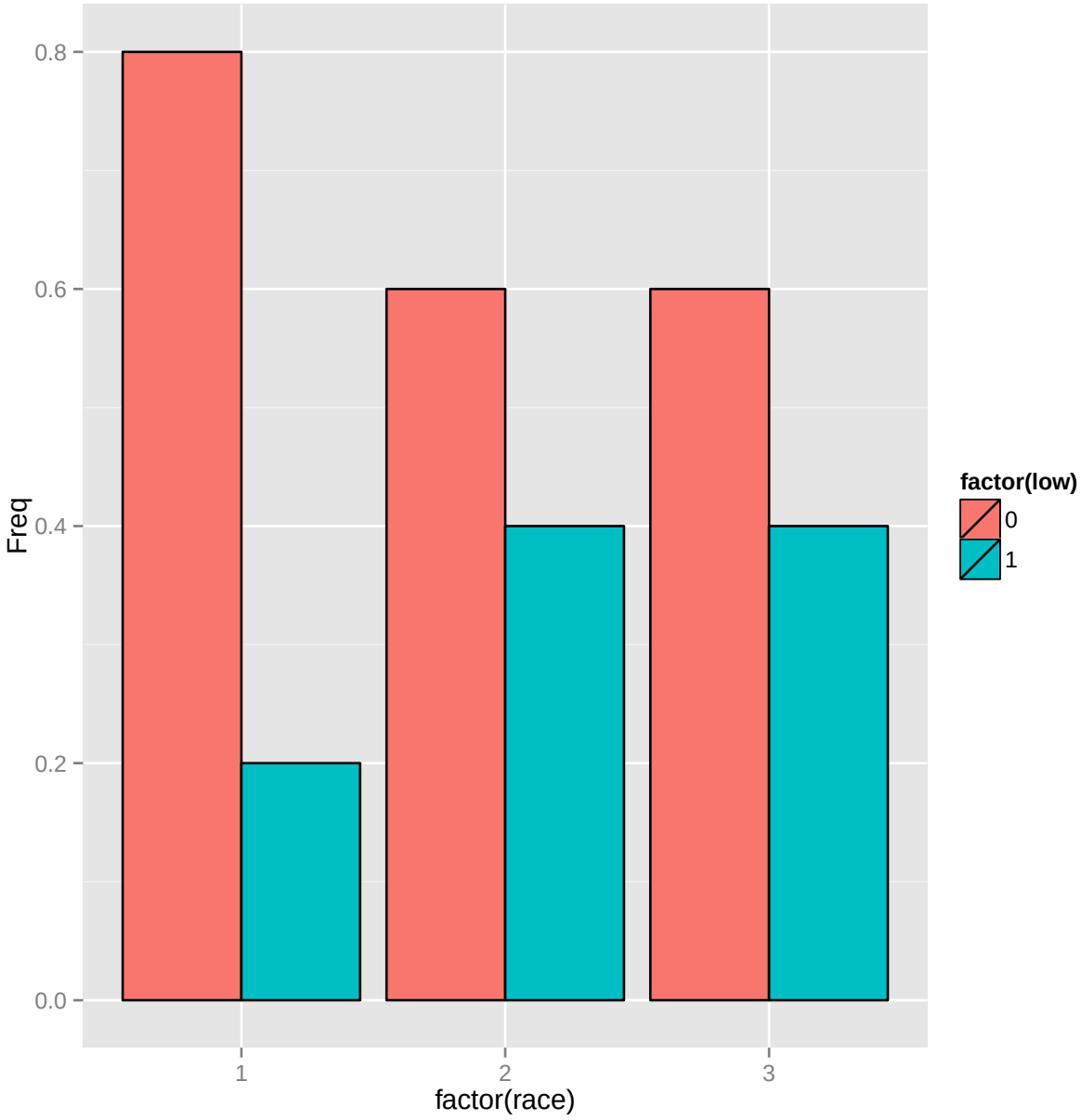
Para poder utilizar los resultados en ggplot2, debemos transformar la tabla a un data.frame:

```
tp <- as.data.frame(round(prop.table(t,1),1))
tp
```

```
##   race low Freq
## 1    1   0  0.8
## 2    2   0  0.6
## 3    3   0  0.6
## 4    1   1  0.2
## 5    2   1  0.4
## 6    3   1  0.4
```

Ahora podemos representar estos porcentages

```
ggplot(tp,aes(x=factor(race),fill=factor(low),Freq))+
  geom_bar(stat="identity",position=position_dodge(),color="black")
```



2.9. Gráficas de medidas e intervalos de confianza

La obtención de gráficos de medias es algo complicada con ggplot2. Para simplificarlo, es necesario cargar una función auxiliar summarySE que se encuentra en la página web: http://wiki.stdout.org/rcookbook/Graphs/ en el apartado Plotting means and error bars. En este apartado utiizaremos el mismo ejemplo que se encuentra más desarrollado en dicha página. Para cargar esta función es necesario tener instalada la libreria plyr.

Empezaremos, calulado los datos necesarios. En este caso, utilizamos los datos ToothGrowth:

```
install.packages("plyr")
```

```
## Installing package into 'C:/Users/User/Documents/R/win-library/3.2'
## (as 'lib' is unspecified)
## Error in contrib.url(repos, "source"): trying to use CRAN without setting a
mirror

library(plyr)

## Warning: package 'plyr' was built under R version 3.2.2
##
## Attaching package: 'plyr'
##
## The following objects are masked from 'package:Hmisc':
##
##     is.discrete, summarize
```

```
## Error in eval(expr, envir, enclos): objeto 'dfc' no encontrado
```

Se puede apreciar que se ha creado un data.frame con la media de longitud por dosis y suplemento. Así mismo, se ha calculado la correspondiente desviación estandard y el valor de la amplitud del intervalo de confianza. Una vez calculados, podemos representarlos fácilmente

```
ggplot(dfc, aes(x=dose, y=len, colour=supp)) + geom_errorbar(aes(ymin=len-ci, ymax=len

## Error in ggplot(dfc, aes(x = dose, y = len, colour = supp)): objeto 'dfc' no
encontrado
```

Un ejemplo que reune muchas de las opciones que pueden utilizarse sería:

```
pd <- position_dodge(.1)
ggplot(dfc, aes(x=dose, y=len, colour=supp, group=supp)) + geom_errorbar(aes(ymin=len-

## Error in ggplot(dfc, aes(x = dose, y = len, colour = supp, group = supp)):
objeto 'dfc' no encontrado

# Legend label, use darker colors
# Use darker colors, lightness=40
# Set y range
# Set tick every 4
# Position legend in bott
```

En algunos casos, se prefiere representar las medias con barras. Podemos hacer lo siguiente. Primero, definiremos la dosis como un factor:

```
dfc2 <- dfc

## Error in eval(expr, envir, enclos): objeto 'dfc' no encontrado

dfc2$dose <- factor(dfc2$dose)

## Error in factor(dfc2$dose): objeto 'dfc2' no encontrado
```

Ahora podemos obtener la gráfica haciendo:

```
ggplot(dfc2, aes(x=dose, y=len, fill=supp)) +
  geom_bar(stat="identity",position=position_dodge()) +
  geom_errorbar(aes(ymin=len-ci, ymax=len+ci),
                width=.2, position=position_dodge(.9))

## Error in ggplot(dfc2, aes(x = dose, y = len, fill = supp)): objeto 'dfc2' no
encontrado

# Width of the error bars
```

Podemos acabar, con un ejemplo más elaborado

```
ggplot(dfc2, aes(x=dose, y=len, fill=supp)) + geom_bar(position=position_dodge(),
    colour="black", size=.3) + geom_errorbar(aes(ymin=len-se, ymax=len+se),
    size=.3, width=.2, position=position_dodge(.9)) + xlab("Dose (mg)") +
  ylab("Tooth length") + scale_fill_hue(name="Supplement type",
  breaks=c("OJ", "VC"), labels=c("Orange juice", "Ascorbic acid")) +
  ggtitle("The Effect of Vitamin C on\nTooth Growth in Guinea Pigs") +
  scale_y_continuous(breaks=0:20*4) + theme_bw()

## Error in ggplot(dfc2, aes(x = dose, y = len, fill = supp)): objeto 'dfc2' no
encontrado

# Use black outlines,
# Thinner lines
# Legend label, use darker colors
```

2.10. Ejemplo 2:

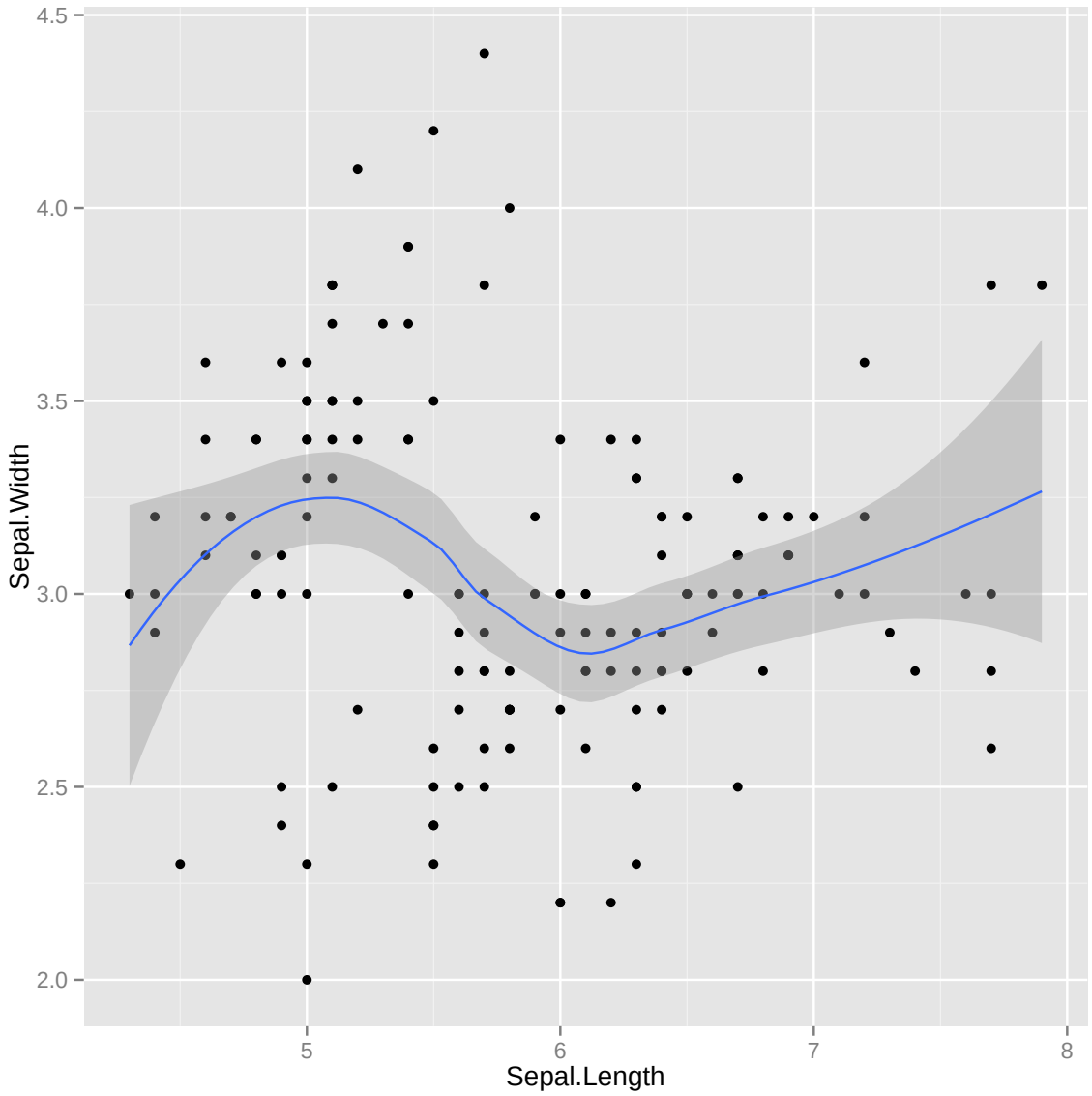
1. **Activa tu directorio de trabajo**

```
getwd()

## [1] "C:/Users/User/Documents/GGPLOT2_TAREA"

setwd("C:/Users/User/Documents/GGPLOT2_TAREA")
```

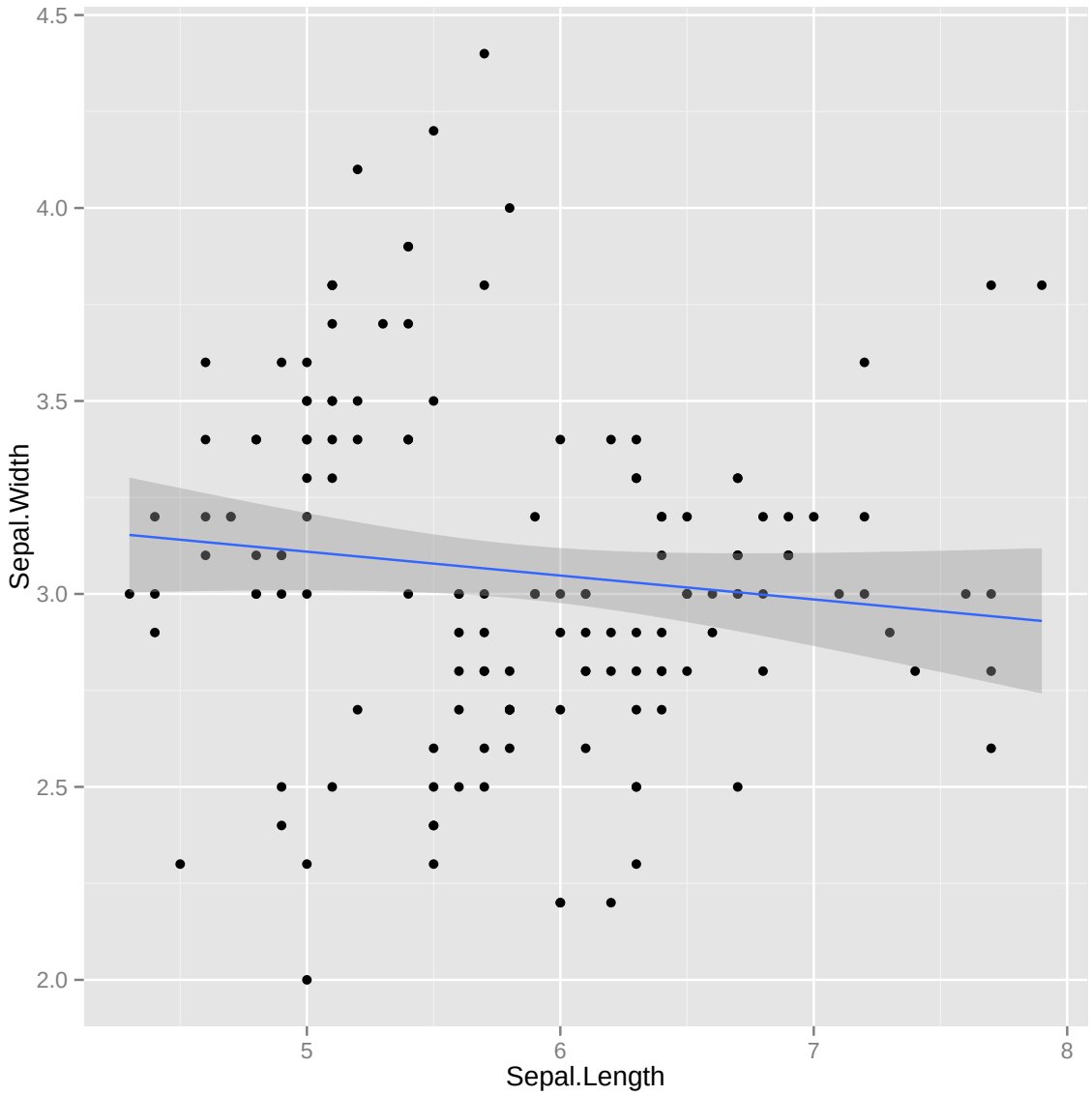
2. **Instalar el paquete ggplot de forma manual o utilizando el menú en R Packages**

3. **Activar la libreria**

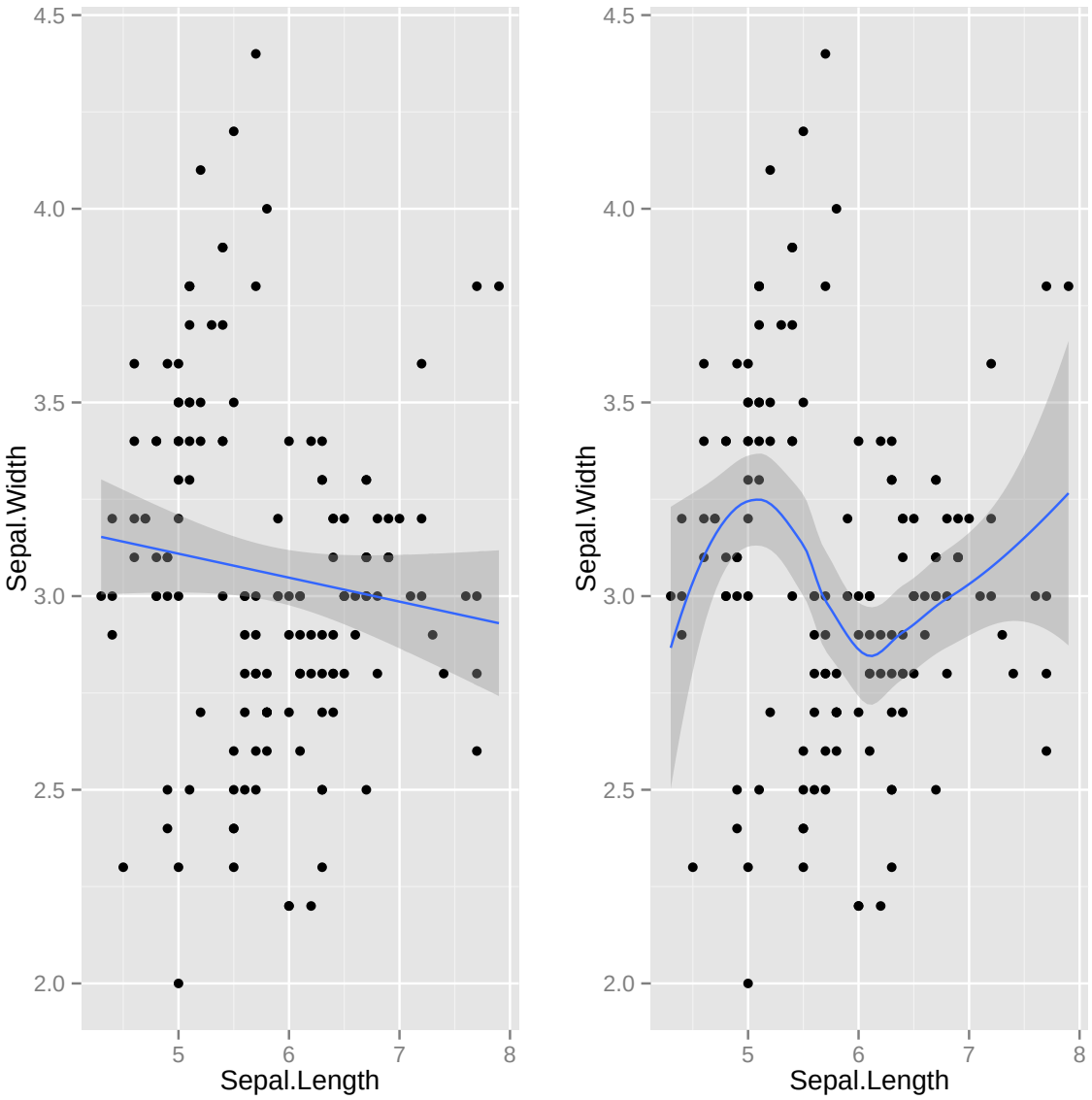
```
library("ggplot2", lib.loc="~/R/win-library/3.2")

ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width)) + geom_point() +
  stat_smooth()

## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
```



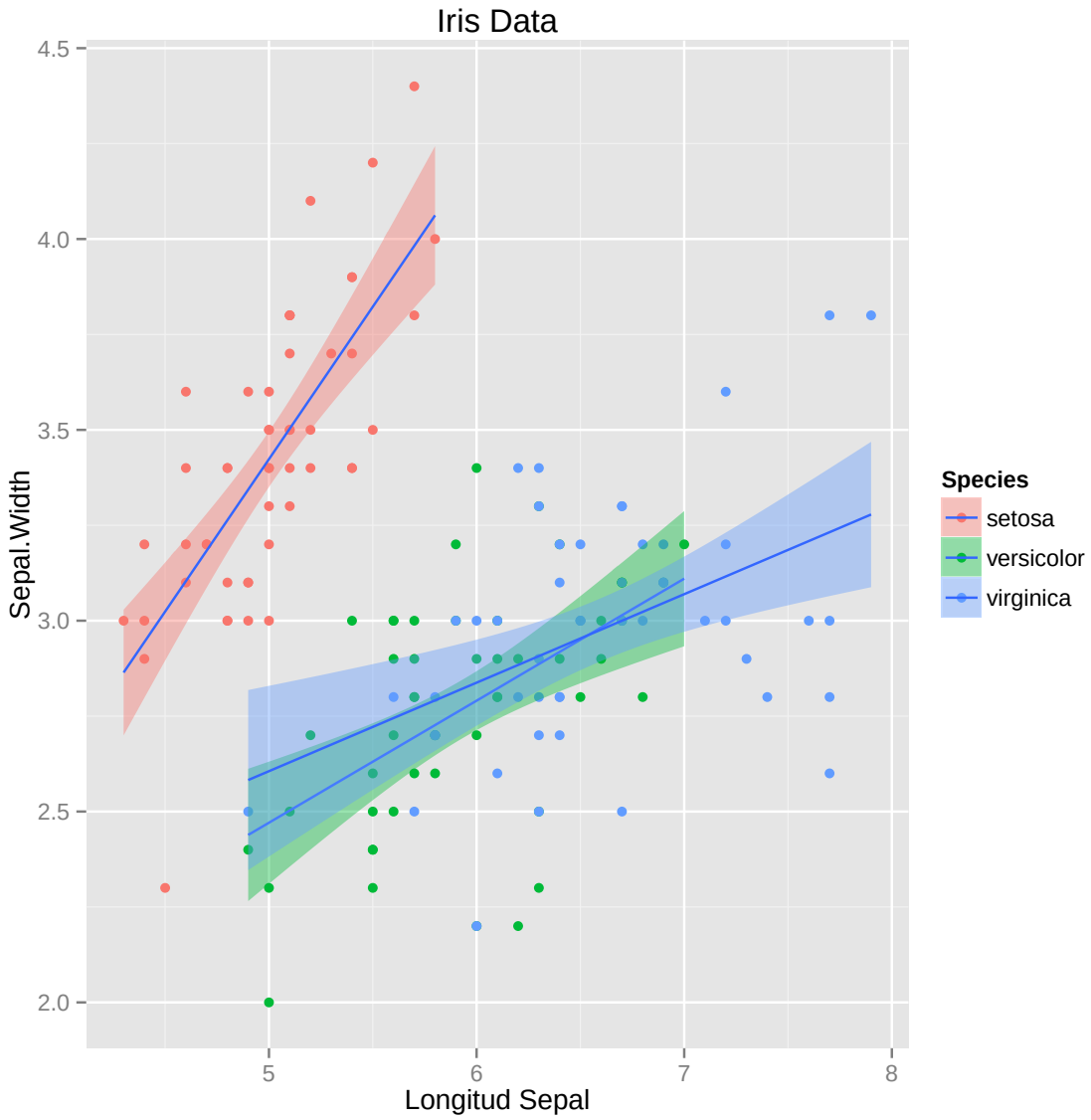

```
p<-ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width))
p+geom_point()+geom_smooth(method='lm')
```



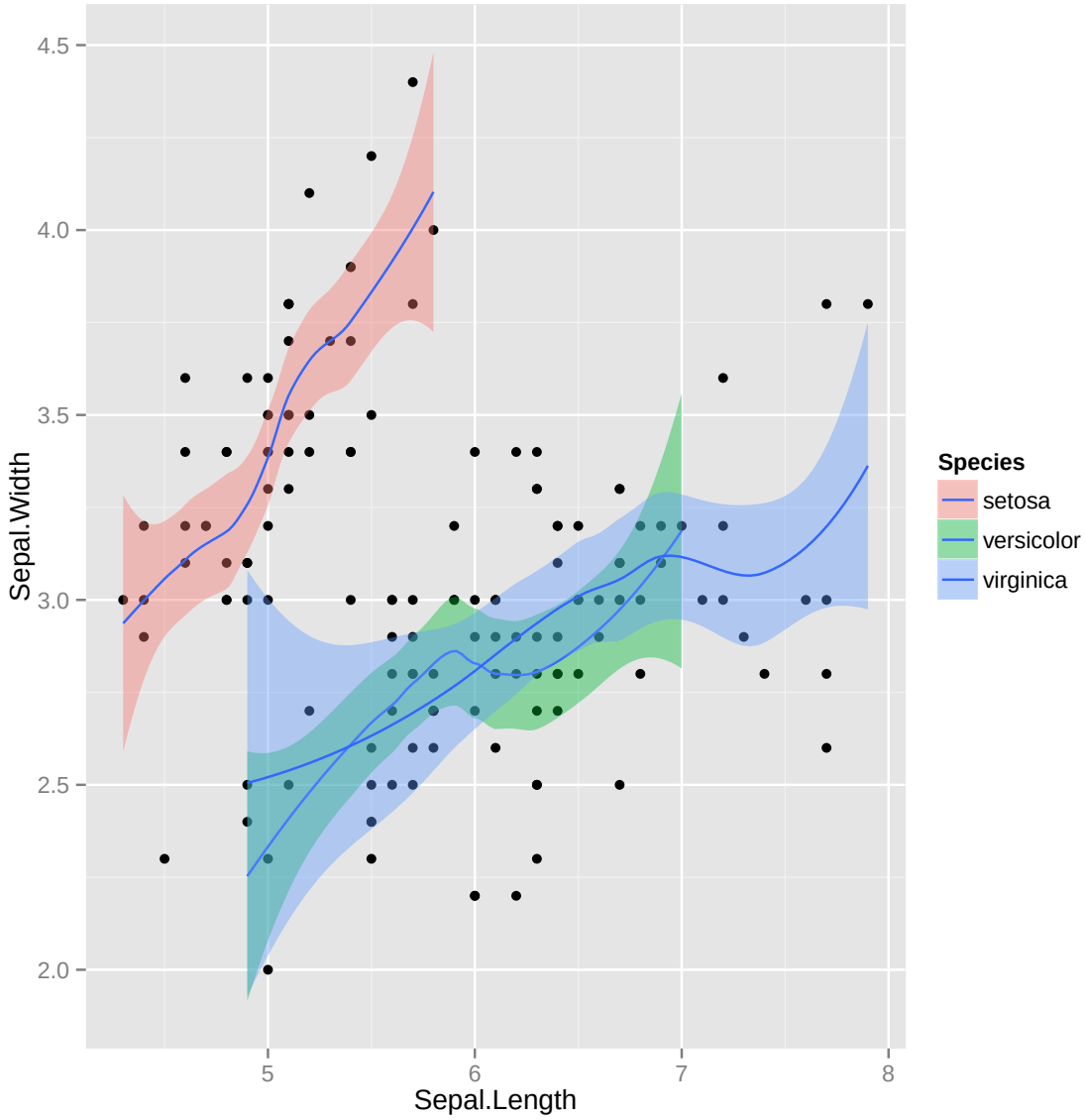

```
library("grid", lib.loc="~/R/win-library/3.2")
vplayout <- function(x, y) viewport(layout.pos.row = x, layout.pos.col = y)
p1<-ggplot(data=iris,aes(x=Sepal.Length,y=Sepal.Width))
grid.newpage()
pushViewport(viewport(layout = grid.layout(1, 2)))
print(p1+geom_point()+geom_smooth(method='lm'), vp = vplayout(1, 1))
print(p1+geom_point()+geom_smooth(), vp = vplayout(1, 2))

## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
```



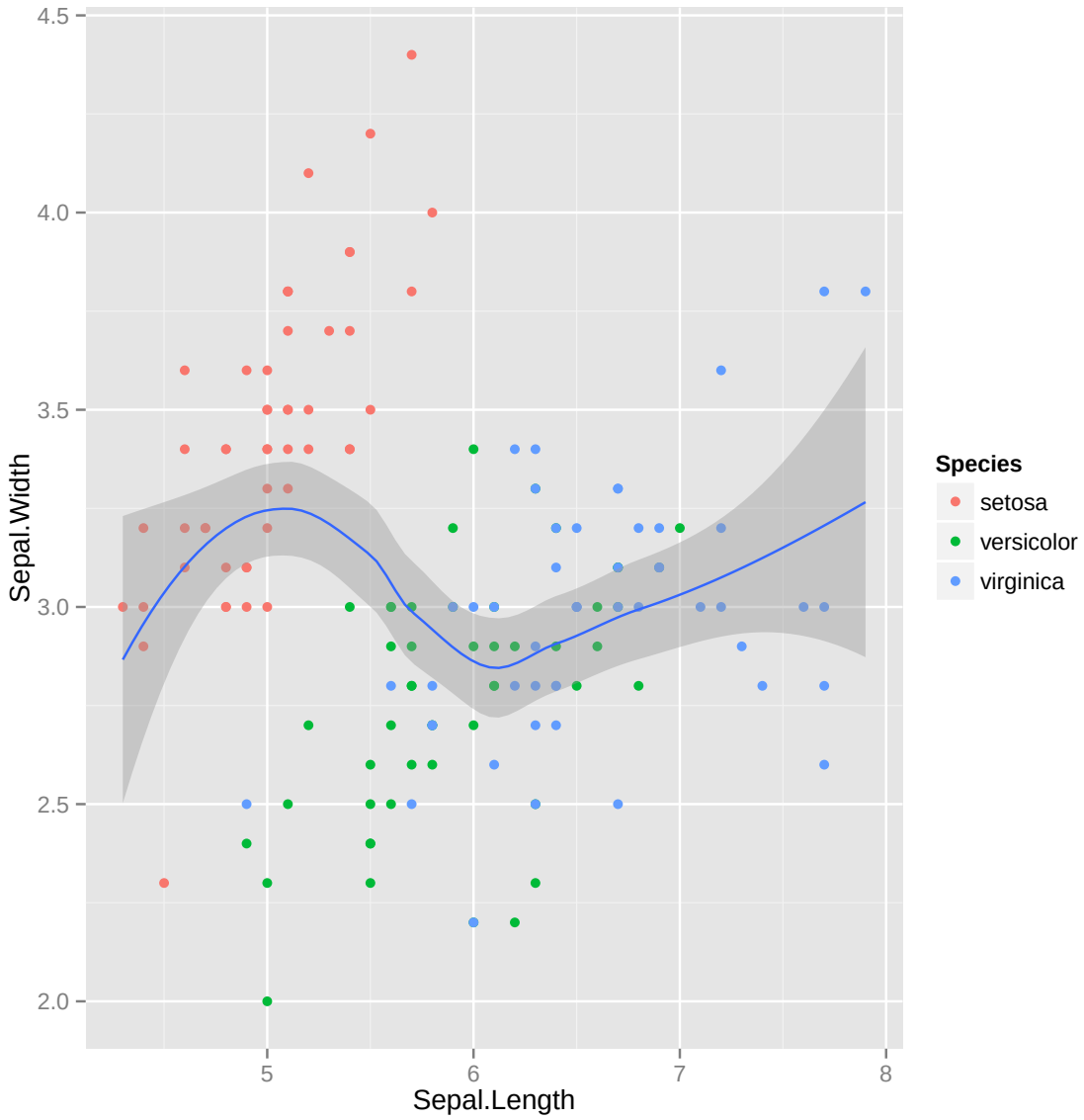

```
p1+geom_point(aes(colour=Species))+
geom_smooth(aes(fill=Species),method='lm')+ labs(title='Iris Data',
                                                 x='Longitud Sepal')
```



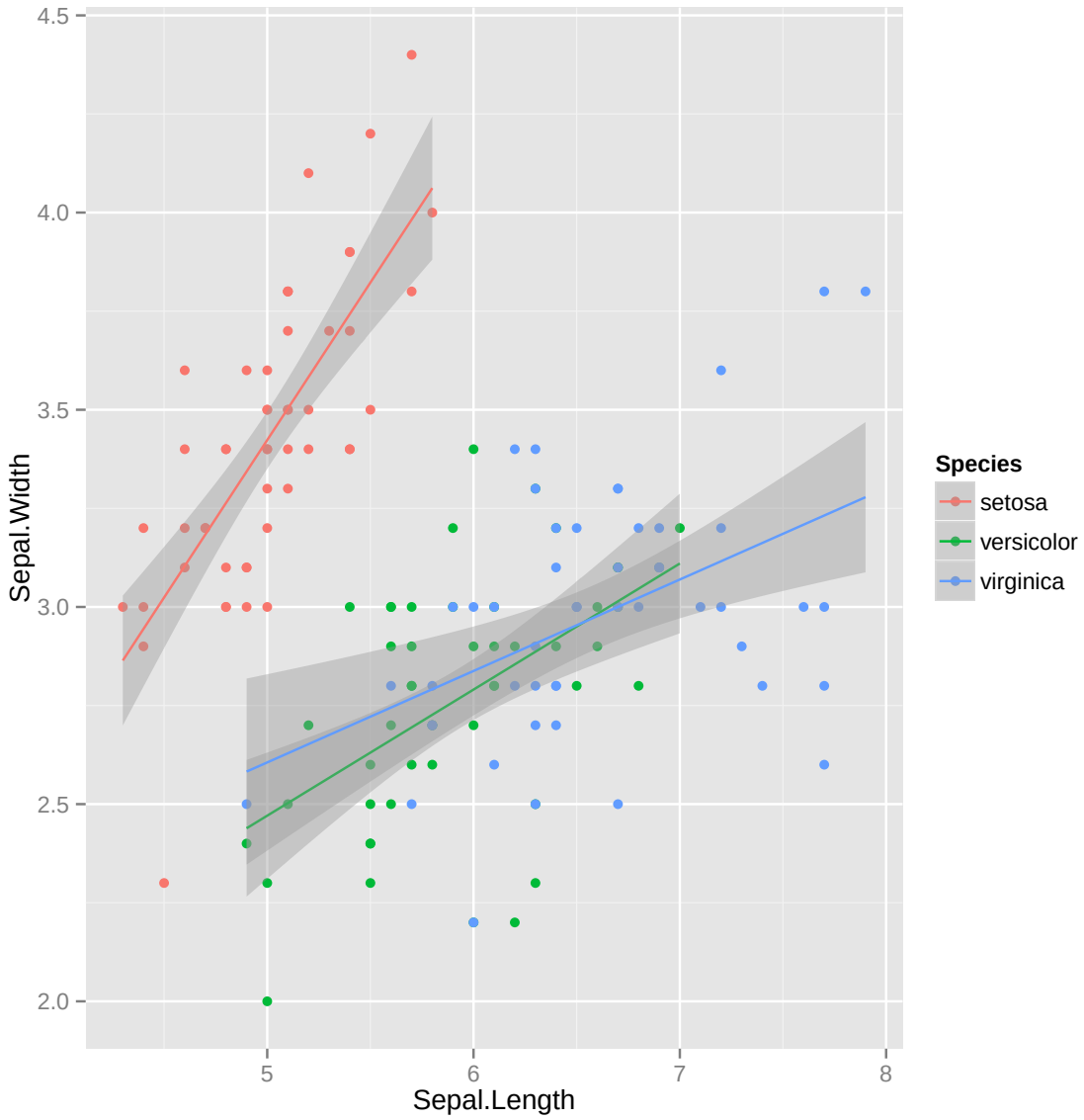

```
p1+geom_point()+
geom_smooth(aes(fill=Species))


## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
```




```
p1+geom_point(aes(colour=Species))+
geom_smooth()

## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
```

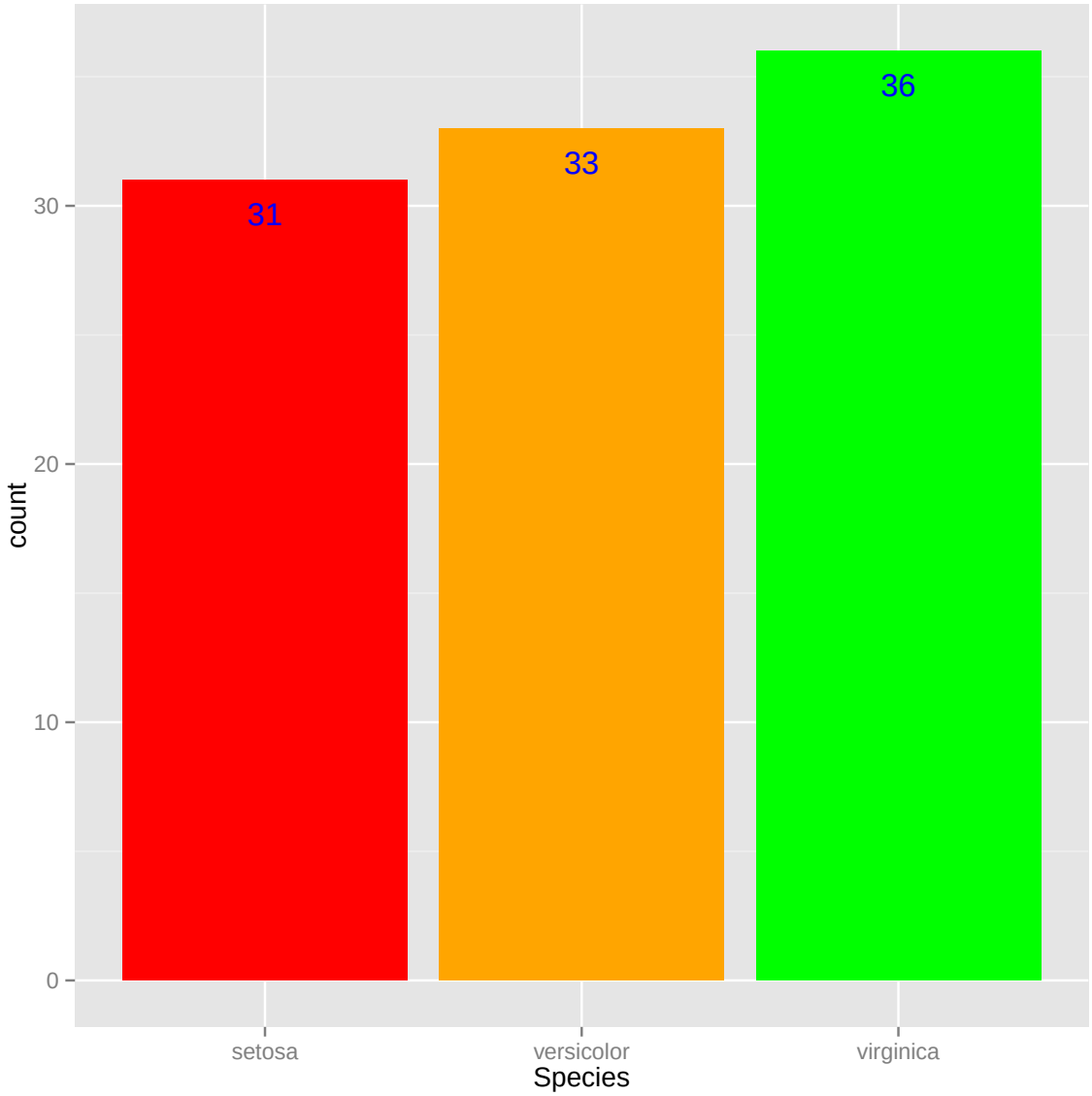



```
p2<-p1+aes(colour=Species)

p2+geom_point()+geom_smooth(method='lm')
```



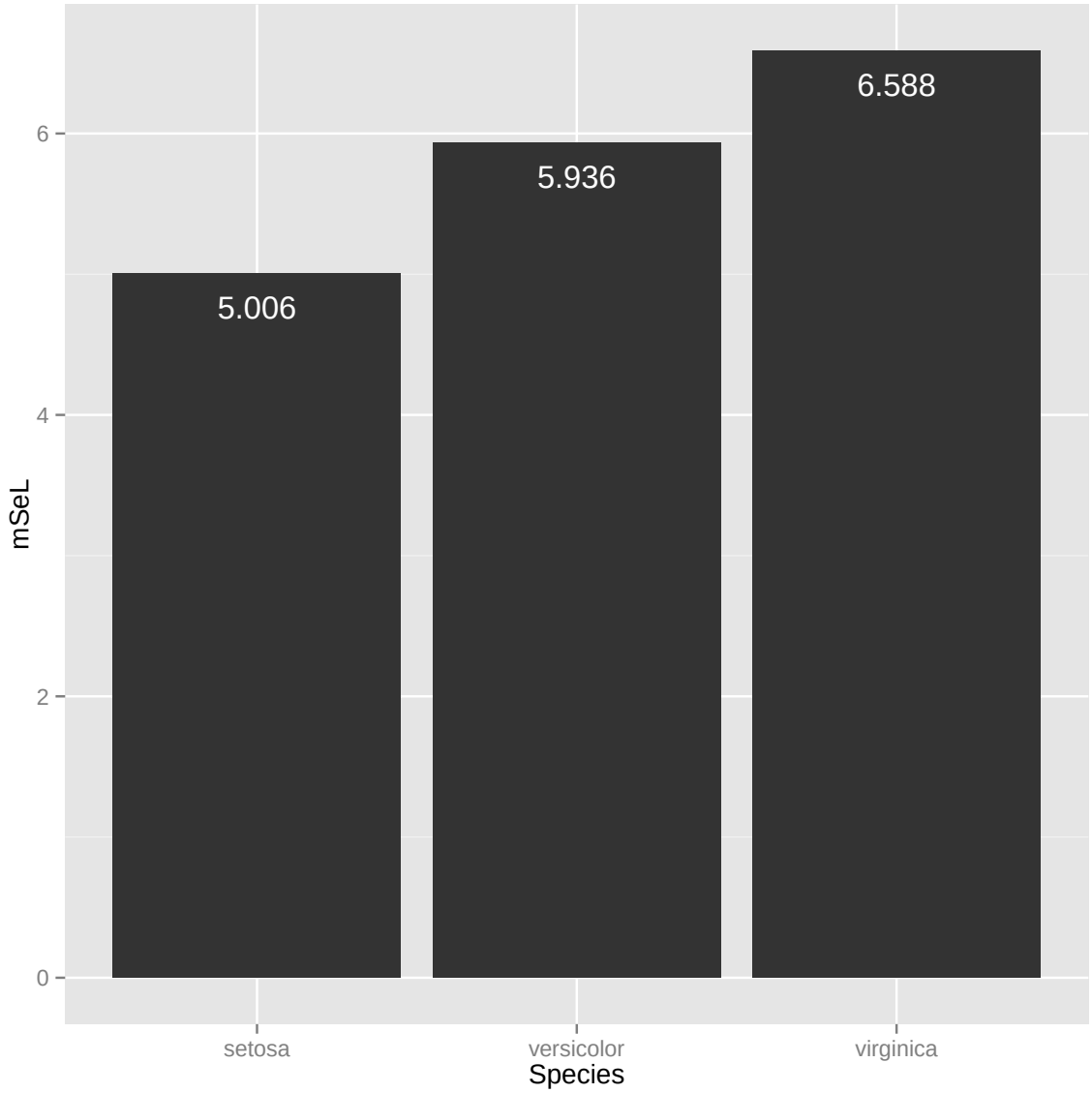
TRANSFORMACIONES ESTADÍSTICA: Utilización para añadir una etiqueta a las barras de datos

```
library("ggplot2", lib.loc="~/R/win-library/3.2")
index<-sample(1:150,100)
subdata<-iris[index,]
p2<-ggplot(subdata,aes(x=Species))+geom_bar(fill=c('red','orange','green'))
p2+stat_bin(geom='text',aes(label=..count..,vjust=2),colour='blue')

## ymax not defined: adjusting position using y instead
```



Cómo podemos representar la media de una variable según la especie

```
library("plyr", lib.loc="~/R/win-library/3.2")
# Calculem la mitjana de Sepal.Length per cada especie
mSeL <- ddply(iris, "Species", summarise, mSeL = mean(Sepal.Length))
p<-ggplot(mSeL, aes(x = Species, y = mSeL)) + geom_bar(stat = "identity")
p+geom_text(aes(label=mSeL),vjust=2,colour='white')
```

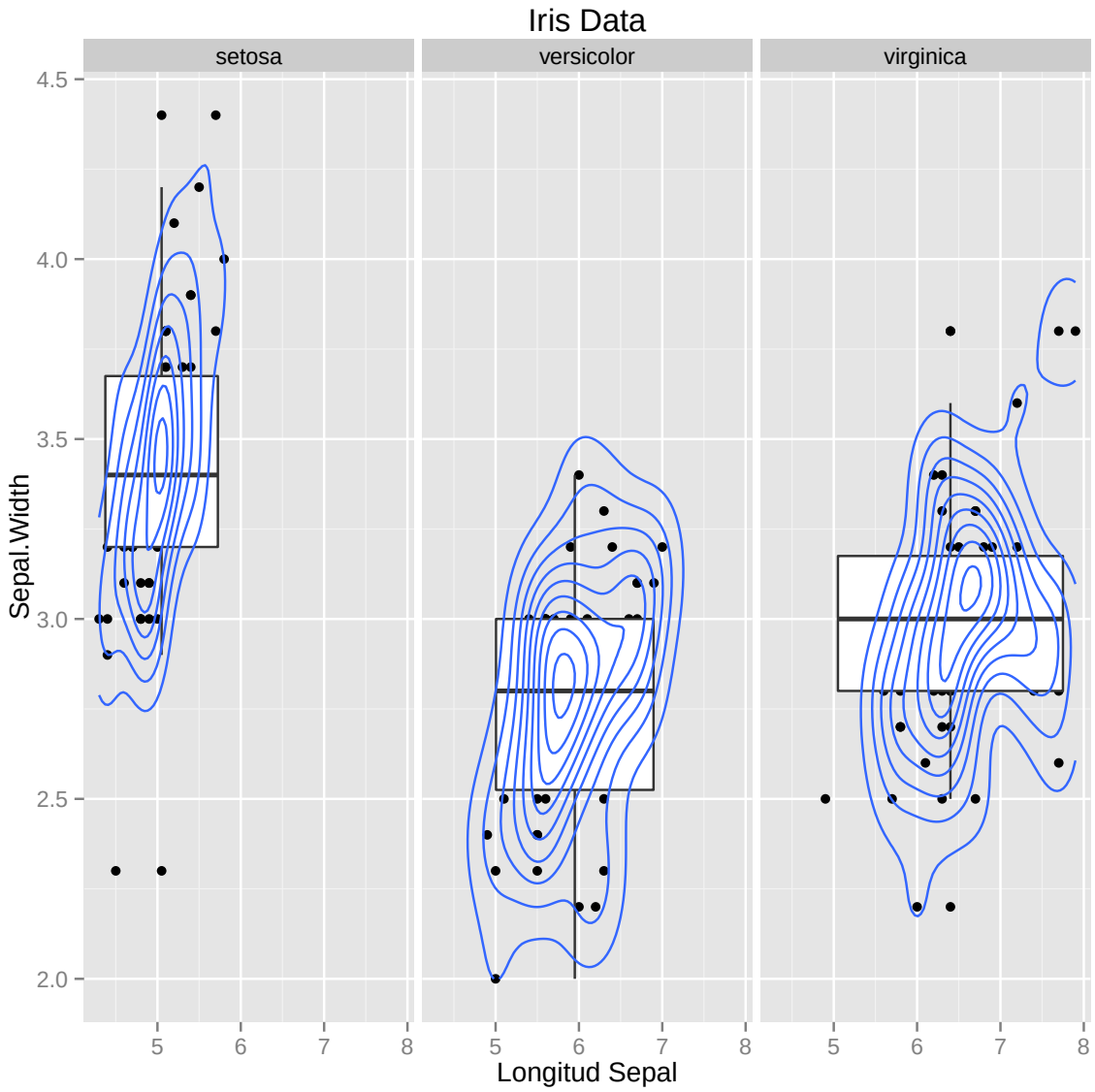


Uso de facets

- Facets es la nomenclatura de ggplot2 para muestras múltiples
- Crea gráficos separados para diferentes subconjuntos de datos definidos por fórmulas

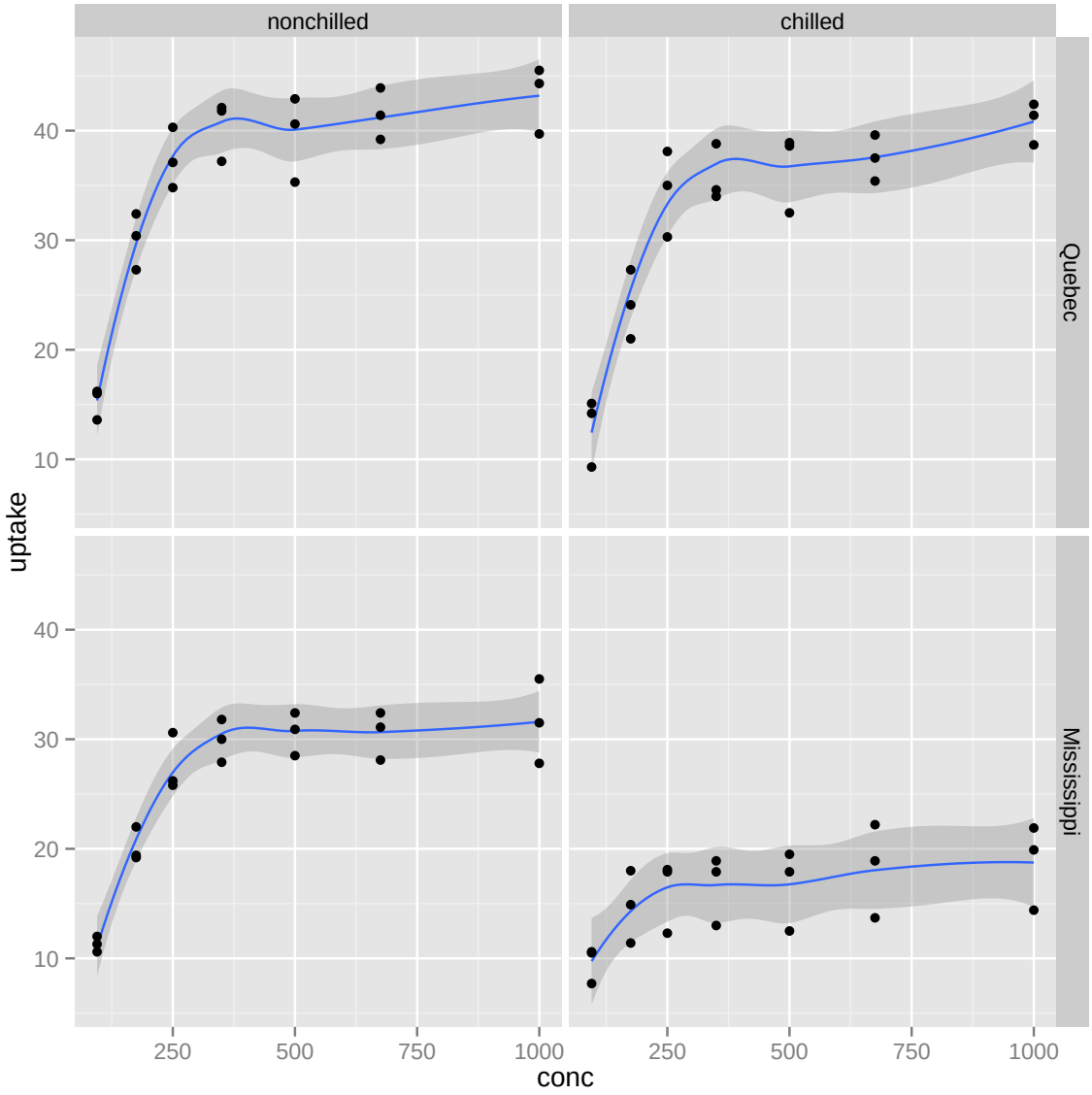
- La principal función es facet$_grid$()

```
p1+geom_point()+geom_boxplot()+geom_density2d()+
  labs(title='Iris Data',x='Longitud Sepal')+
  facet_grid(~Species)
```




```
library("ggplot2", lib.loc="~/R/win-library/3.2")
ggplot(CO2,aes(x=conc,y=uptake))+geom_smooth()+geom_point()+
  facet_grid(Type~Treatment)
```

```
## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
```



Control del aspecto con Scales: características

- Las Scales permiten modificar
- position
- color and fill
- size
- shape
- line type

La sintaxis es scale$_{<}$*aesthetic* $>_{<}$ *type*

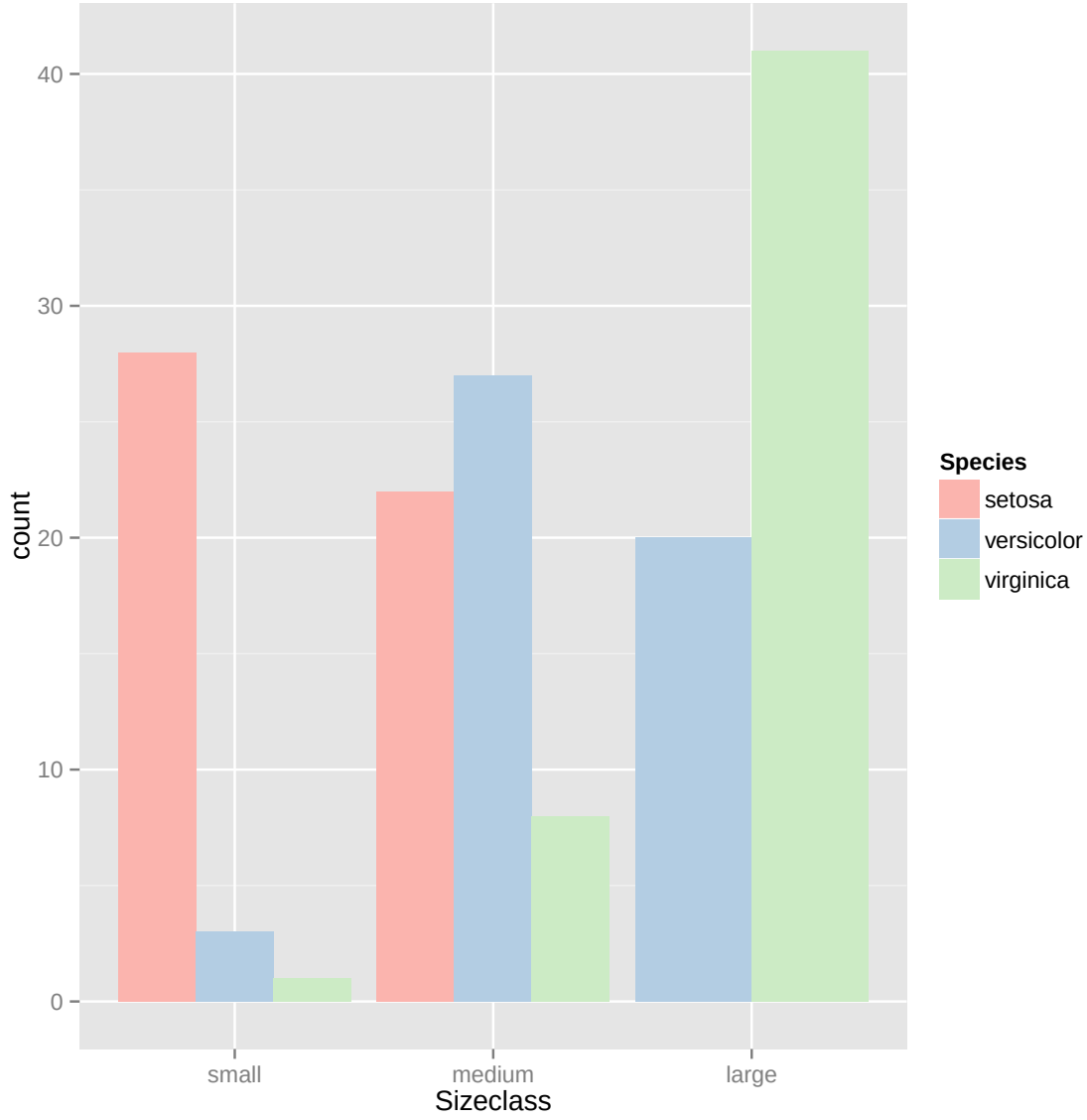
Los principales argumentos para modificar son:*name*

- limits
- breaks

- labels

Modificación del color

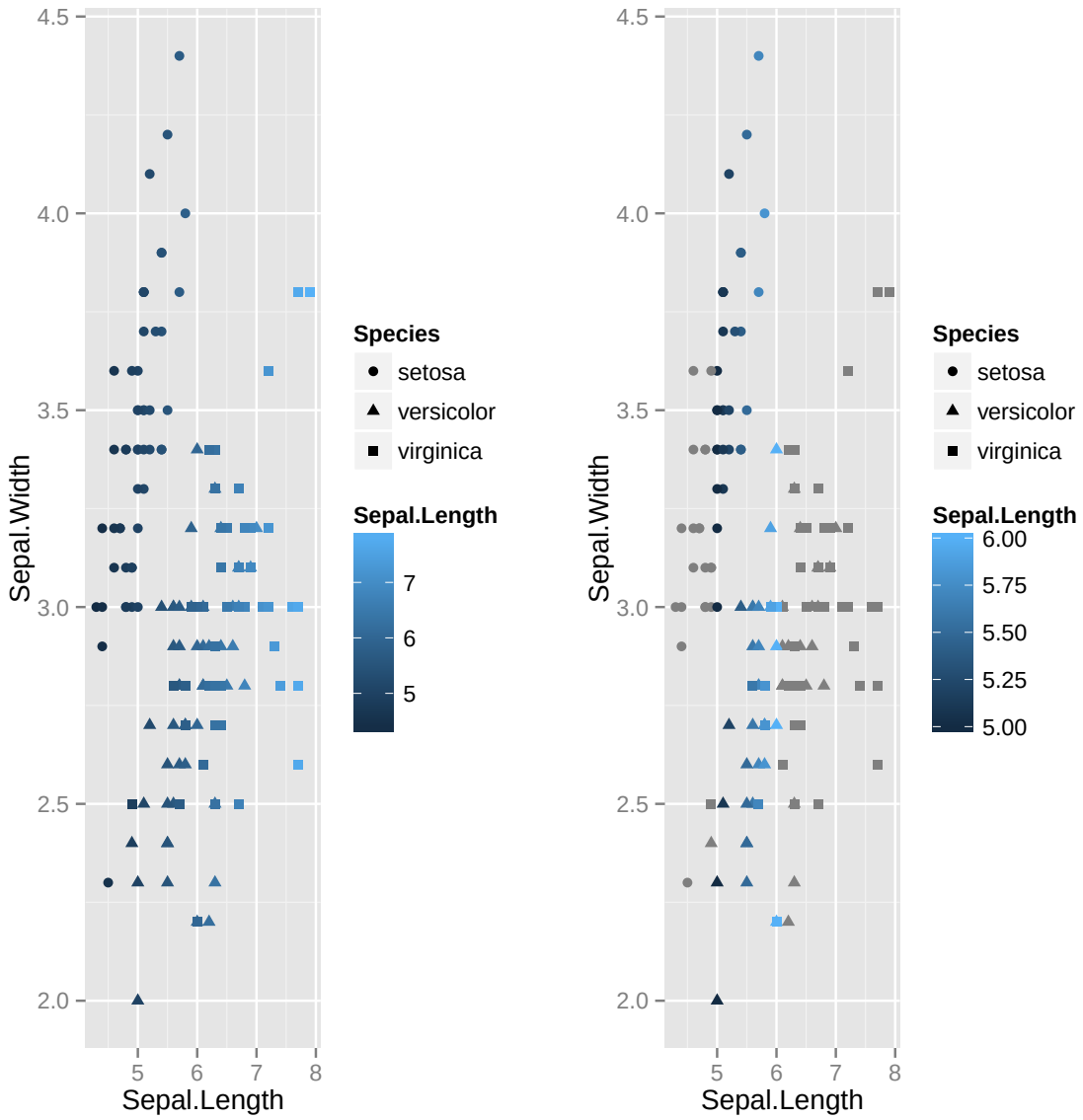
```
iris$Sizeclass <- cut(iris$Sepal.Length, breaks = c(0, 5, 6, Inf),
                      labels = c("small", "medium", "large"))
p<-ggplot(iris,aes(x=Sizeclass,fill=Species))+geom_bar(position='dodge')+
scale_fill_brewer(palette="Pastel1")
p
```



Utilización de las Scales

```
library("grid", lib.loc="~/R/win-library/3.2")
p<-ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width))
grid.newpage()
pushViewport(viewport(layout = grid.layout(1, 2)))
```

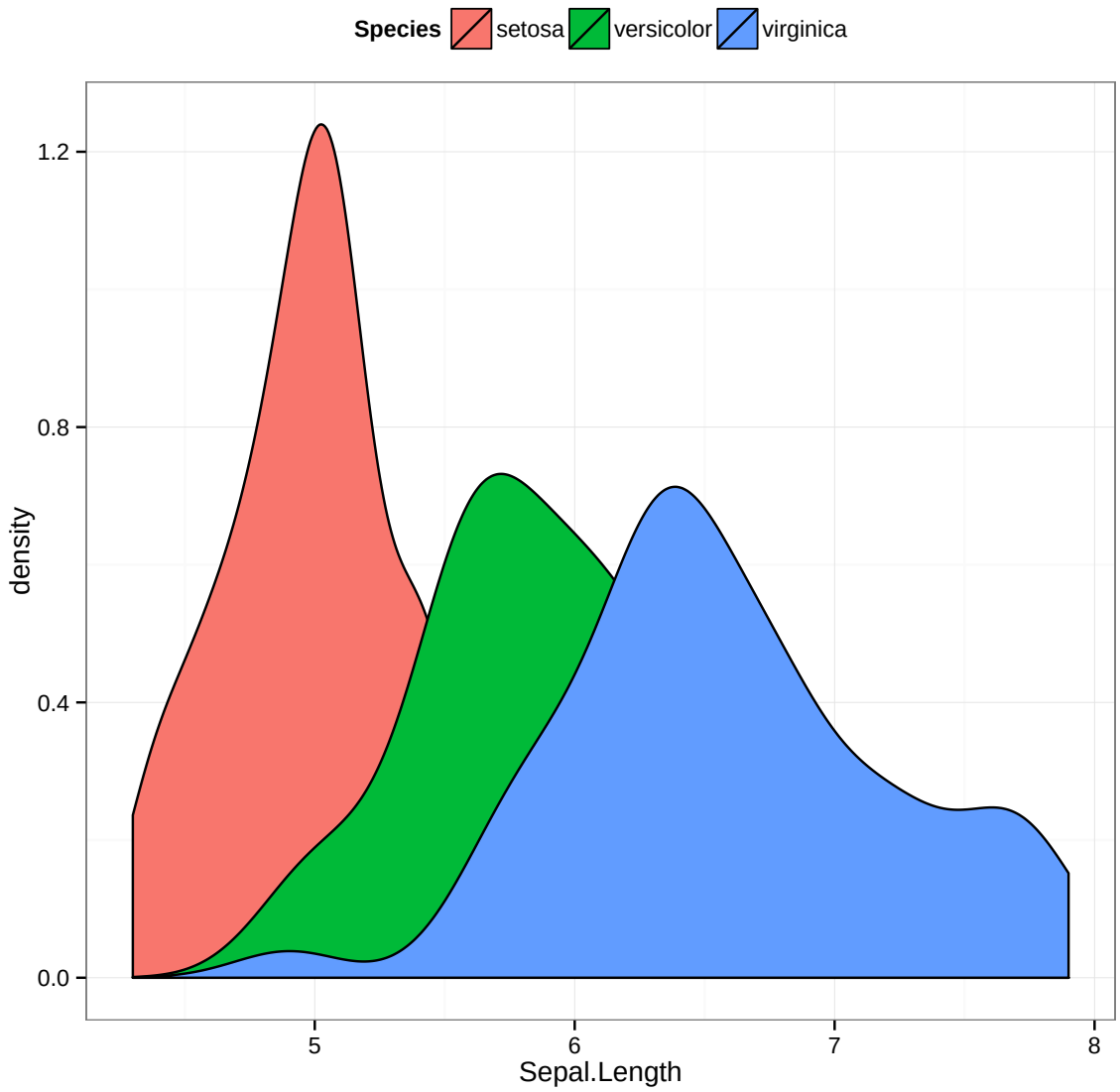
```
print(p+geom_point(aes(colour=Sepal.Length,shape=Species)),
      vp = vplayout(1, 1))
print(p+geom_point(aes(colour=Sepal.Length,shape=Species))+
        scale_colour_gradient(limits=c(5,6)), vp = vplayout(1, 2))
```



Geoms y Stats asociadas a ggplot

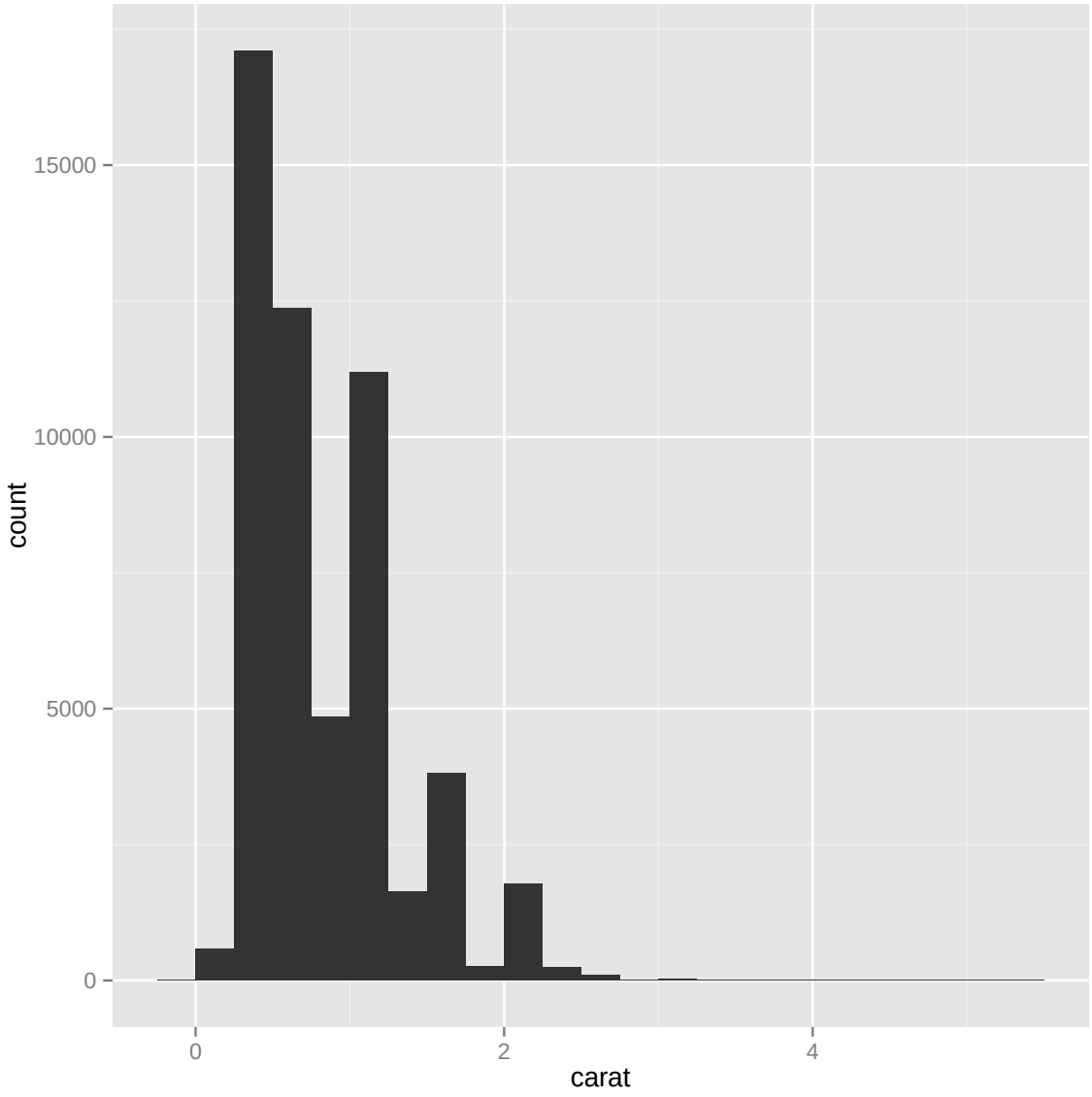
La geom density y la utilización del theme

```
p2<-ggplot(iris,aes(x=Sepal.Length))
p2+geom_density(aes(fill=Species))+theme_bw()+theme(legend.position='top')
```

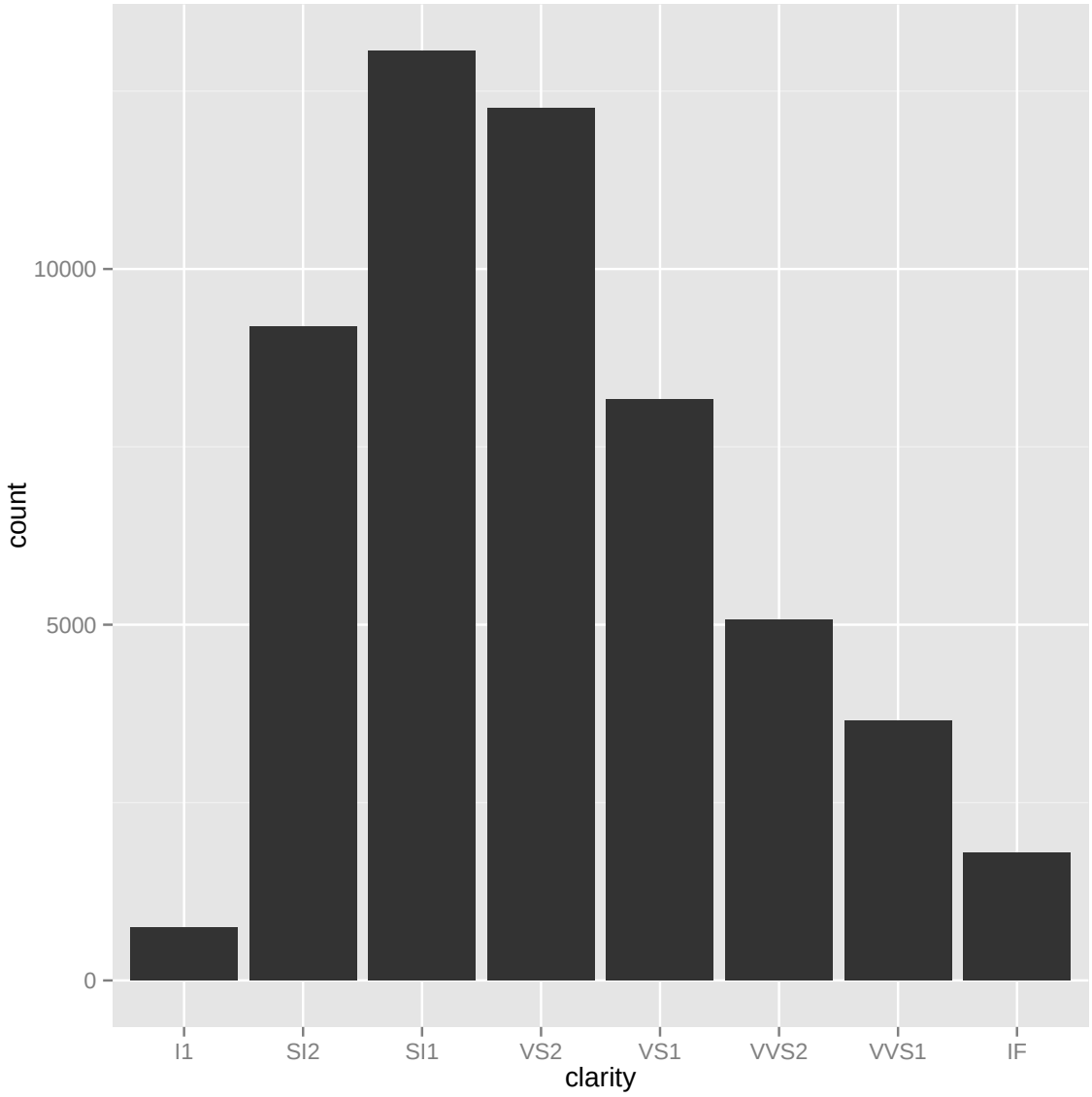


2.11. Ejemplo 3:

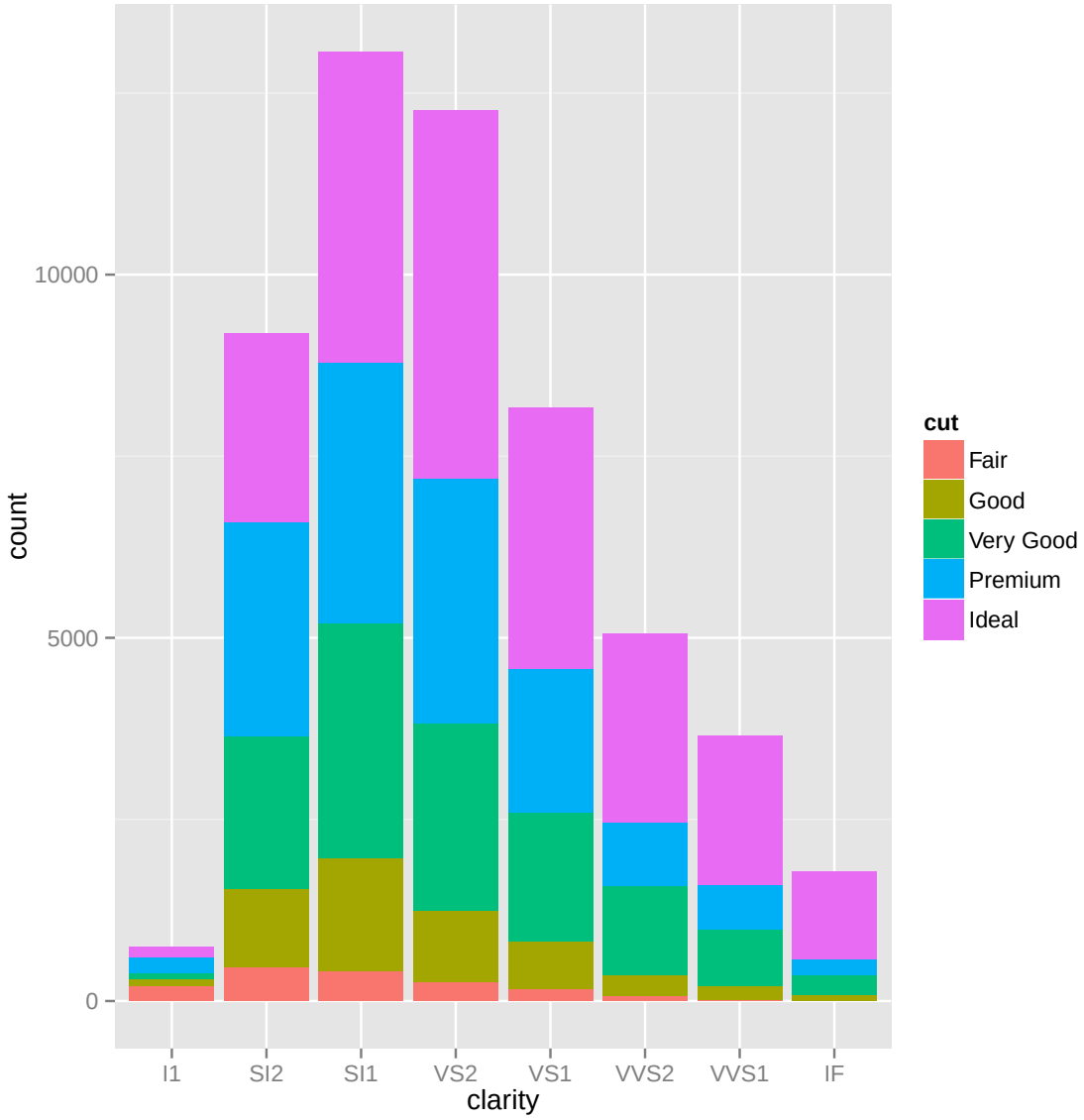
```
library("ggplot2", lib.loc="~/R/win-library/3.2")

# 1 HISTOGRAMA CON BIND ESPECIFICO
qplot(x=carat, binwidth=0.25, data = diamonds)
```



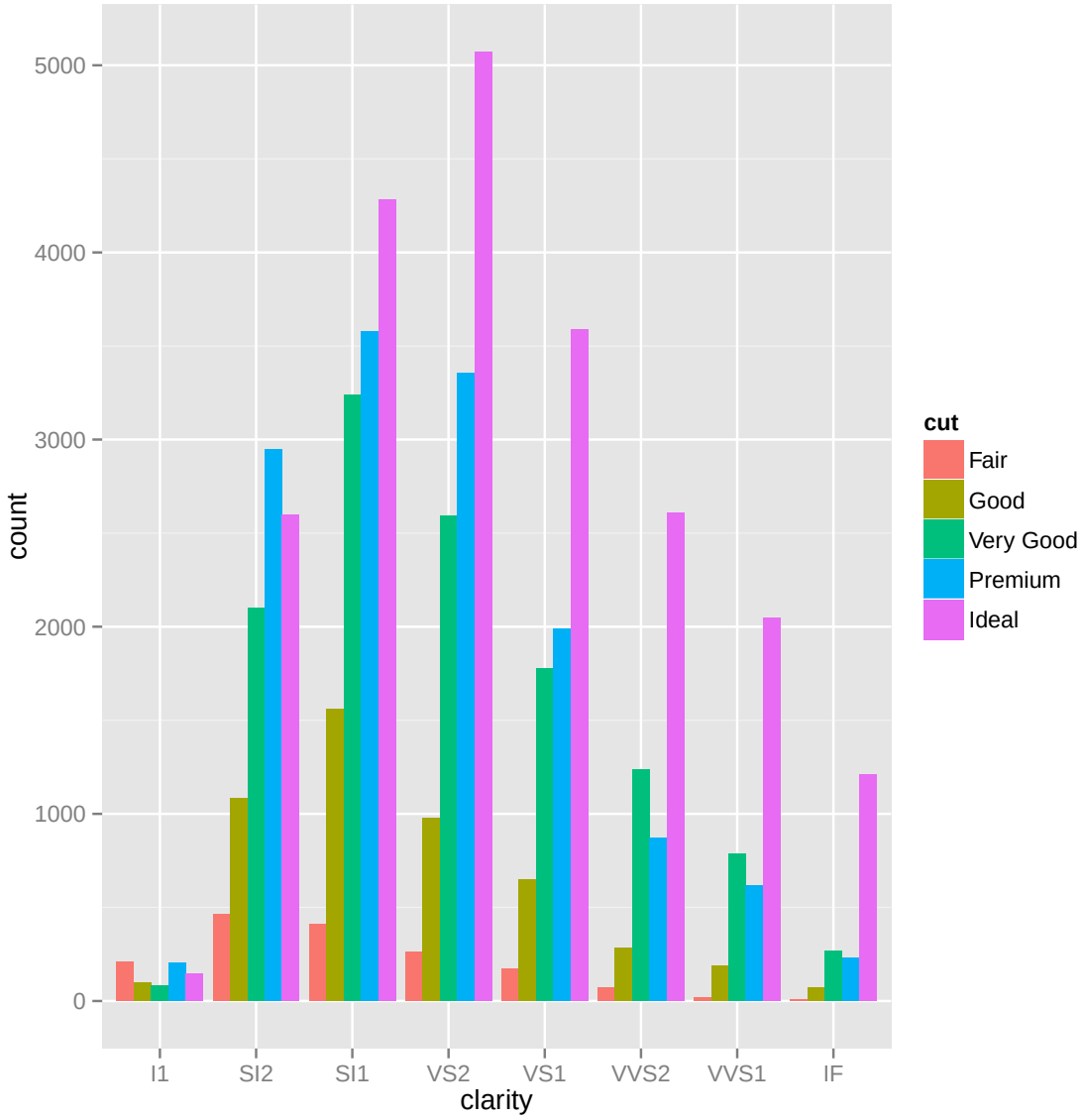

```
# 2 GRAFICO DE BARRAS
qplot(x=clarity, data = diamonds)
```



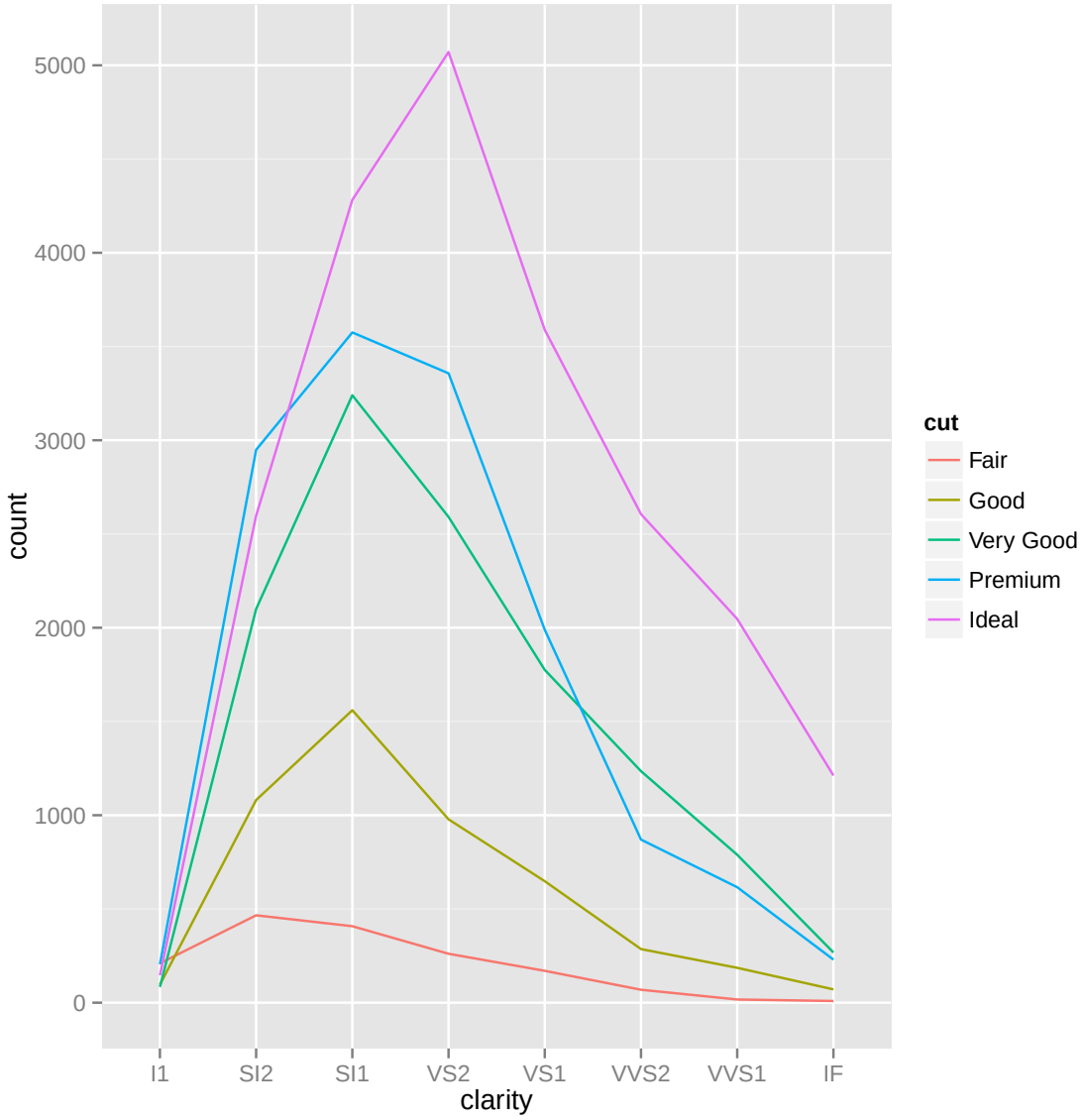

```
# 3 GRAFICO DE BARRAS APILADO
qplot(x=clarity, data=diamonds, geom="bar", fill=cut, position="stack")
```



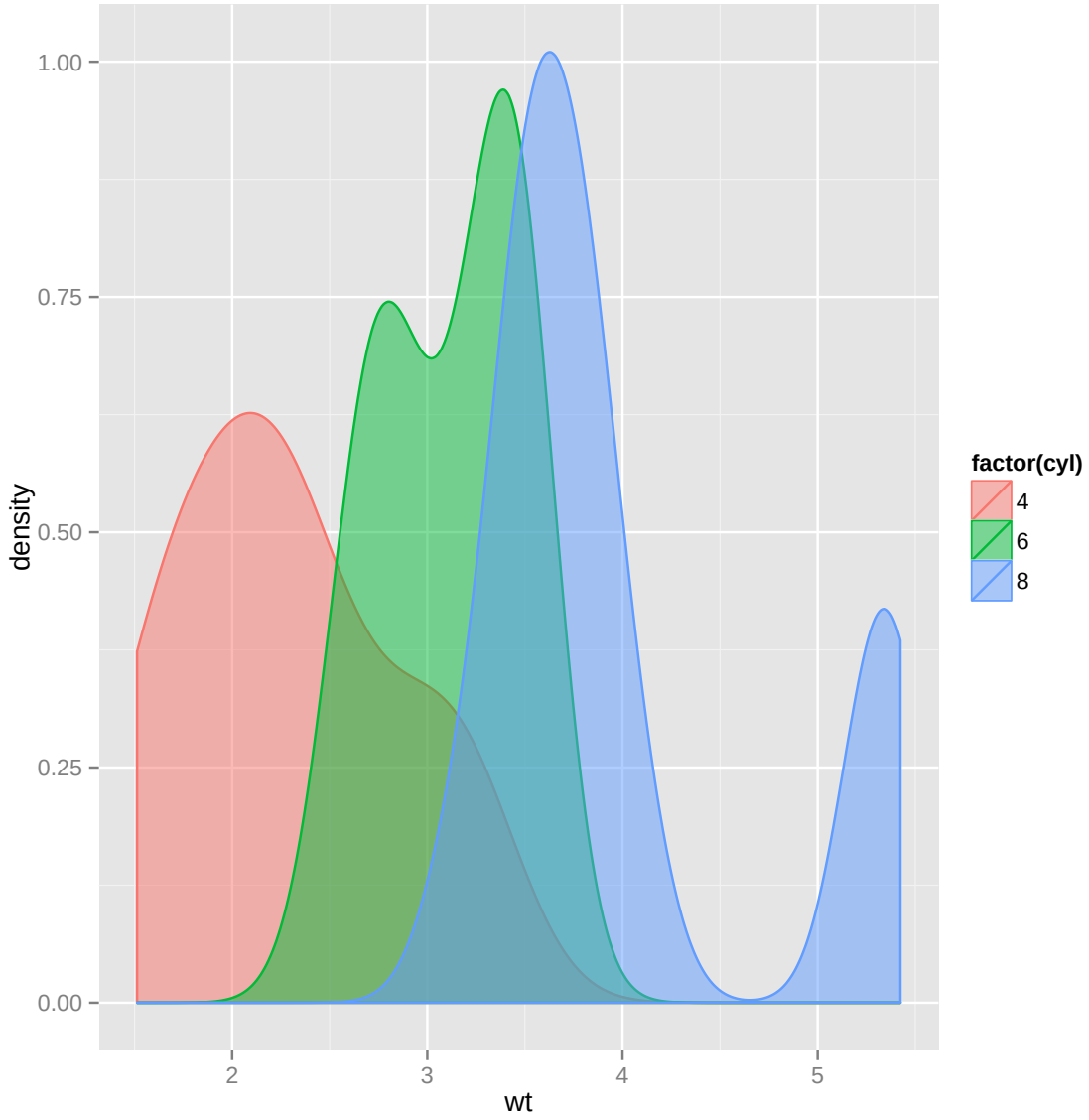

```
# 4 GRAFICO DE BARRA SIN APILAR
qplot(x=clarity, data=diamonds, geom="bar", fill=cut, position="dodge")
```



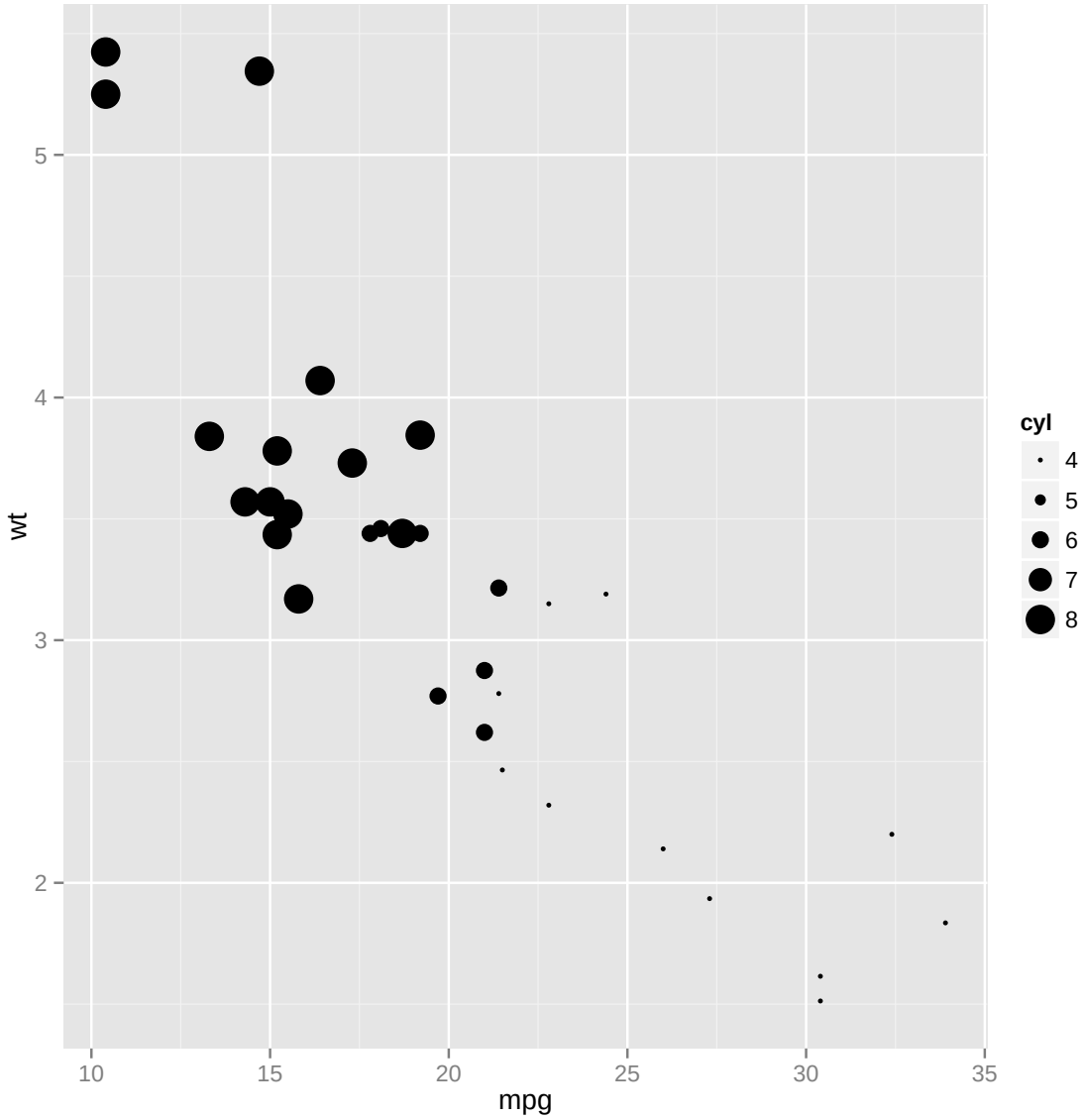

```
# 5 GRAFICO DE LINEA DE FRECUENCIA POR CLASE
qplot(x=clarity, geom="freqpoly", group=cut, colour=cut,data=diamonds)
```



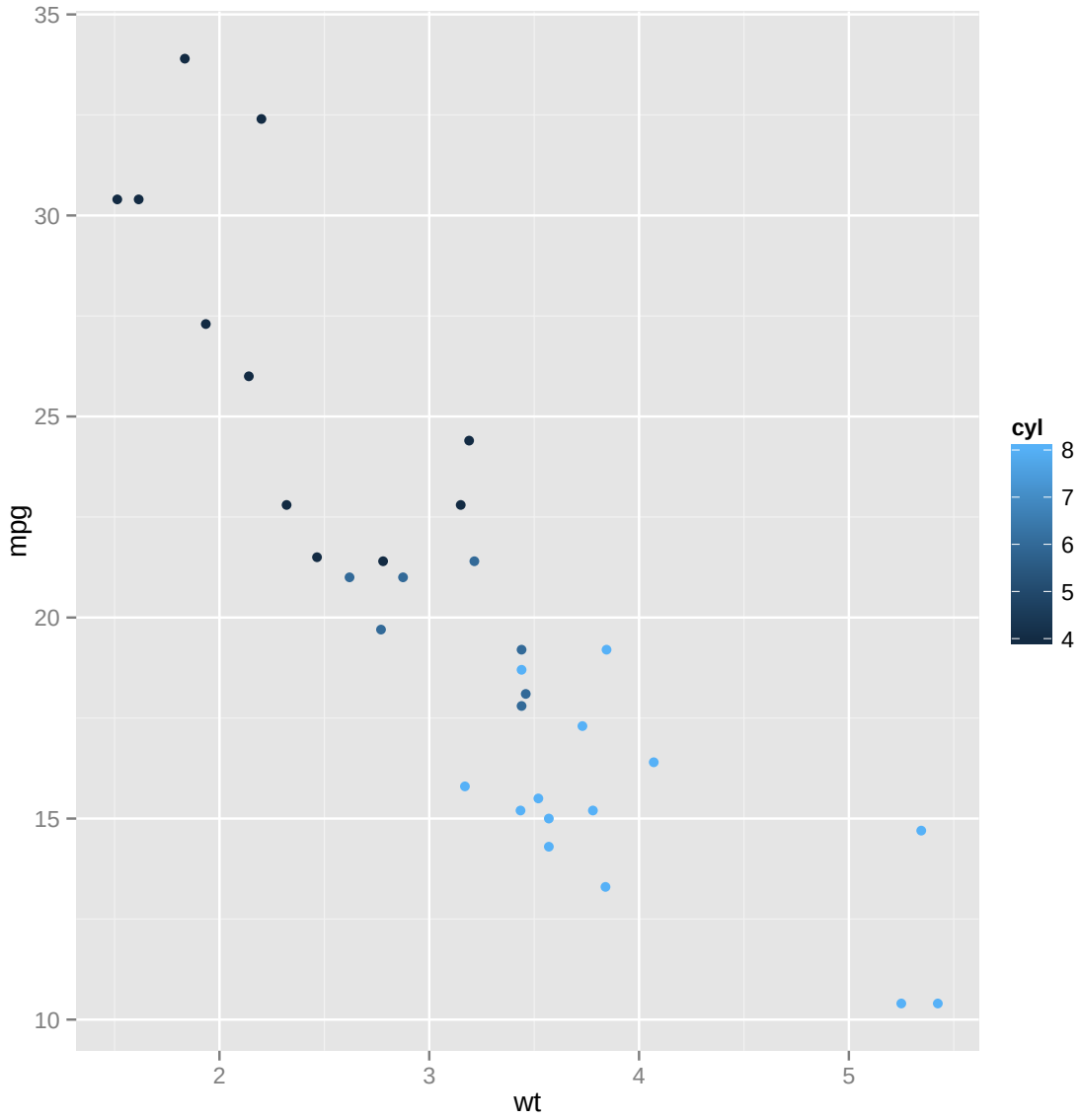

```
# 6 GRAFICO DE AREA
qplot(x=wt, geom="density", group=factor(cyl), colour=factor(cyl),
      fill=factor(cyl),alpha=I(.5),data=mtcars)
```



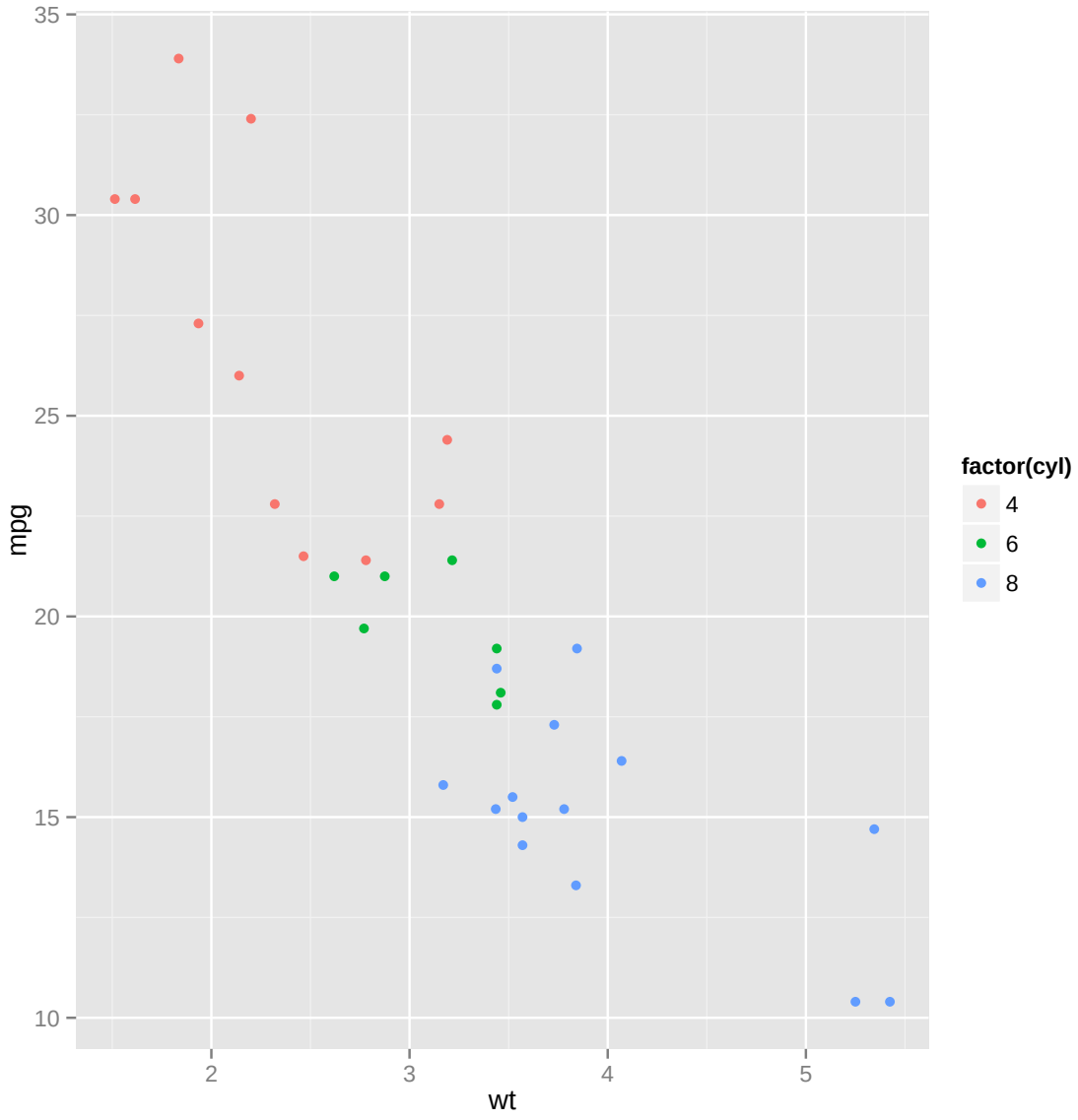

```
# 7 GRAFICO DE DISPERSION CON TAMA?O
qplot(x=mpg, y=wt, size=cyl,data = mtcars)
```



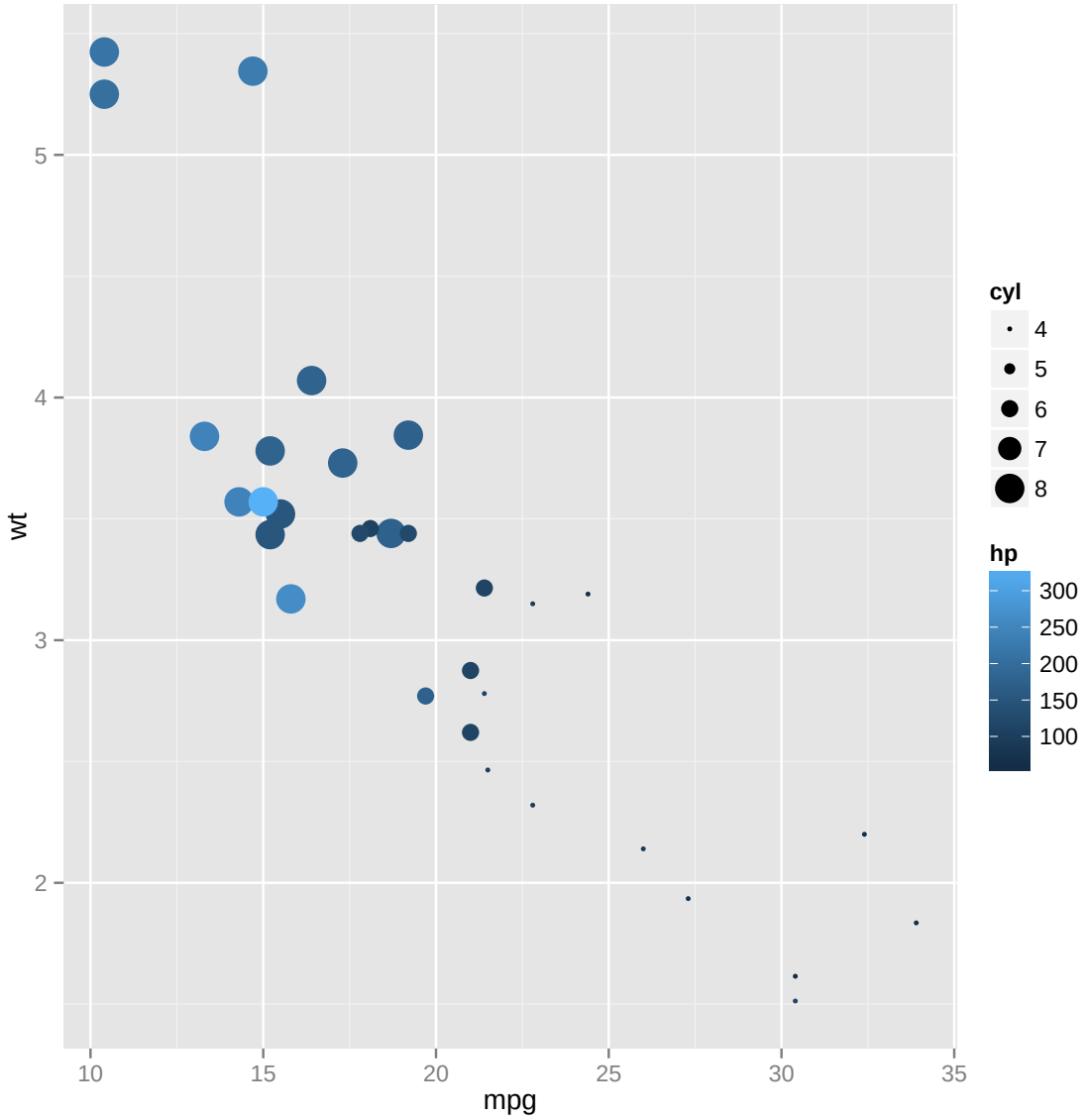

```
# 8 GRAFICO CON COLOR GRADIENTE
qplot(x=wt, y=mpg, color=cyl,data=mtcars)
```



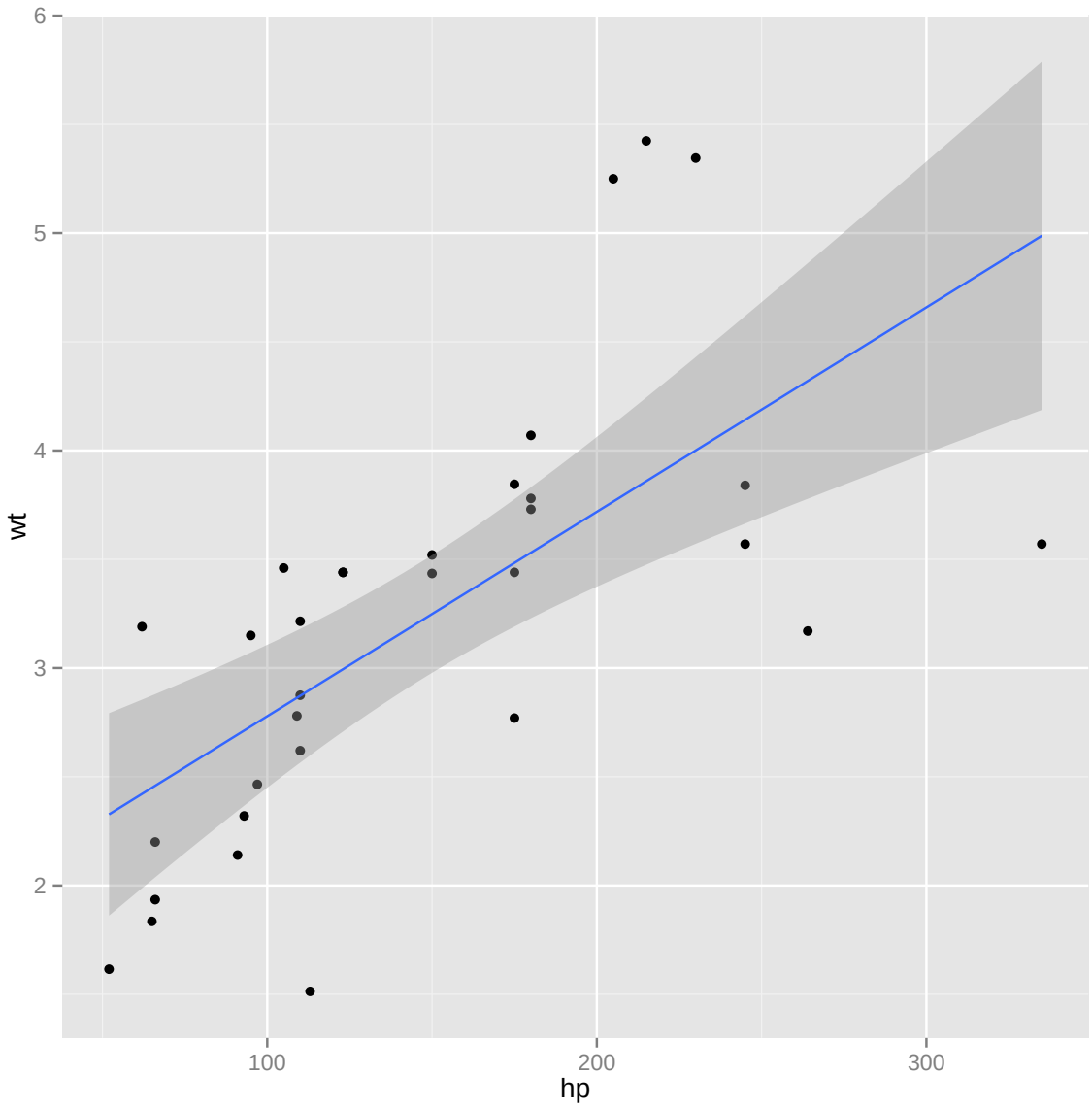

```
# 9 GRAFICO CON COLOR POR CLASE
qplot(x=wt, y=mpg, color=factor(cyl),data=mtcars)
```



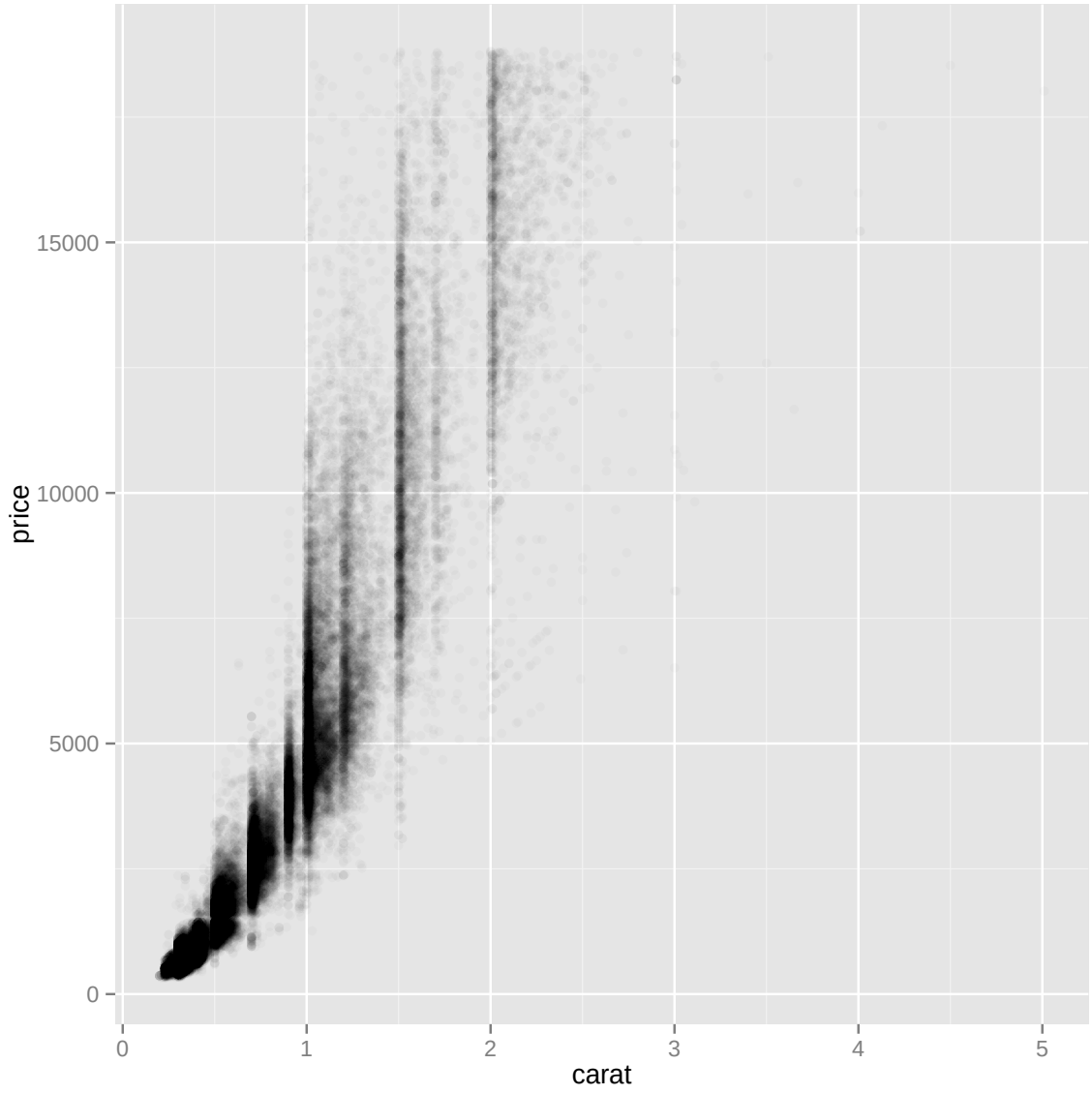

```
# 10 GRAFICO CON COLOR GRADIENTE + TAMA?O
qplot(x=mpg, y=wt, size=cyl,  colour=hp, data = mtcars)
```



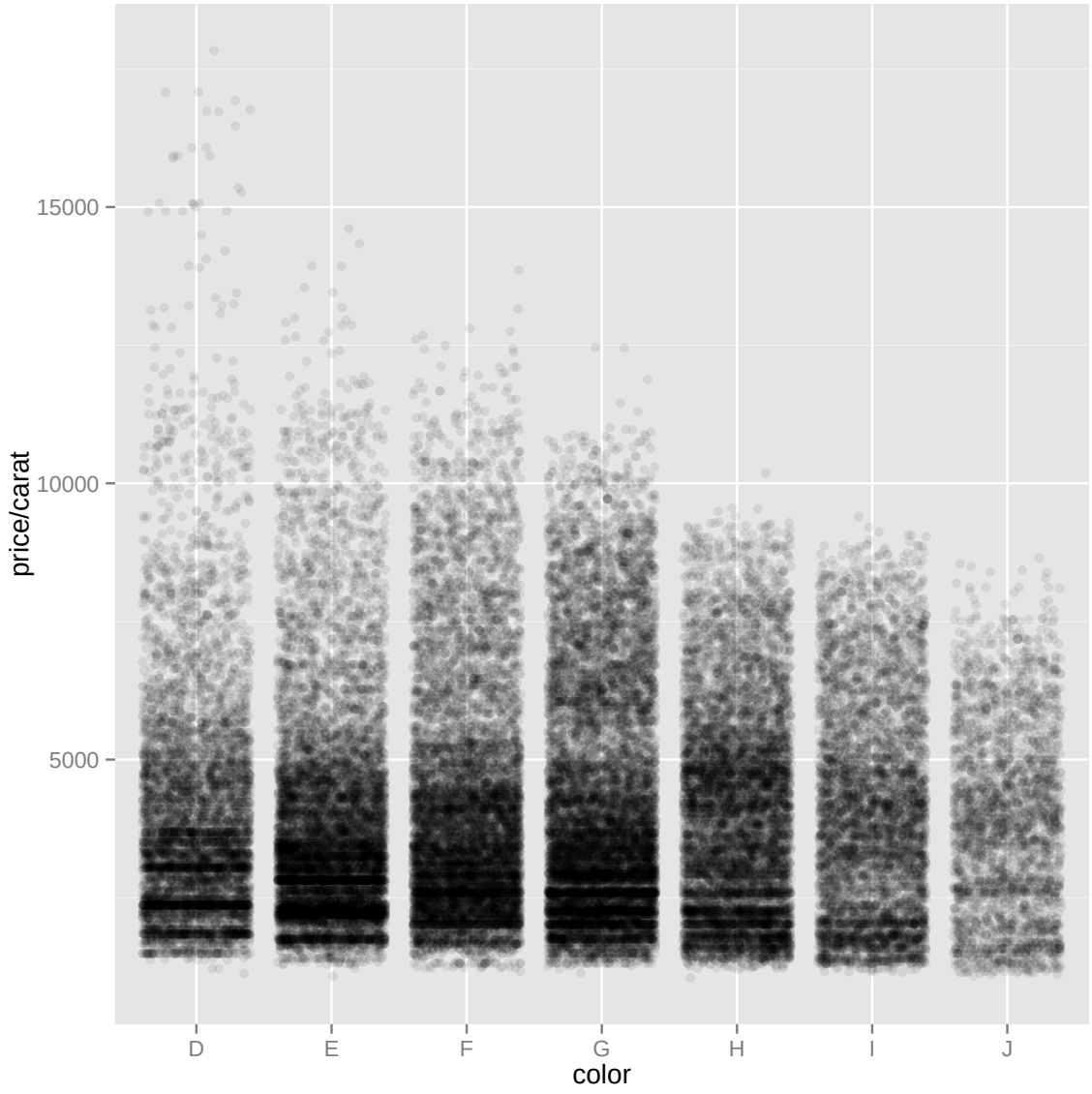

```
# 11 GRAFICO DISPERSION CON AREA DE ERROR DE LA LINEA DE REGRESION
qplot(x=hp, y=wt, data=mtcars, geom=c("point","smooth"), method="lm")
```



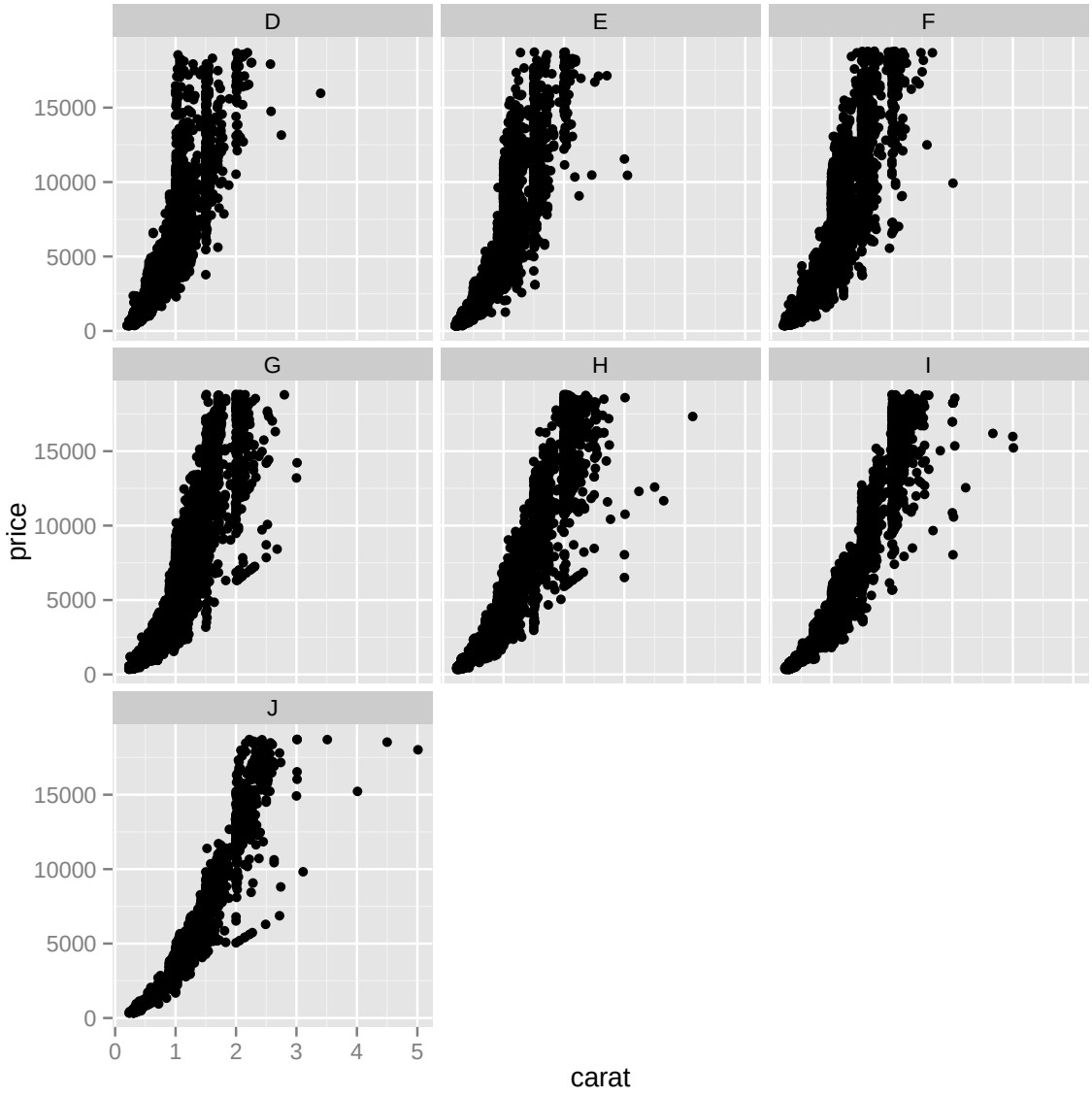

```
# 12 GRAFICO DISPERSION CON COLOR TRANSPARENTE EN LOS PUNTOS
qplot(x=carat,y=price, alpha=I(0.02), data = diamonds)
```



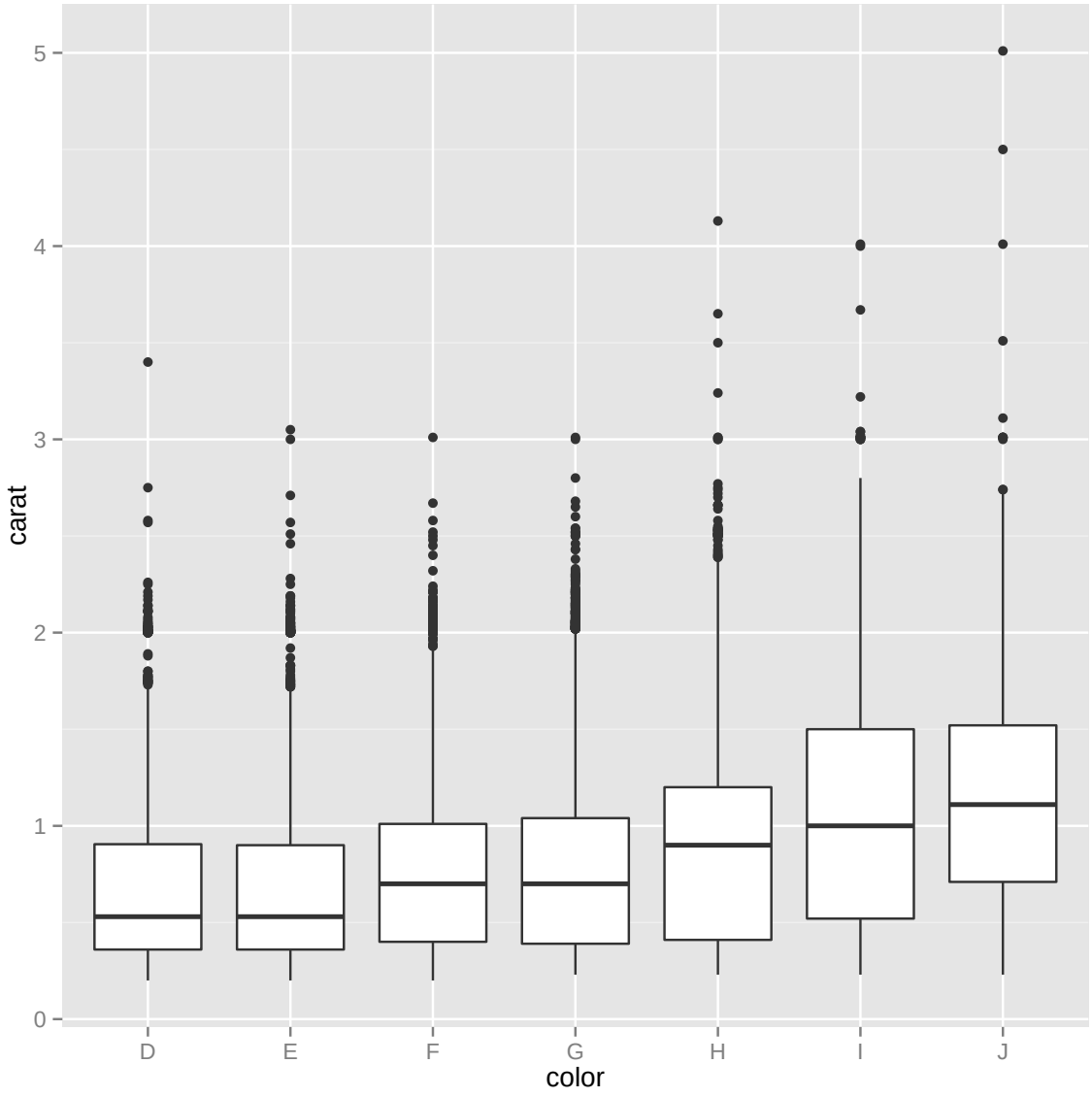

```
# 13 GRAFICO DE DISPERSION CON VARIABLE DISCRETA.
# JITTER DA UN VALOR ALEATORIO A ACA PUNTO DENTRO DE LA DIIMENSION
qplot(x=color, y=price/carat, data = diamonds, geom = "jitter",alpha = I(0.08))
```




```
# 14 GRAFICO DE DISPERSION SEGMENTADO
qplot(x=carat, y=price, facets = ~color, data = diamonds)
```



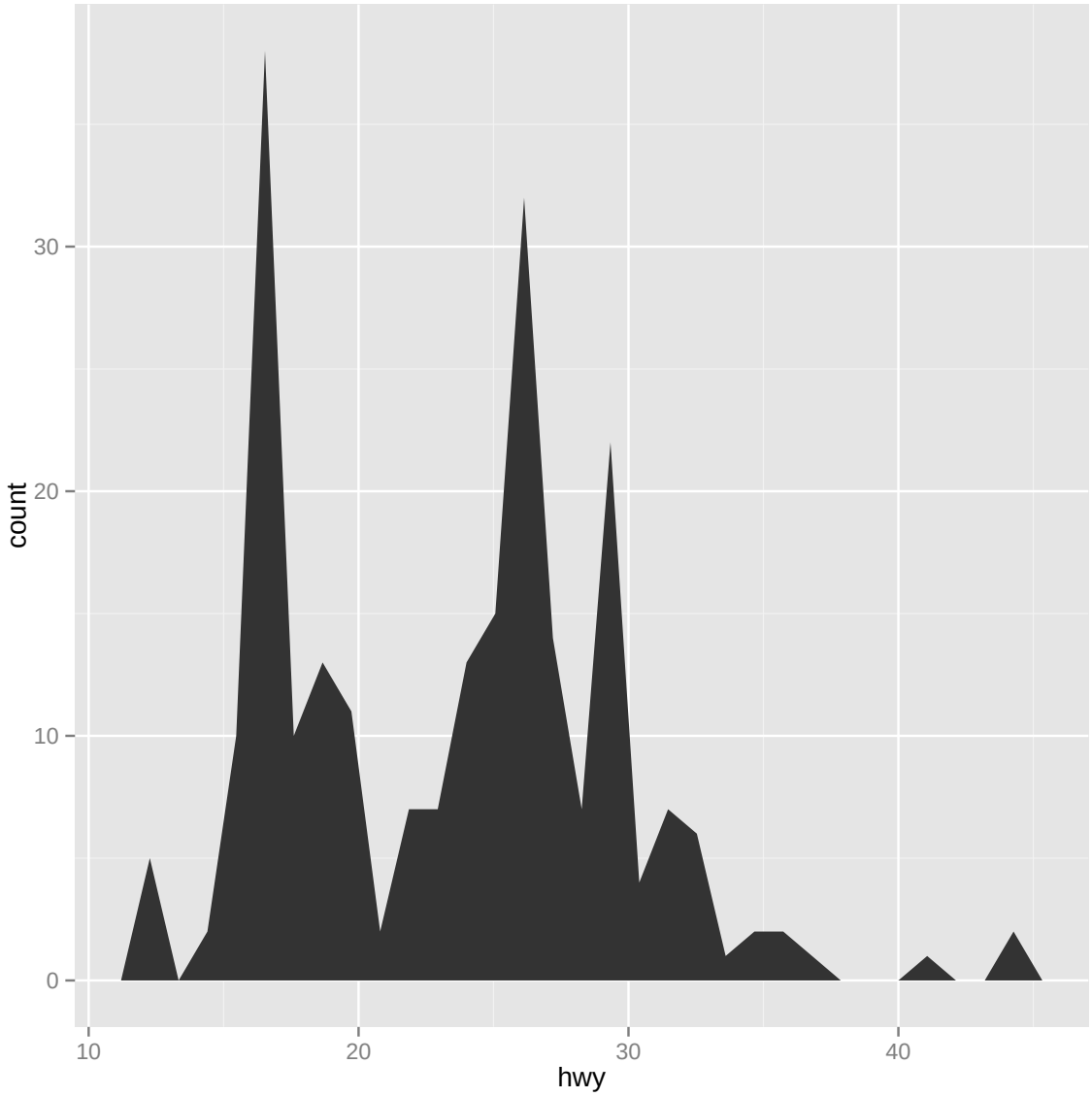

```
# 15 BOXPLOT
qplot(x=color, y=carat, data = diamonds, geom = "boxplot")
```



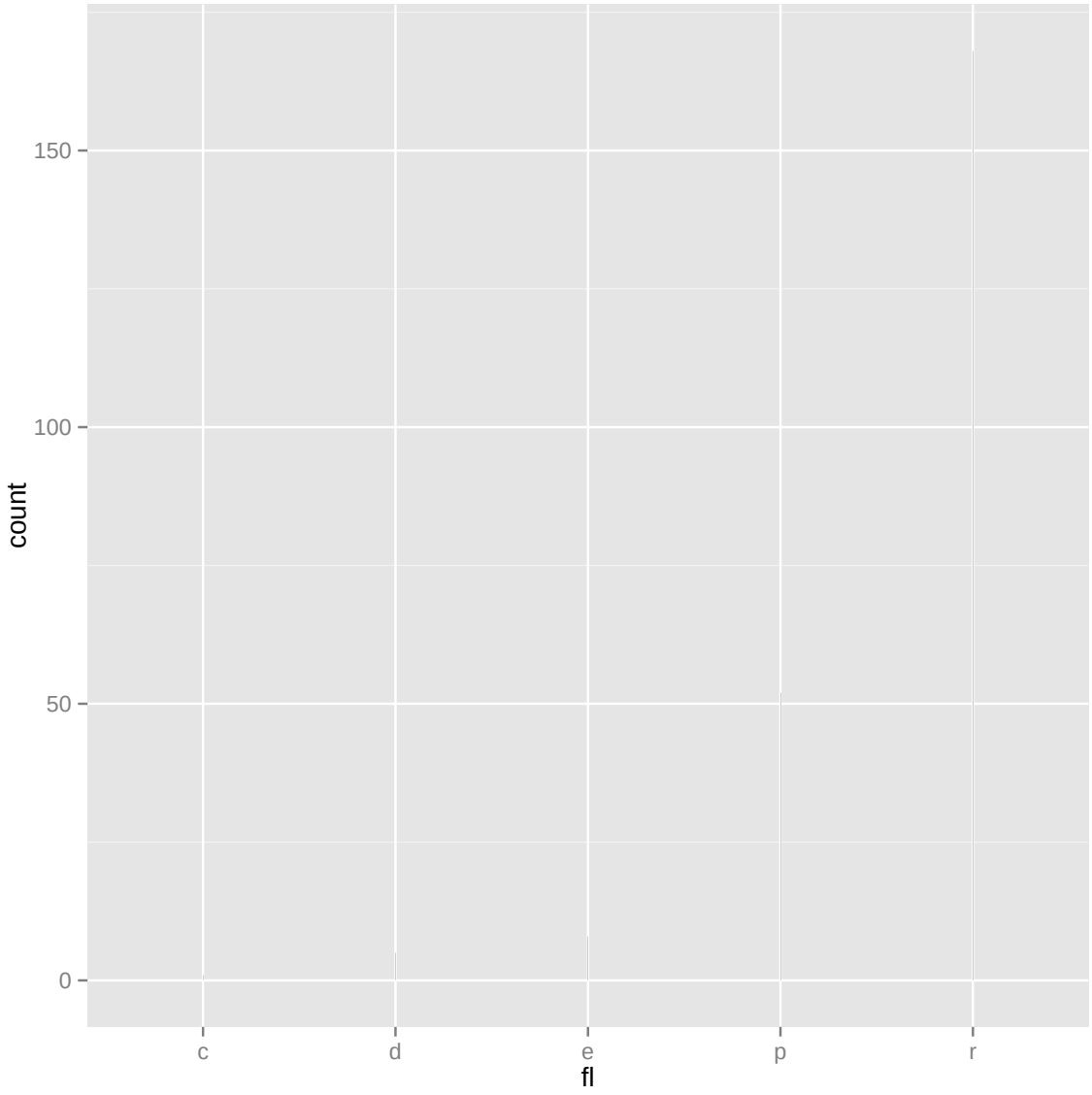
2.12. Ejemplo 4:

Geoms - Usa una geom para representar los datos, usa las propiedades estÃ©ticas de la geom para representar variables. Cada funciÃ³n devuelve una capa.

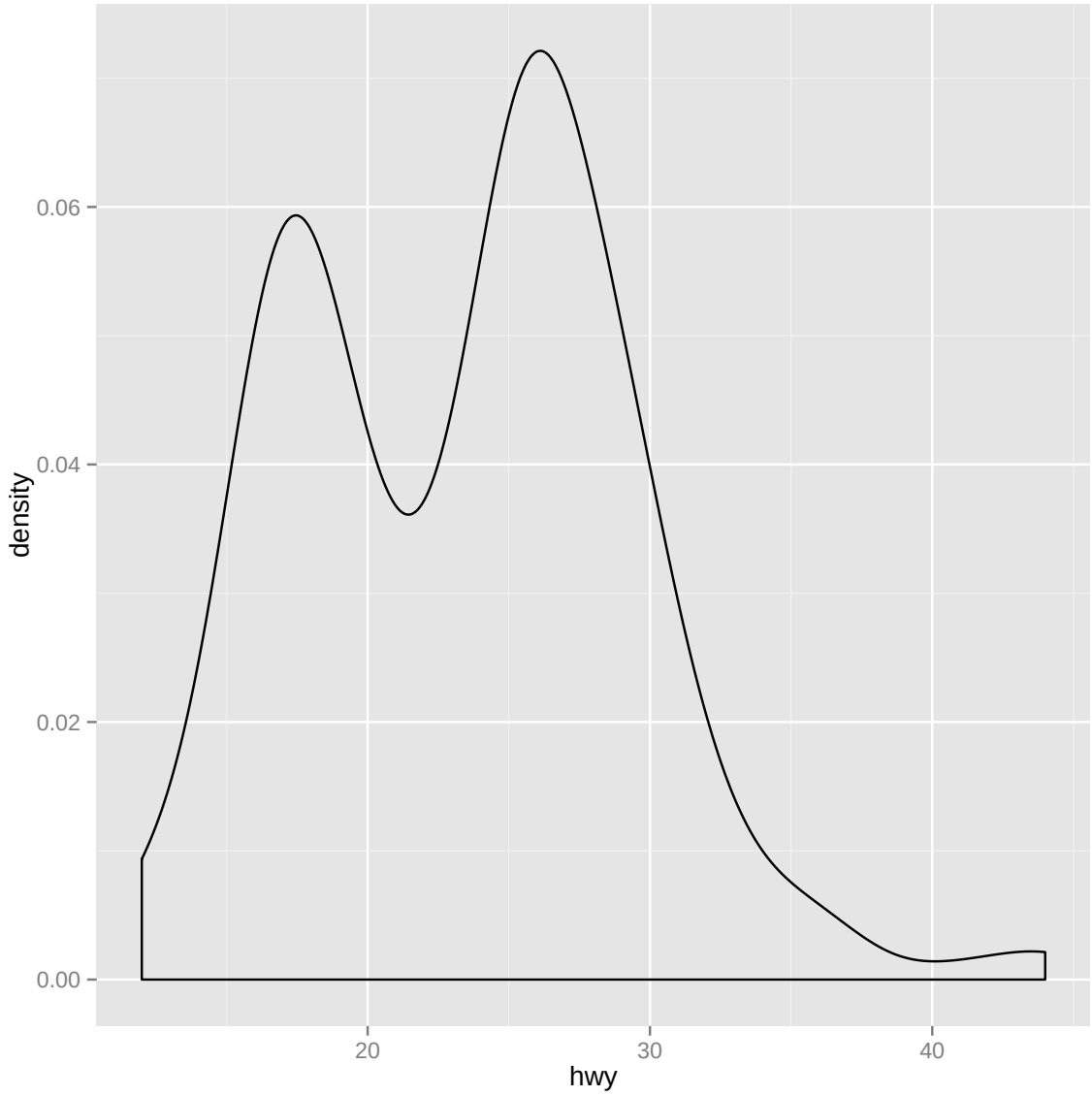
```
#########################################################################
# Una Variable
# ContÃnua

a <- ggplot(mpg, aes(hwy))
b<- ggplot(mpg, aes(fl))
a + geom_area(stat = "bin") # x, y, alpha, color, fill, linetype, size)

## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust
this.
```



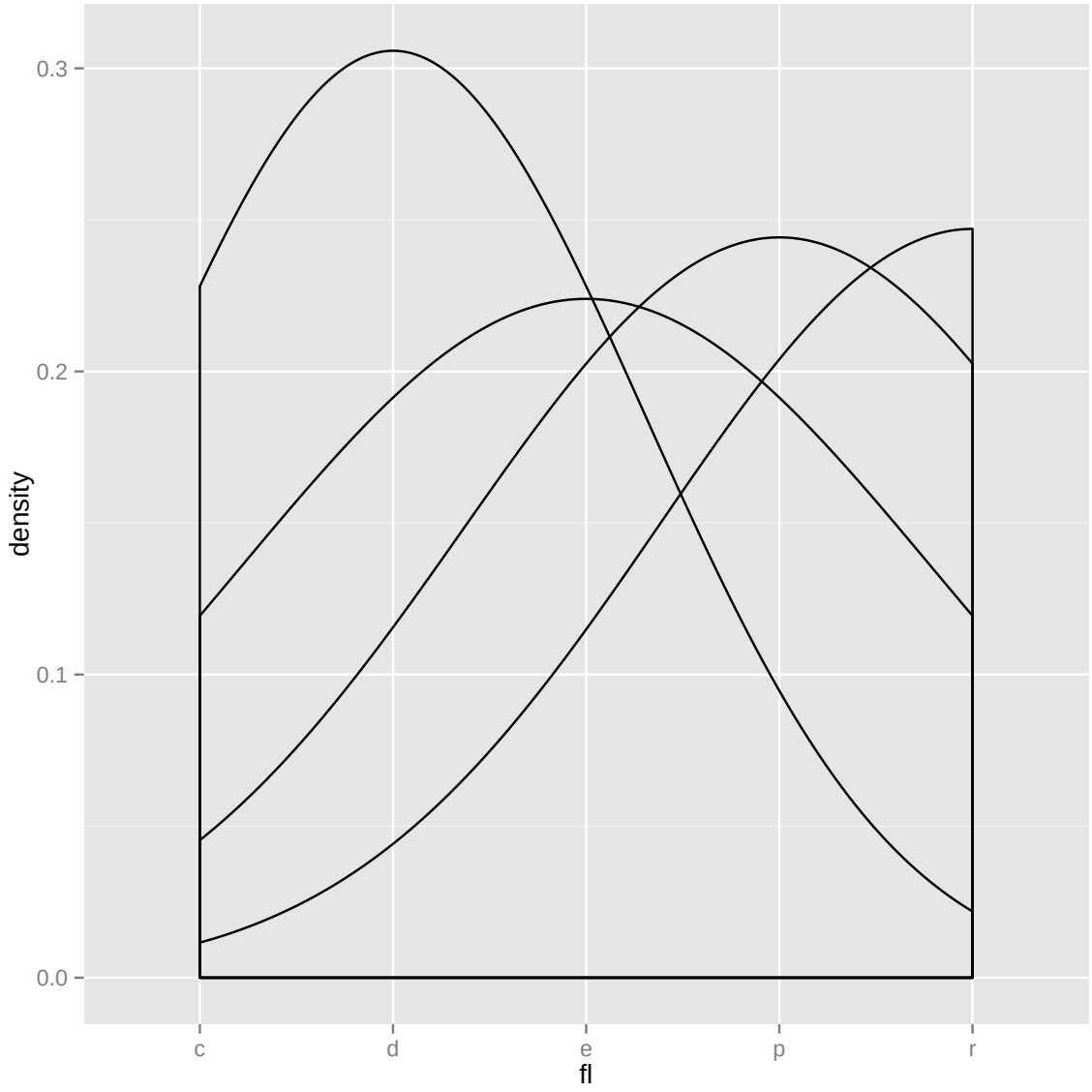

```
b<- ggplot(mpg, aes(fl))
b + geom_area(stat = "bin")
```



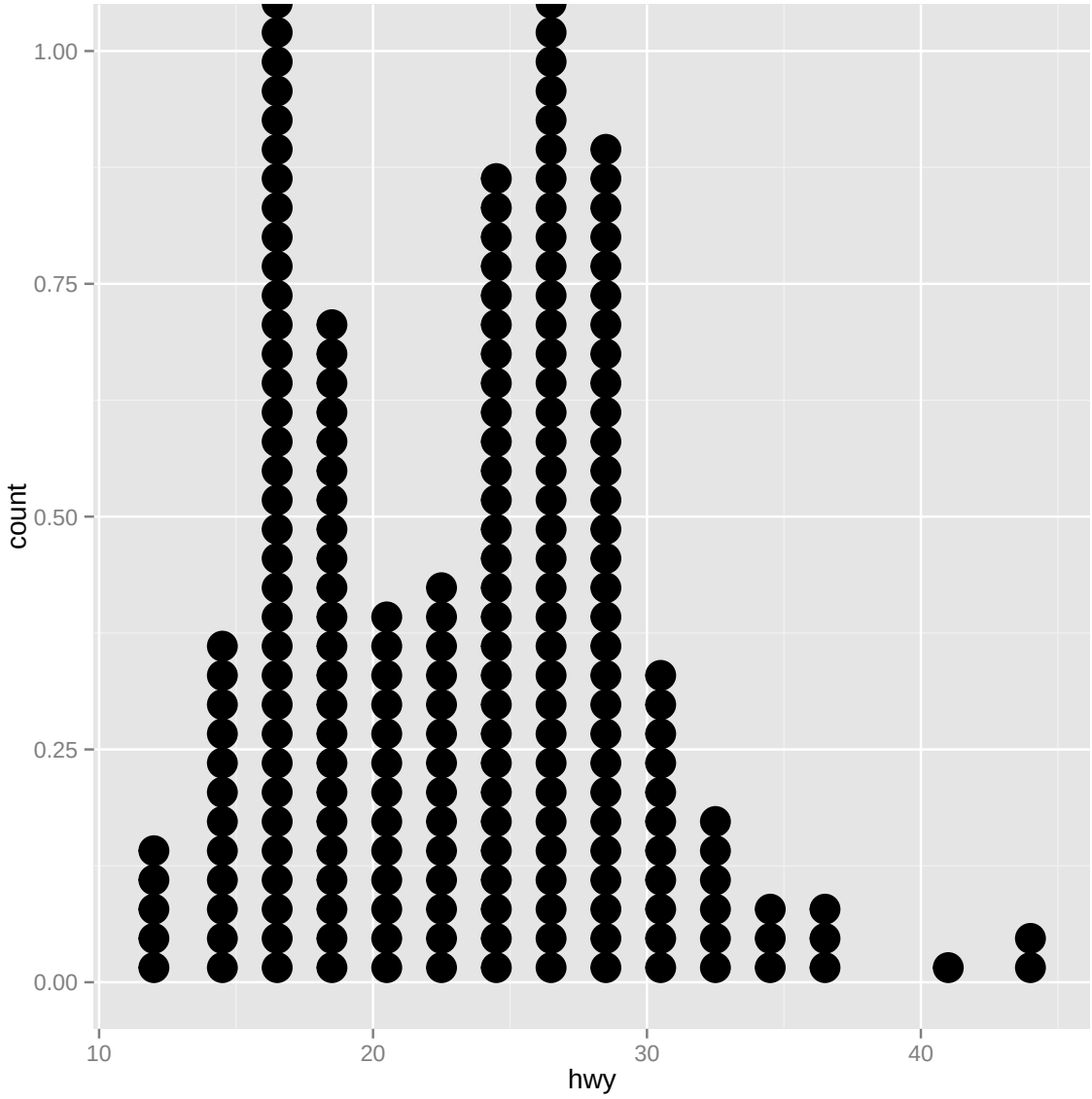

```
a + geom_density(kernel = "gaussian") # x, y, alpha, color, fill, linetype, size
```



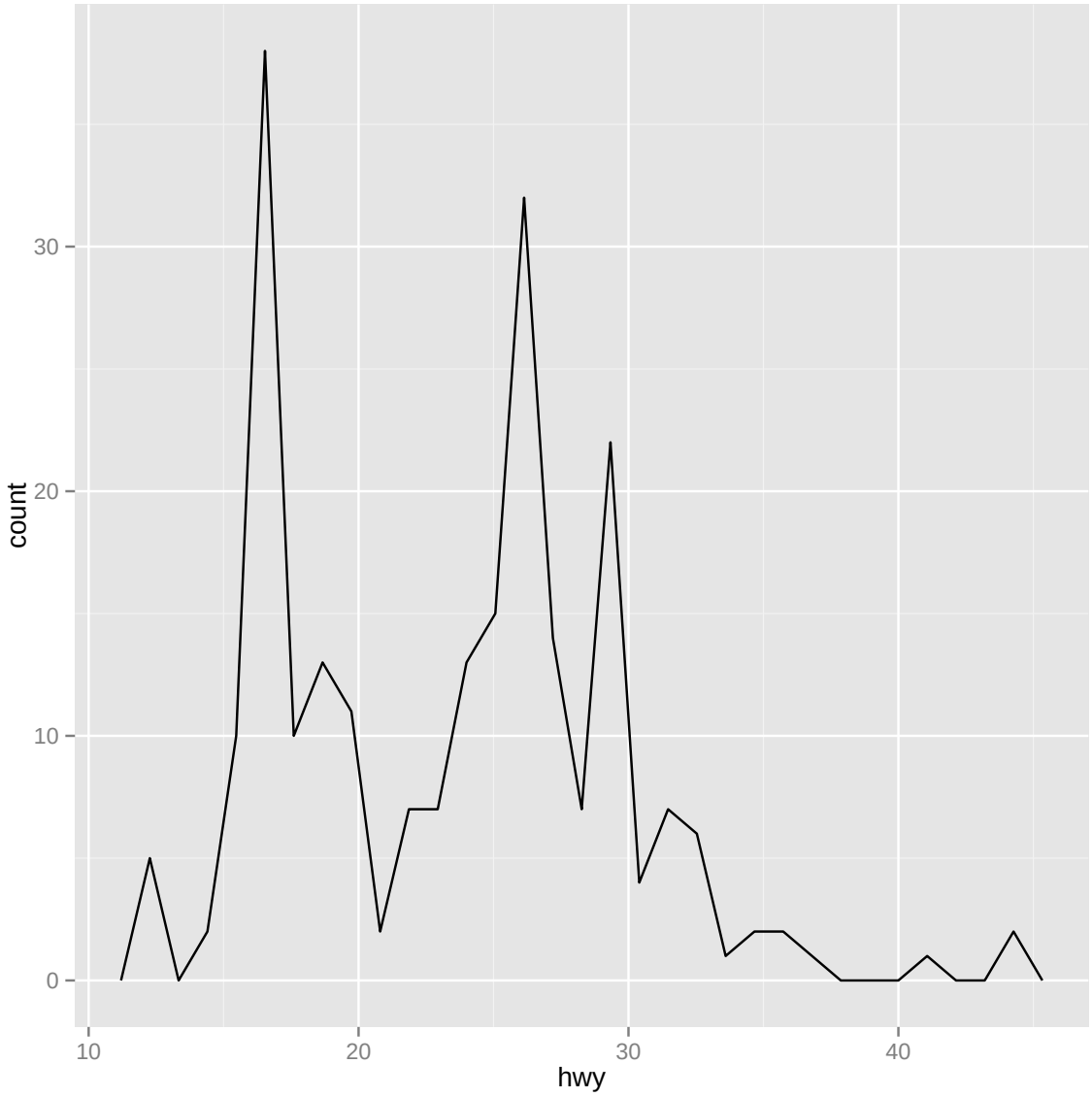

```
b + geom_density(kernel="gaussian")
```




```
a + geom_dotplot() # x, y, alpha, color, fill)

## stat_bindot: binwidth defaulted to range/30. Use 'binwidth = x' to adjust
this.
```

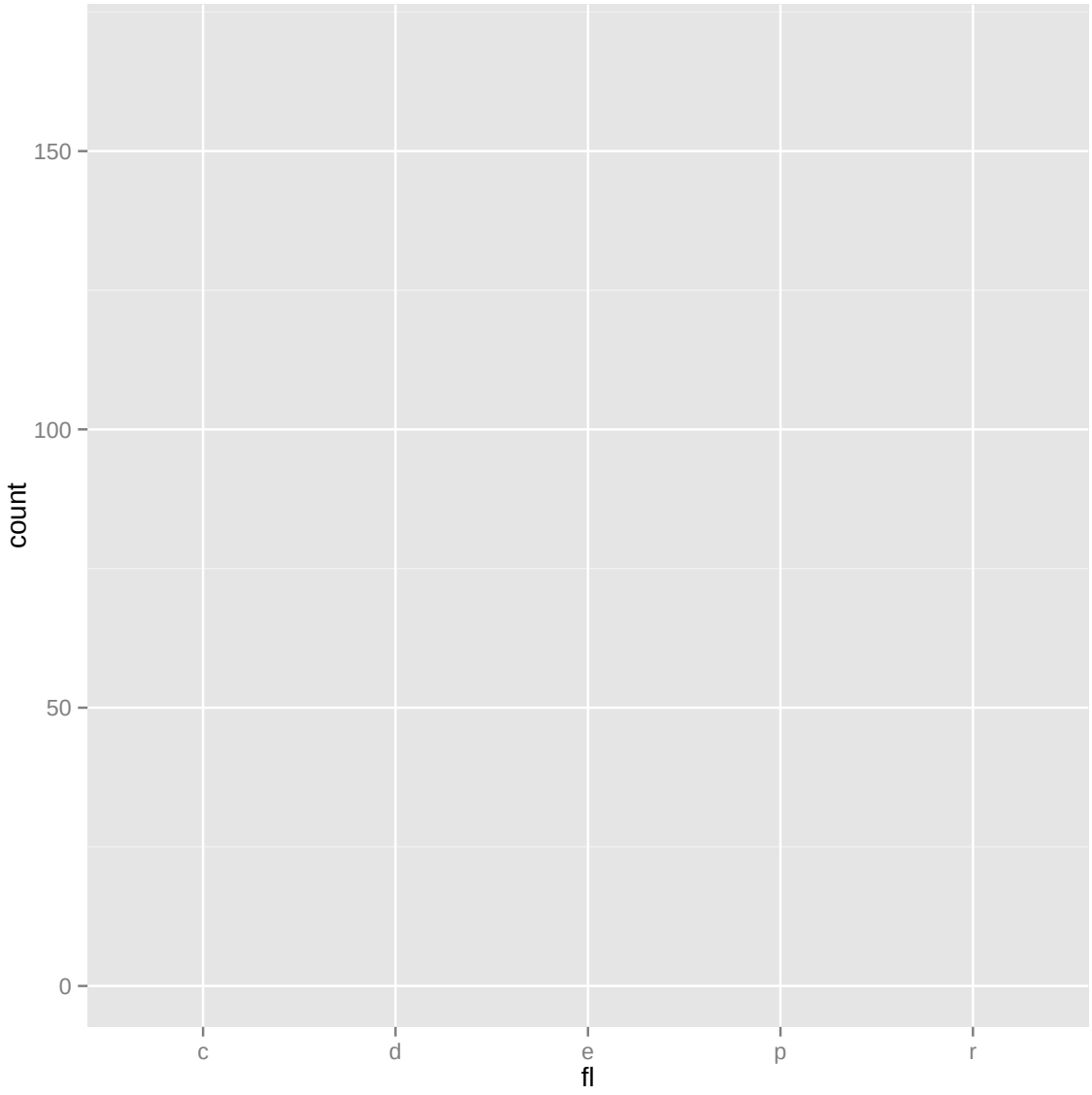



```
a + geom_freqpoly() # x, y, alpha, color, linetype, size)

## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust
this.
```

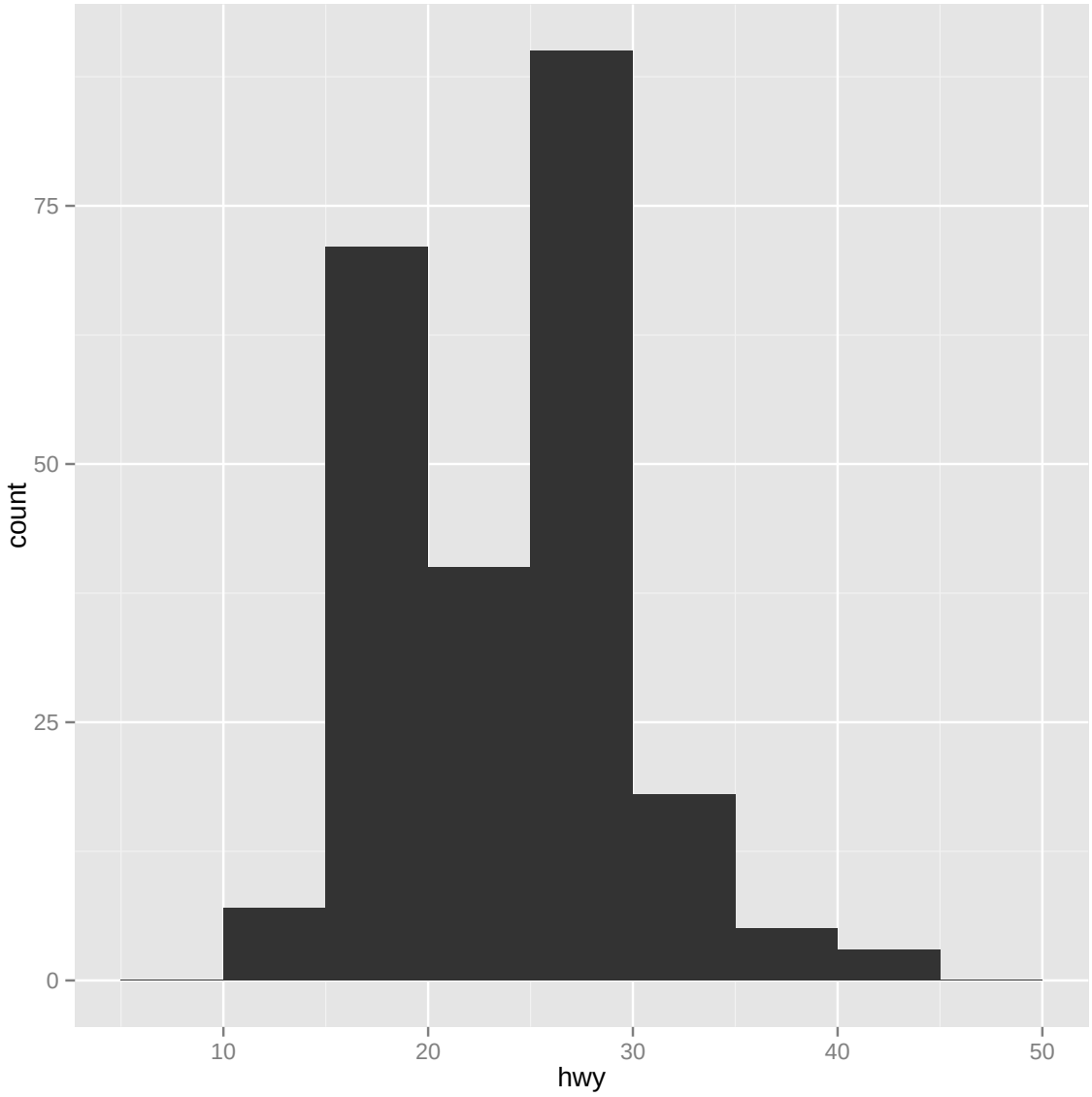



```
b + geom_freqpoly()
```

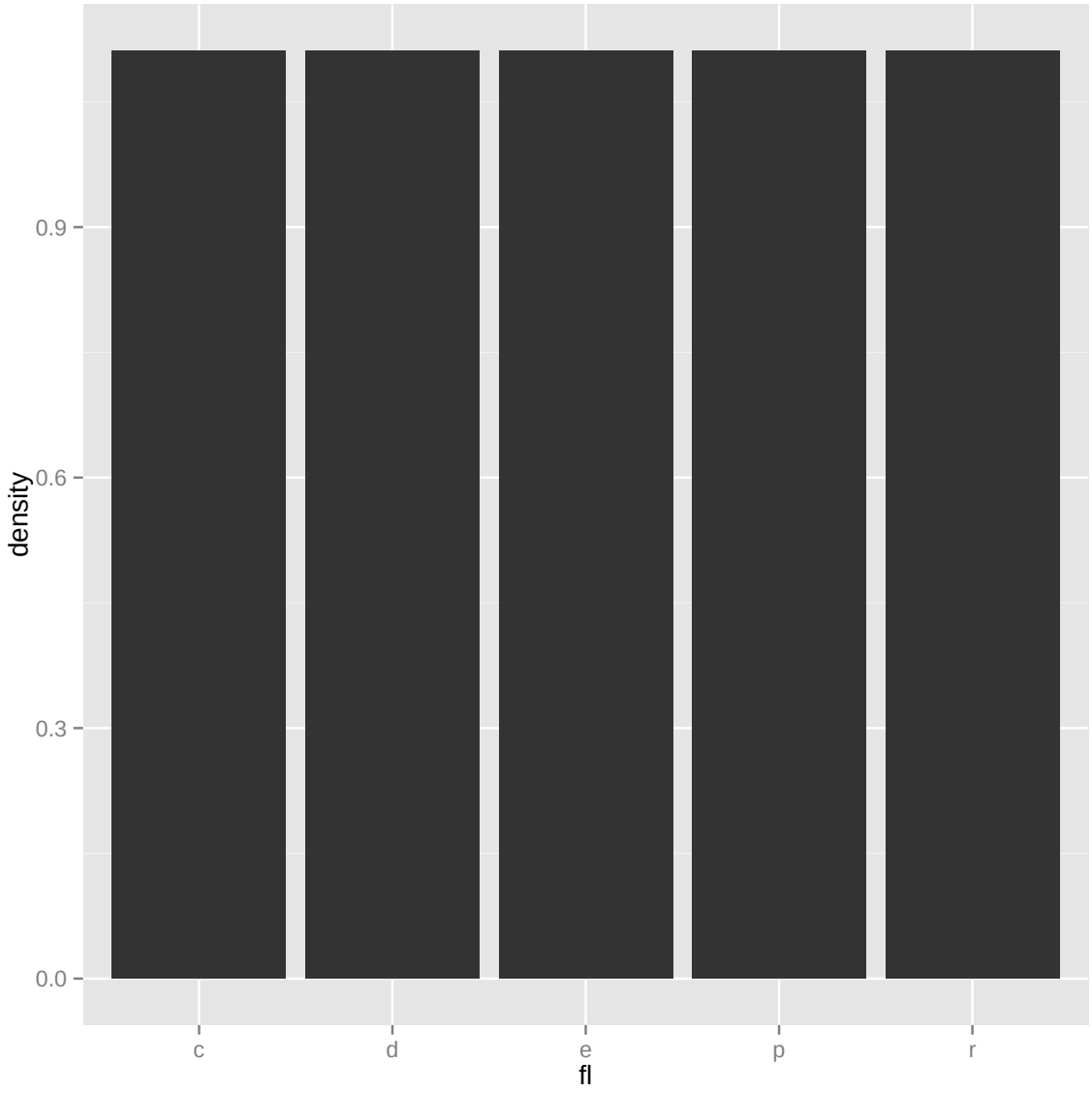
```
## geom_path: Each group consist of only one observation. Do you need to
adjust the group aesthetic?
## geom_path: Each group consist of only one observation. Do you need to
adjust the group aesthetic?
## geom_path: Each group consist of only one observation. Do you need to
adjust the group aesthetic?
## geom_path: Each group consist of only one observation. Do you need to
adjust the group aesthetic?
## geom_path: Each group consist of only one observation. Do you need to
adjust the group aesthetic?
```



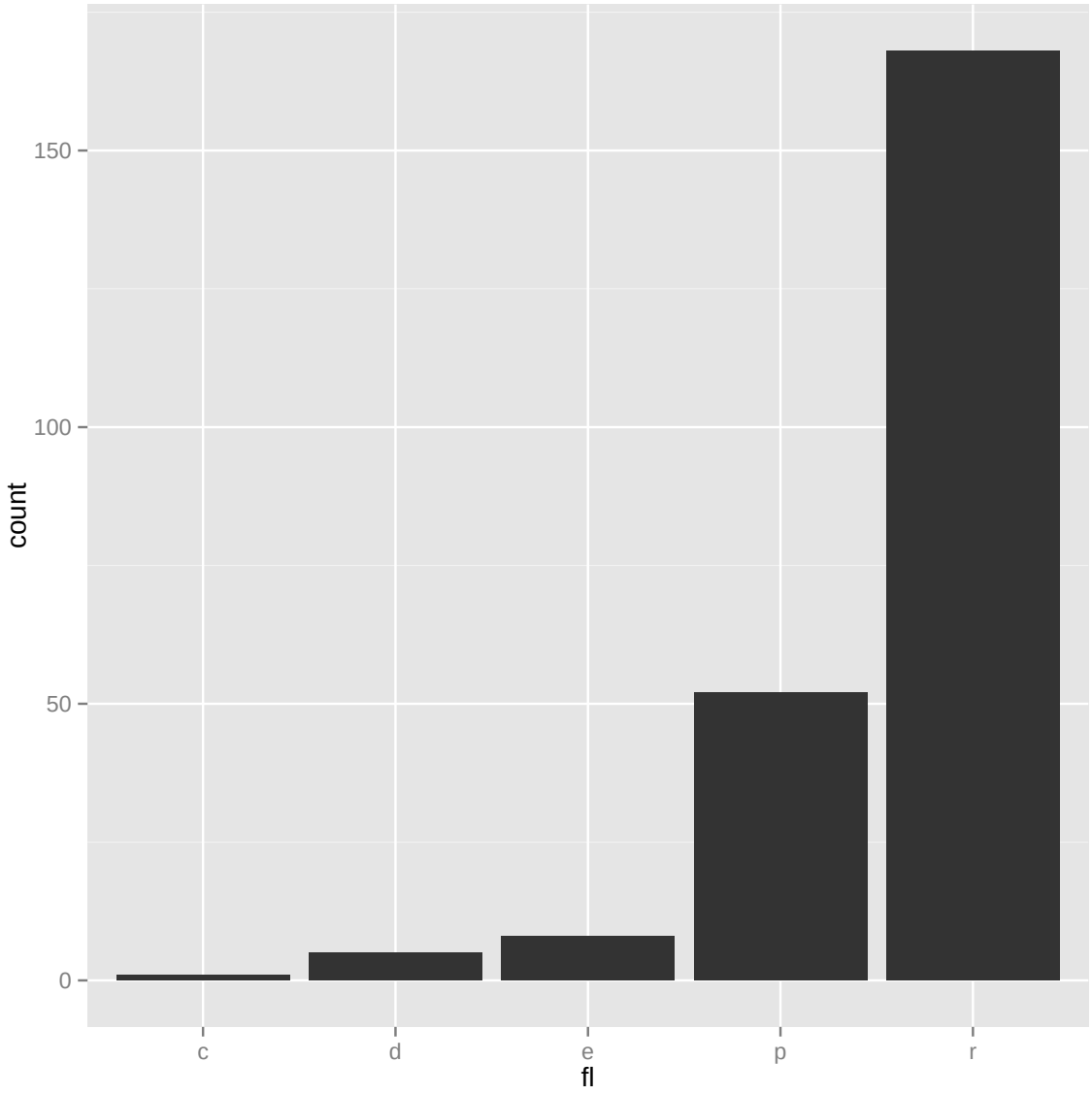

```
a + geom_histogram(binwidth = 5) # x, y, alpha, color, fill, linetype, size, wei
```



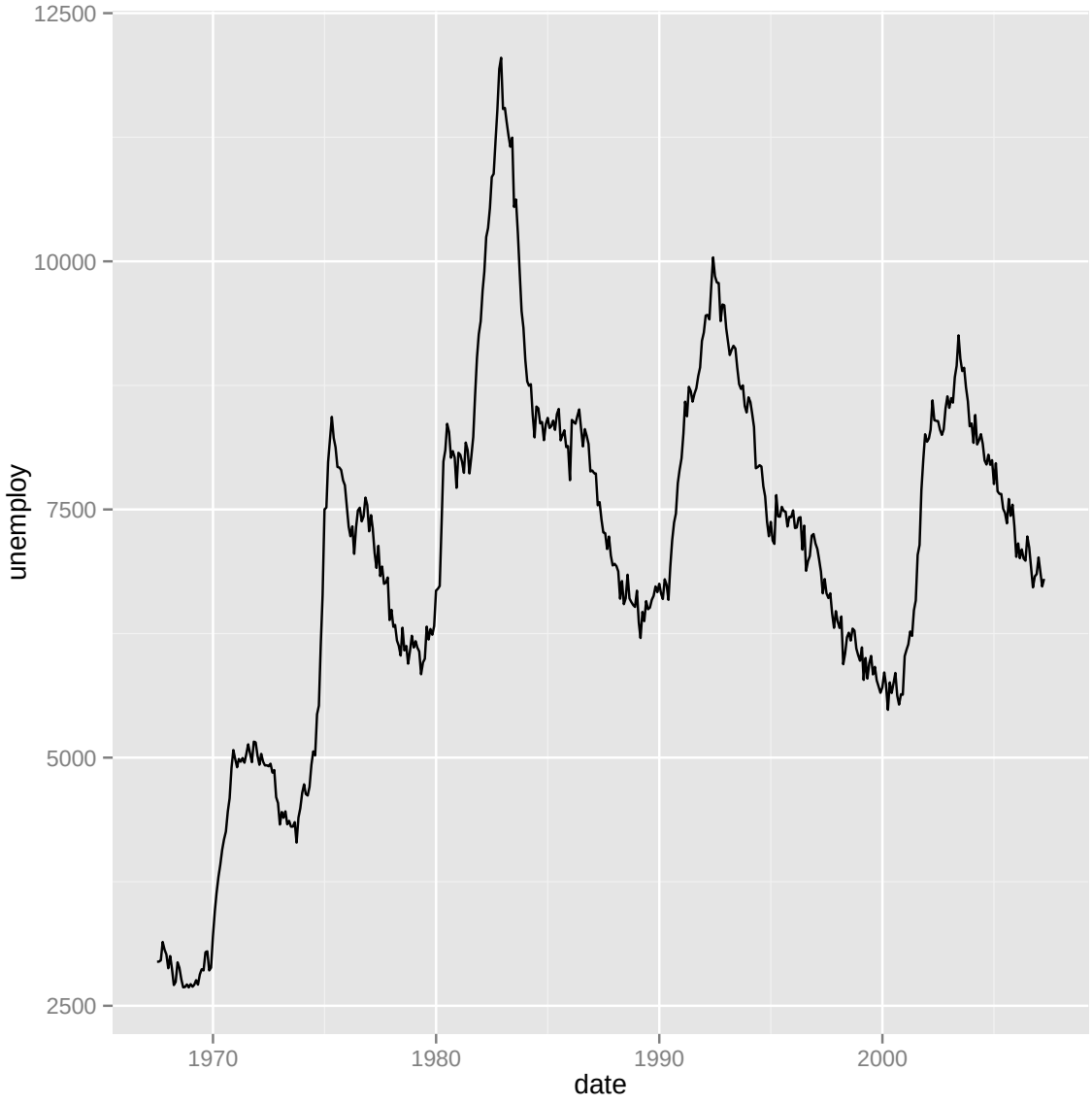

```
b + geom_histogram(aes(y = ..density..))
```



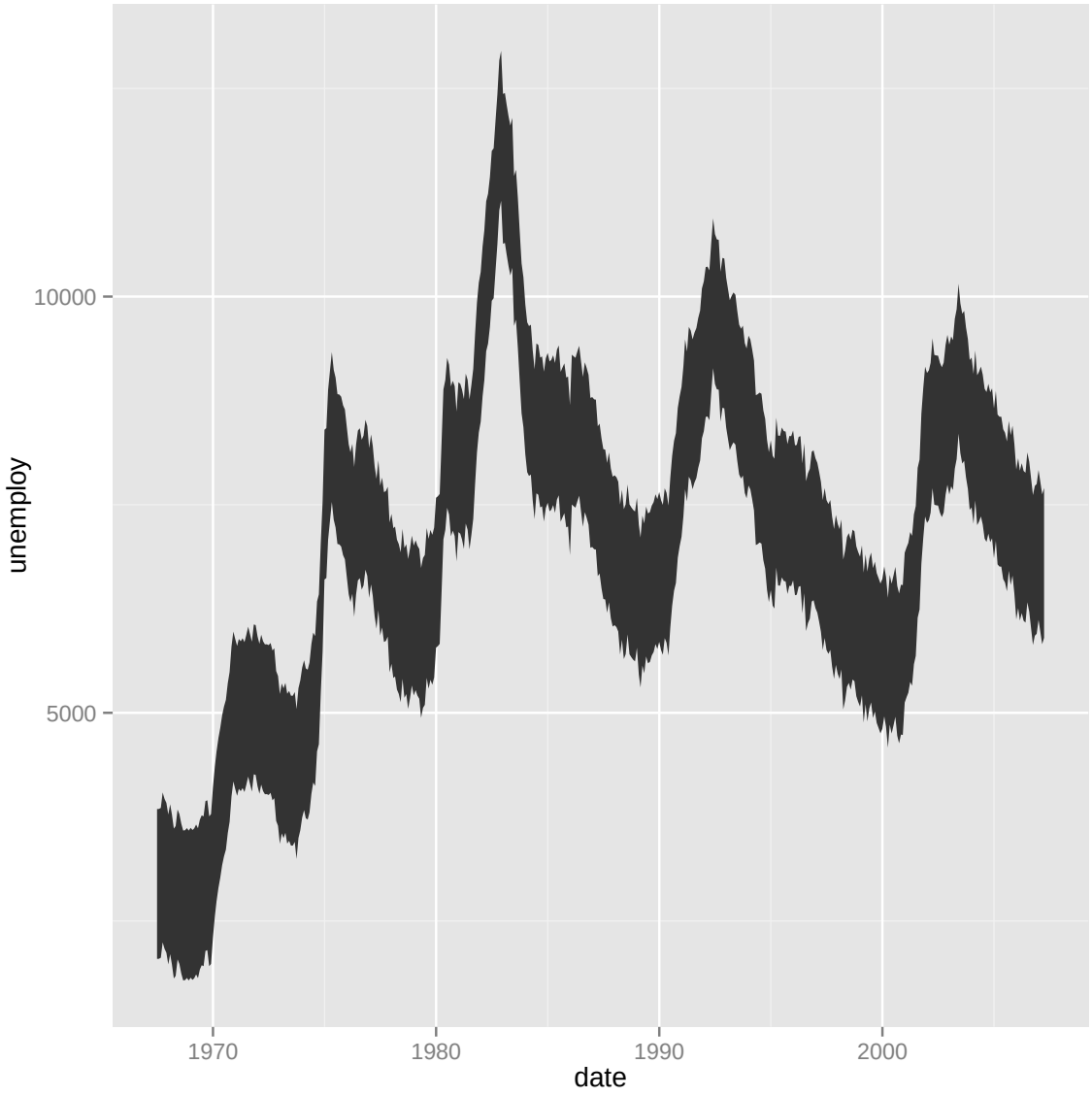

```r
# Discreta
b <- ggplot(mpg, aes(fl))
b + geom_bar() # x, alpha, color, fill, linetype, size, weight)
```



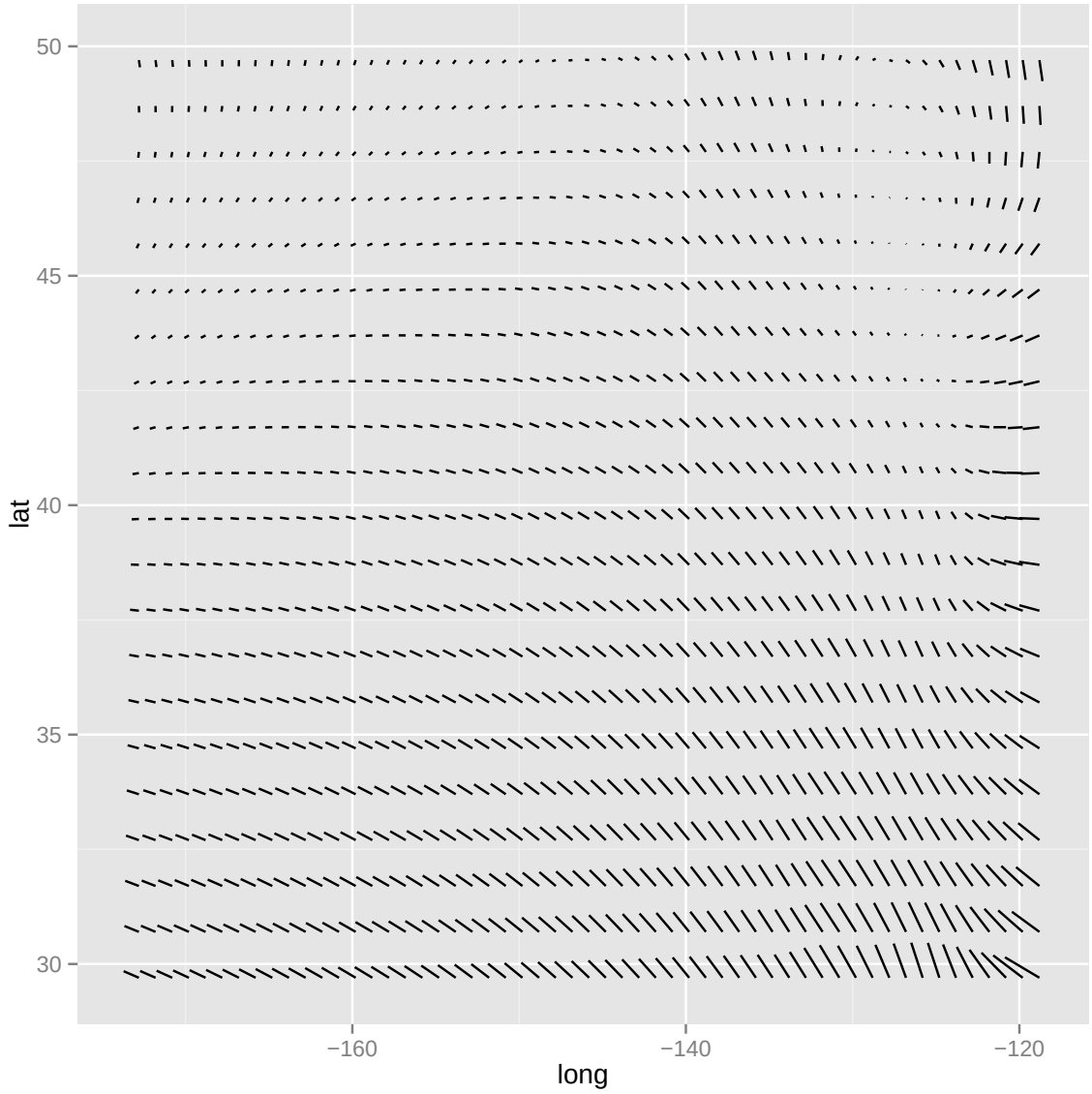

```
##########################################################################

##########################################################################
# Primitivas Graficas
c <- ggplot(map, aes(long, lat))

## Error in ggplot(map, aes(long, lat)): objeto 'map' no encontrado

c + geom_polygon(aes(group = group)) # x, y, alpha, color, fill, linetype, size

## Error in c + geom_polygon(aes(group = group)): argumento no-numÃ©rico
para operador binario

d <- ggplot(economics, aes(date, unemploy))
d + geom_path(lineend="butt", linejoin="round", linemitre=1)
```



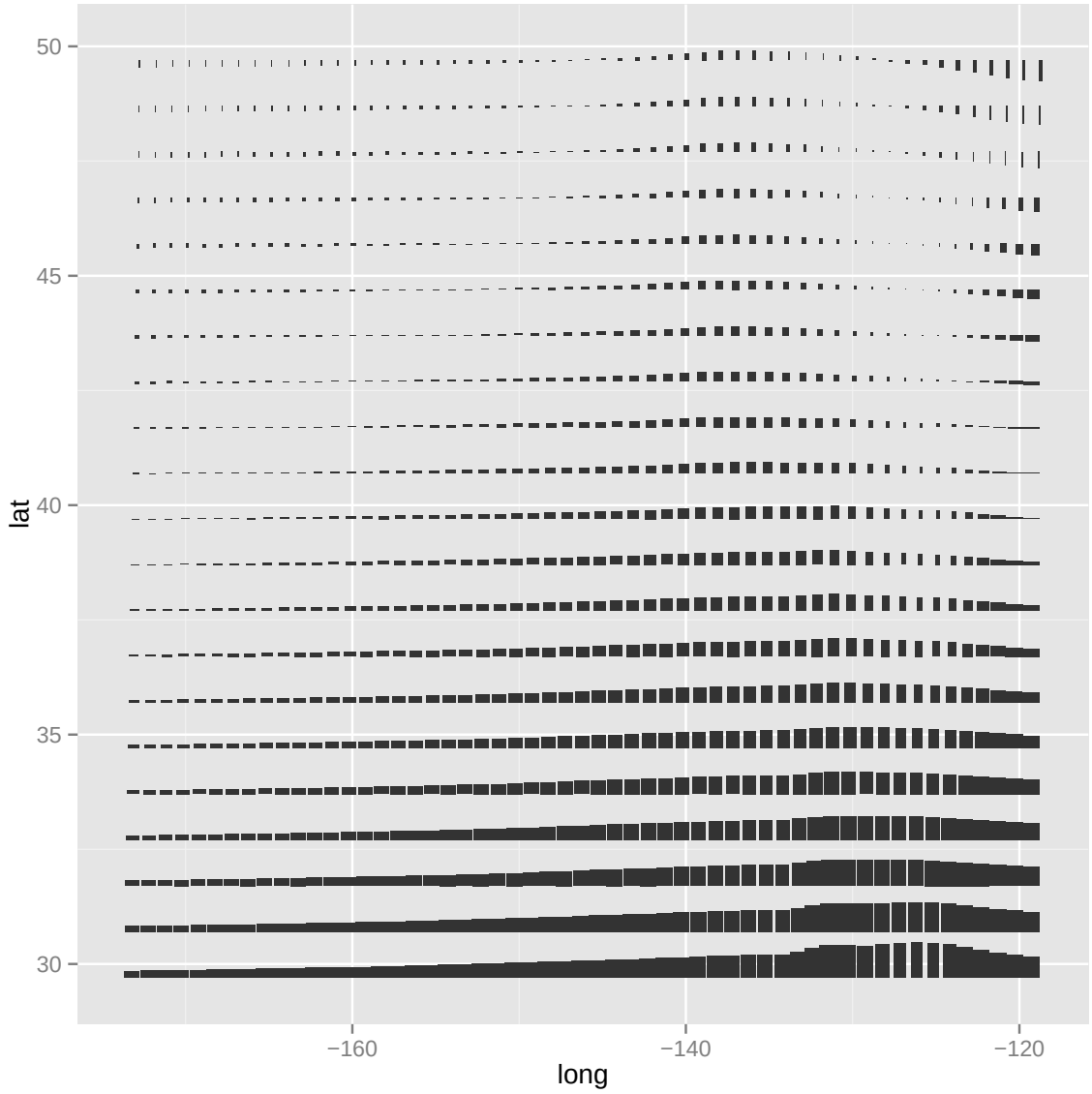

```
# x, y, alpha, color, linetype, size
d + geom_ribbon(aes(ymin=unemploy - 900, ymax=unemploy + 900))
```



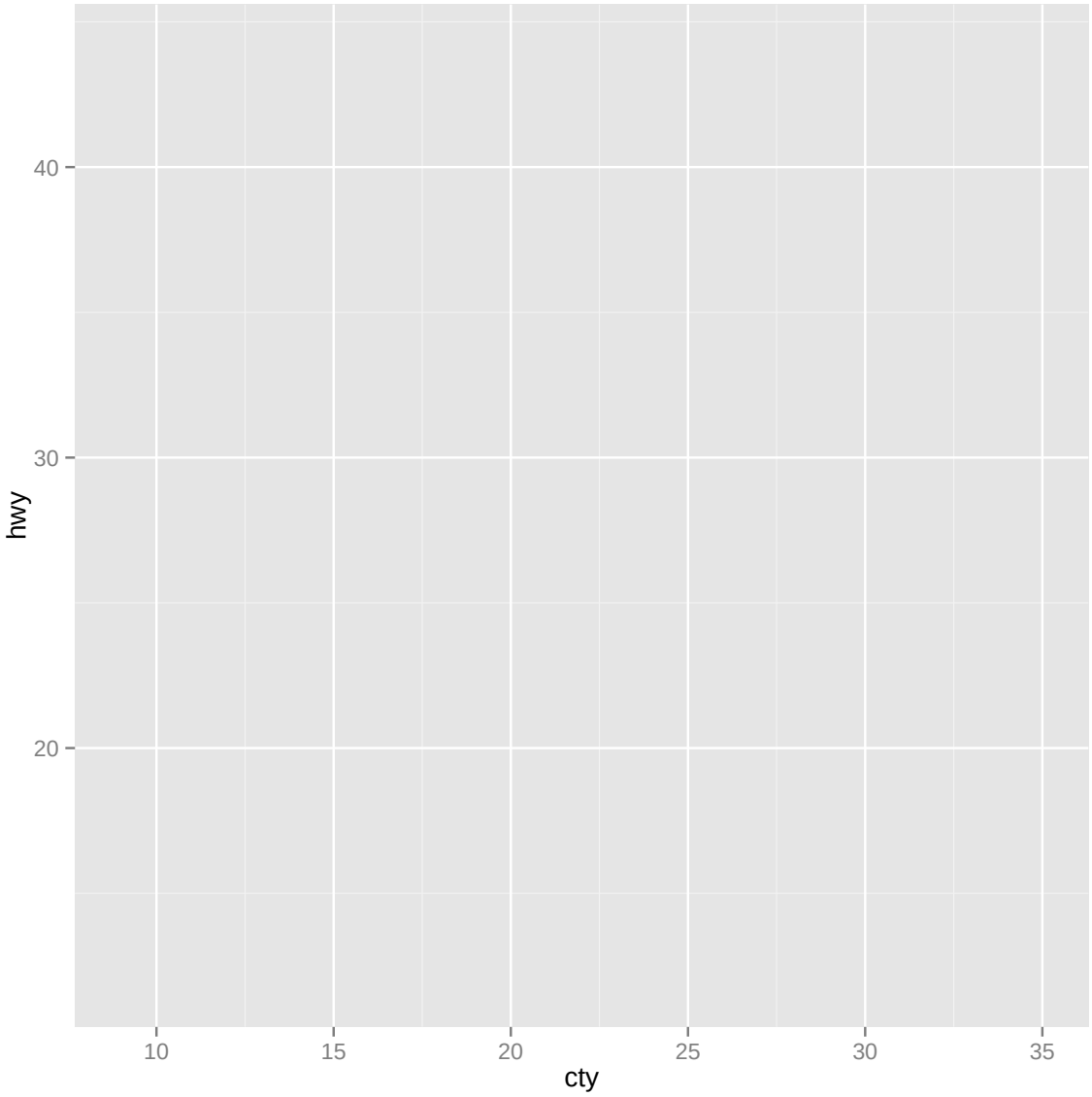

```
#x, ymax, ymin, alpha, color, fill, linetype, siz

e <- ggplot(seals, aes(x = long, y = lat))
e + geom_segment(aes(xend = long + delta_long, yend = lat + delta_lat))
```



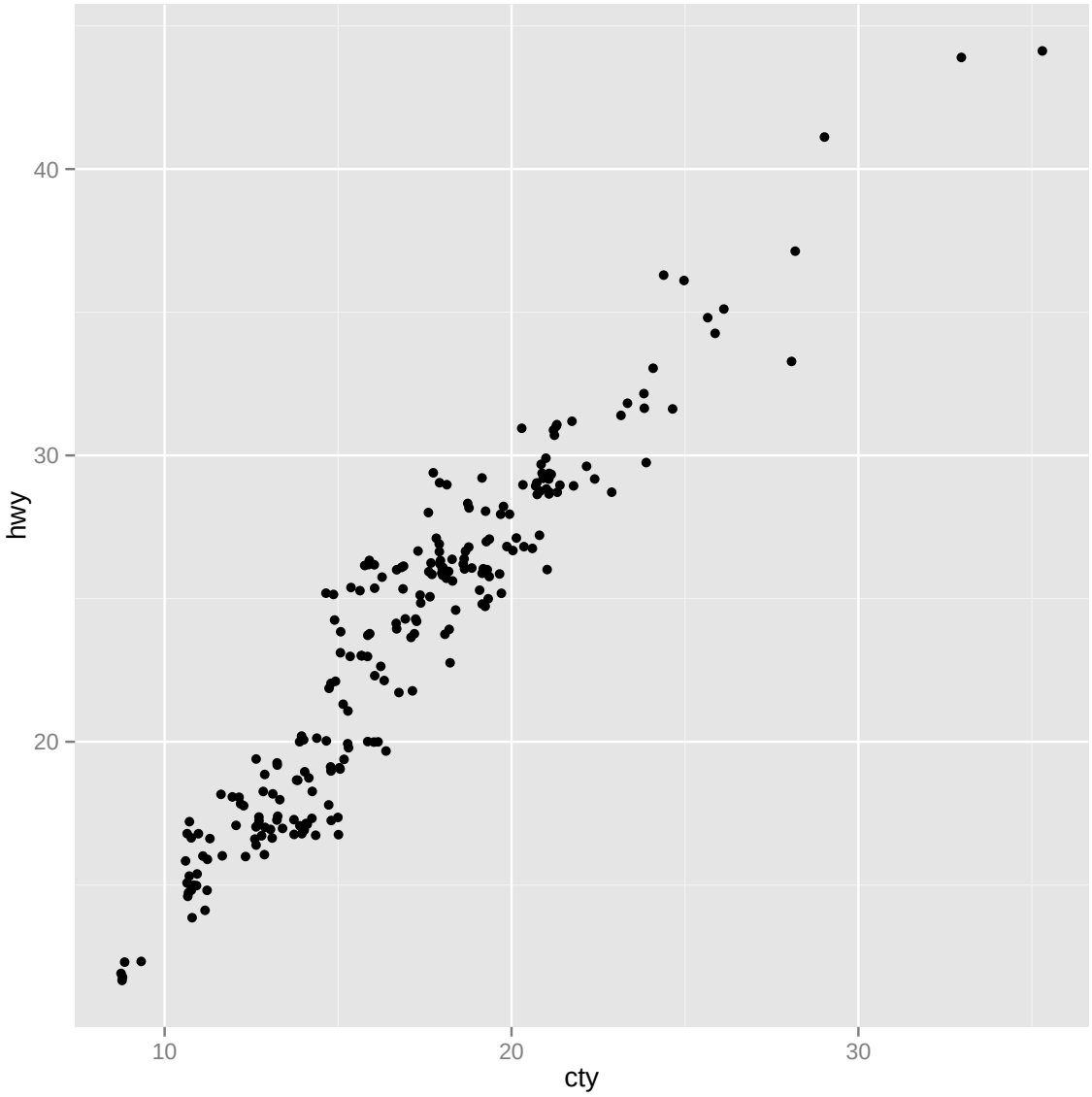

```
e + geom_rect(aes(xmin = long, ymin = lat, xmax= long + delta_long, ymax = lat +
```



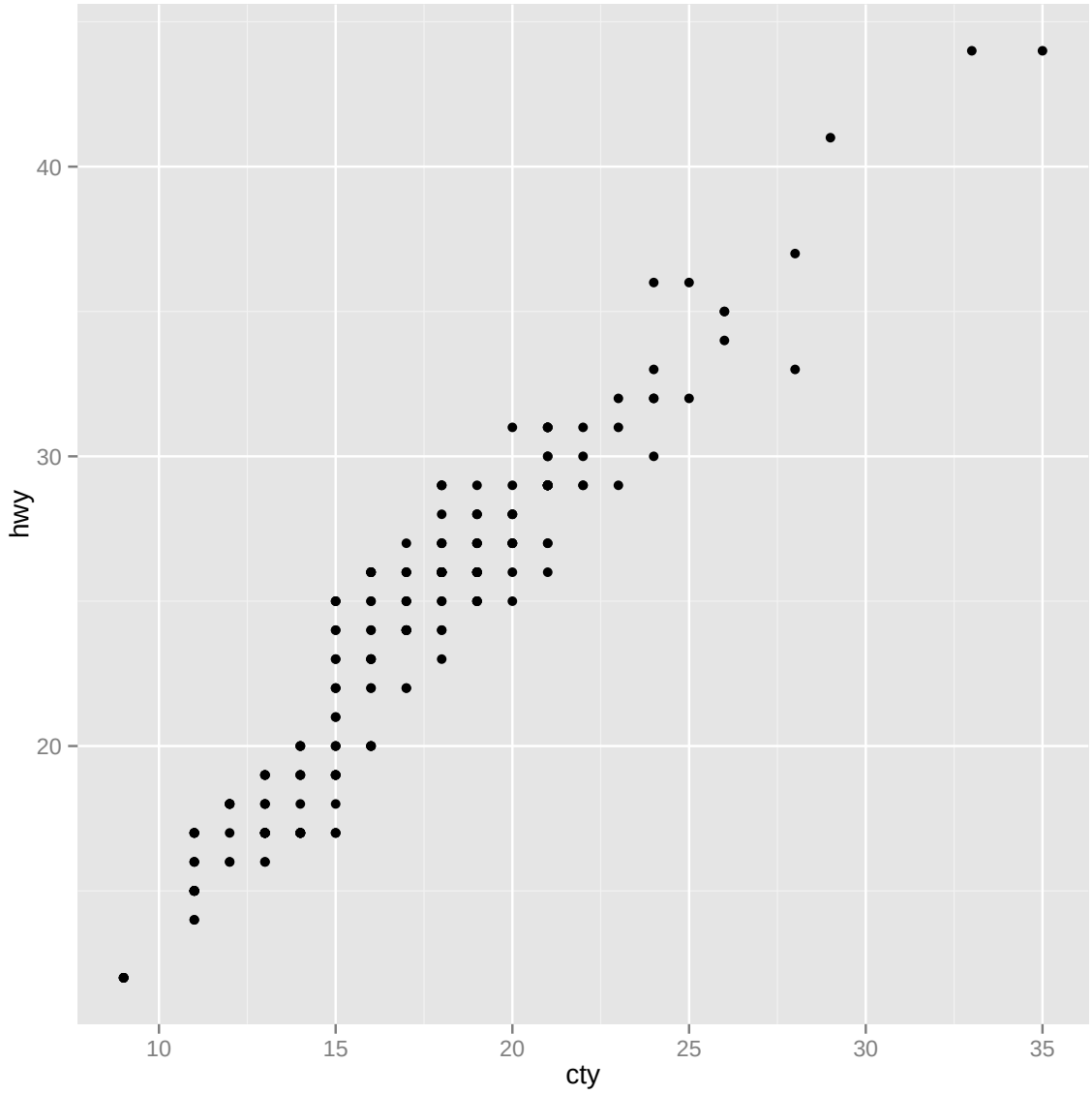

```
##########################################################################

##########################################################################
# Dos variables
#Continua X, Continua Y

f <- ggplot(mpg, aes(cty, hwy))
f + geom_blank()
```



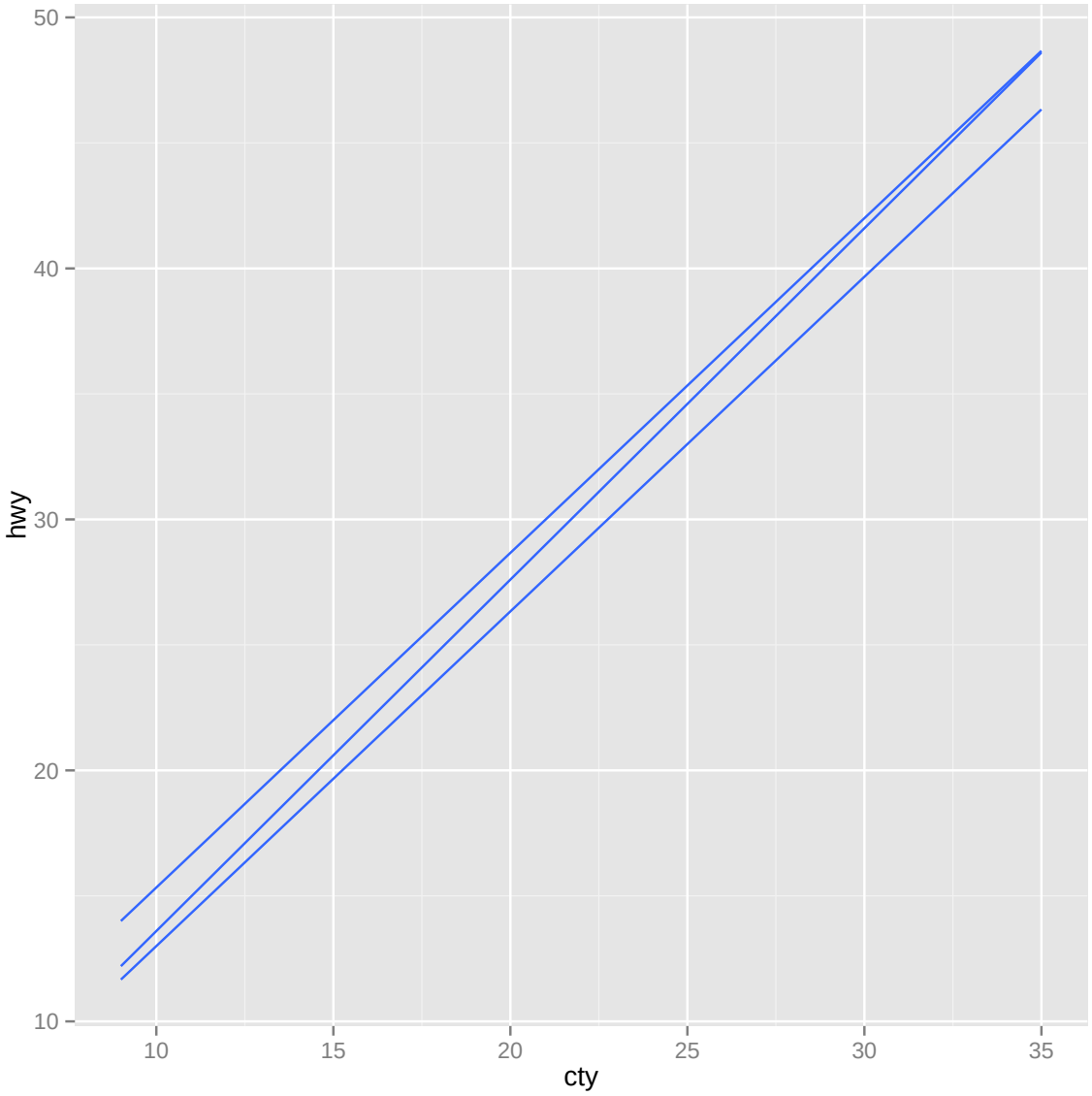

```
f + geom_jitter()
```



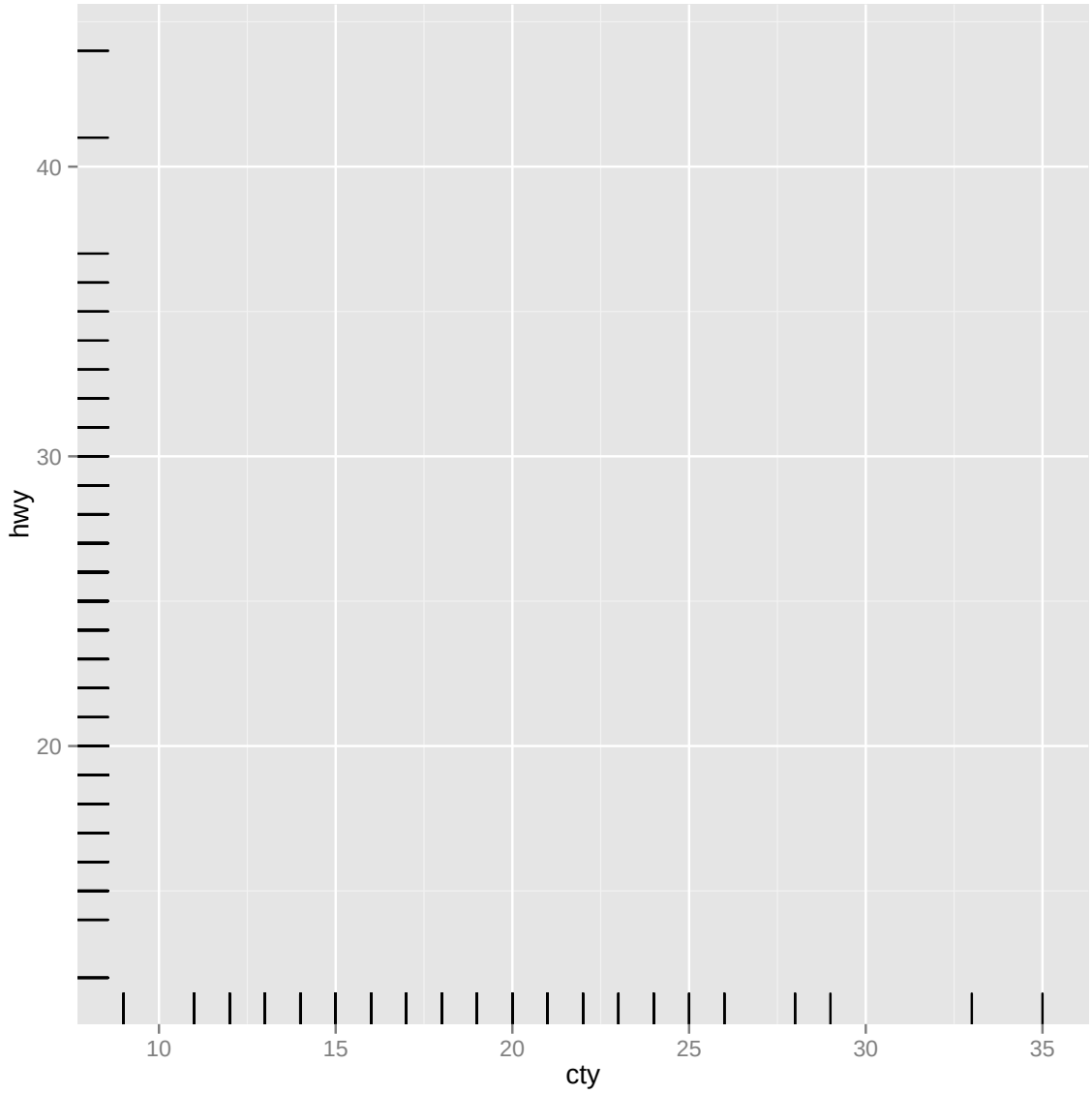

```
# x, y, alpha, color, fill, shape, size
f + geom_point()
```




```
# x, y, alpha, color, fill, shape, size
f + geom_quantile()

## Smoothing formula not specified. Using: y ~ x
```

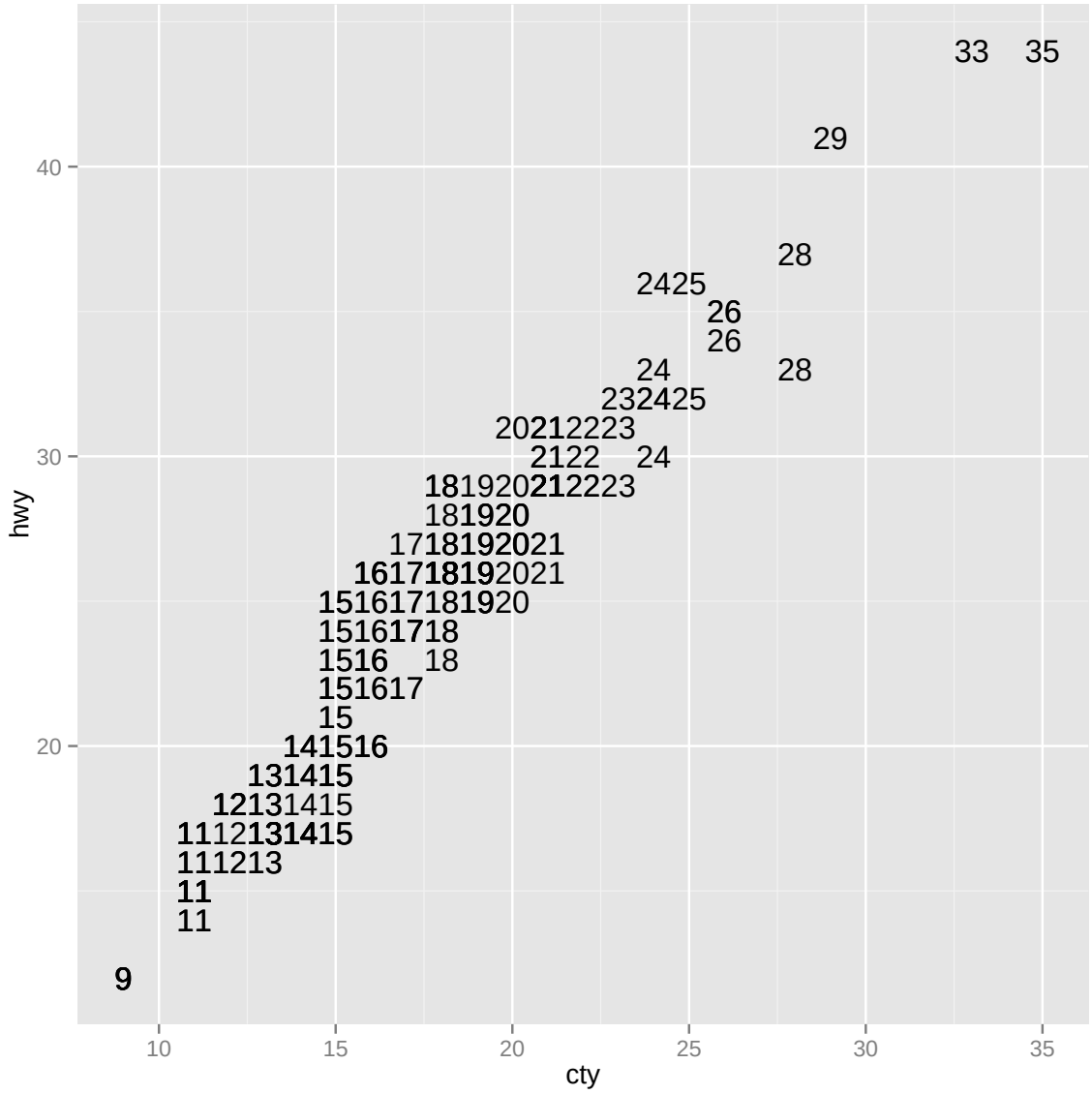



```
# x, y, alpha, color, linetype, size, weight
f + geom_rug(sides = "bl")
```

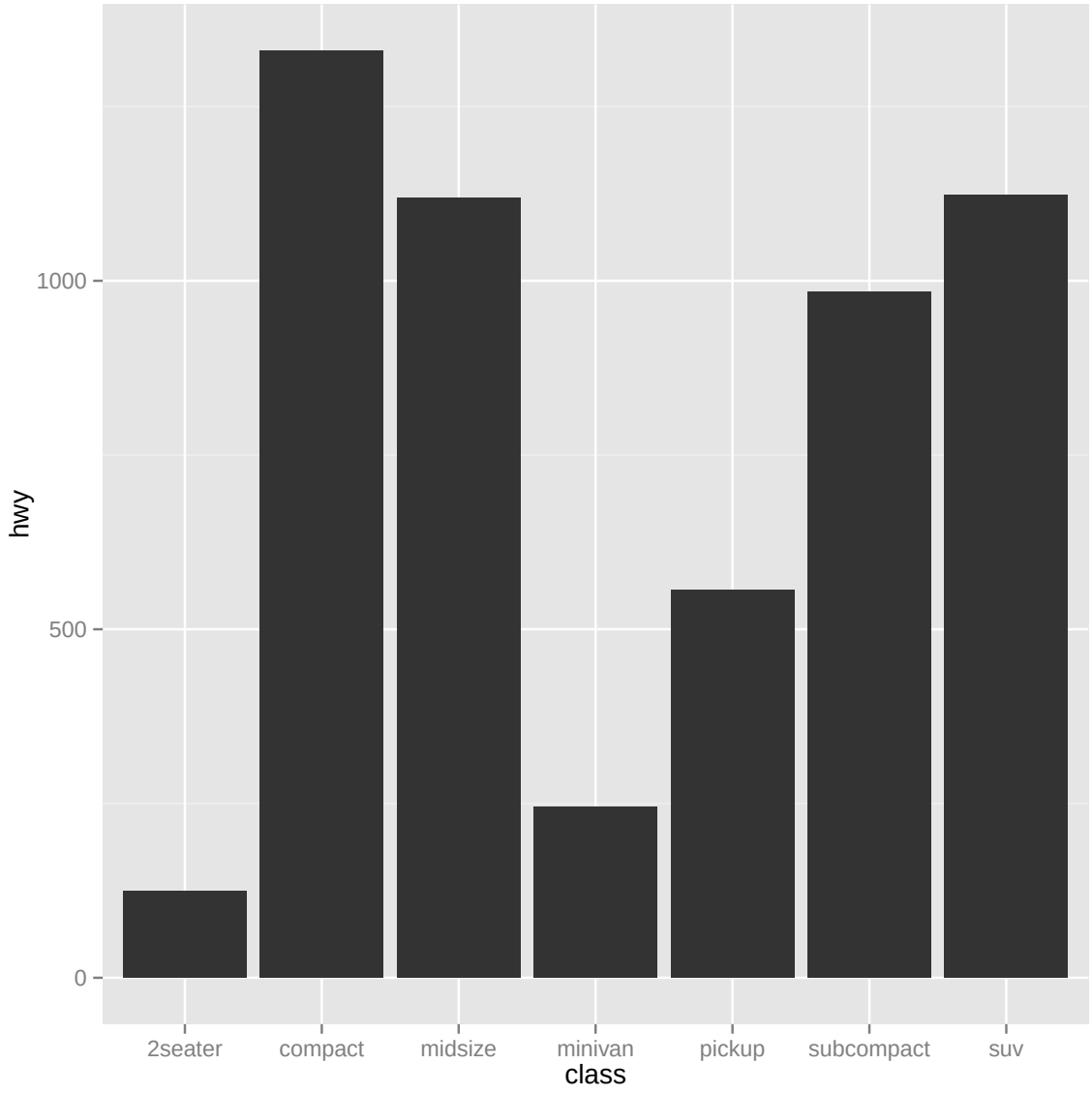



```
# alpha, color, linetype, size
f + geom_smooth(model = lm)

## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
```

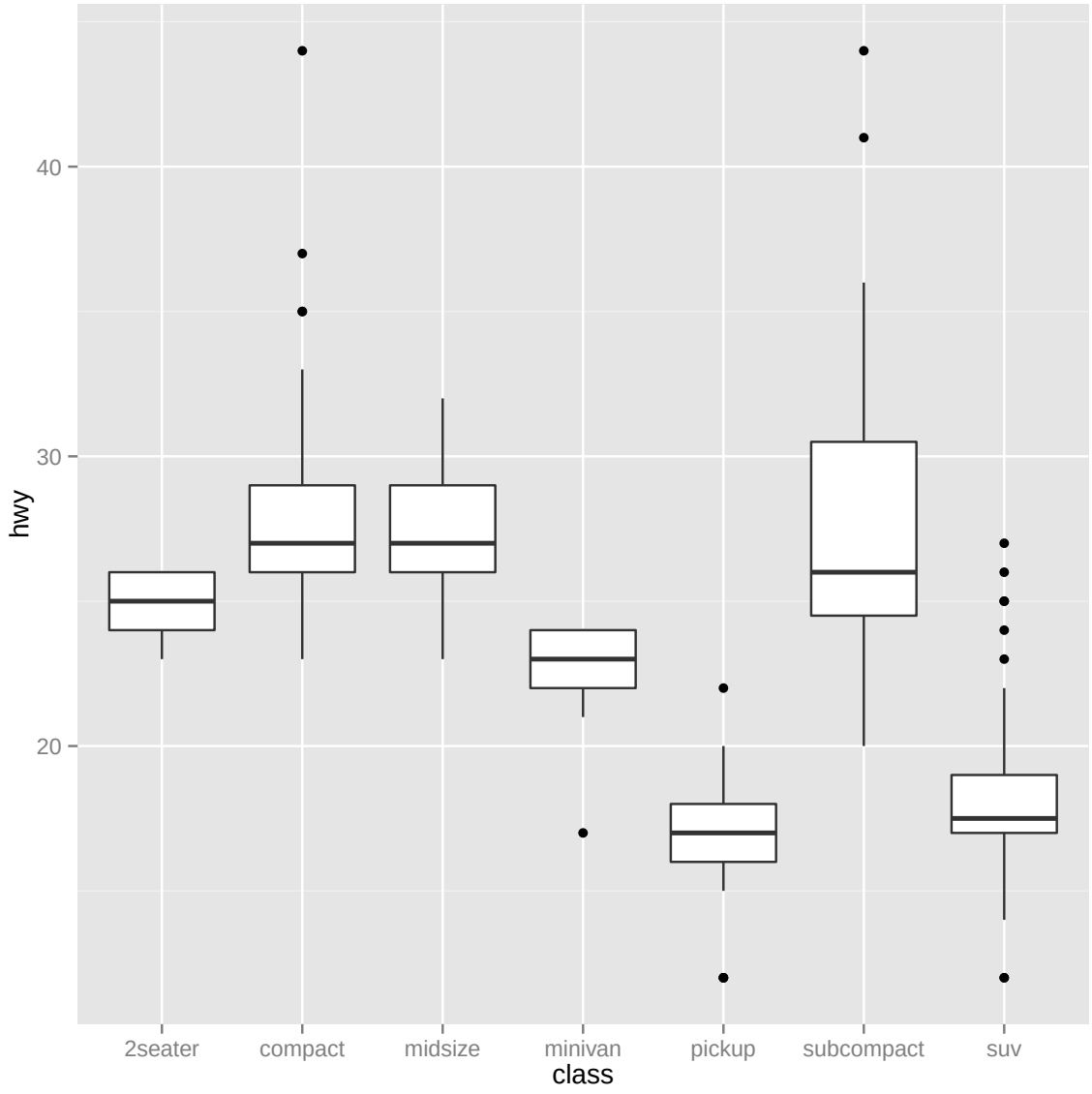

```
# x, y, alpha, color, fill, linetype, size, weight
f + geom_text(aes(label = cty))
```



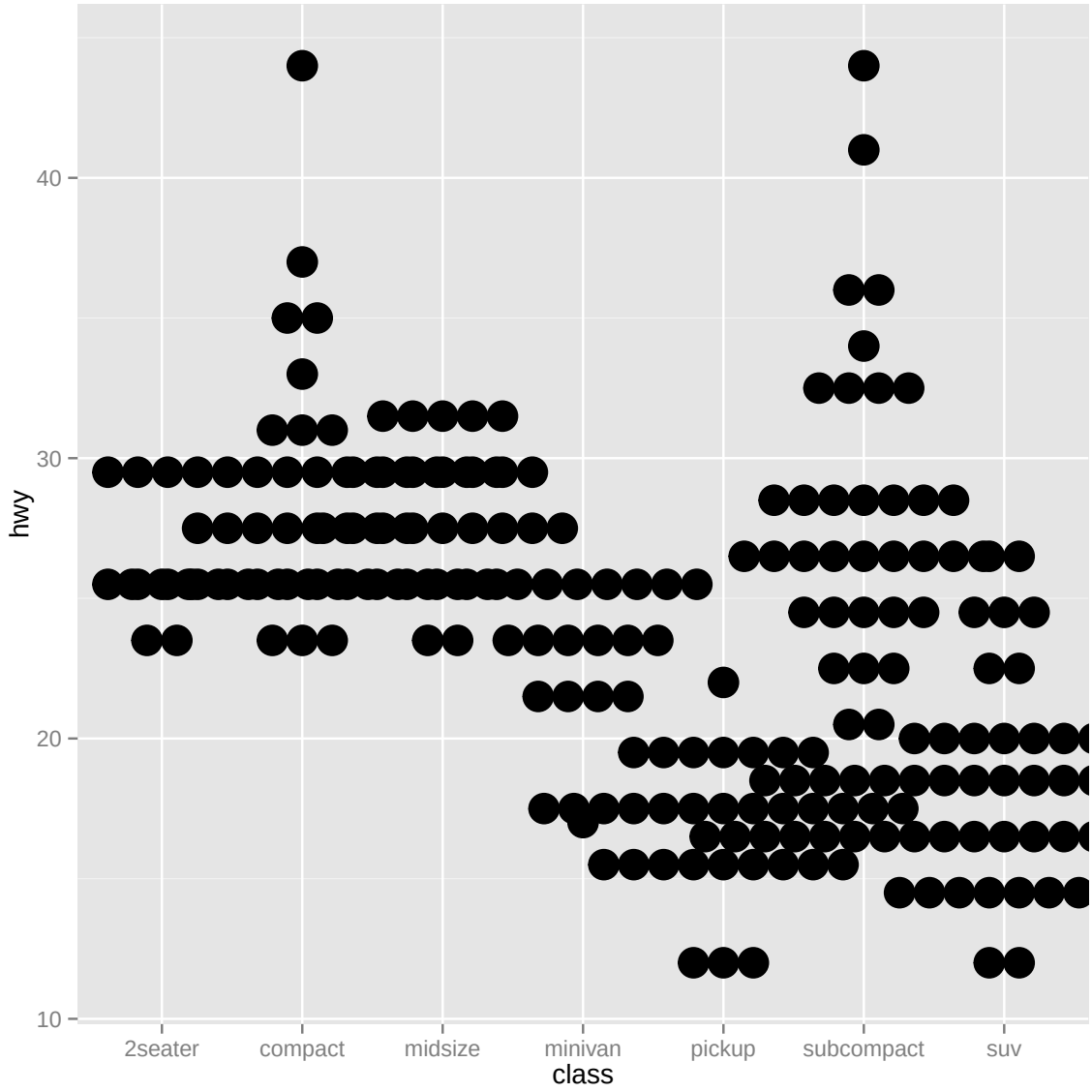

```
# x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size,

# Discreta X, Continua Y
g <- ggplot(mpg, aes(class, hwy))
g + geom_bar(stat = "identity")
```



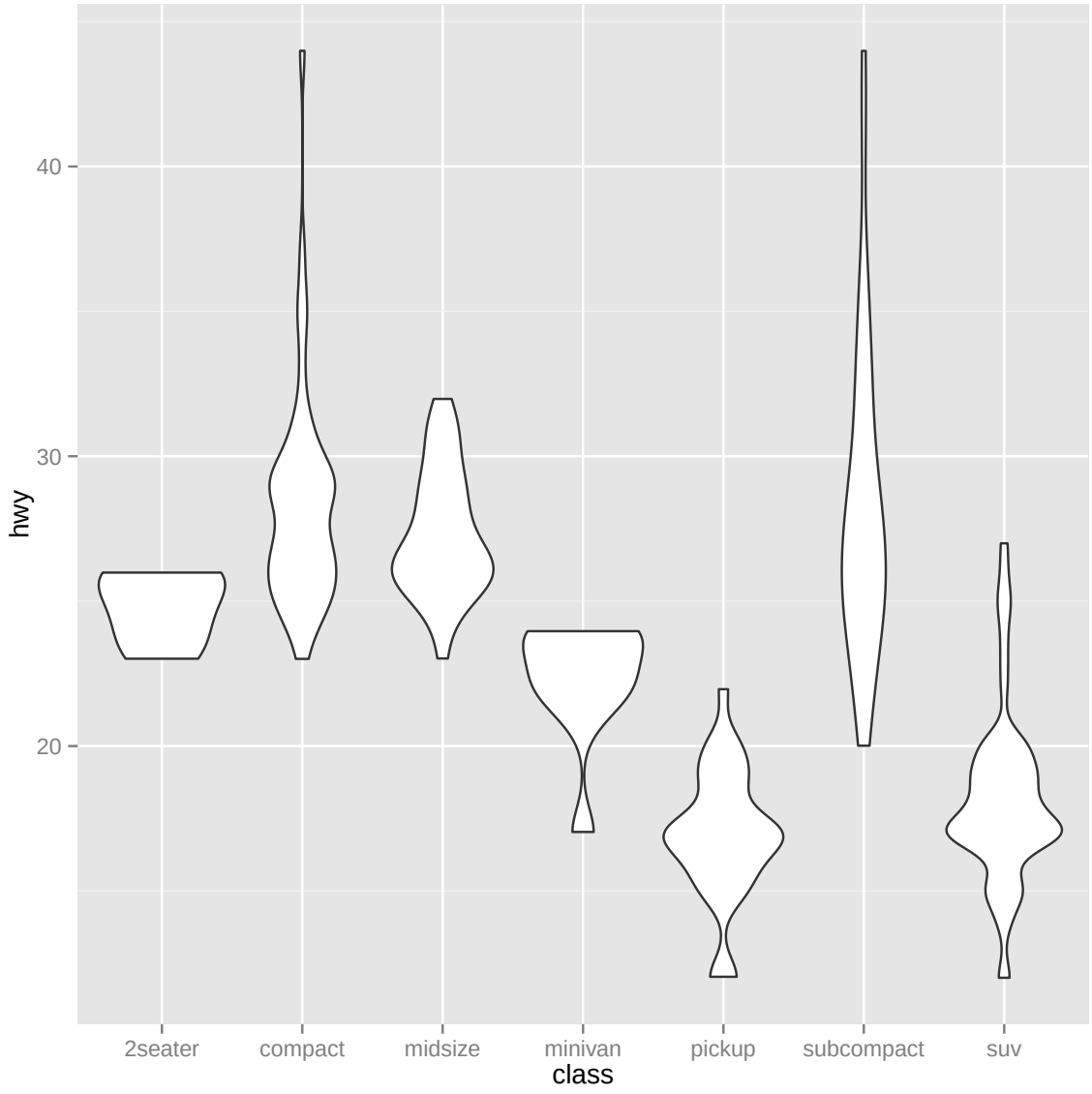

```
#x, y, alpha, color, fill, linetype, size, weight
g + geom_boxplot()
```



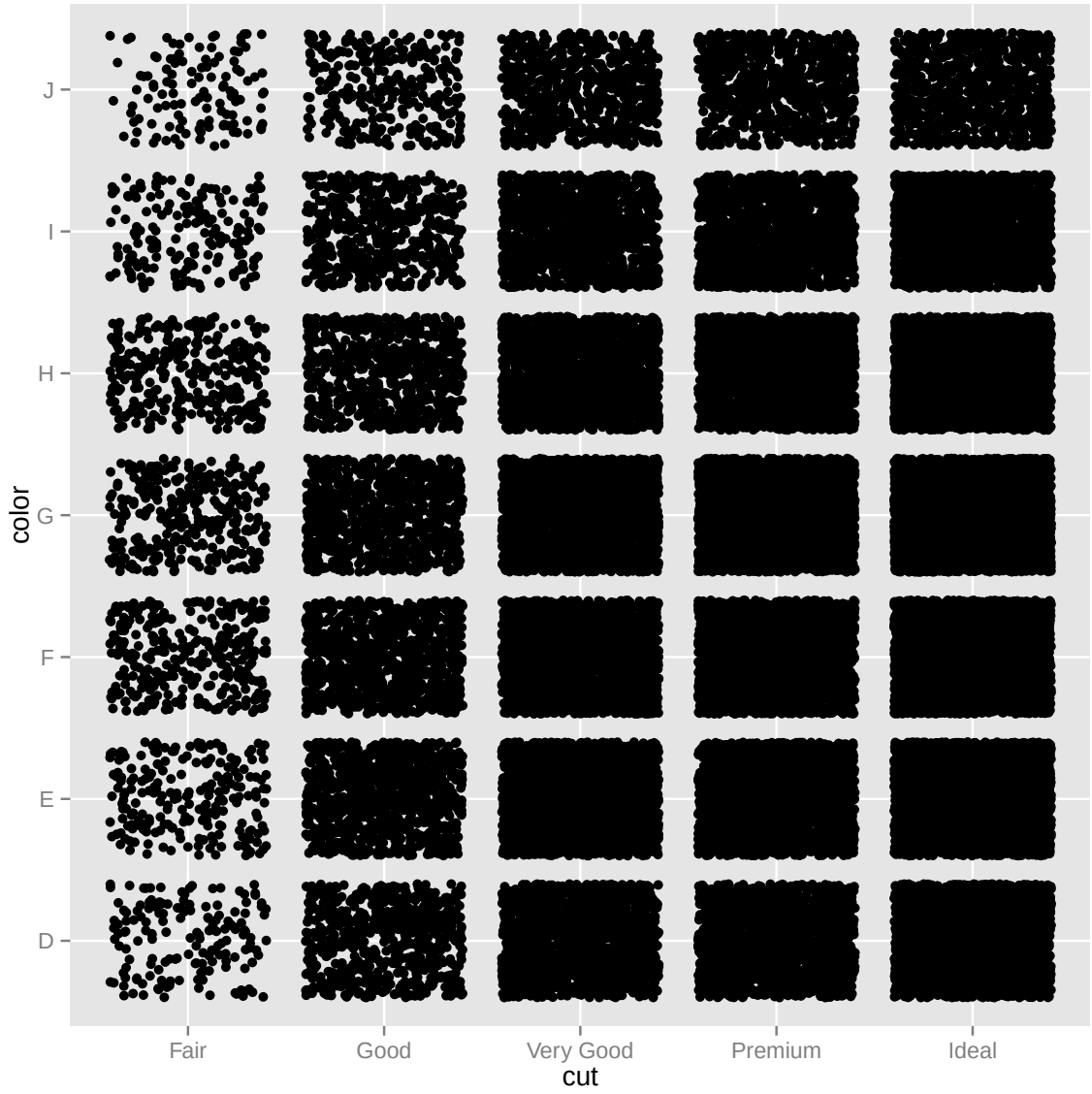

```
#lower, middle, upper, x, ymax, ymin, alpha, color, fill, linetype, shape, size
g + geom_dotplot(binaxis = "y",
stackdir = "center")

## stat_bindot: binwidth defaulted to range/30. Use 'binwidth = x' to adjust
this.
```




```
#x, y, alpha, color, fill
g + geom_violin(scale = "area")
```

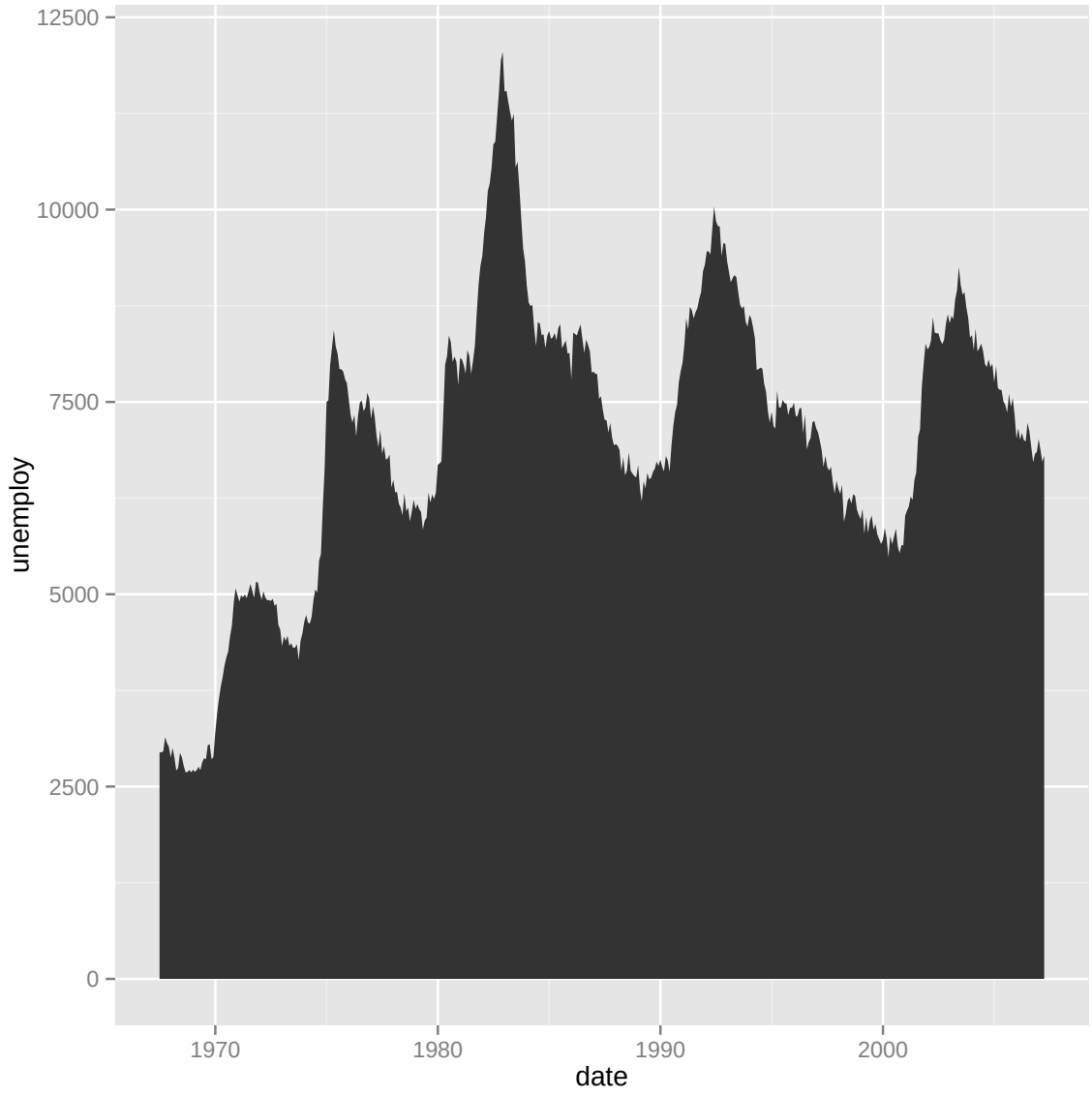



```
#x, y, alpha, color, fill, linetype, size, weight

#Discreta X, Discreta Y
h <- ggplot(diamonds, aes(cut, color))
h + geom_jitter()
```

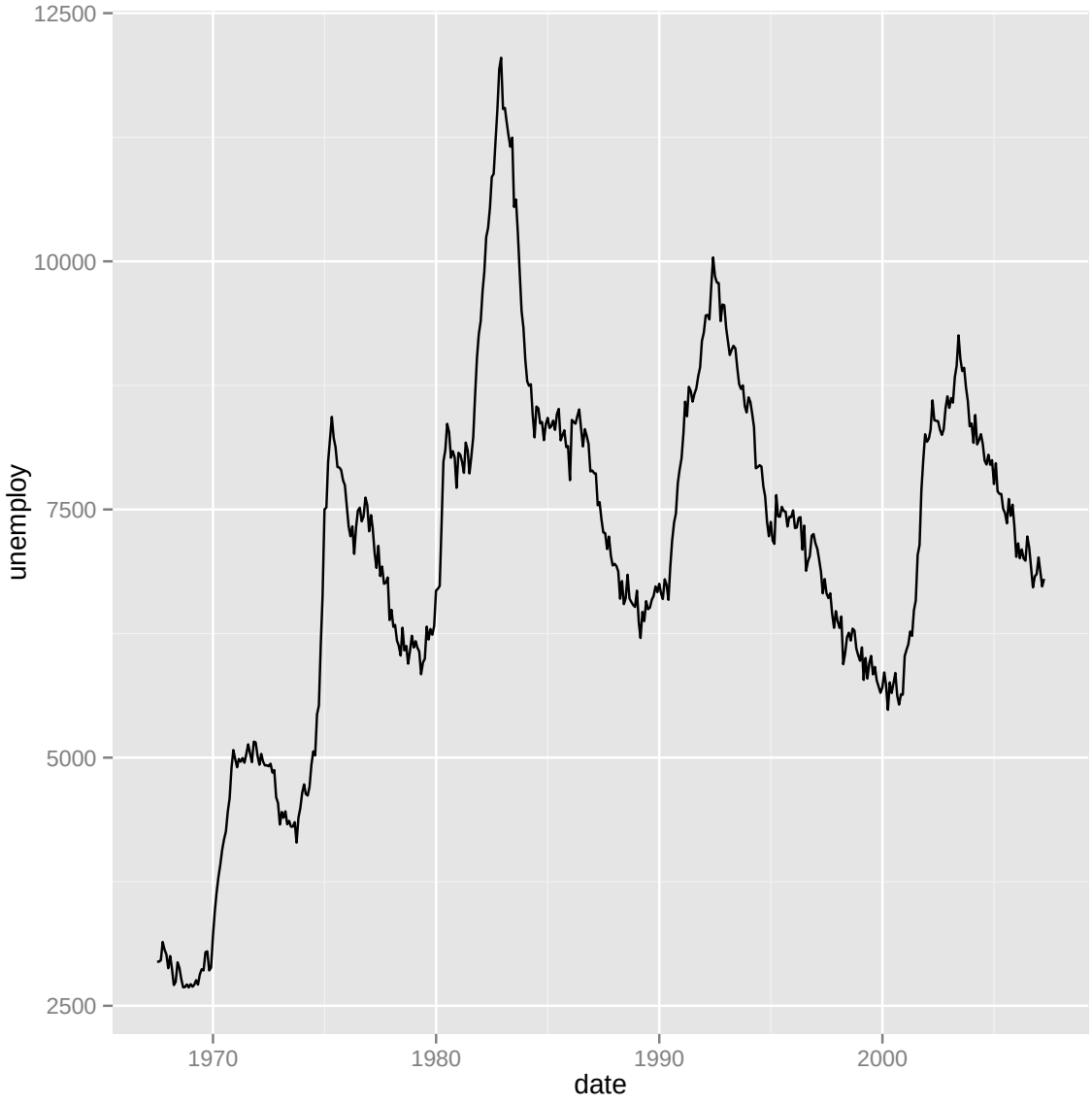



```
#x, y, alpha, color, fill, shape, size

#DistribuciÃ³n Bivariada Continua
i <- ggplot(movies, aes(year, rating))
i + geom_bin2d(binwidth = c(5, 0.5))

## Error in eval(expr, envir, enclos): objeto 'year' no encontrado

#xmax, xmin, ymax, ymin, alpha, color, fill,linetype, size, weight
i + geom_density2d()

## Error in eval(expr, envir, enclos): objeto 'year' no encontrado

#x, y, alpha, colour, linetype, size
i + geom_hex()
```

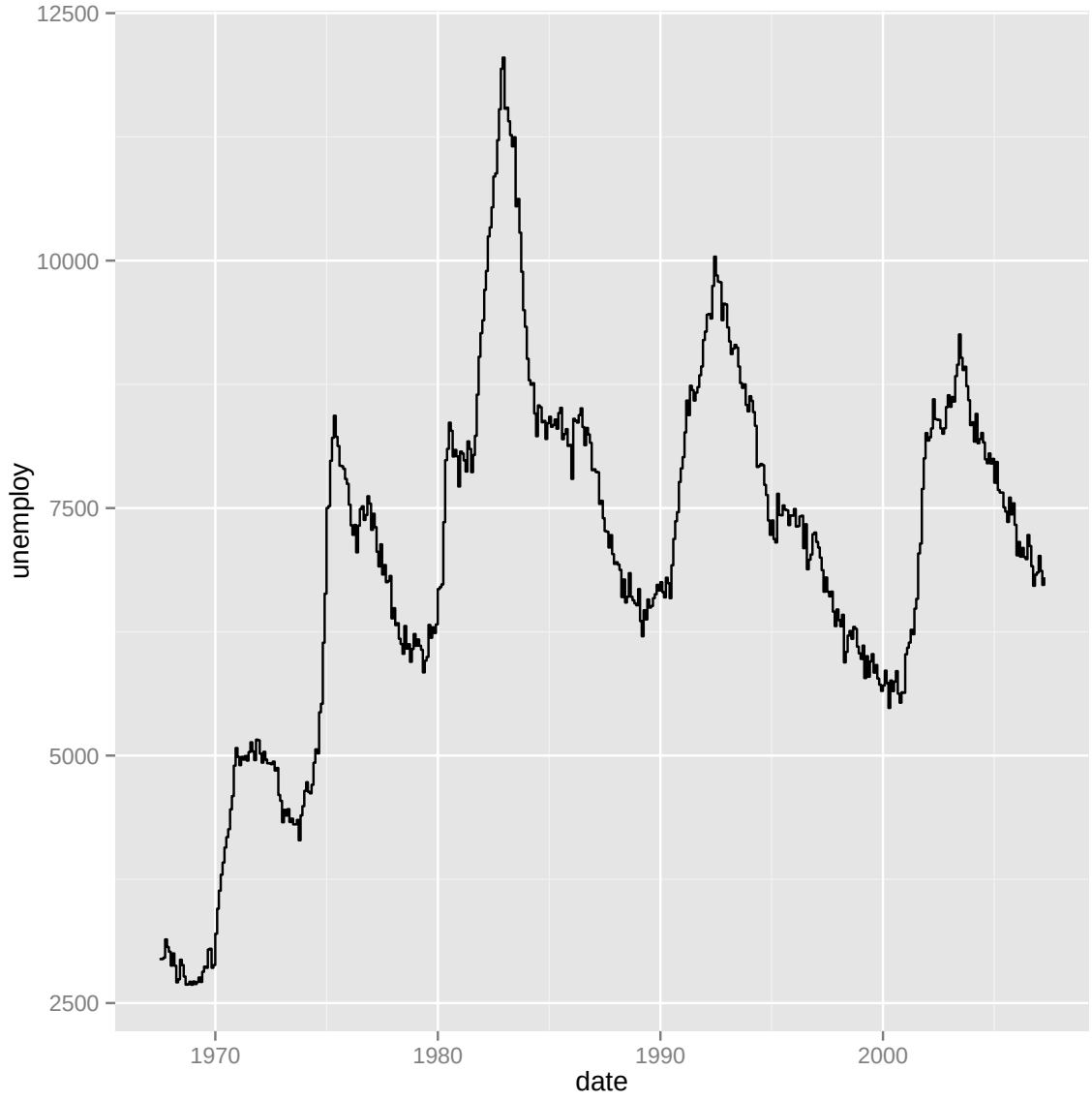
```
## Error in eval(expr, envir, enclos): objeto 'year' no encontrado

#x, y, alpha, colour, fill size

#Funcion Continua
j <- ggplot(economics, aes(date, unemploy))
j + geom_area()
```



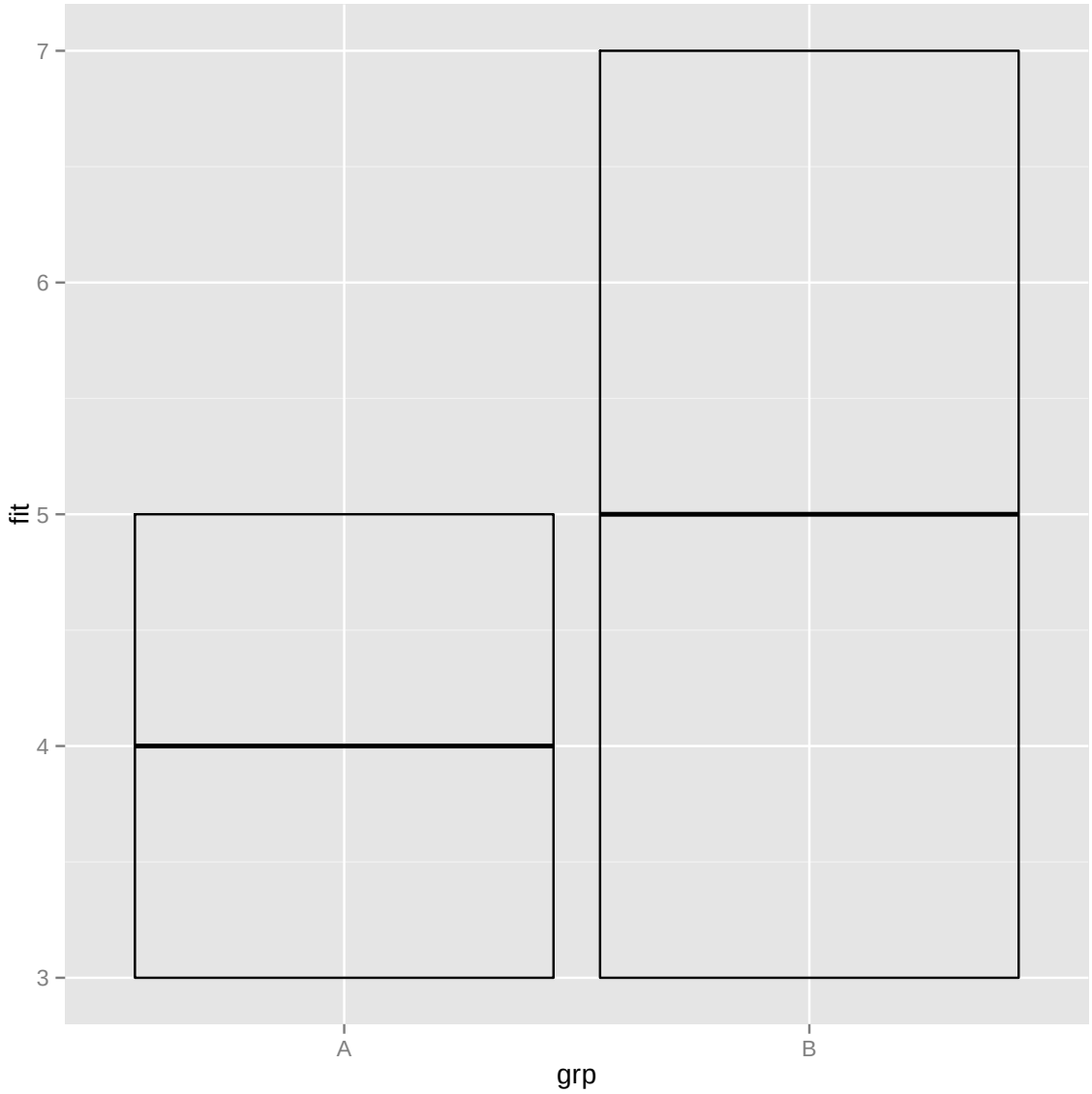

```
#x, y, alpha, color, fill, linetype, size
j + geom_line()
```



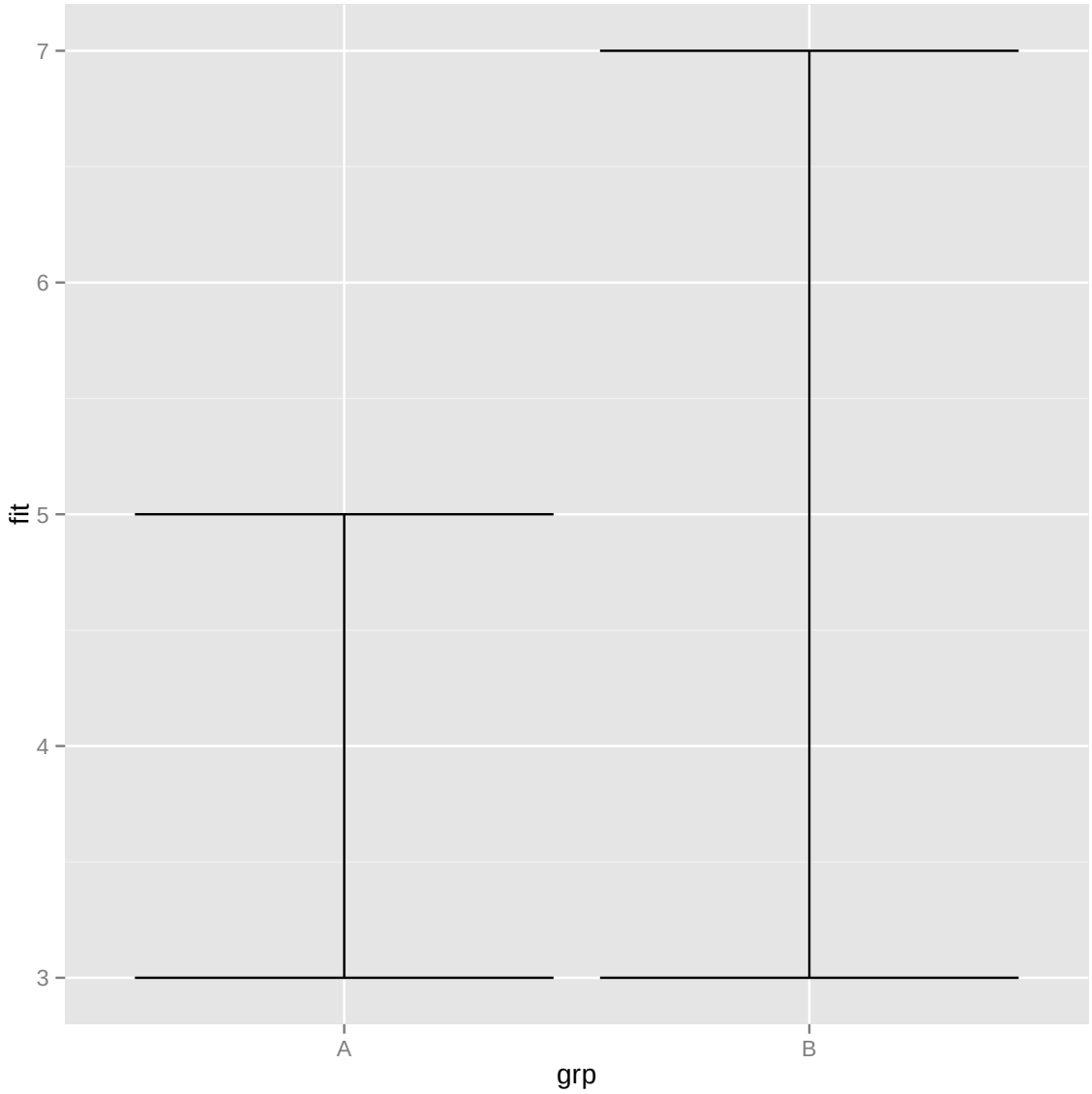

```
#x, y, alpha, color, linetype, size
j + geom_step(direction = "hv")
```



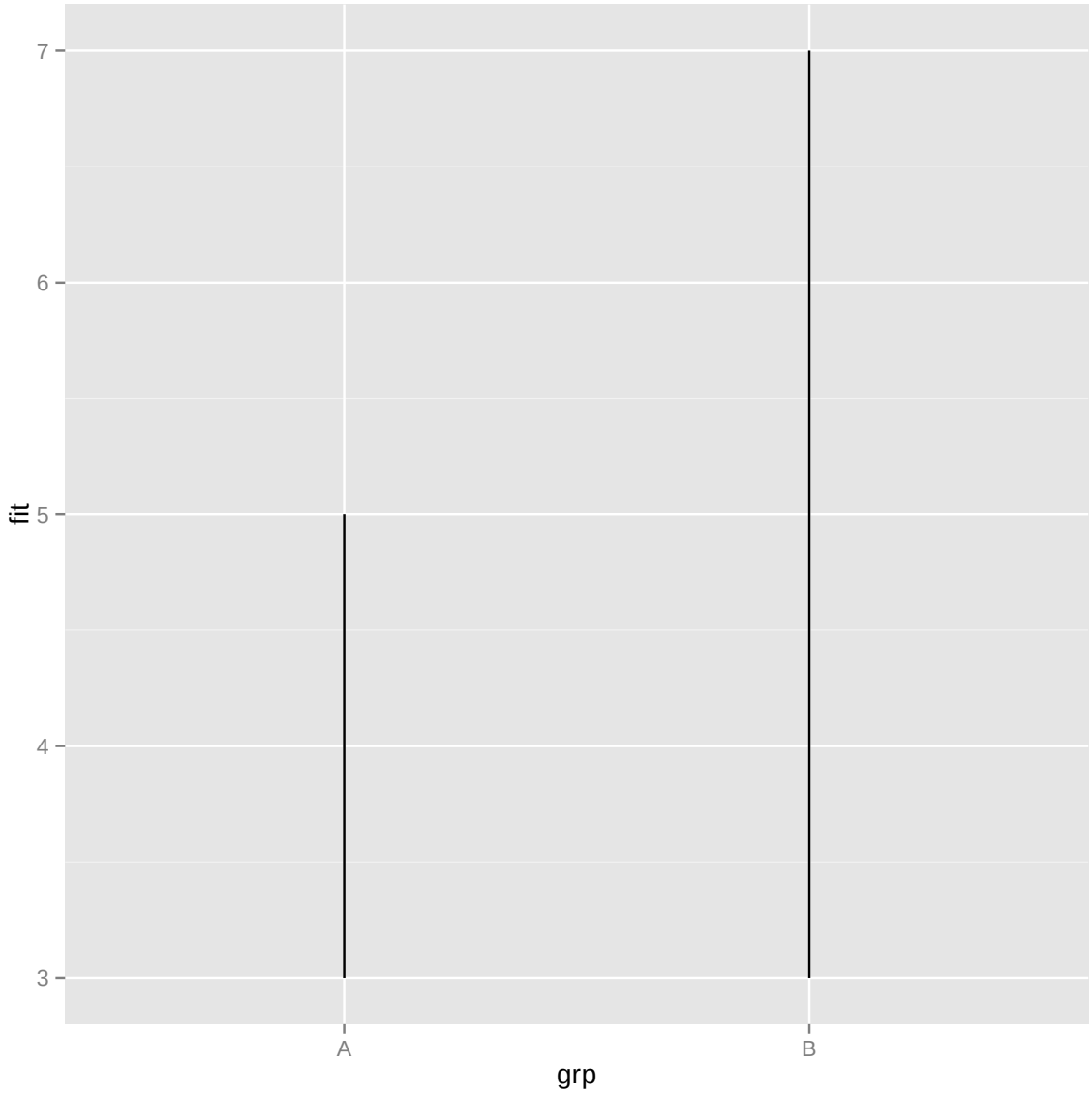

```r
#x, y, alpha, color, linetype, size

# Visualizando el error
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
k <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))
k + geom_crossbar(fatten = 2)
```



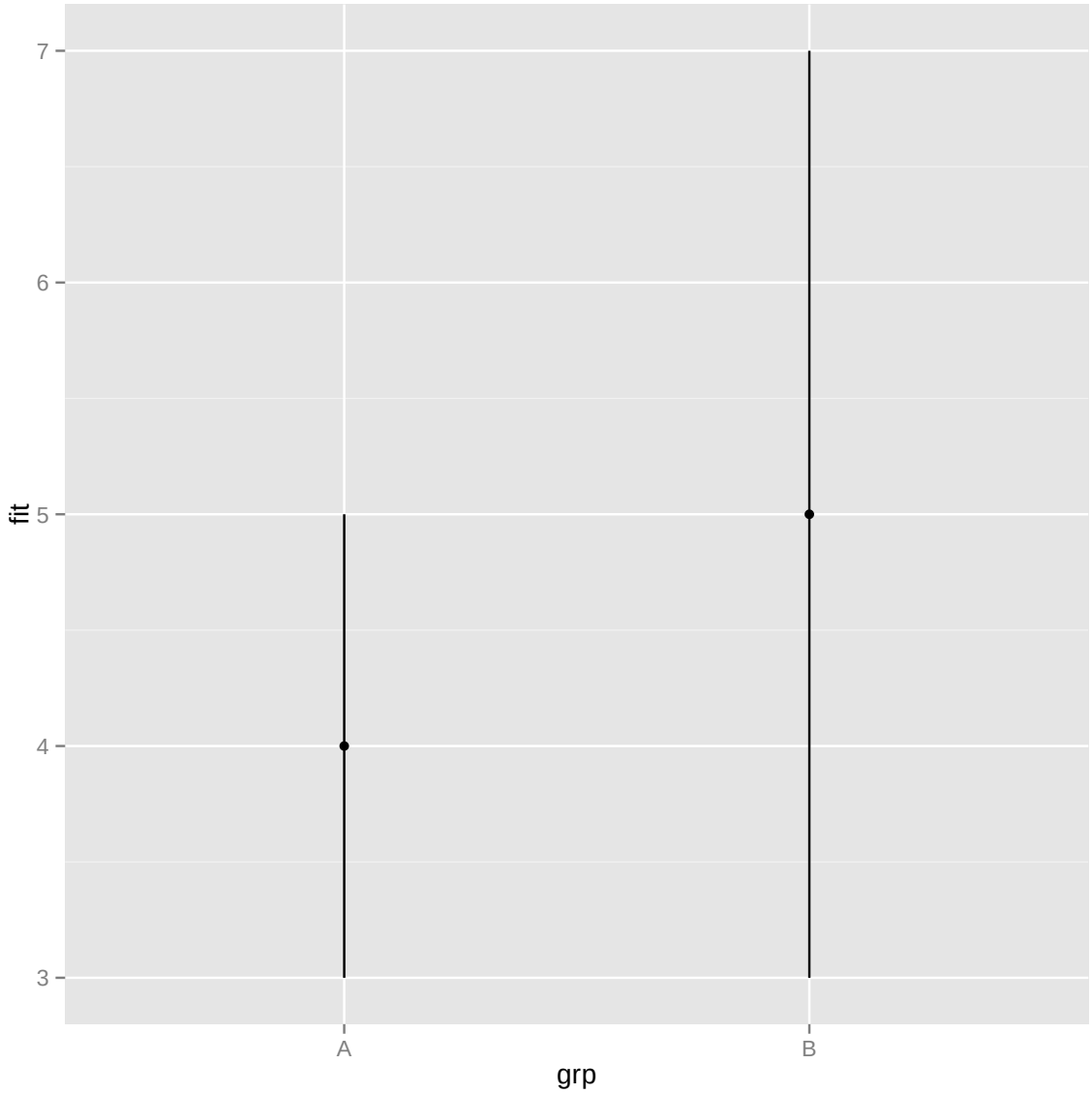

```
#x, y, ymax, ymin, alpha, color, fill, linetype, size
k + geom_errorbar()
```




```
#x, ymax, ymin, alpha, color, linetype, size, width (also geom_errorbarh())
k + geom_linerange()
```

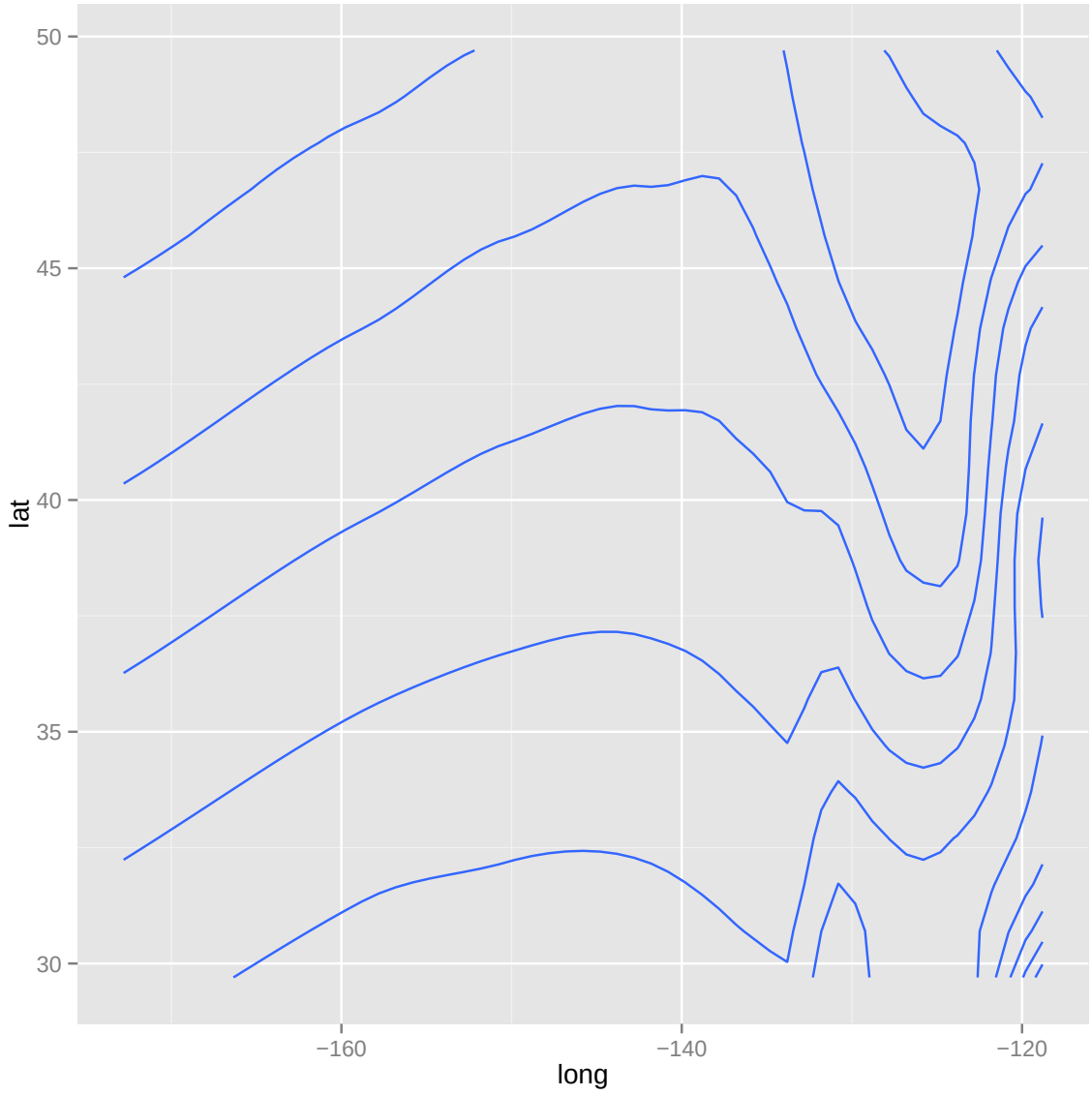



```
#x, ymin, ymax, alpha, color, linetype, size
k + geom_pointrange()
```

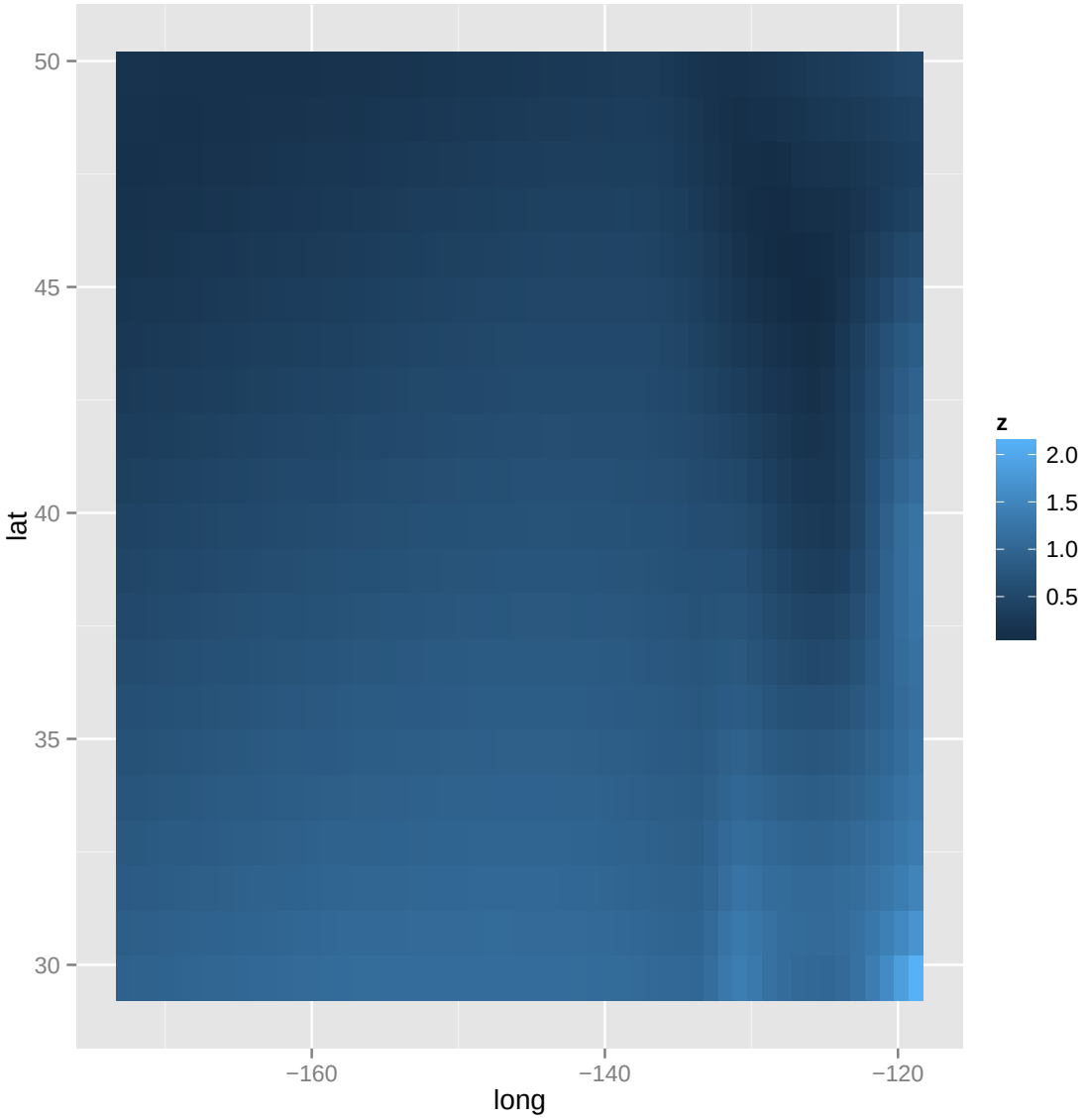



```
#x, y, ymin, ymax, alpha, color, fill, linetype, shape, size

#Mapas
data <- data.frame(murder =USArrests$Murder, state = tolower(rownames(USArrests))
map <- map_data("state")

## Error: maps package required for this functionality. Please install and
try again.

l <- ggplot(data, aes(fill = murder))
l + geom_map(aes(map_id = state), map = map) +
expand_limits(x = map$long, y = map$lat)

## Error in is.data.frame(map): objeto 'map' no encontrado
```

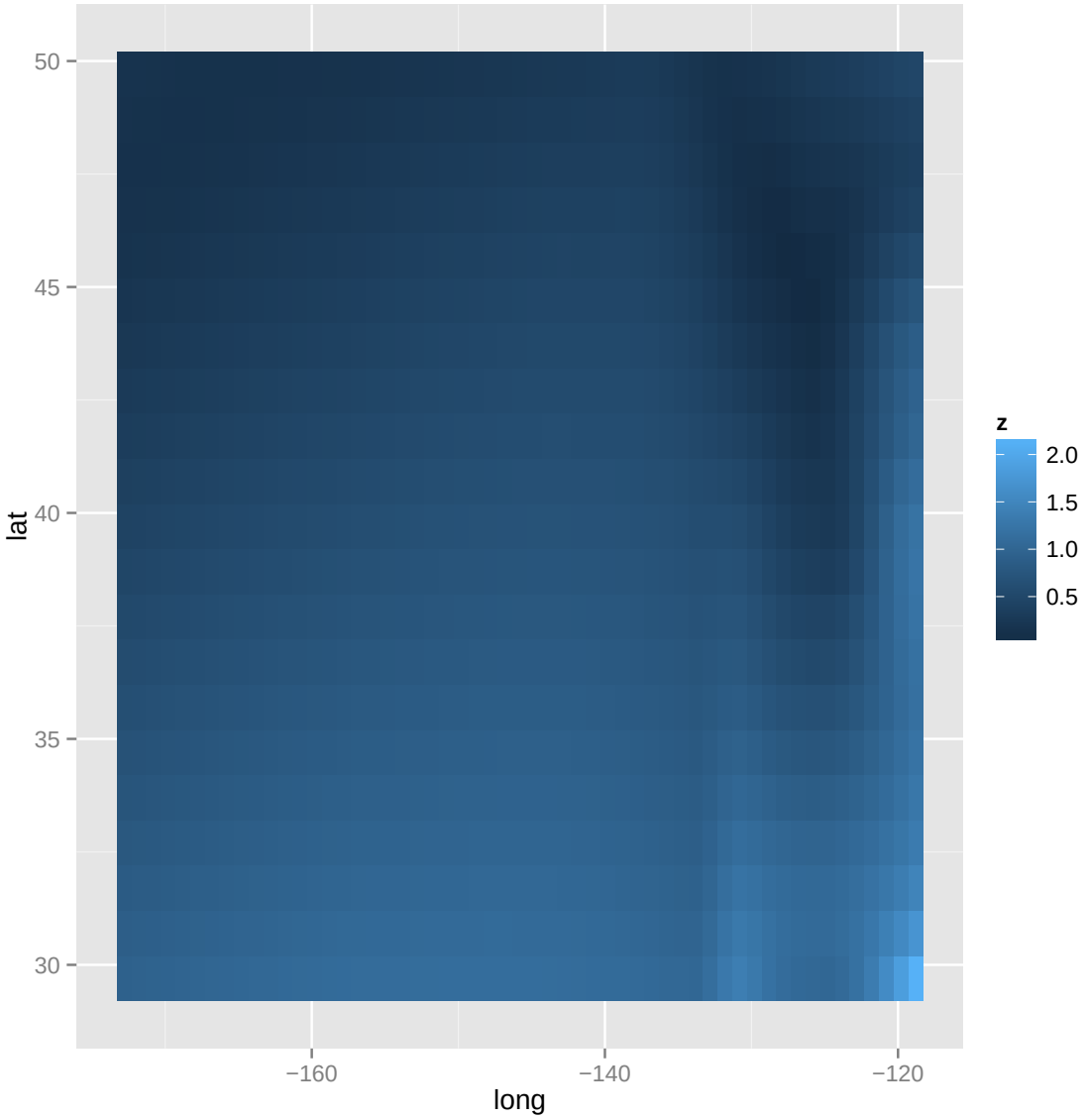
```
#map_id, alpha, color, fill, linetype, size
##########################################################################

##########################################################################
#Tres Variables
seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))
m <- ggplot(seals, aes(long, lat))
m + geom_contour(aes(z = z))
```




```
#x, y, z, alpha, colour, linetype, size, weight seals$z
m + geom_raster(aes(fill = z), hjust=0.5,
vjust=0.5, interpolate=FALSE)
```



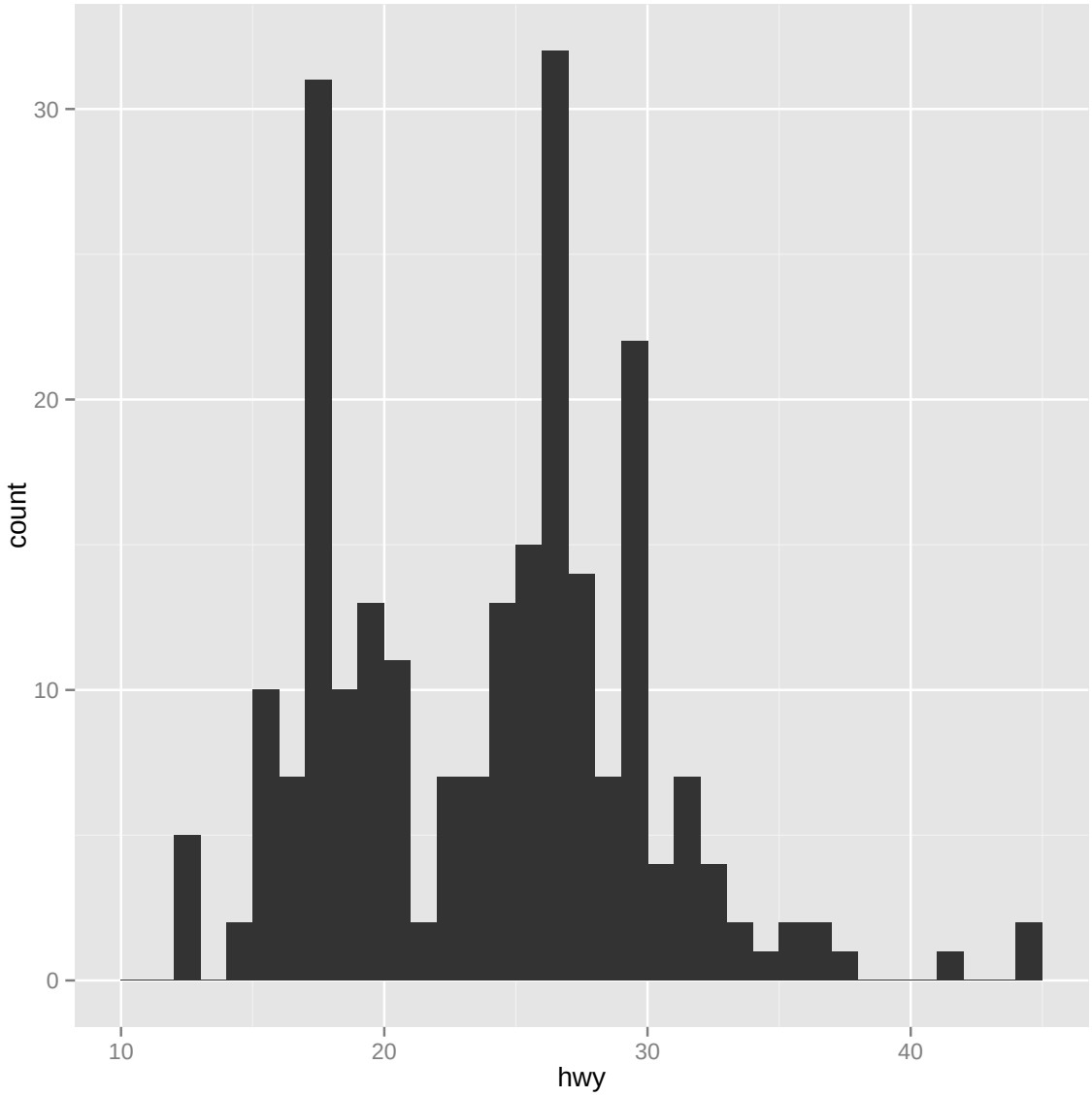

```
#x, y, alpha, fill
m + geom_tile(aes(fill = z))
```



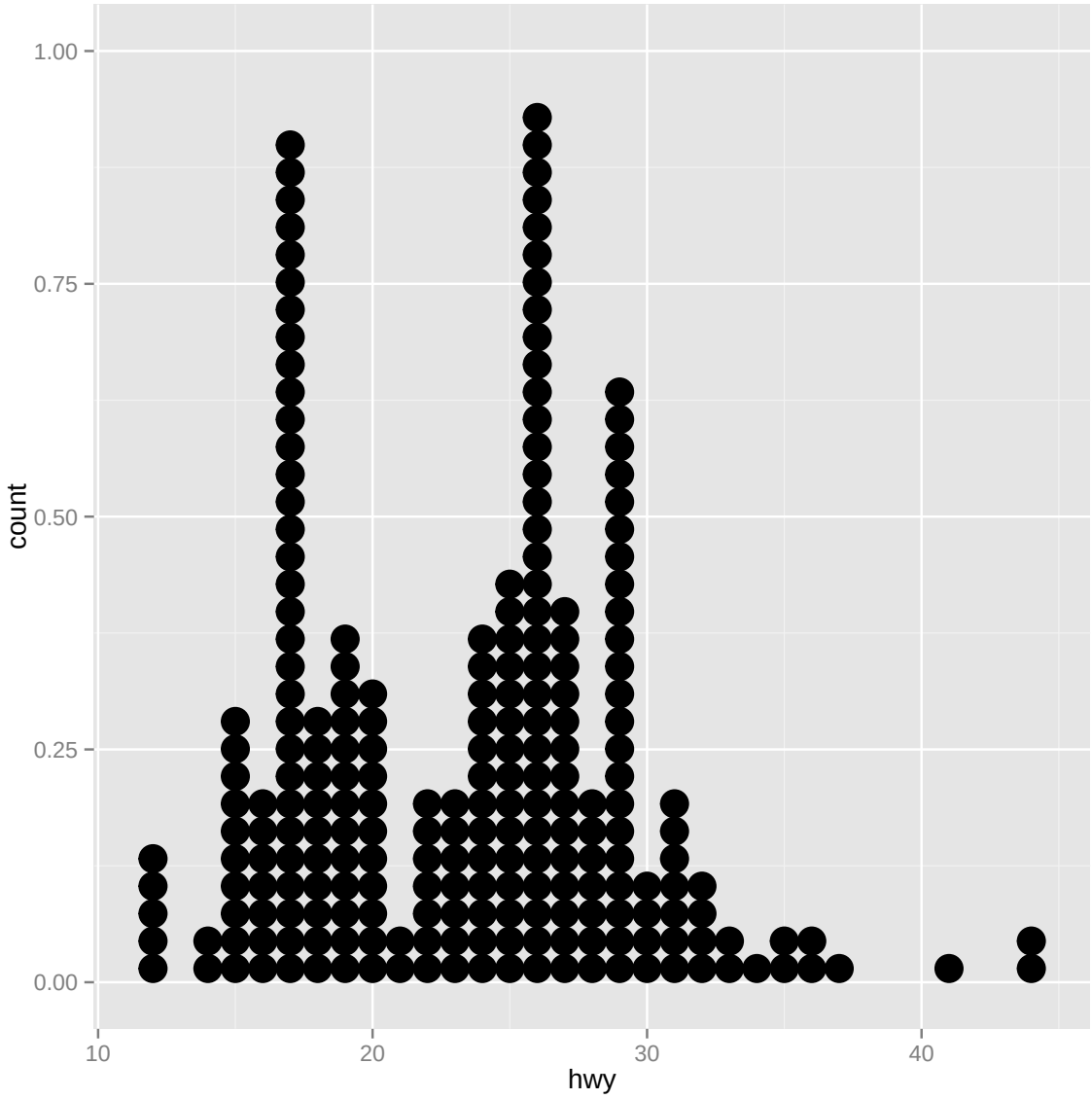

```
#x, y, alpha, color, fill, linetype, size
```

Stats Una forma alternativa de crear una capa

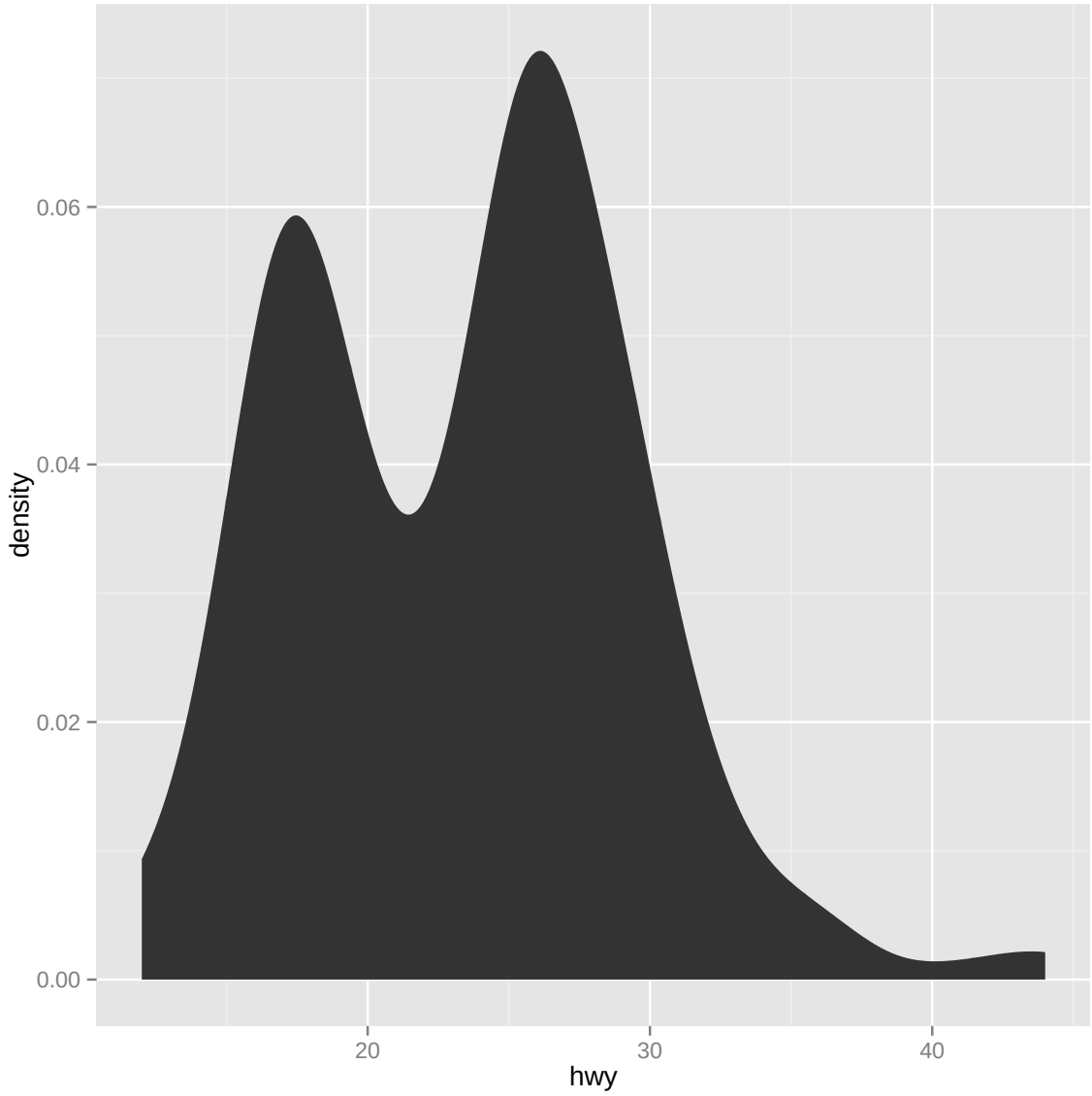
```
a + stat_bin(binwidth = 1, origin = 10)
```



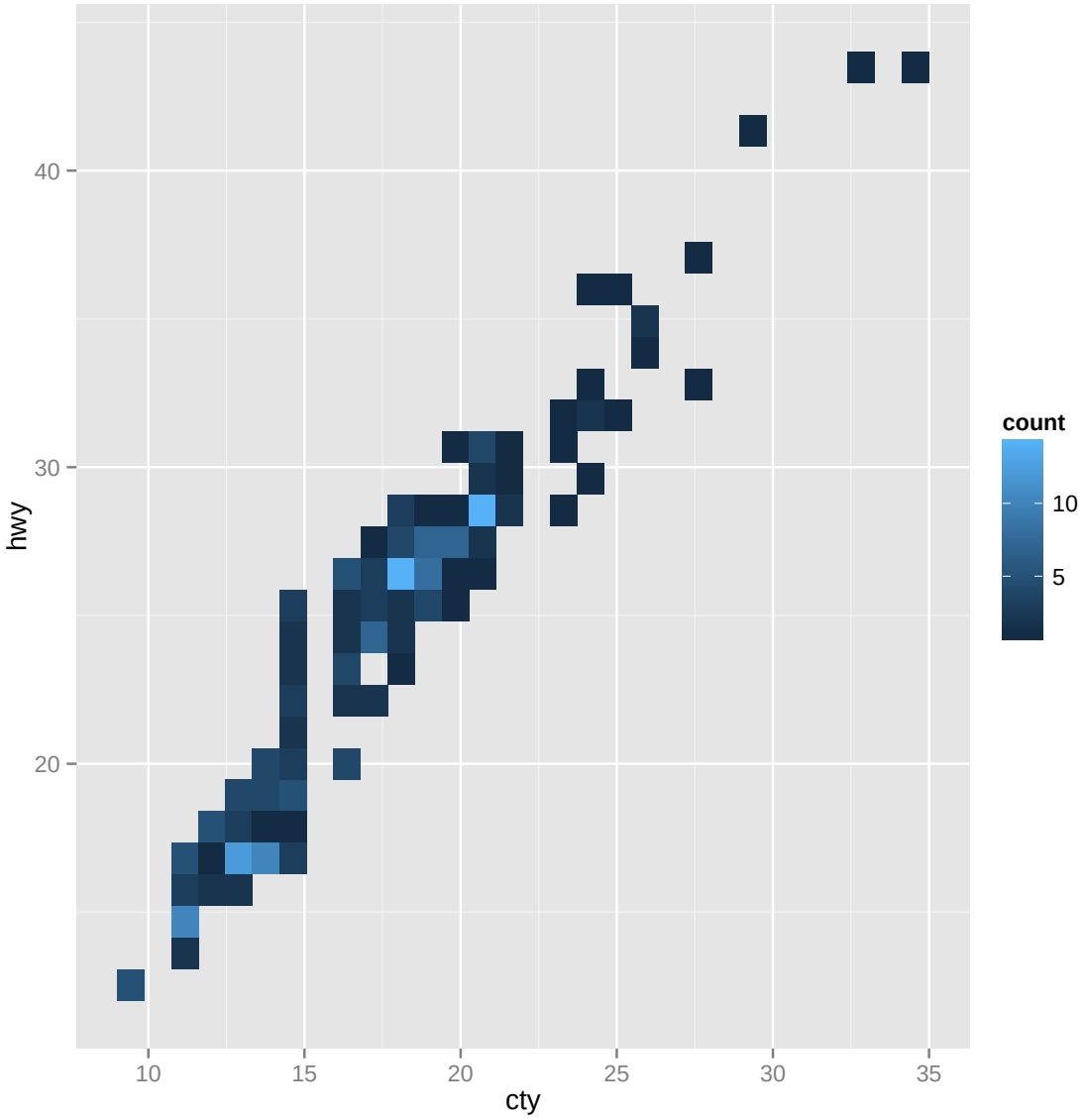

```
#x, y | ..count.., ..ncount.., ..density.., ..ndensity..
a + stat_bindot(binwidth = 1, binaxis = "x")
```




```
#x , y, | ..count.., ..ncount..
a + stat_density(adjust = 1, kernel = "gaussian")
```

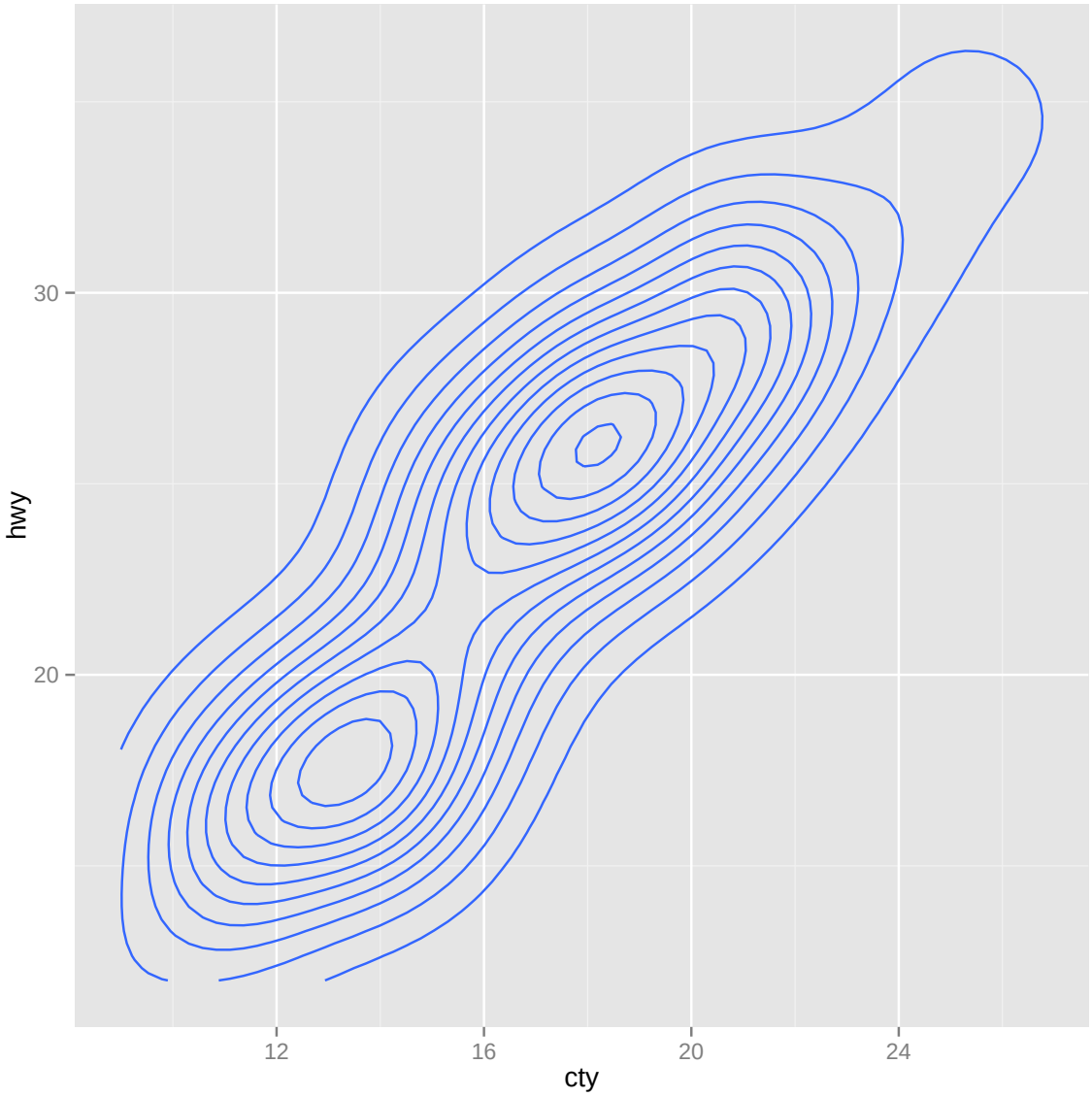



```
#x, y, | ..count.., ..density.., ..scaled..
f + stat_bin2d(bins = 30, drop = TRUE)
```

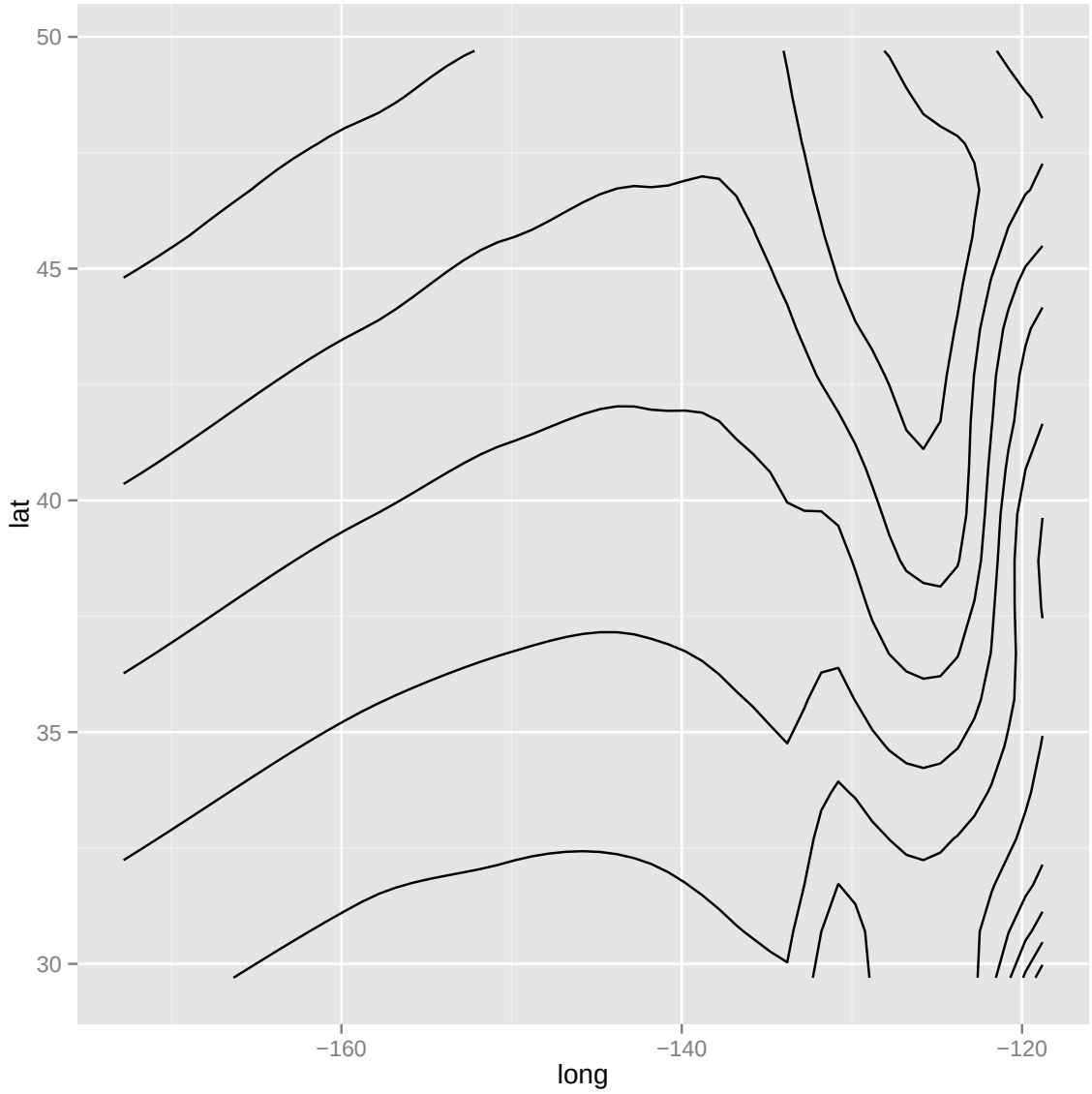



```
#x, y, fill | ..count.., ..density..
f + stat_binhex(bins = 30)
```

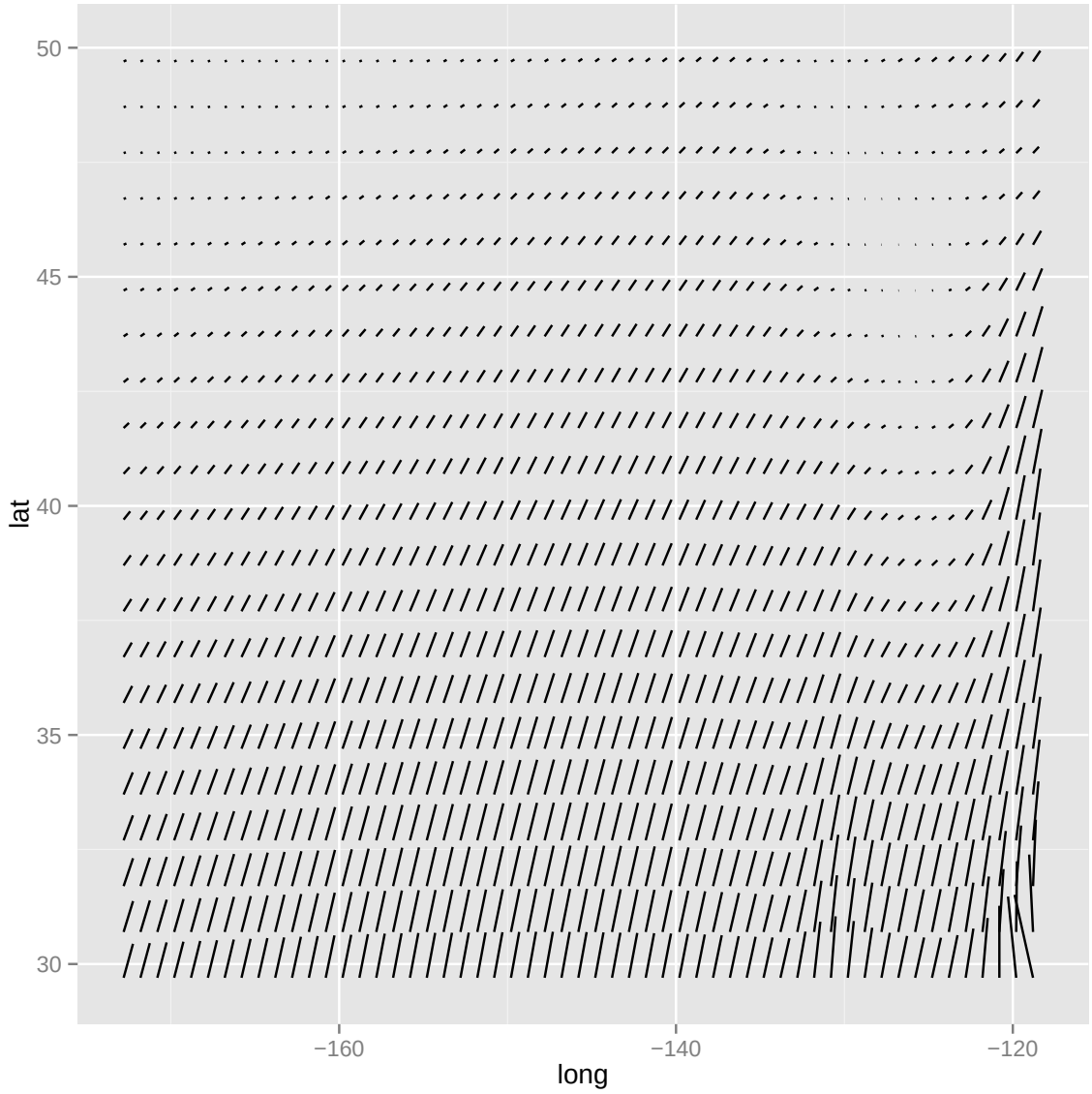

```
#x, y, fill / ..count.., ..density..
f + stat_density2d(contour = TRUE, n = 100)
```



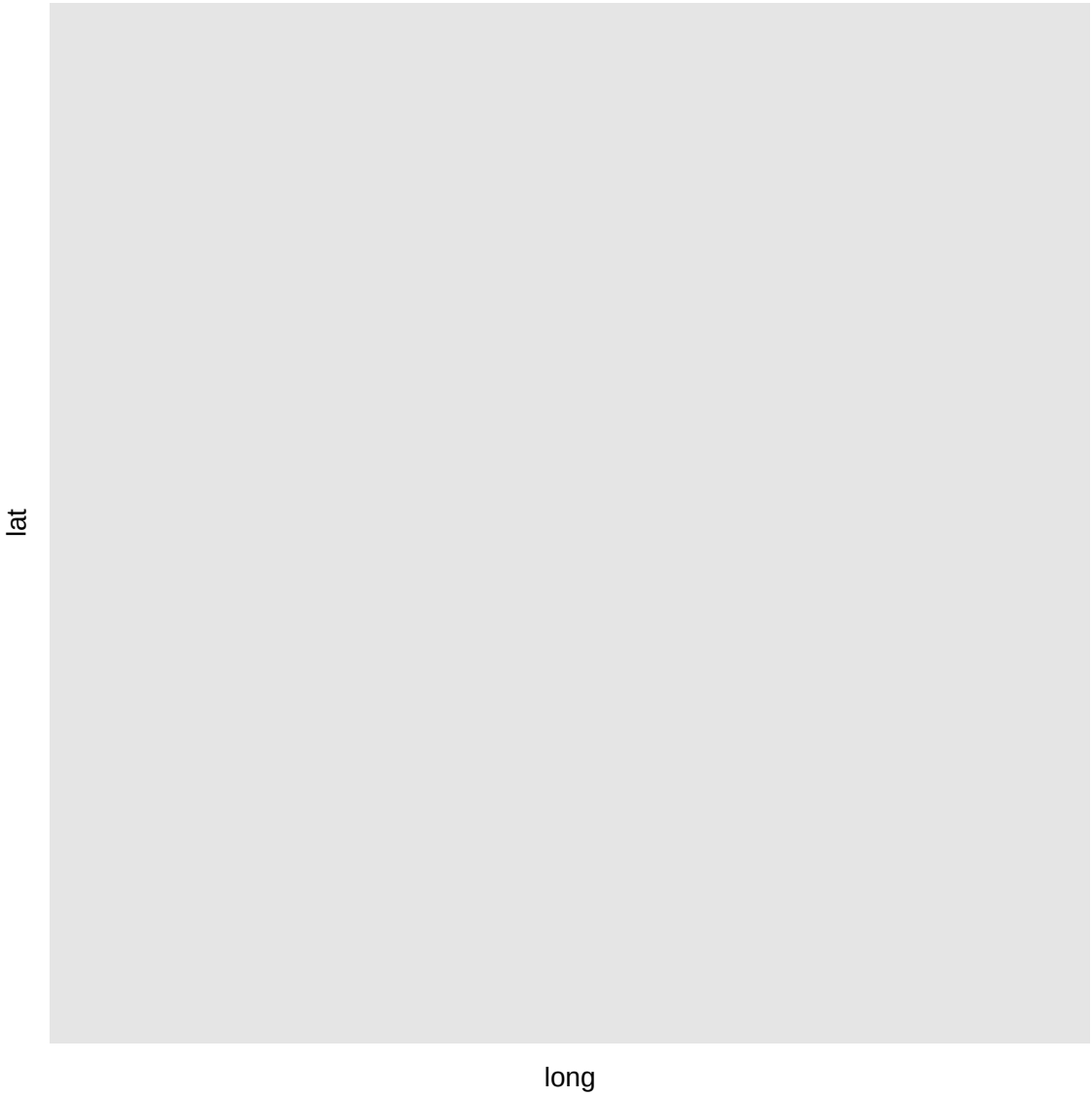

```
#x, y, color, size | ..level..
m + stat_contour(aes(z = z))
```



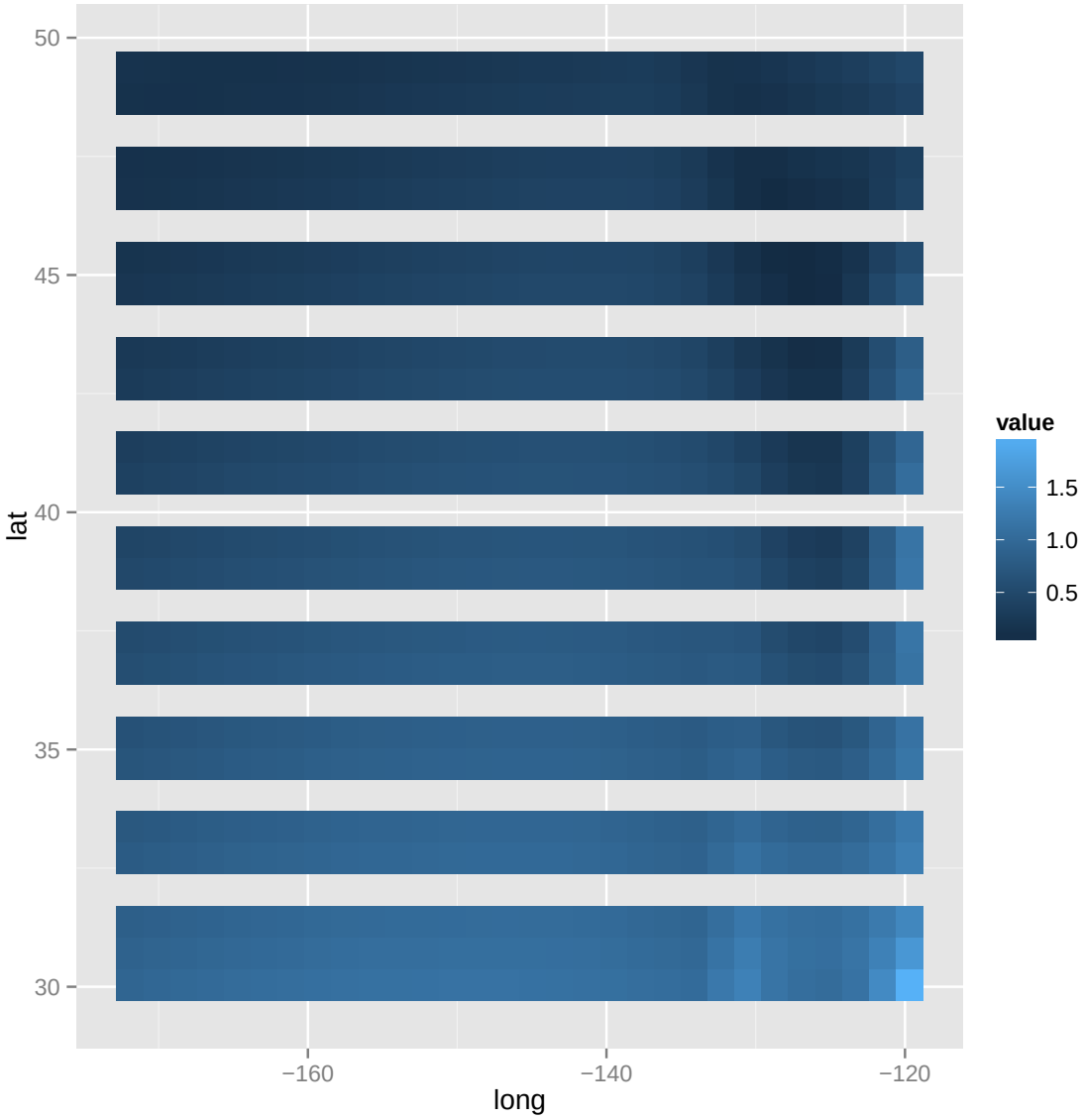

```
#x, y, z, order | ..level..
m+ stat_spoke(aes(radius= z, angle = z))
```



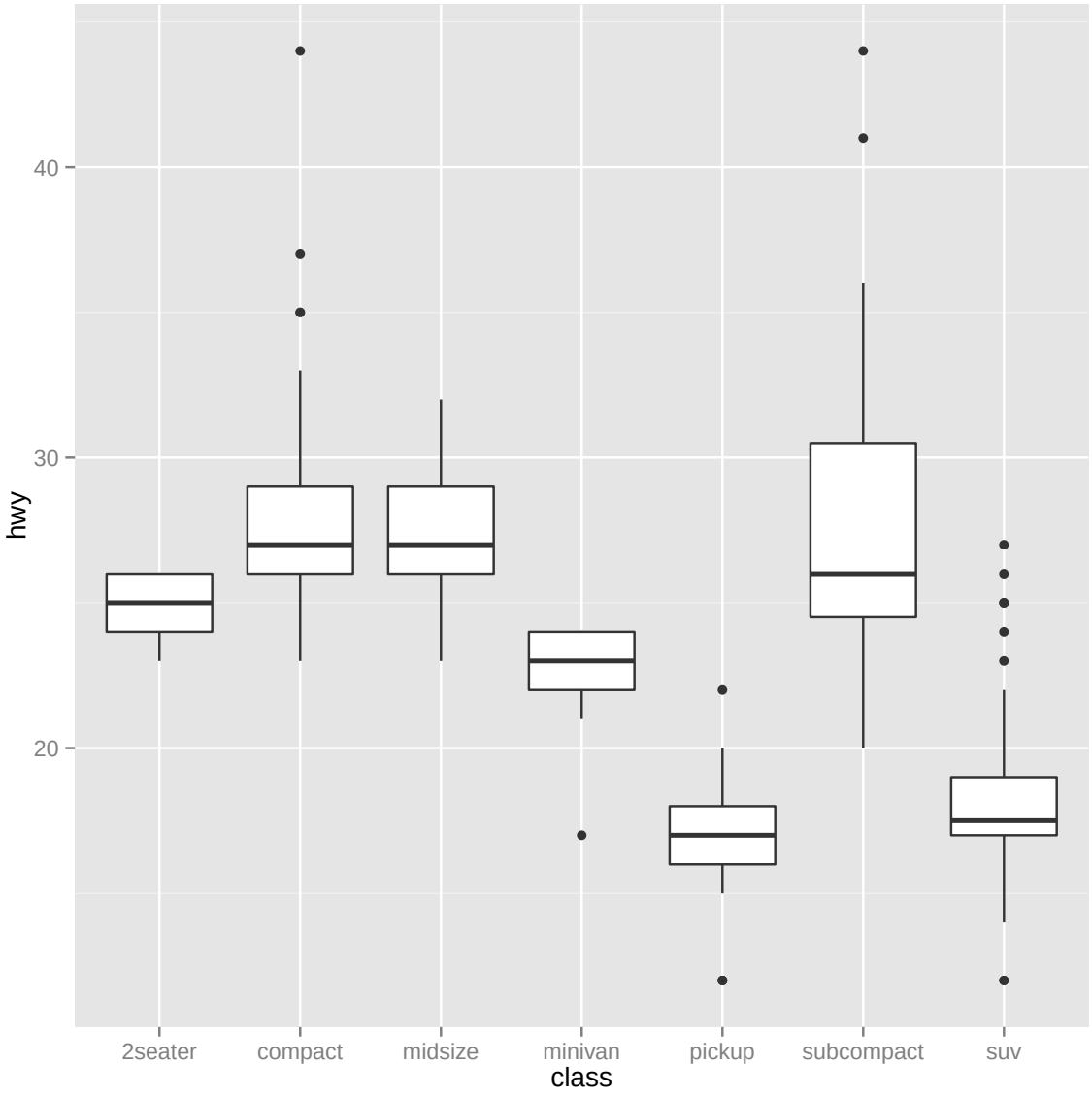

```
#angle, radius, x, xend, y, yend | ..x.., ..xend.., ..y.., ..yend..
m + stat_summary_hex(aes(z = z), bins = 30, fun = mean)
```



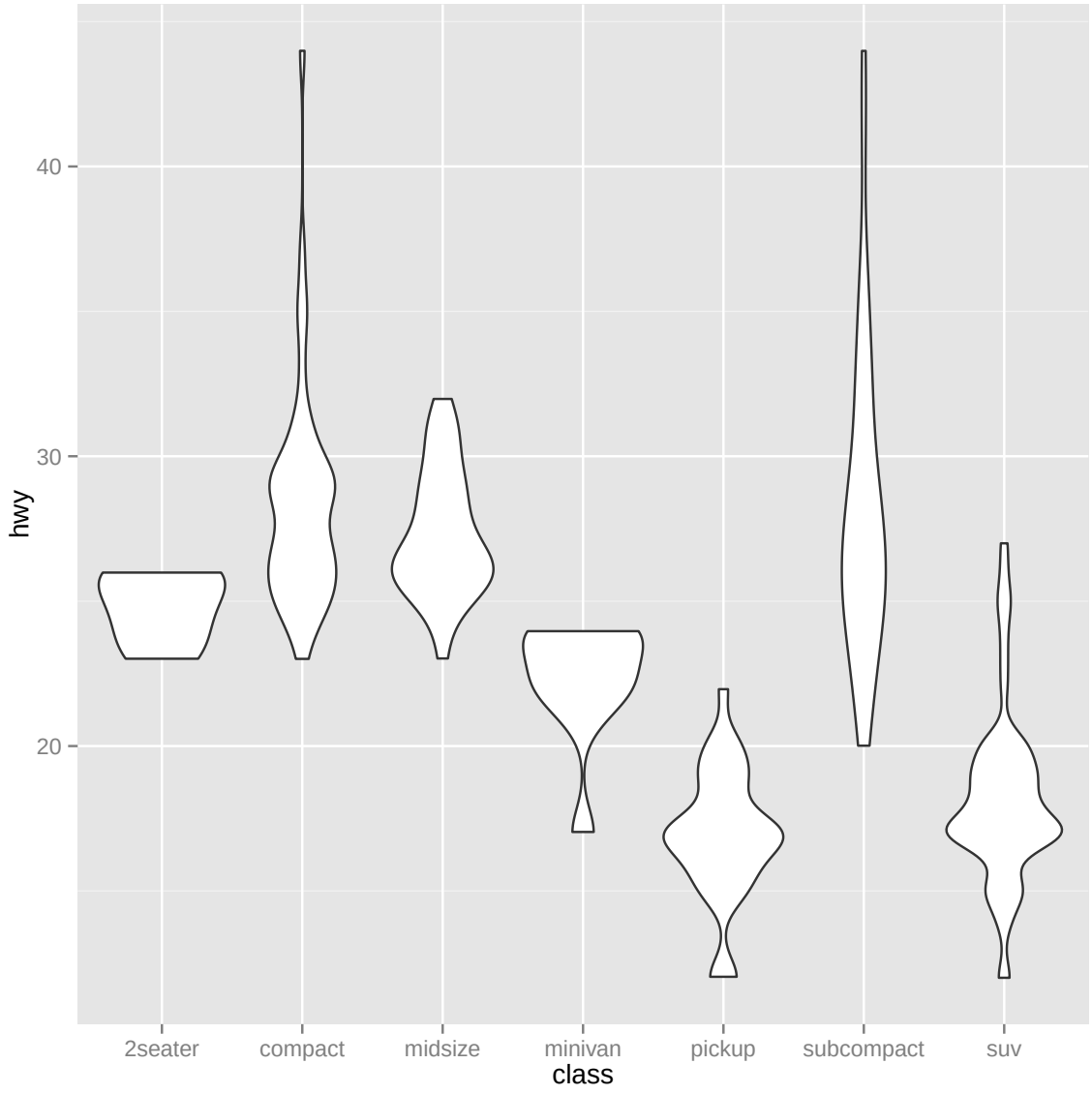

```
#x, y, z, fill | ..value..
m + stat_summary2d(aes(z = z), bins = 30, fun = mean)
```



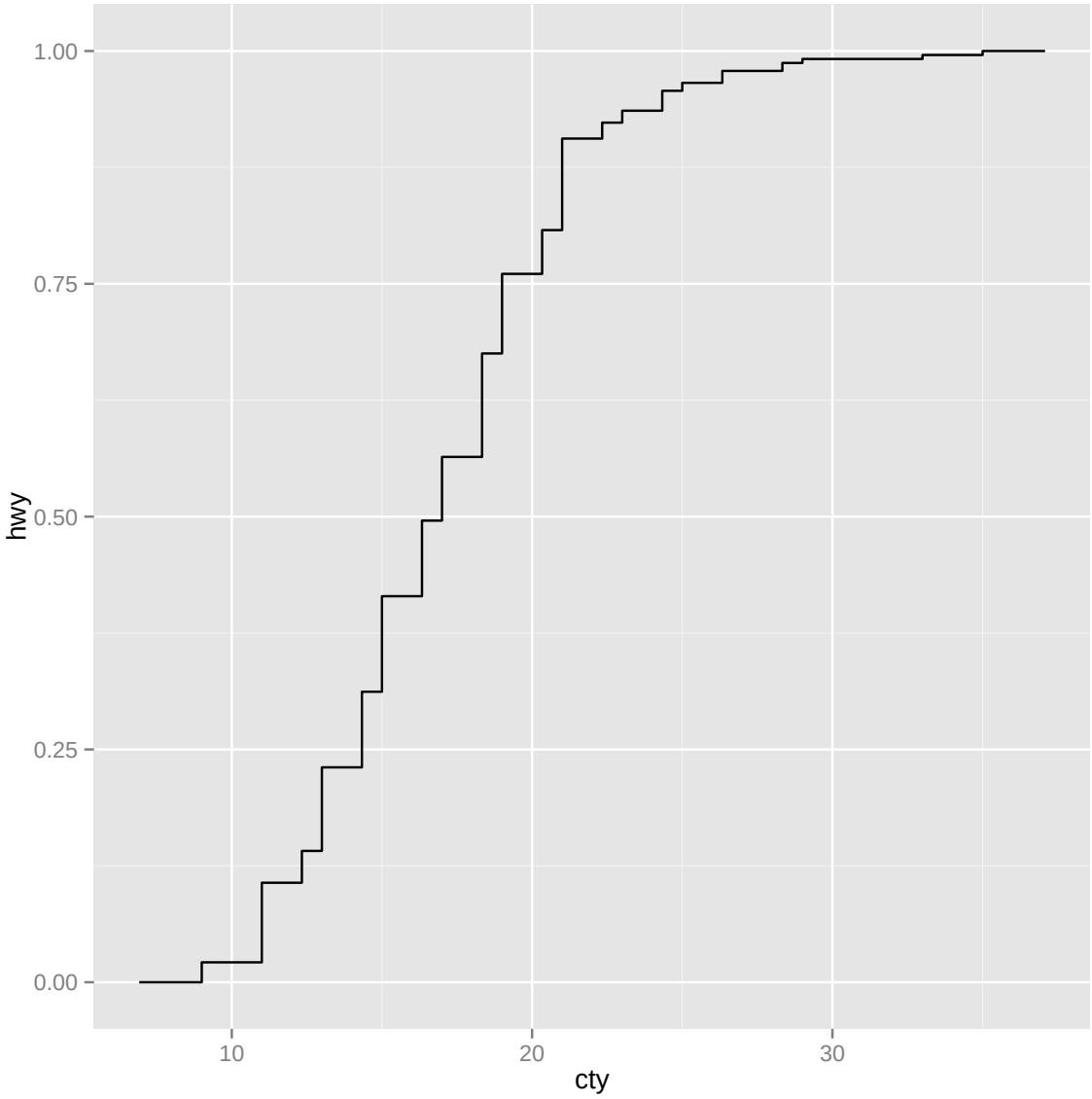

```
#x , y, z, fill | ..value..
g + stat_boxplot(coef = 1.5)
```



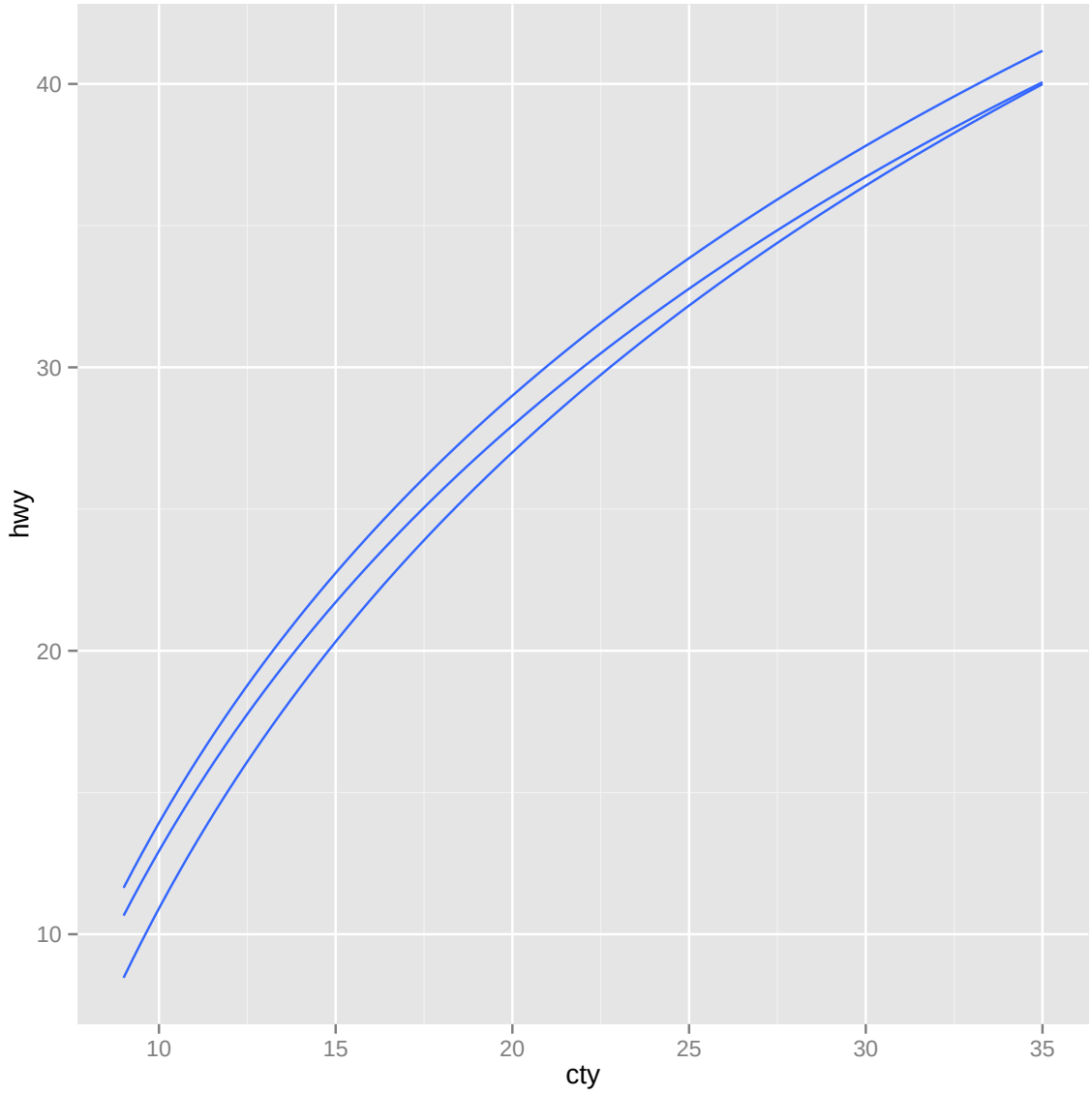

```
#x, y | ..lower.., ..middle.., ..upper.., ..outliers..
g + stat_ydensity(adjust = 1, kernel = "gaussian", scale = "area")
```



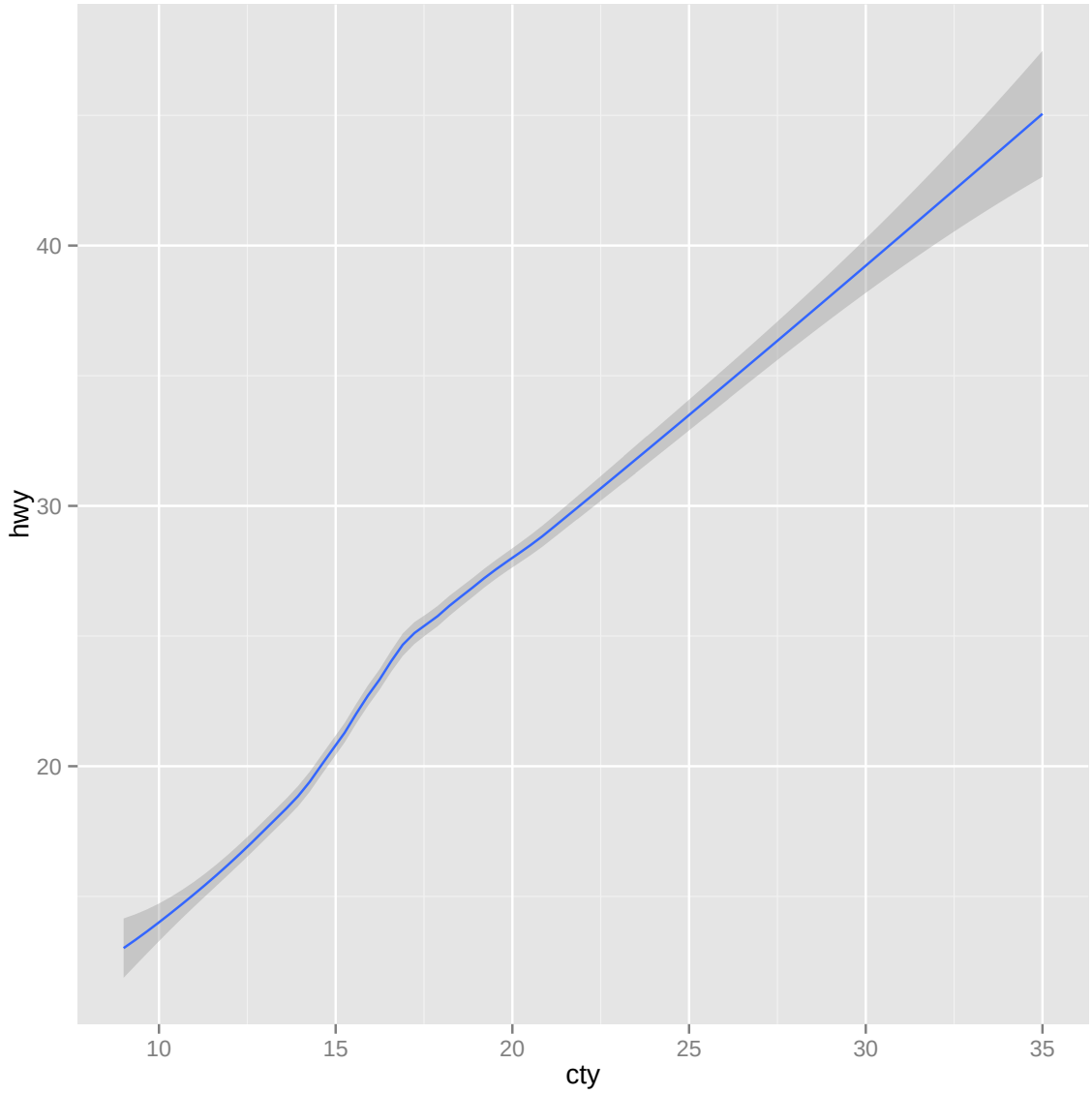

```
#x, y | ..density.., ..scaled.., ..count.., ..n.., ..violinwidth.., ..width..
f + stat_ecdf(n = 40)
```



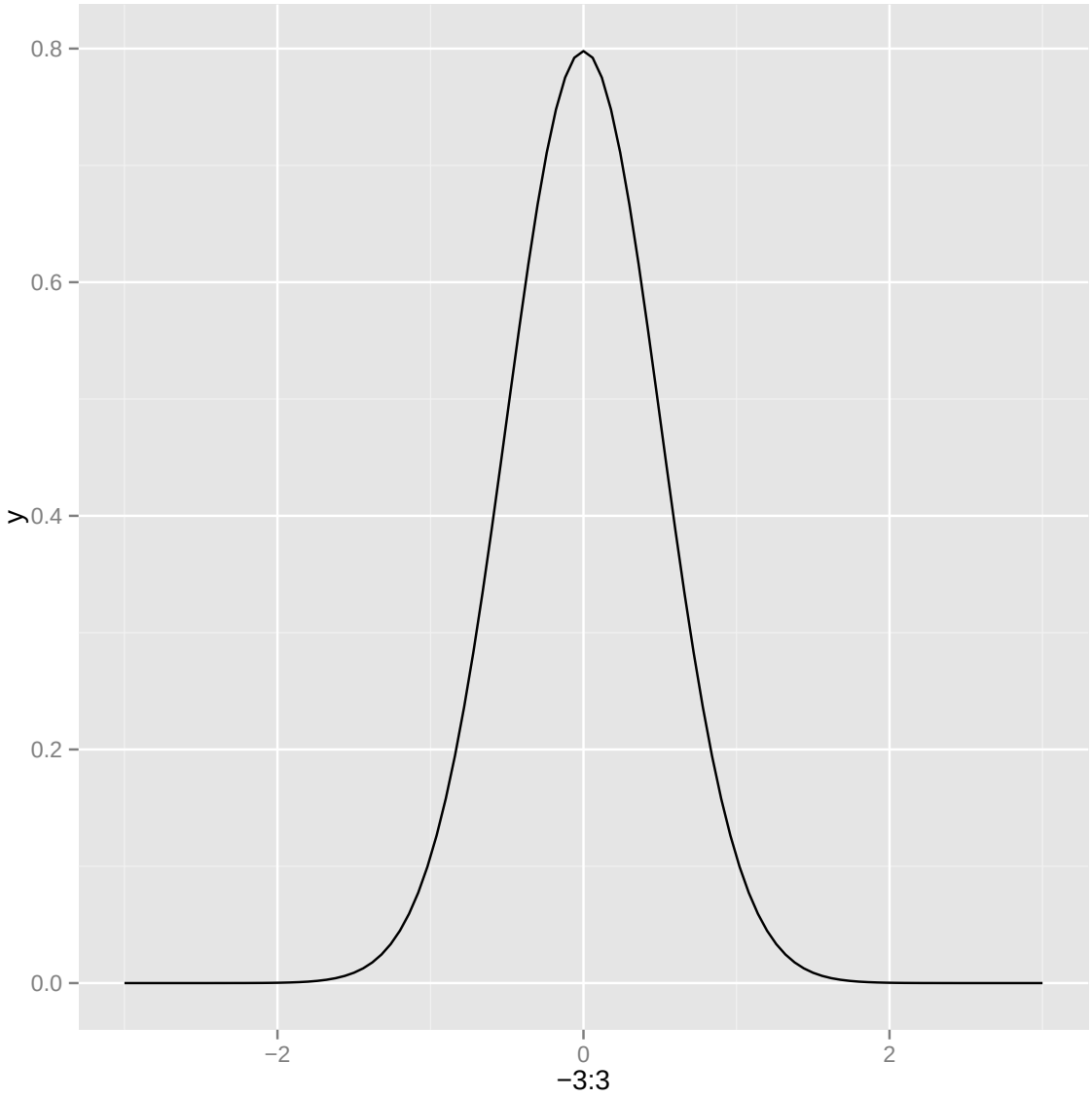

```
#x, y | ..x.., ..y..
f + stat_quantile(quantiles = c(0.25, 0.5, 0.75), formula = y ~ log(x),
method = "rq")
```




```
#x, y | ..quantile.., ..x.., ..y..
f + stat_smooth(method = "auto", formula = y ~ x, se = TRUE, n = 80,
fullrange = FALSE, level = 0.95)

## geom_smooth: method="auto" and size of largest group is <1000, so using
loess. Use 'method = x' to change the smoothing method.
```



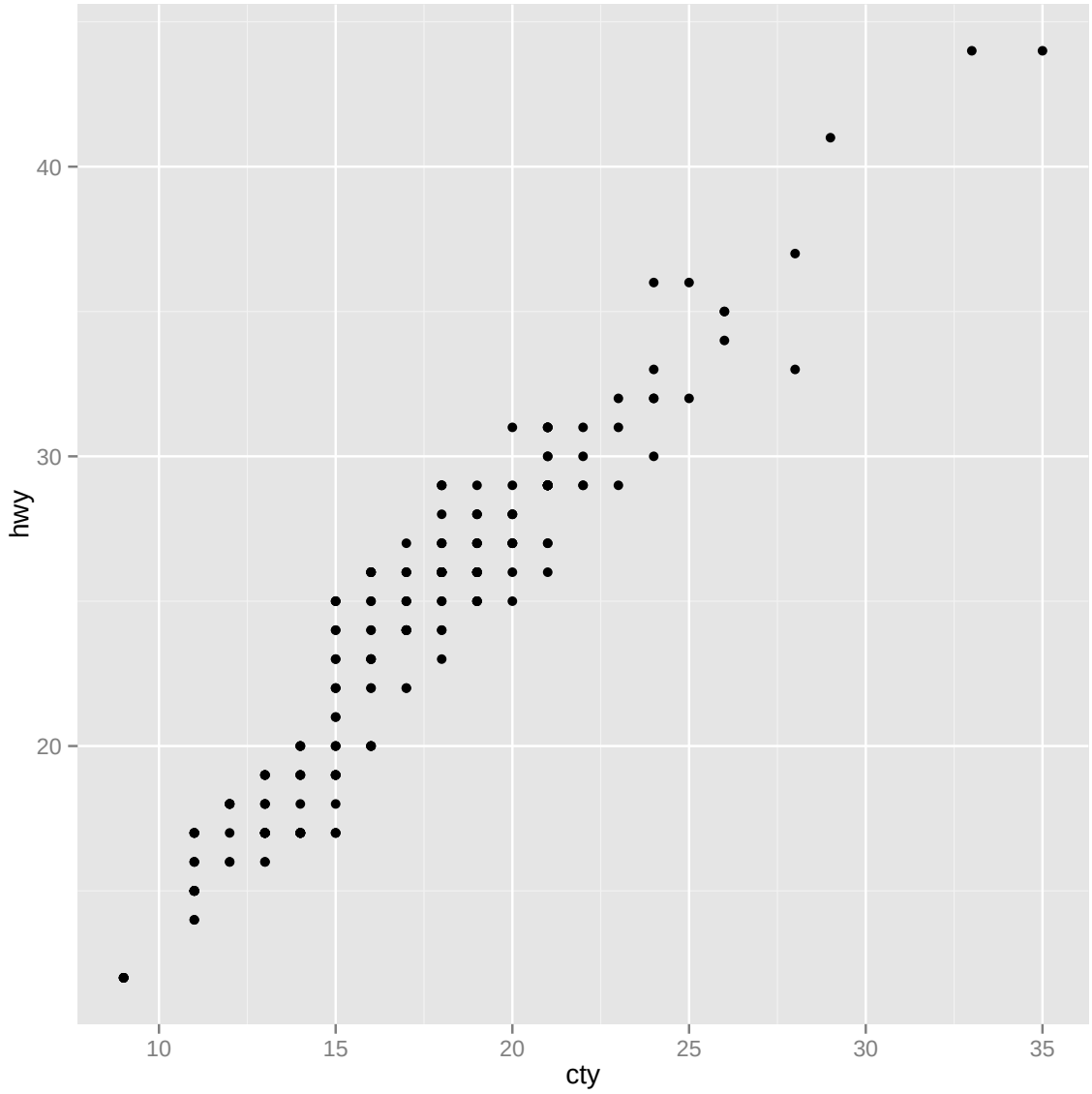

```
#x, y | ..se.., ..x.., ..y.., ..ymin.., ..ymax..
ggplot() + stat_function(aes(x = -3:3),
fun = dnorm, n = 101, args = list(sd=0.5))
```



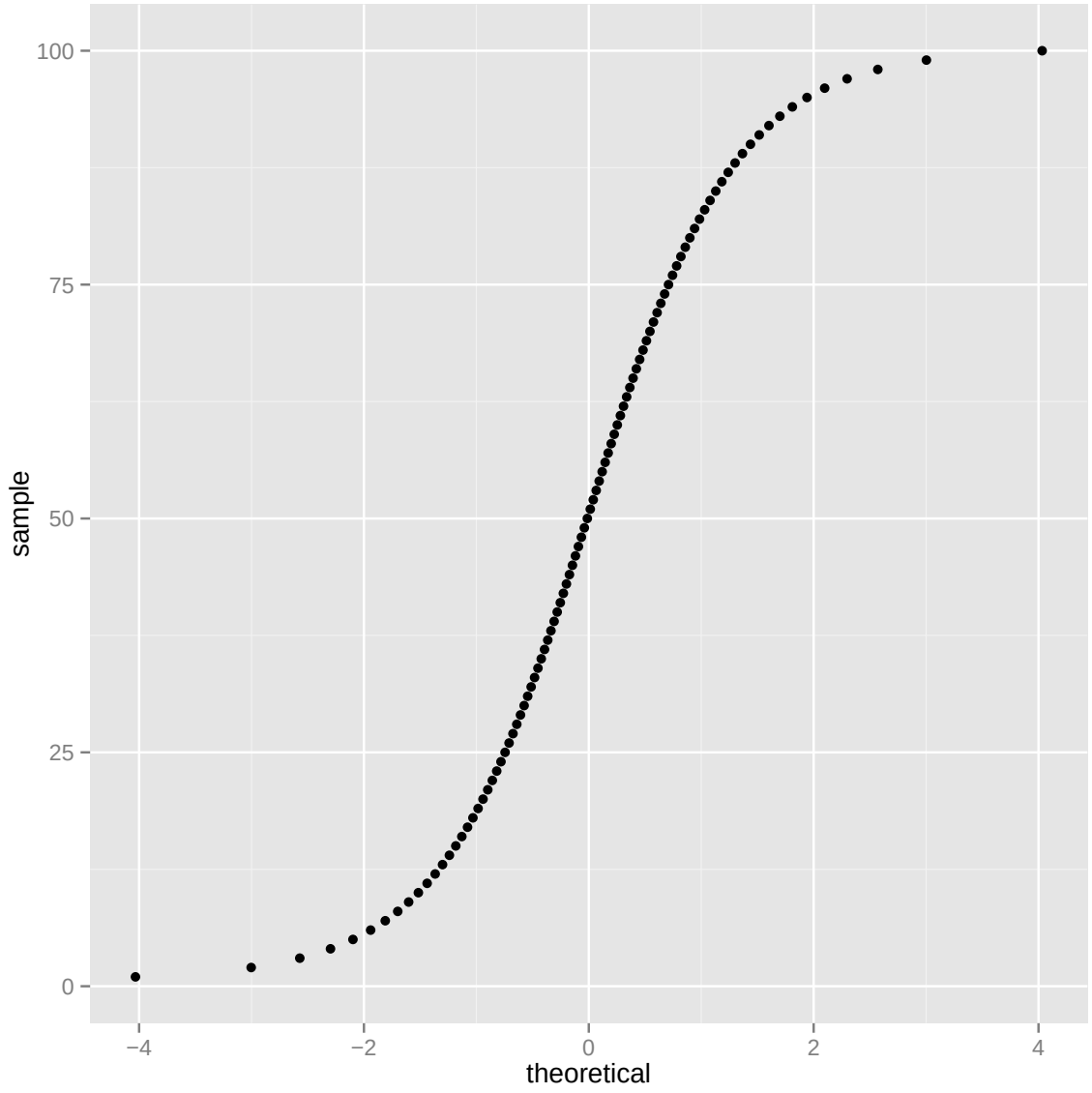

```
x | ..y..
```

```
## Error in eval(expr, envir, enclos): objeto 'x' no encontrado
```

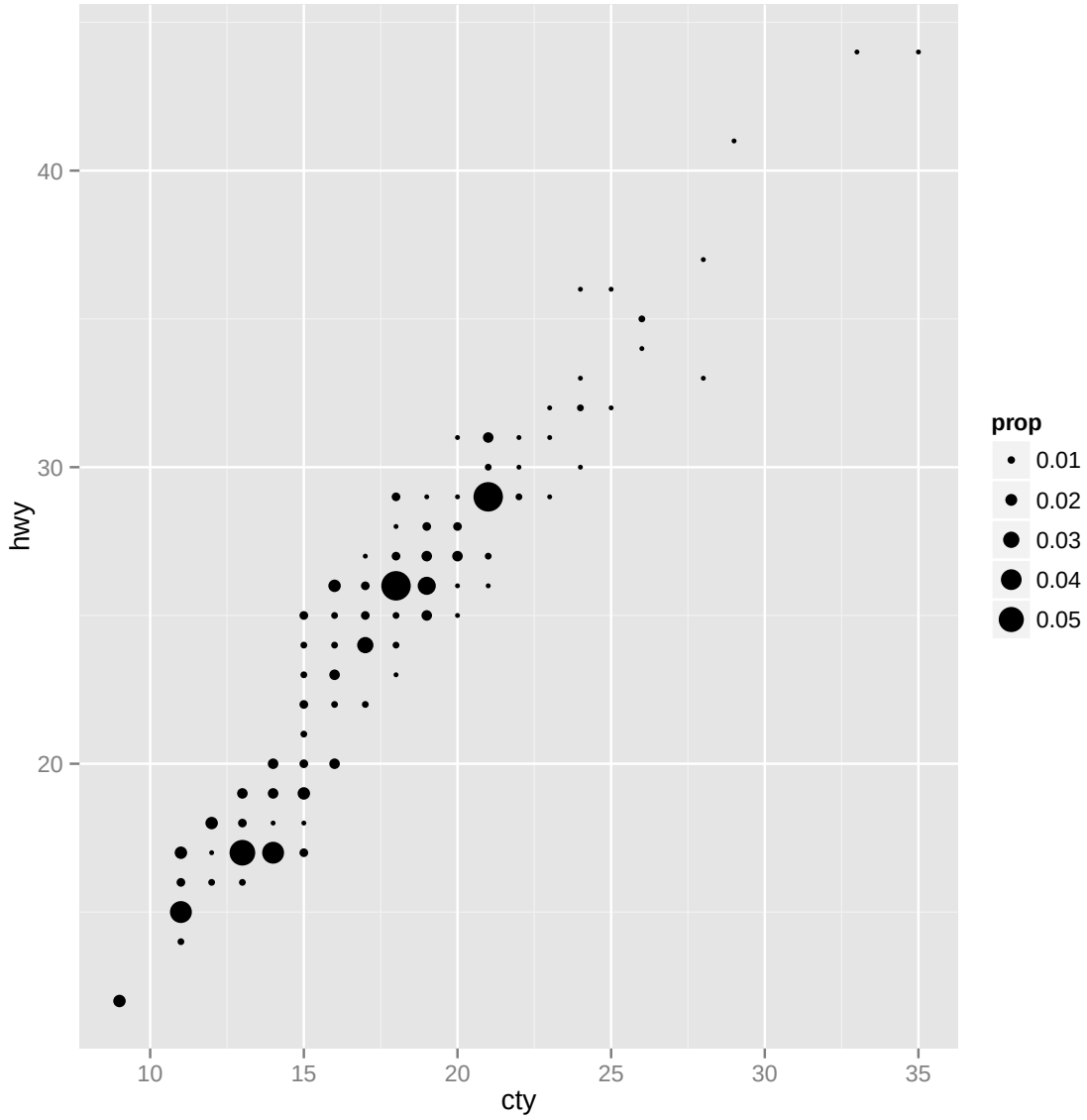
```
f + stat_identity()
```



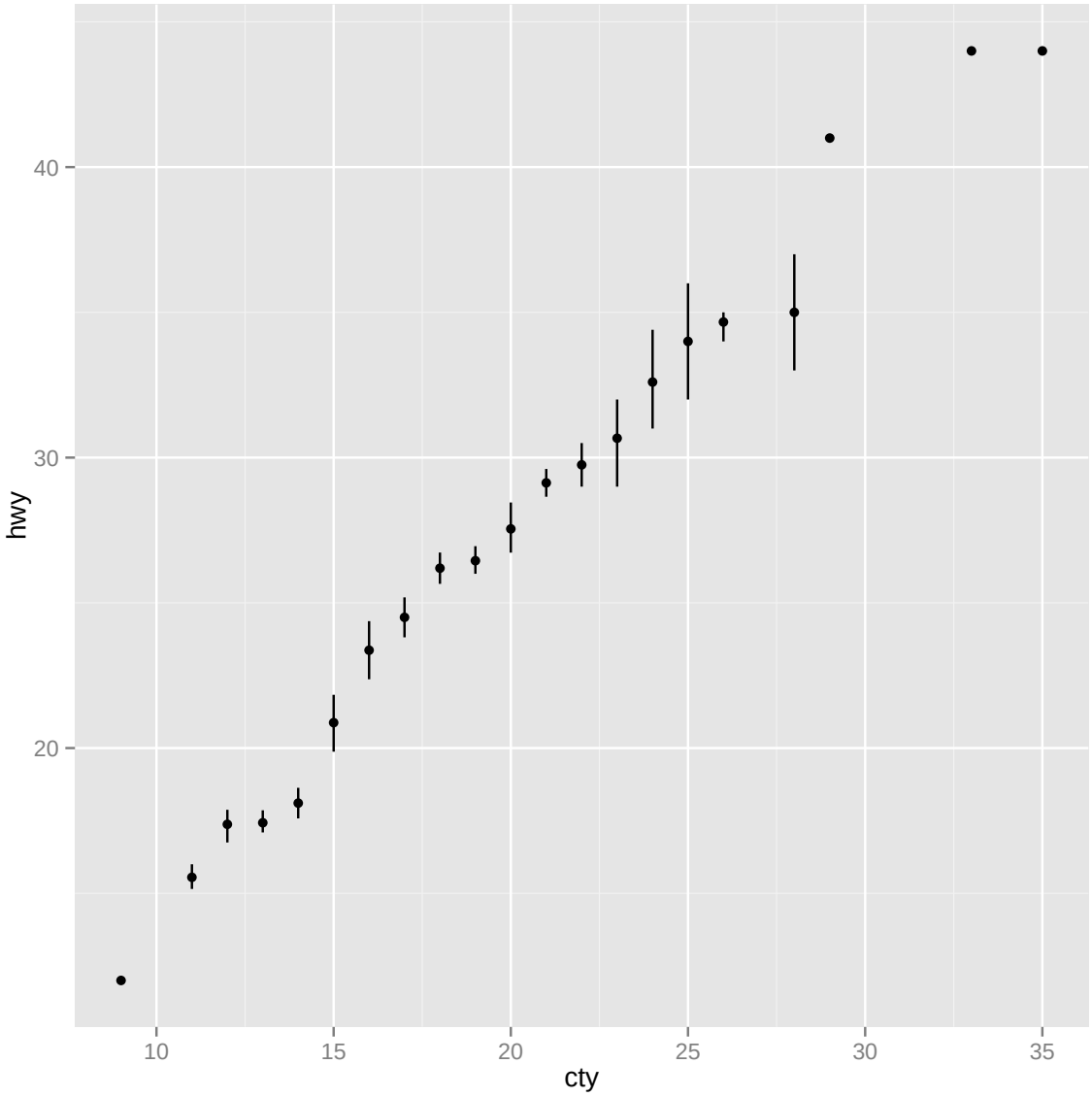

```
ggplot() + stat_qq(aes(sample=1:100), distribution = qt,
dparams = list(df=5))
```



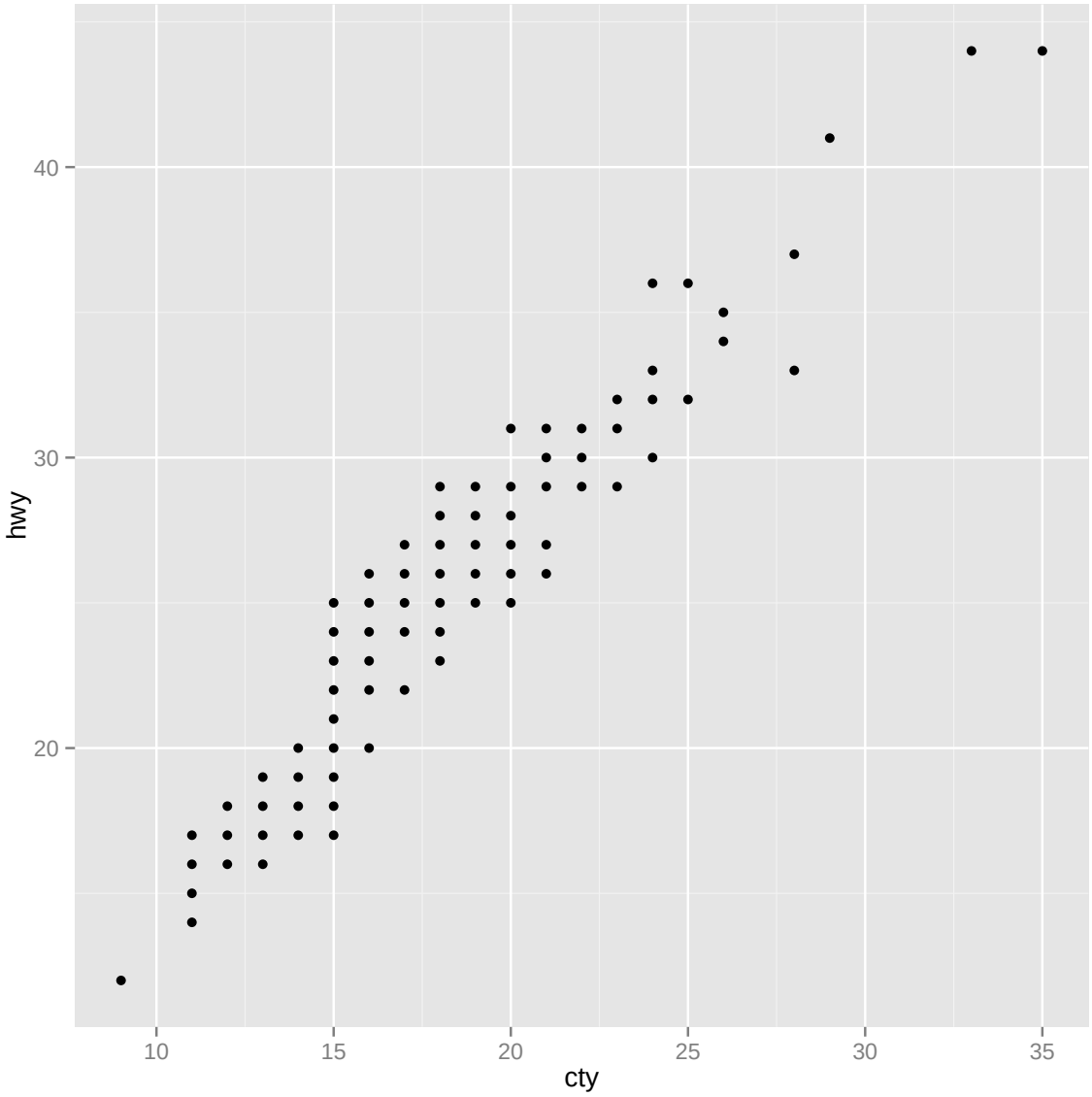

```
#sample, x, y | ..x.., ..y..
f + stat_sum()
```




```
#x, y, size | ..size..
f + stat_summary(fun.data = "mean_cl_boot")

## Warning: Removed 3 rows containing missing values (geom_segment).
```

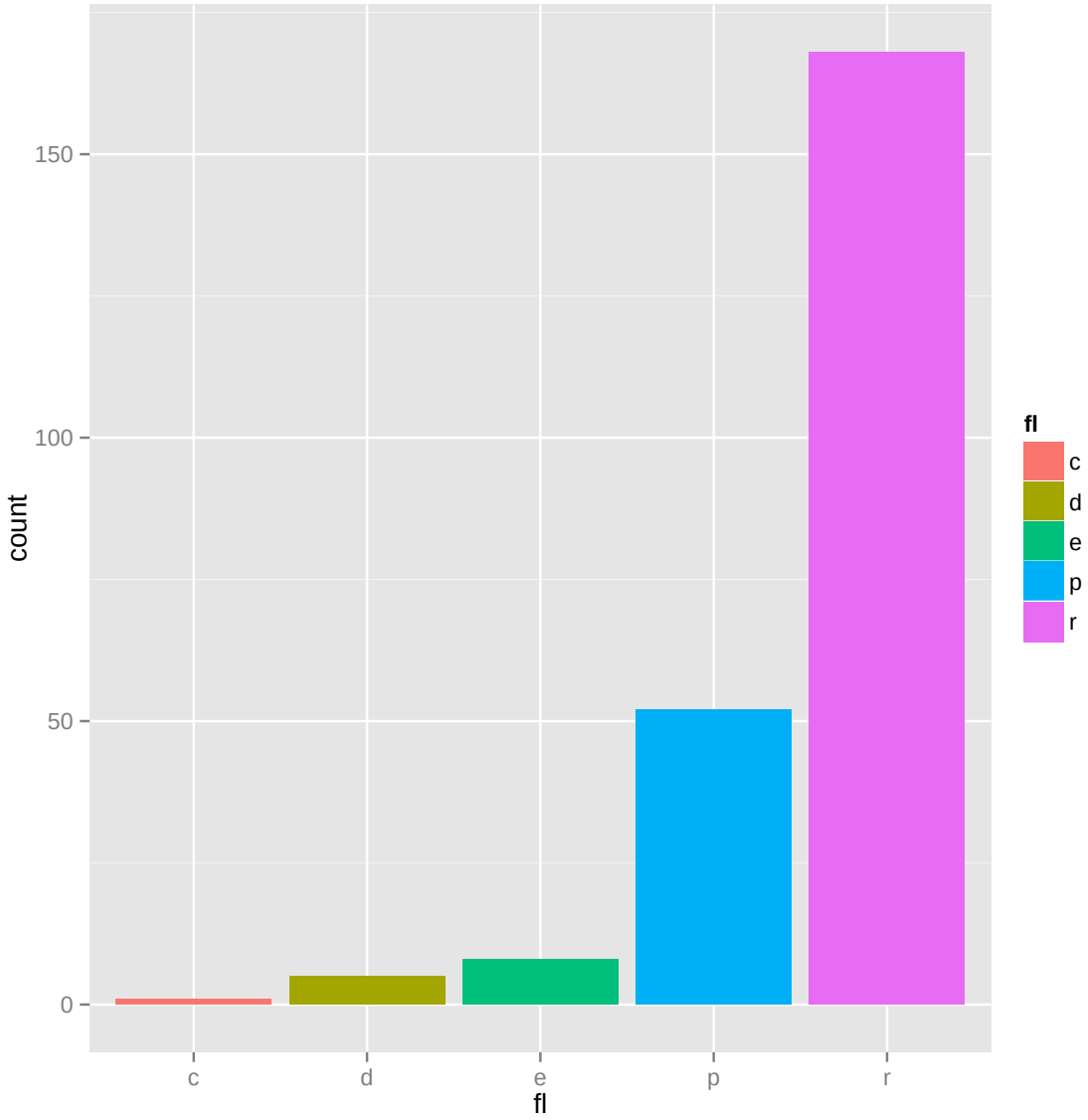



```
f + stat_unique()
```

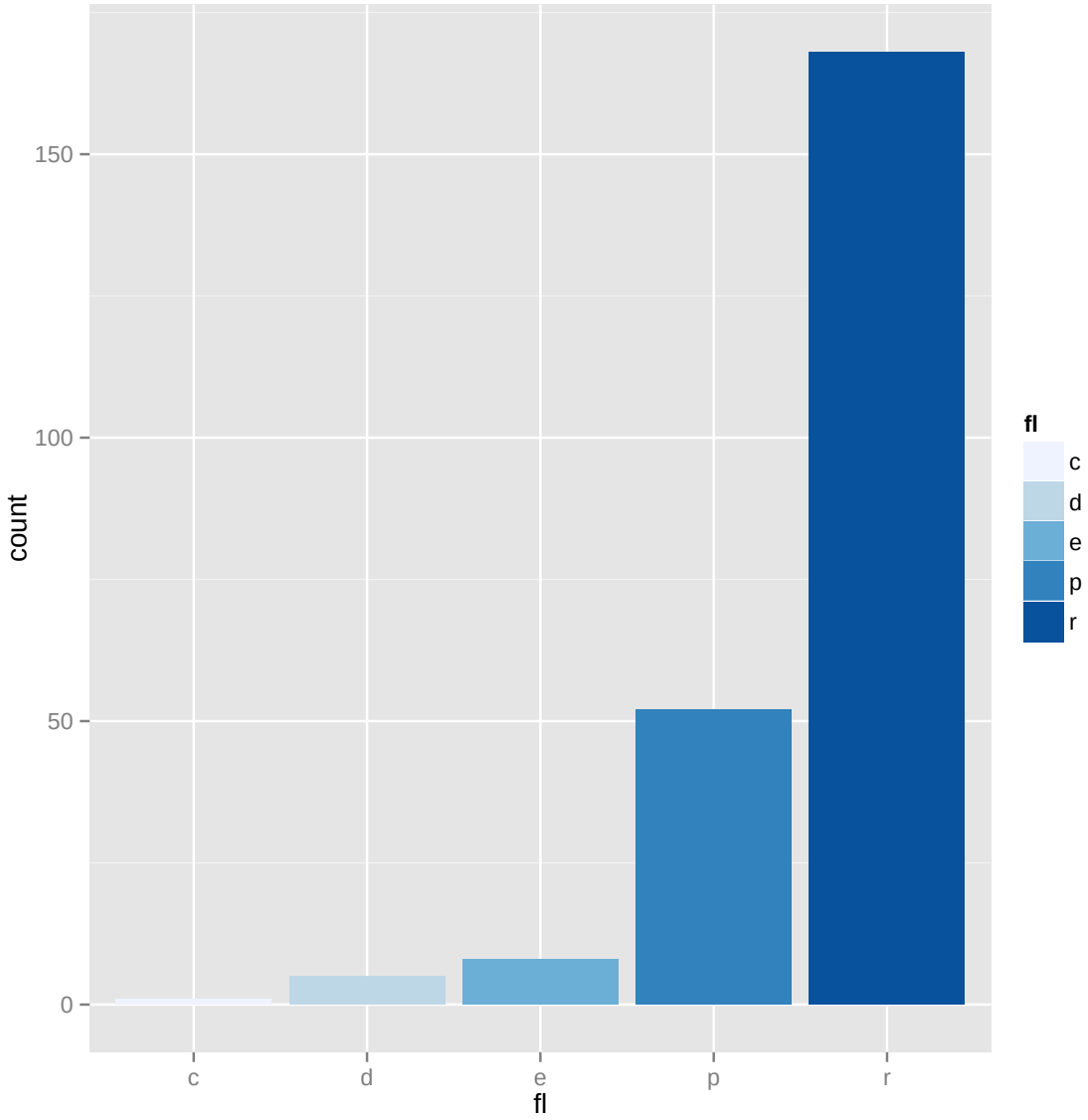


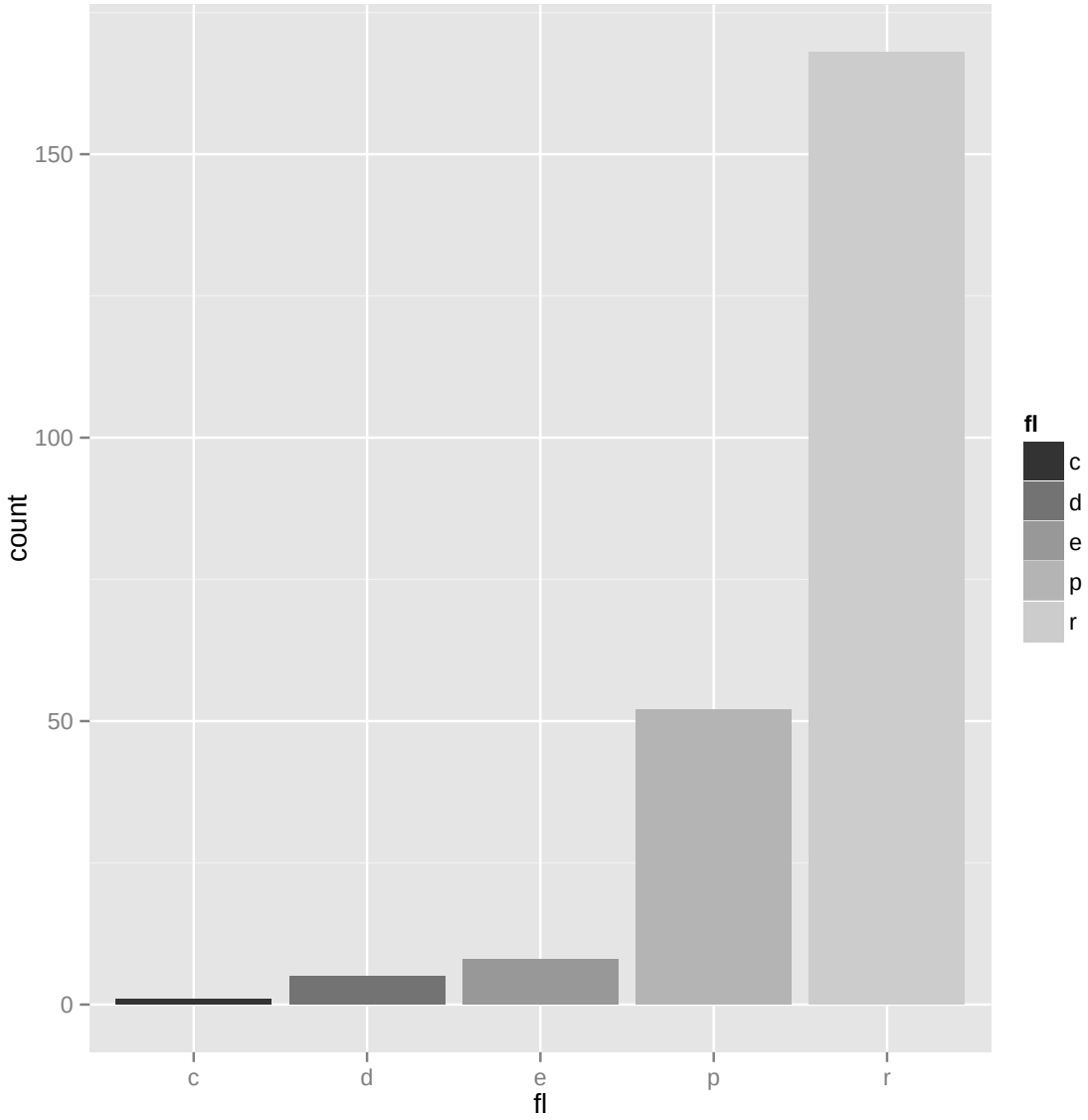
Escalas (Scales)

```
n <- b + geom_bar(aes(fill = fl))
n
```

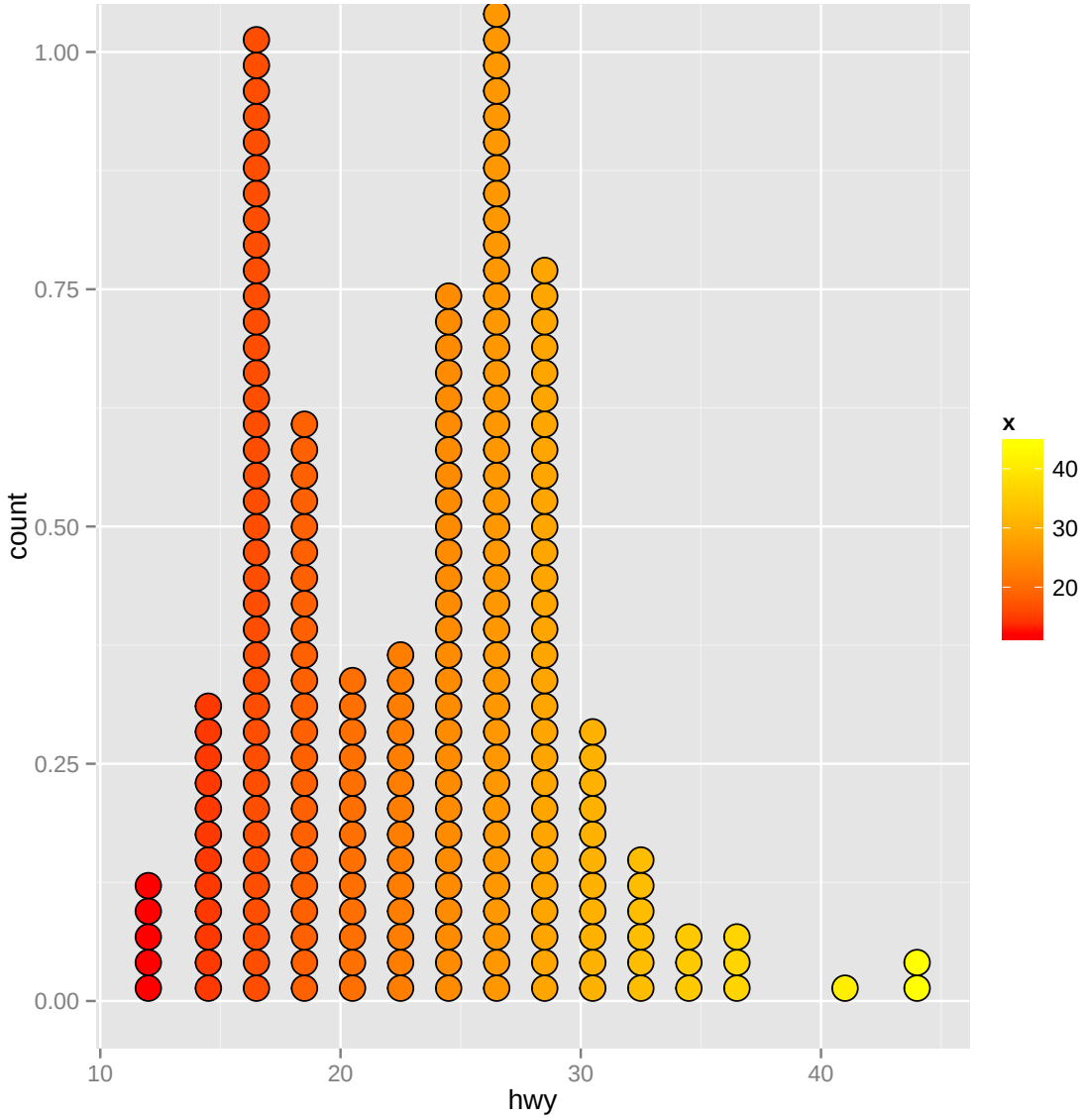



```
#Escalas de color y relleno
n <- b + geom_bar(aes(fill = fl))
n + scale_fill_brewer(palette = "Blues")
```

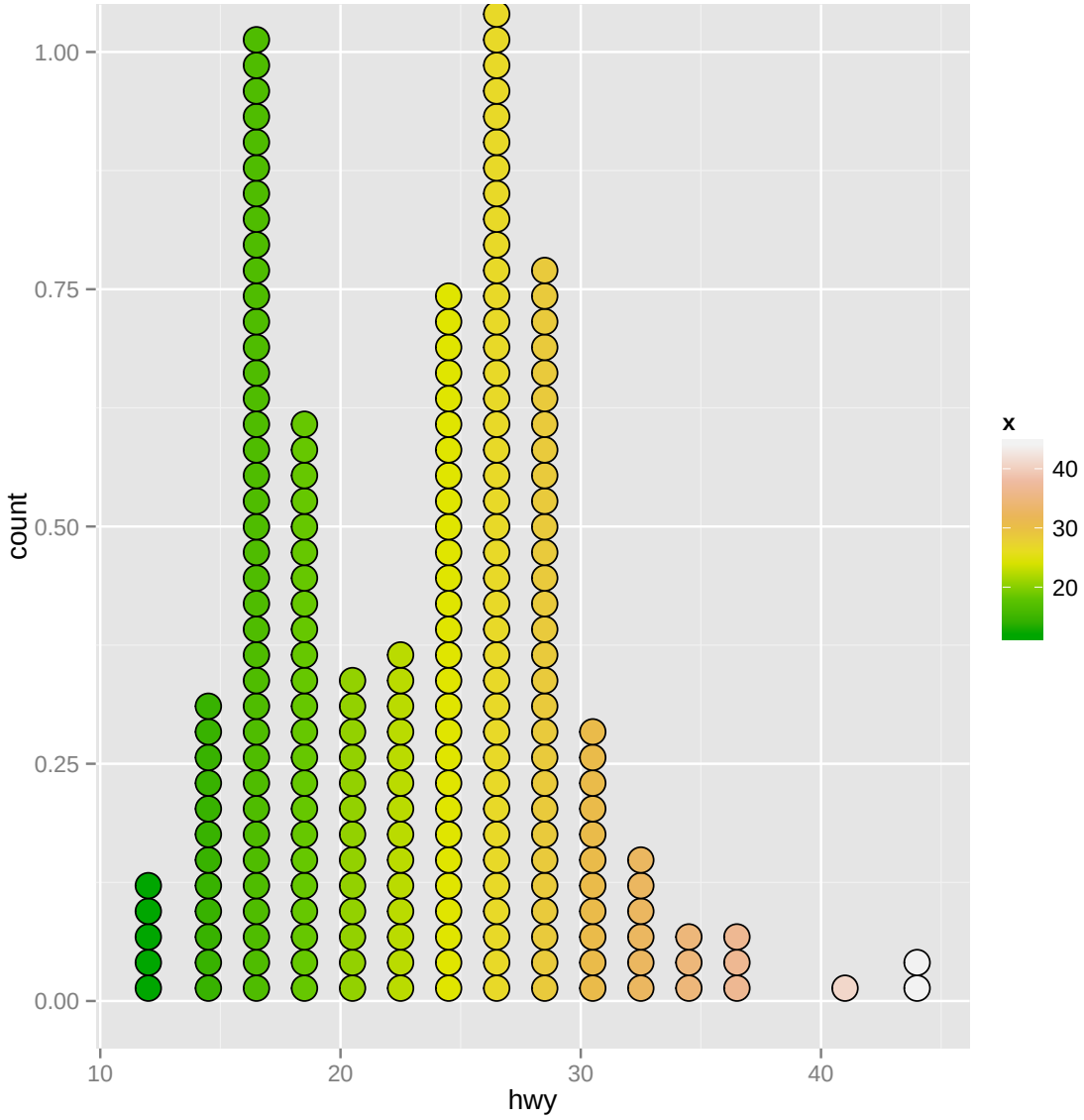

150
100
50
0
count
c
d
e
p
r
fl
fl
c
d
e
p
r



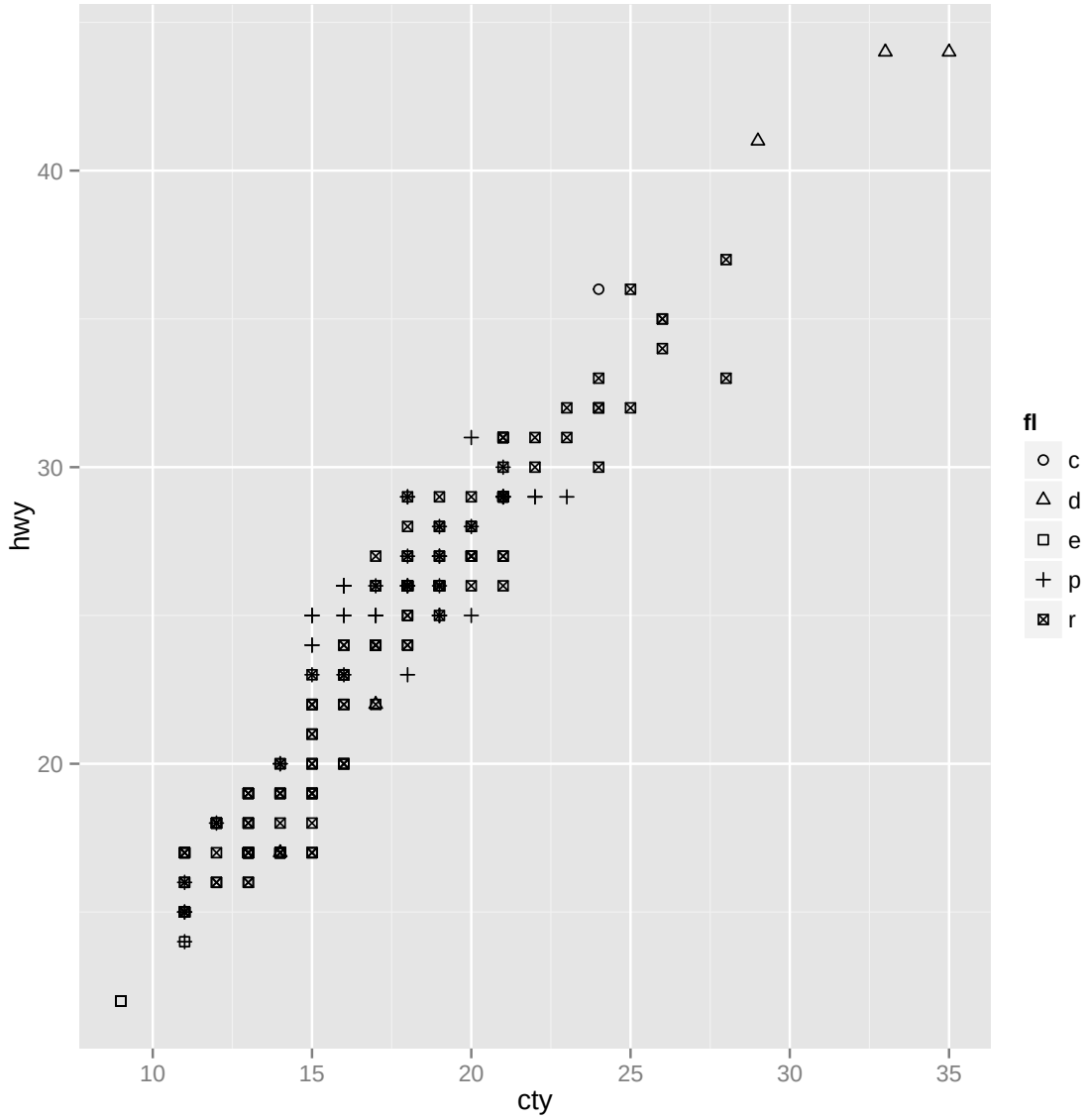

```
o <- a + geom_dotplot(aes(fill = ..x..))
o + scale_fill_gradient(low = "red", high = "yellow")

## stat_bindot: binwidth defaulted to range/30. Use 'binwidth = x' to adjust
this.
```



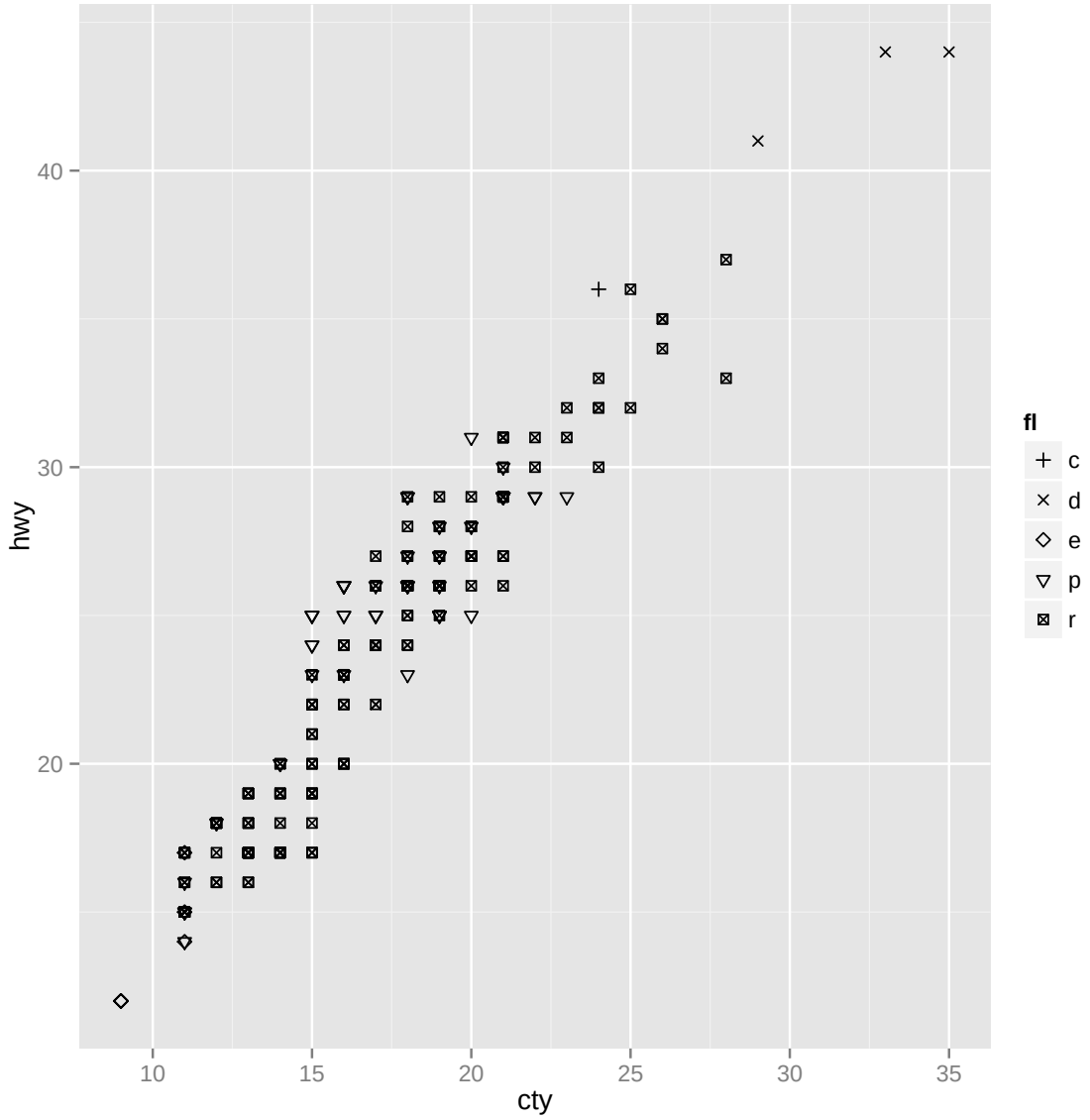

```
o + scale_fill_gradient2(low = "red", hight = "blue",mid = "white", midpoint = 25

## Error in continuous_scale("fill", "gradient2", div_gradient_pal(low, mid,
: unused argument (hight = "blue")

o + scale_fill_gradientn(colours = terrain.colors(6))

## stat_bindot: binwidth defaulted to range/30. Use 'binwidth = x' to adjust
this.
```




```
#TambiÃ©n: rainbow(),
#heat.colors(), topo.colors(),
#cm.colors(),
#RColorBrewer::brewer.pal()

#Formas para las Escalas
p <- f + geom_point(aes(shape = fl))
p + scale_shape(solid = FALSE)
```




```
p + scale_shape_manual(values = c(3:7))
```




```
#Se muestran valores de
#la forma a la derecha del
#grÃ¡fico

#tamaÃ±o de Escalas
q <- f + geom_point(aes(size = cyl))
q + scale_size_area(max = 6)

## Error in continuous_scale("size", "area", palette = abs_area(max_size),
: unused argument (max = 6)

#Valor mapeado al Ã¡rea del cÃrculo(no el radio)
```