semana 1

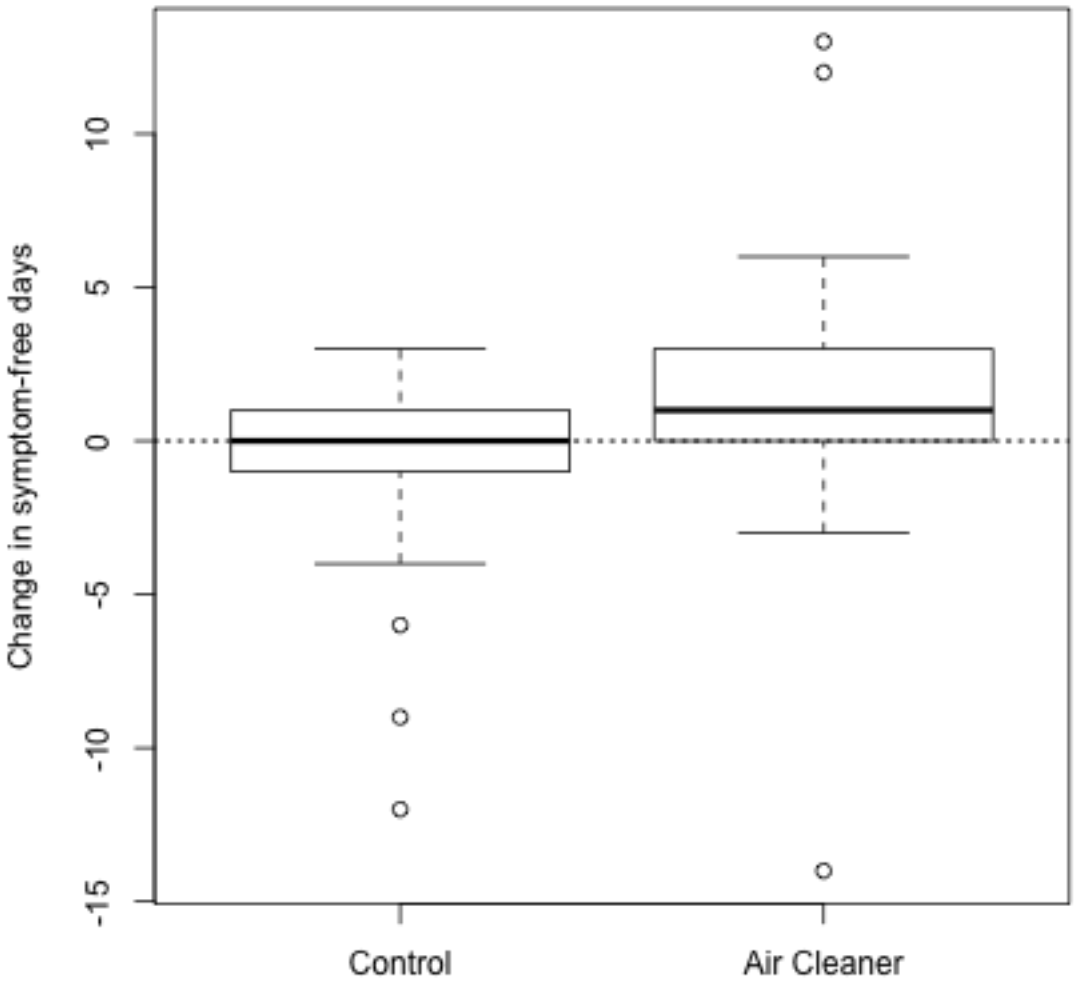
Luis Ambrocio

15/8/2021

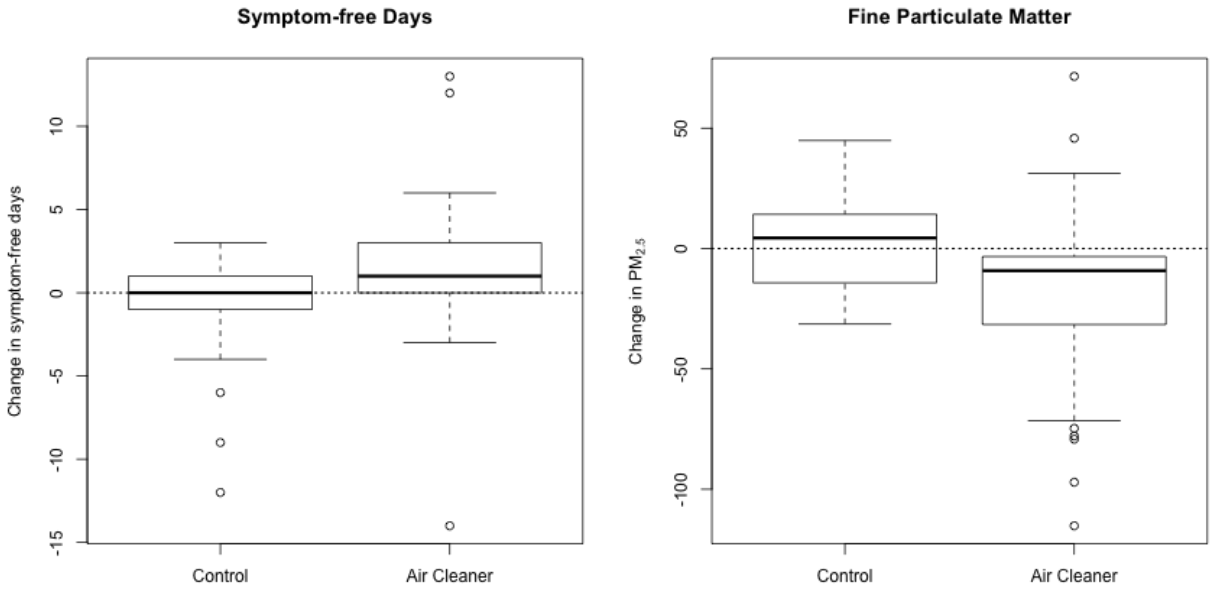
Contents

Principios de los graficos analiticos

- Principio 1: Mostrar comparaciones
  - La evidencia de una hipótesis es siempre * relativa * a otra hipótesis en competencia.
  - Pregunte siempre "¿Comparado con qué?"

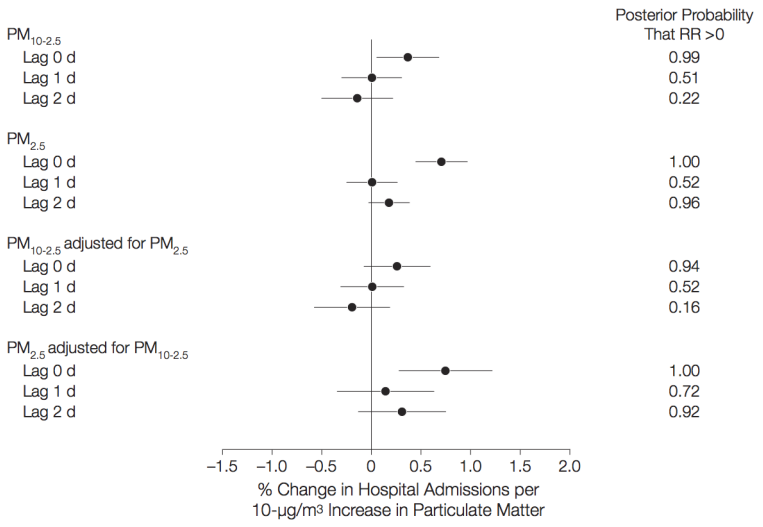


- Principio 2: Mostrar causalidad, mecanismo, explicación, estructura sistemática.
  - ¿Cuál es su marco causal para pensar en una pregunta?



- Principio 3: Mostrar datos multivariados
  - Multivariante = más de 2 variables
  - El mundo real es multivariado.
  - Necesidad de "escapar de la llanura"

Figure 2. Percentage Change in Emergency Hospital Admissions Rate for Cardiovascular Diseases per a 10-µg/m³ Increase in Particulate Matter



Estimates are on average across 108 counties. $PM_{2.5}$ indicates particulate matter is 2.5 µm or less in aerodynamic diameter; PM_{10}, particulate matter is 10 µm or less in aerodynamic diameter; $PM_{10\text{-}2.5}$, particulate matter is greater than 2.5 µm and 10 µm or less in aerodynamic diameter; RR, relative risk. Error bars indicate 95% posterior intervals.

- Principio 4: Integración de evidencia
  - Integra completamente palabras, números, imágenes, diagramas
  - Los gráficos de datos deben hacer uso de muchos modos de presentación de datos.

  - No dejes que la herramienta dirija el análisis.
- Principio 5: Describe y documenta la evidencia con etiquetas, escalas, fuentes, etc. apropiadas.
  - Un gráfico de datos debe contar una historia completa que sea creíble.
- Principio 6: El contenido es el rey
  - Las presentaciones analíticas finalmente se mantienen o caen dependiendo de la calidad, relevancia e integridad de su contenido.

Gráficos exploratorios

¿Por qué utilizamos gráficos en el análisis de datos?

- Entender las propiedades de los datos
- Para encontrar patrones en los datos
- Sugerir estrategias de modelado
- Para "depurar" análisis
- Comunicar resultados

Características de los gráficos exploratorios

- Se hacen rapidamente
- Se hacen un gran número
- El objetivo es la comprensión personal.
- Los ejes/leyendas generalmente se limpian (más adelante)
- El color/tamaño se utilizan principalmente para información

ejemplo

Contaminación del aire en los Estados Unidos

- La Agencia de Protección Ambiental de los Estados Unidos (EPA) establece estándares nacionales de calidad del aire ambiental para la contaminación del aire exterior.
  - Estándares nacionales de calidad del aire ambiental de EE. UU.
- Para la contaminación por partículas finas (PM2.5), la "media anual, promediada durante 3 años" no puede exceder los 12 $\mu g/m^3$.
- Los datos sobre PM2.5 diario están disponibles en el sitio web de la EPA de EE. UU.
  - Sistema de calidad del aire de la EPA
- **Pregunta**: ¿Hay condados en los EE. UU. Que excedan ese estándar nacional para la contaminación por partículas finas?

PM2.5 promedio anual promediado durante el período 2008 a 2010

```
pollution<-read.csv("D:/luism/Documents/courses/04_ExploratoryAnalysis/semana 1/exploratoryGraphs/data/a
head(pollution)
```

```
##        pm25  fips region longitude latitude
## 1  9.771185 01003   east -87.74826 30.59278
## 2  9.993817 01027   east -85.84286 33.26581
## 3 10.688618 01033   east -87.72596 34.73148
## 4 11.337424 01049   east -85.79892 34.45913
## 5 12.119764 01055   east -86.03212 34.01860
## 6 10.827805 01069   east -85.35039 31.18973
```

Resúmenes simples de datos: Una dimensión

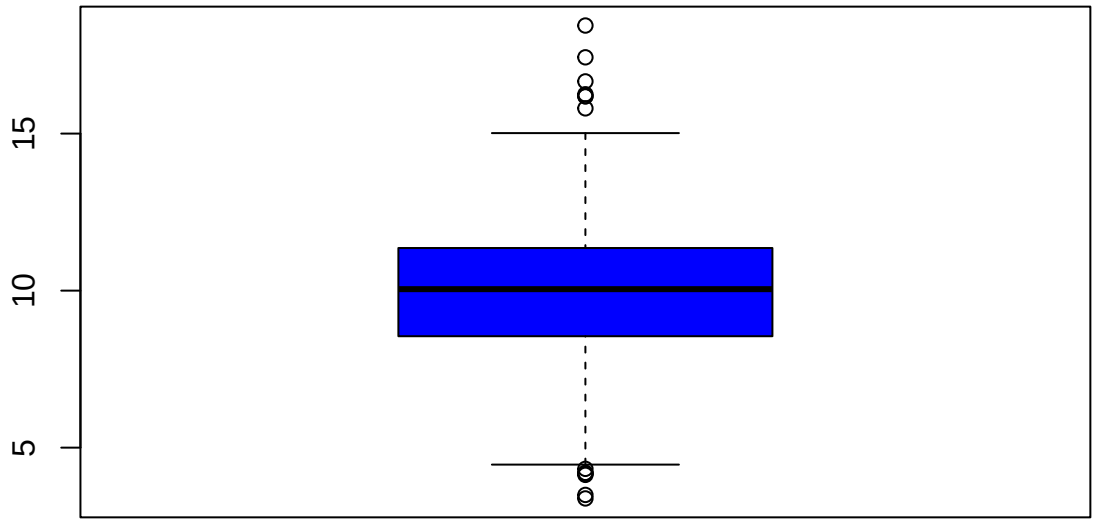
- Resumen de cinco números

```
summary(pollution$pm25)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.383   8.549  10.047   9.836  11.356  18.441
```

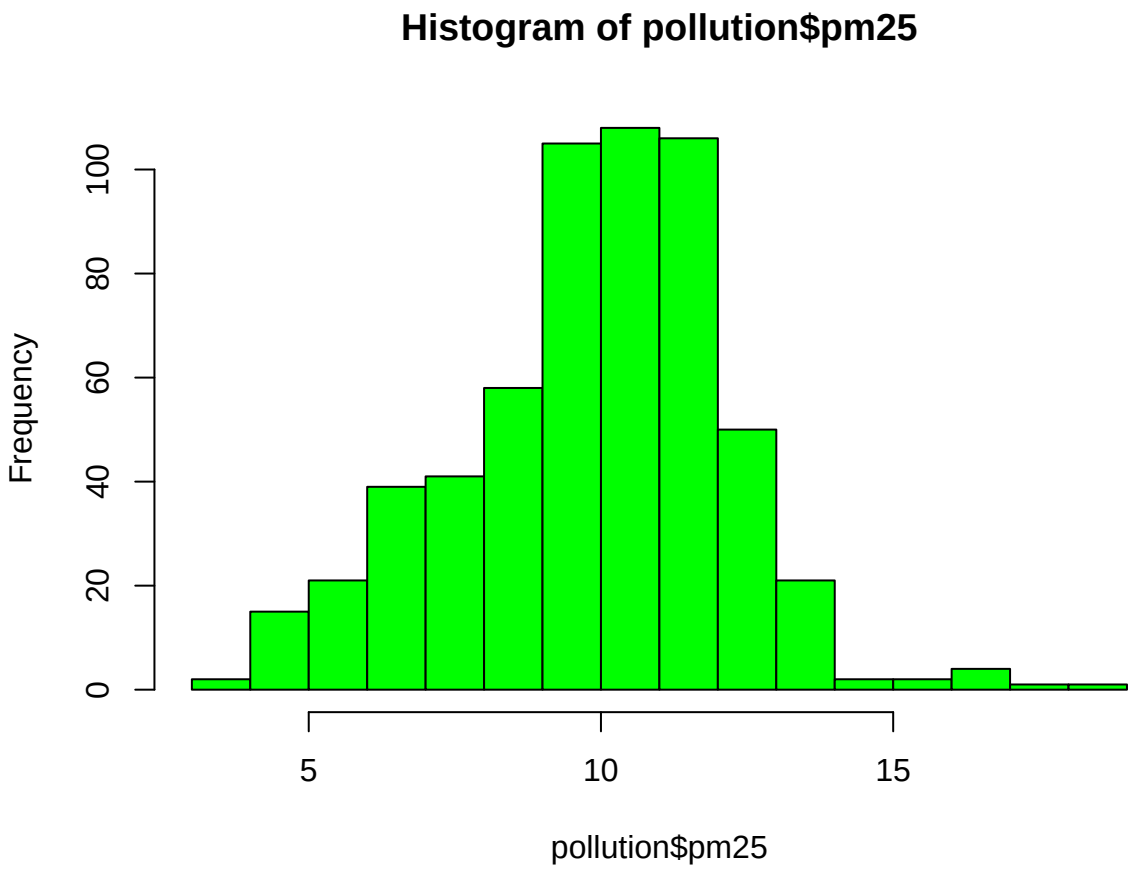
- Diagramas de caja

```
boxplot(pollution$pm25, col = "blue")
```

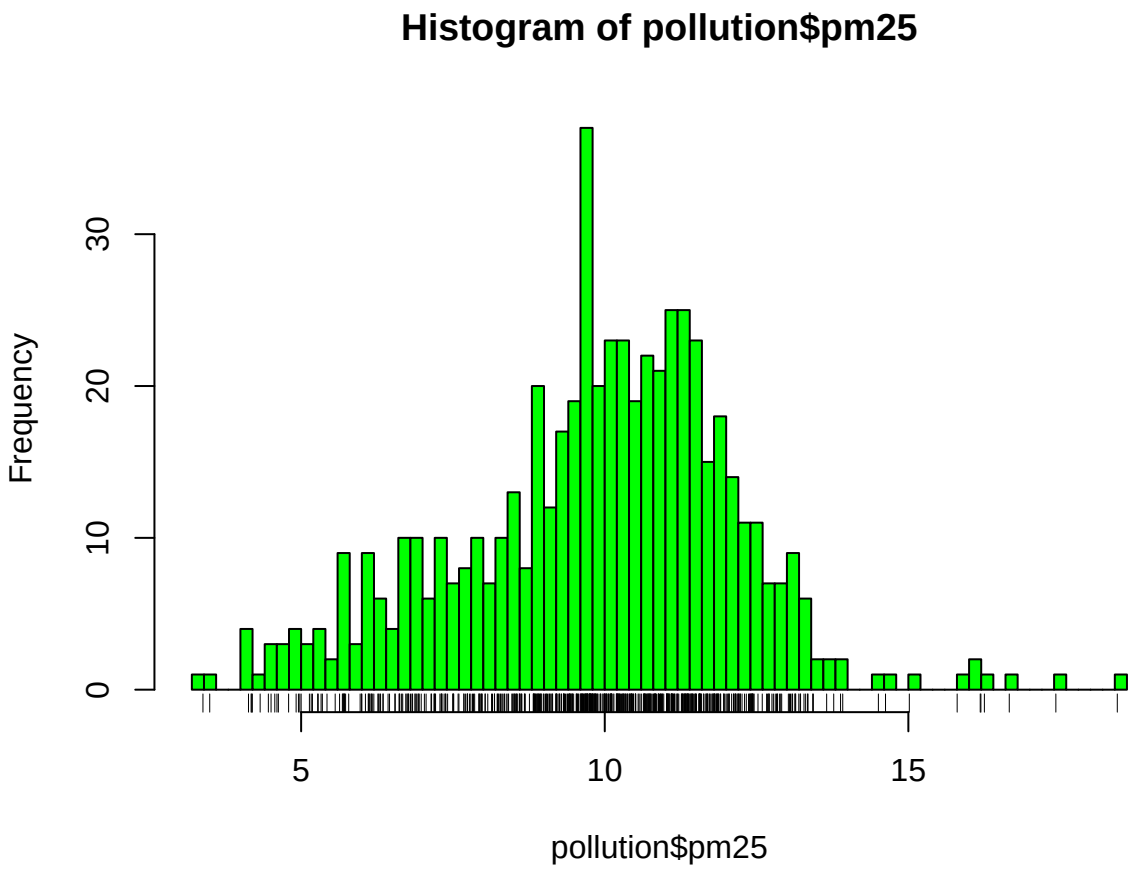


- Histogramas

```
hist(pollution$pm25, col = "green")
```

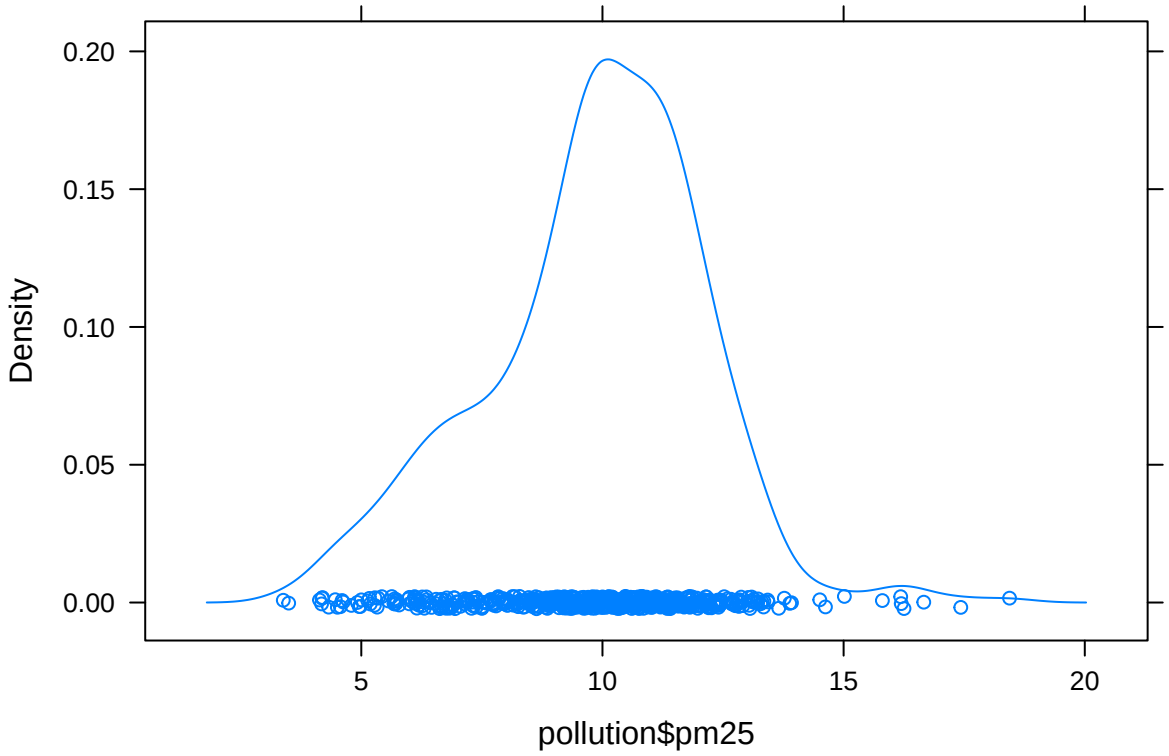



```
hist(pollution$pm25, col = "green", breaks = 100)
rug(pollution$pm25)
```



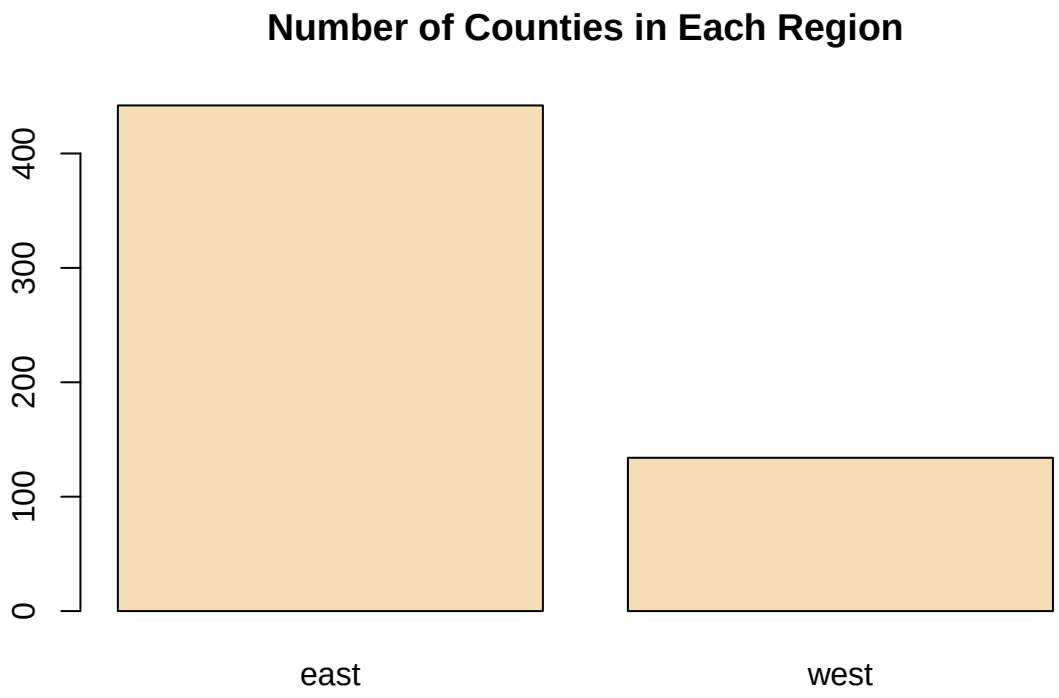
- Gráfico de densidad

```
library(lattice)
densityplot(pollution$pm25)
```



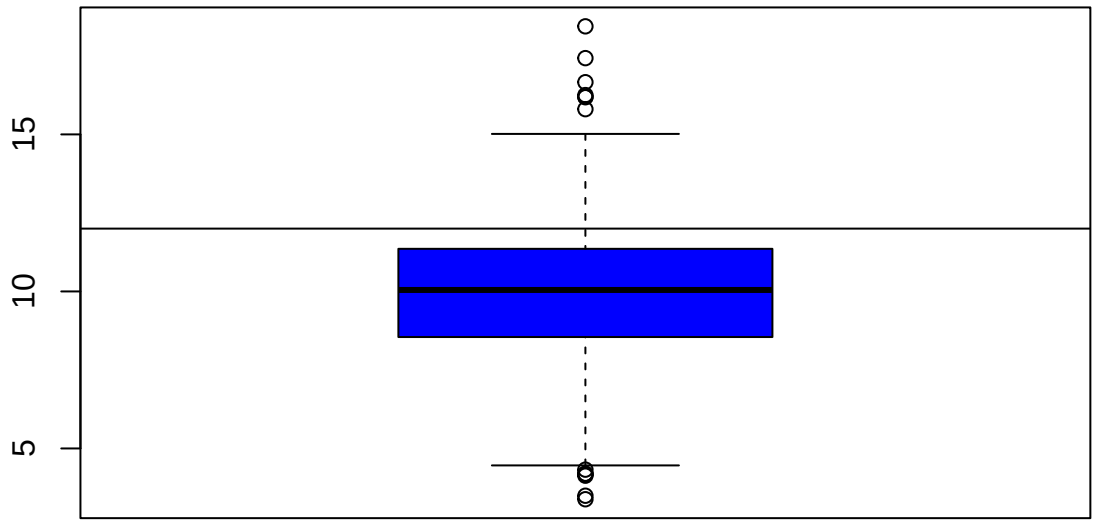
- Gráfico de barras

```
barplot(table(pollution$region), col = "wheat", main = "Number of Counties in Each Region")
```

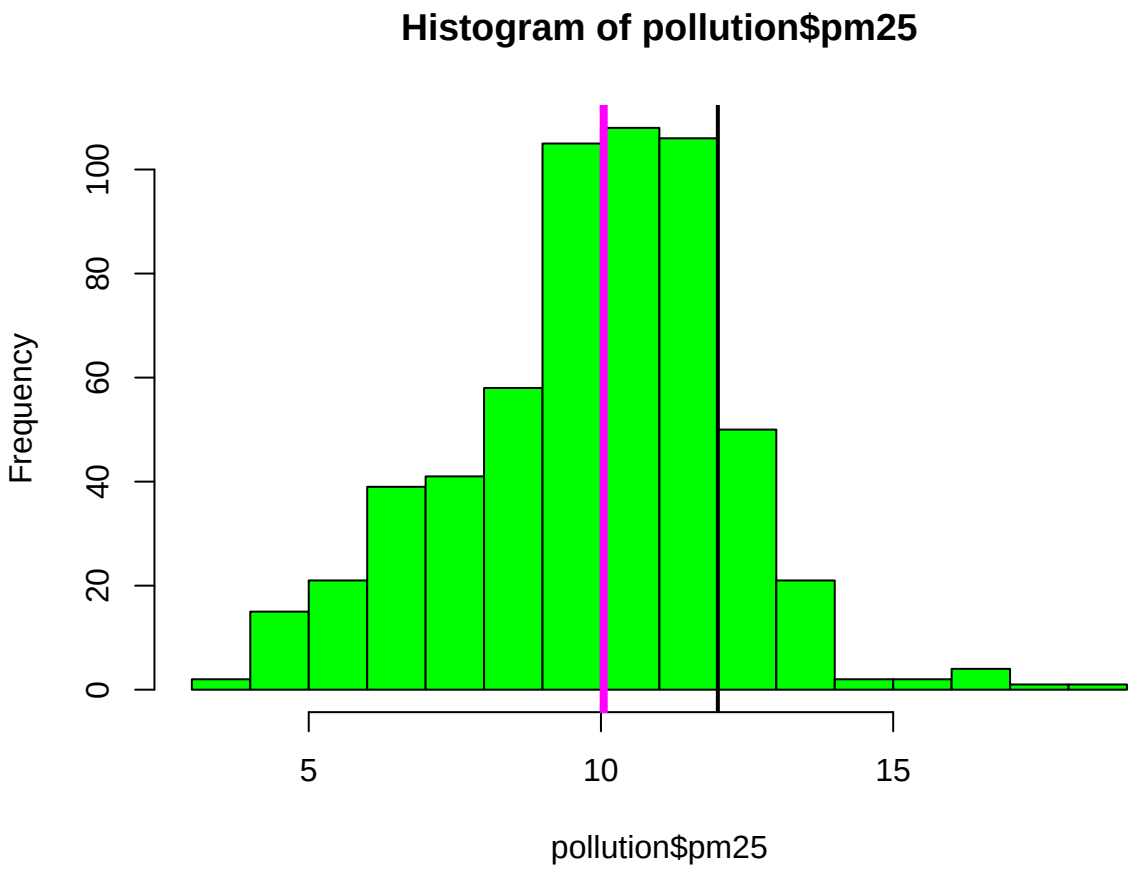


agregando elementos a los graficos

```
boxplot(pollution$pm25, col = "blue")
abline(h = 12)
```




```
hist(pollution$pm25, col = "green")
abline(v = 12, lwd = 2)
abline(v = median(pollution$pm25), col = "magenta", lwd = 4)
```



Resúmenes simples de datos: Dos dimensiones

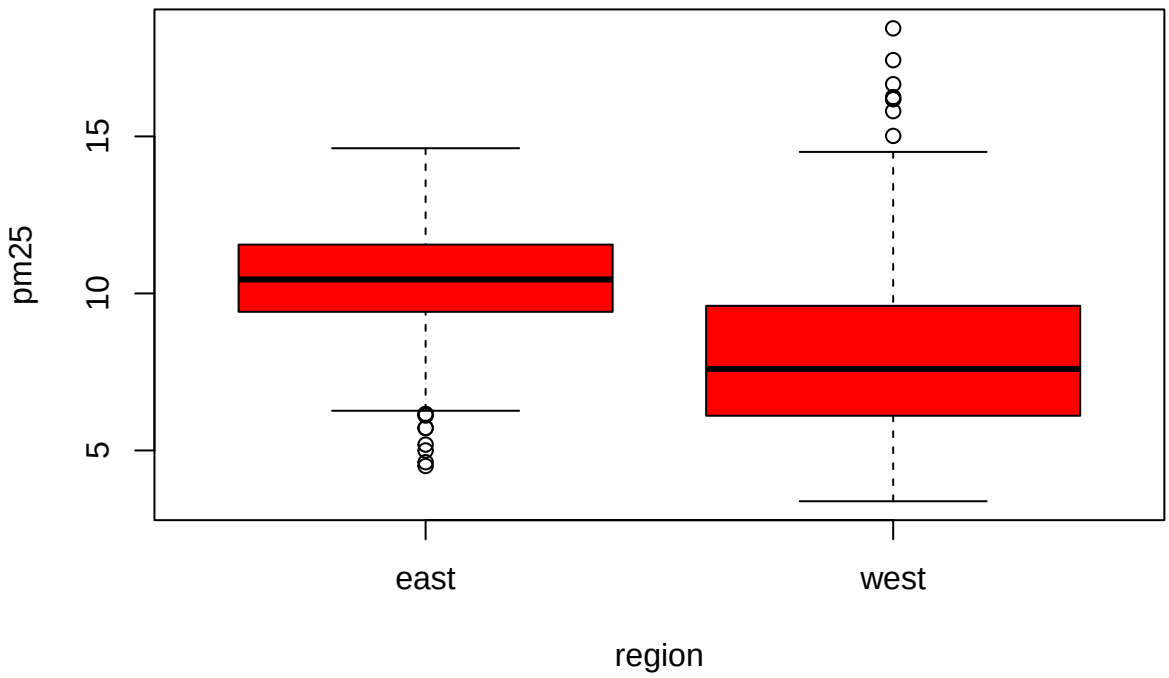
- Gráficos múltiples/1-D superpuestos (Lattice / ggplot2)
- Gráfico de dispersión
- Gráficos de dispersión suaves

> 2 dimensiones

- Gráficos 2-D superpuestos/múltiples; coplots
- Use color, tamaño, forma para agregar dimensiones
- Parcelas giratorias
- Gráficos reales en 3-D (no tan útiles)

Multiples Boxplots

```
boxplot(pm25 ~ region, data = pollution, col = "red")
```



Multiples Histogramas

```
par(mfrow = c(2, 1), mar = c(4, 4, 2, 1))
hist(subset(pollution, region == "east")$pm25, col = "green")
hist(subset(pollution, region == "west")$pm25, col = "green")
```

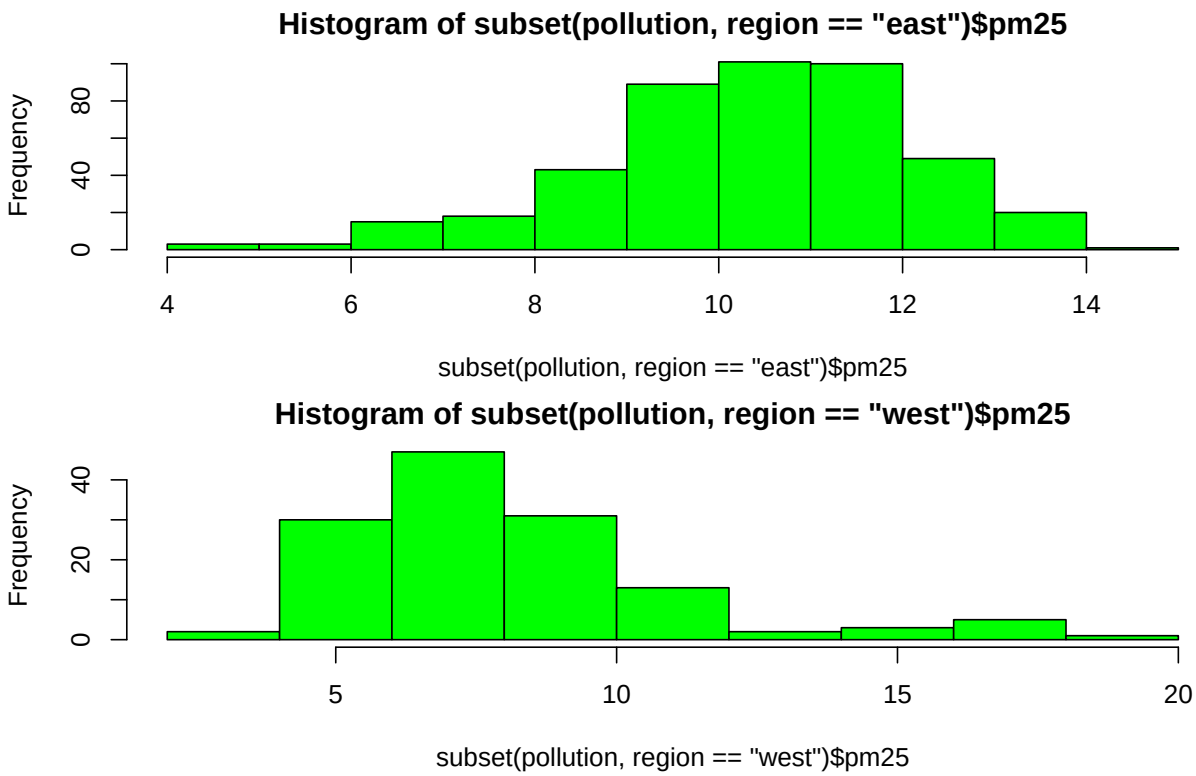


Grafico de dispersion

```
with(pollution, plot(latitude, pm25))
abline(h = 12, lwd = 2, lty = 2)
```

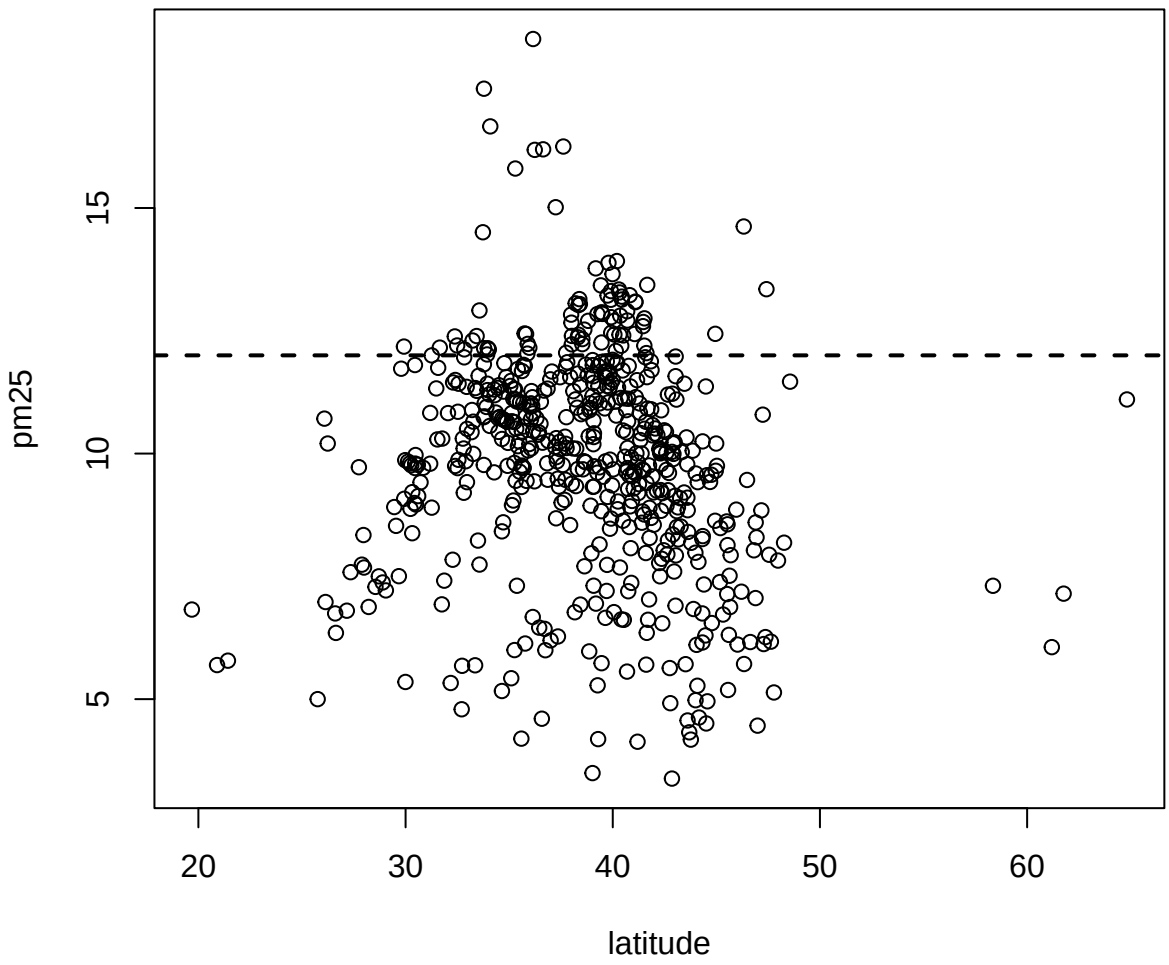
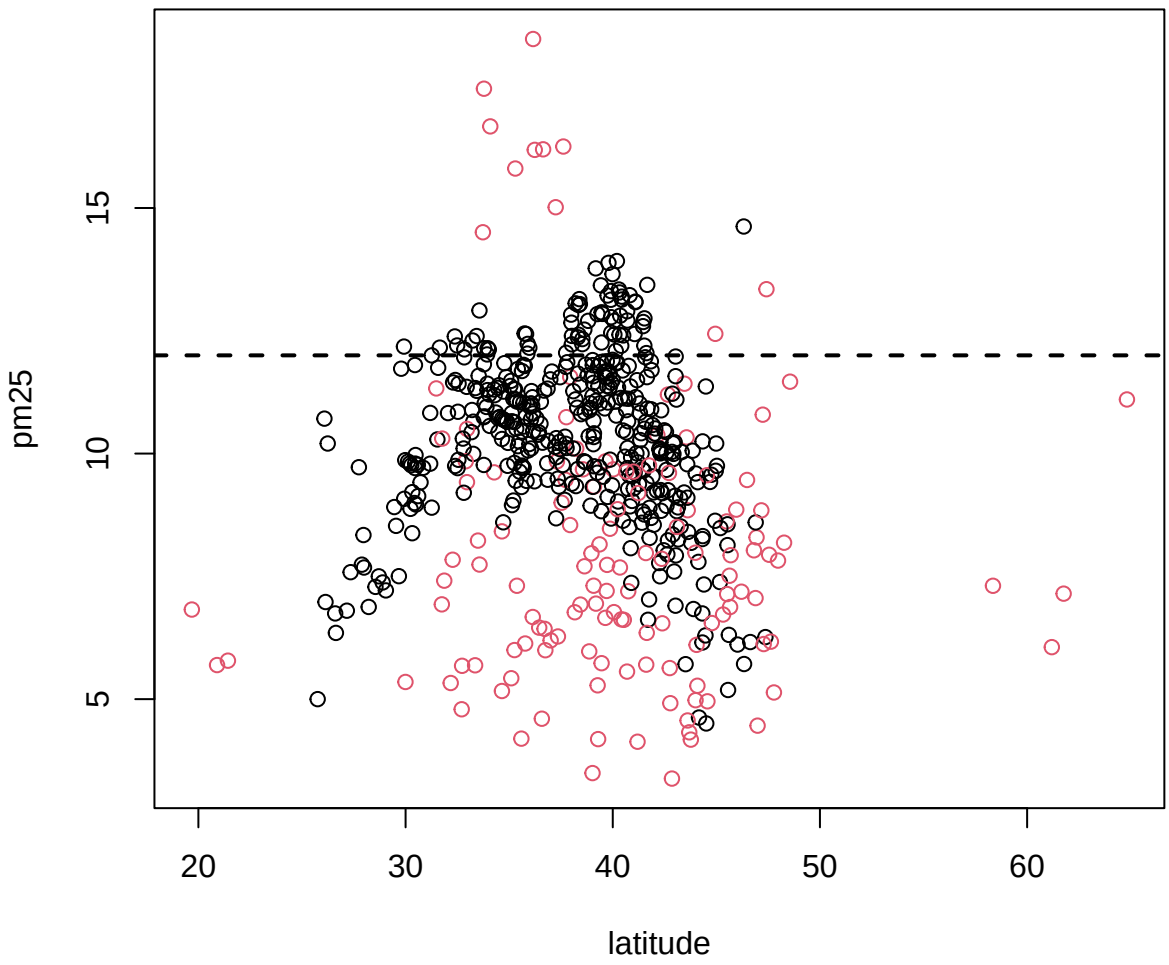


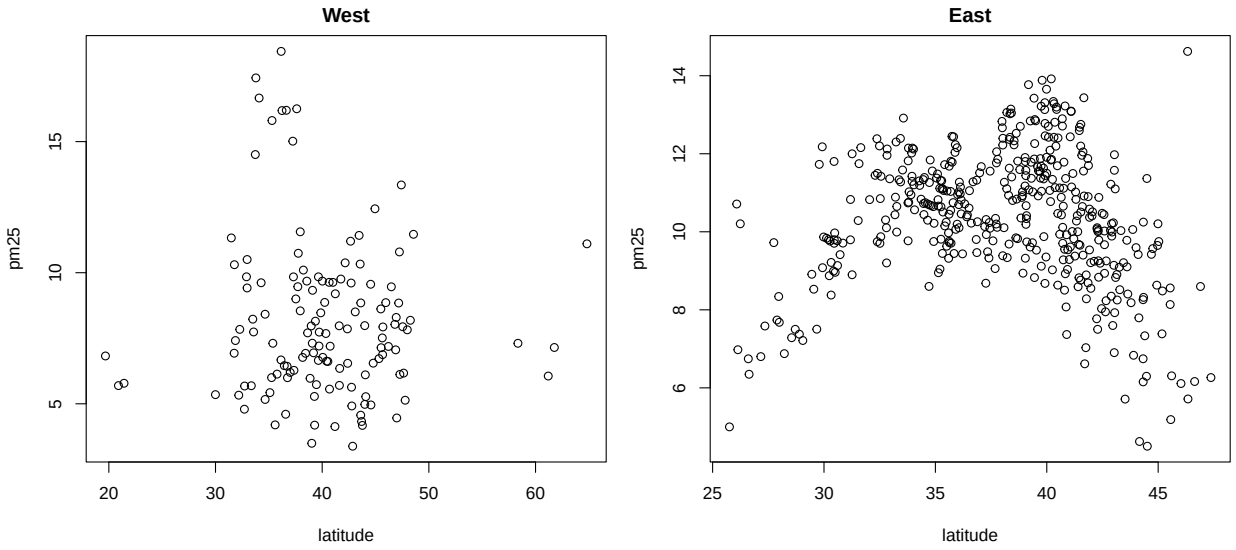
Grafico de dispersion usando color

```
with(pollution, plot(latitude, pm25, col = region))
abline(h = 12, lwd = 2, lty = 2)
```



Multiples graficos de dispersion

```
par(mfrow = c(1, 2), mar = c(5, 4, 2, 1))
with(subset(pollution, region == "west"), plot(latitude, pm25, main = "West"))
with(subset(pollution, region == "east"), plot(latitude, pm25, main = "East"))
```



otros recursos

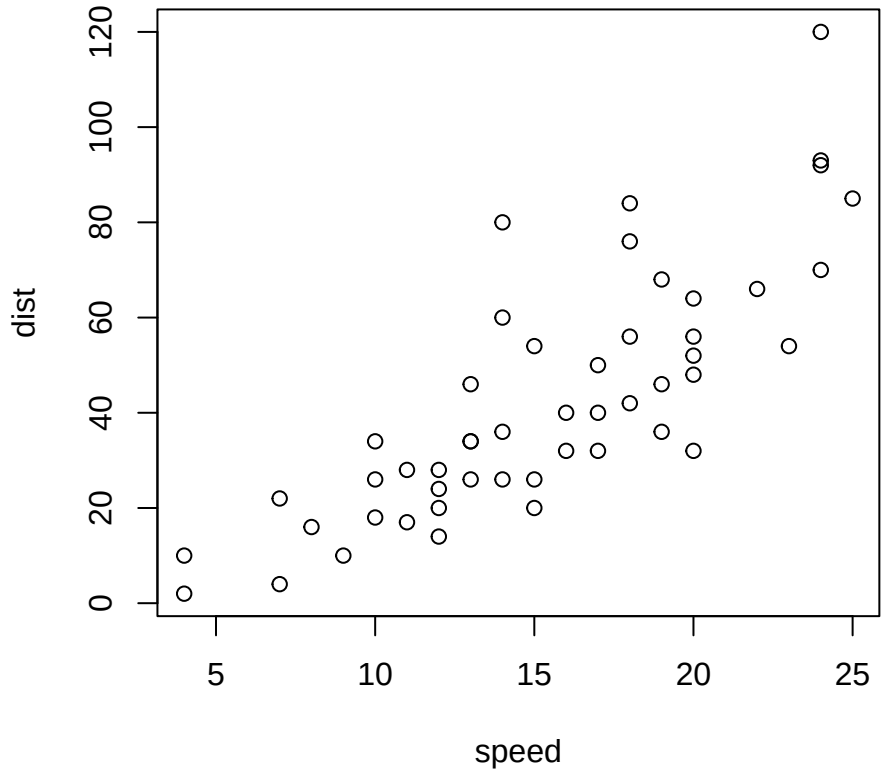
- R Graph Gallery
- R Bloggers}

graficado de sistemas en R

El sistema de trazado base

- Modelo de "paleta del artista"
- Comience con un lienzo en blanco y aumente a partir de ahí
- Comience con la función de plot (o similar)
- Utilice funciones de anotación para agregar/modificar (`texto`,`líneas`, `puntos`,`eje`)
- Conveniente, refleja cómo pensamos en la construcción de parcelas y el análisis de datos.
- No se puede volver una vez que la trama ha comenzado (es decir, para ajustar los márgenes); Necesito planificar con anticipación
- Difícil de "traducir" a otros una vez que se ha creado una nueva trama (sin "lenguaje" gráfico)
- Plot es solo una serie de comandos R

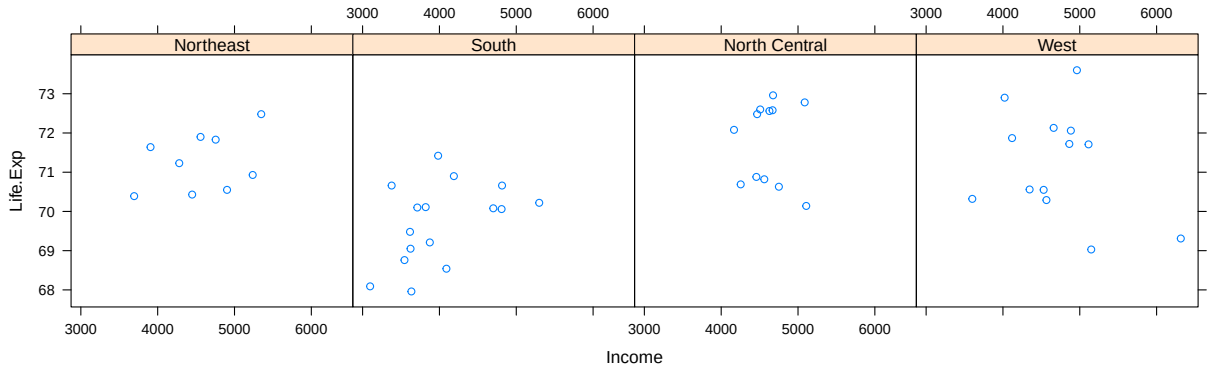
```
library(datasets)
data(cars)
with(cars, plot(speed, dist))
```



El sistema de lattice

- Los gráficos se crean con una sola llamada de función (`xyplot`,`bwplot`,etc.)
- Más útil para condicionar tipos de gráficos: observar cómo cambia y con x en los niveles de z
- Cosas como márgenes/espaciado se establecen automáticamente porque toda la parcela es especificado de una vez
- Bueno para poner muchas parcelas en una pantalla
- A veces es incómodo especificar un gráfico completo en una sola llamada de función
- La anotación en la trama no es especialmente intuitiva
- Uso de funciones de panel y subíndices difíciles de manejar y requiere una preparación intensa
- No se puede "agregar" a la trama una vez creada

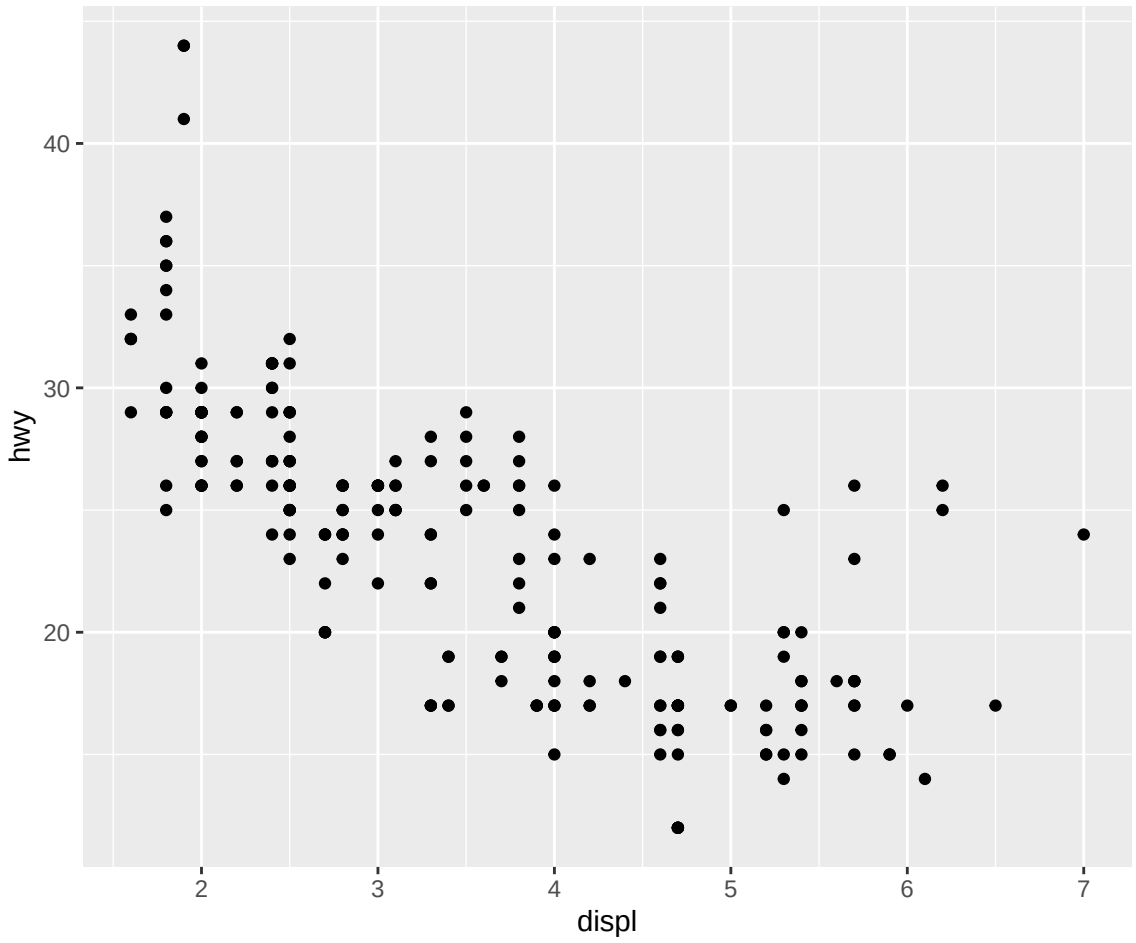
```
library(lattice)
state <- data.frame(state.x77, region = state.region)
xyplot(Life.Exp ~ Income | region, data = state, layout = c(4, 1))
```



El sistema ggplot2

- Divide la diferencia entre la base y lattice de varias formas
- Se ocupa automáticamente de los espacios, el texto, los títulos, pero también le permite para anotar "agregando" a una trama
- Similitud superficial con lattice pero generalmente más fácil / más intuitivo de usar
- El modo predeterminado ofrece muchas opciones (pero aún puede personalizar al deseo de tu corazón)

```
library(ggplot2)
data(mpg)
qplot(displ, hwy, data = mpg)
```



sistema base

El proceso de hacer una trama

Al hacer una trama, primero se deben hacer algunas consideraciones (no necesariamente en este orden):

- ¿Dónde se hará la trama? ¿En la pantalla? ¿En un archivo?
- ¿Cómo se utilizará la trama?
  - ¿La trama se visualiza temporalmente en la pantalla?
  - ¿Se presentará en un navegador web?
  - ¿Eventualmente terminará en un papel que podría imprimirse?
  - ¿Lo estás usando en una presentación?
- ¿Hay una gran cantidad de datos en la trama? ¿O son solo algunos puntos?
- ¿Necesita poder cambiar el tamaño del gráfico de forma dinámica?
- ¿Qué sistema de gráficos utilizará: base, lattice o ggplot2? Por lo general, no se pueden mezclar.
- Los gráficos base generalmente se construyen por partes, con cada aspecto de la trama manejado por separado a través de una serie de llamadas a funciones; Esto a menudo es conceptualmente más simple y permite que el trazado refleje el proceso de pensamiento.
- Los gráficos de lattice generalmente se crean en una sola llamada de función, por lo que todos los

parámetros gráficos deben especificarse a la vez; especificar todo a la vez permite a R calcular automáticamente los espacios y tamaños de fuente necesarios.

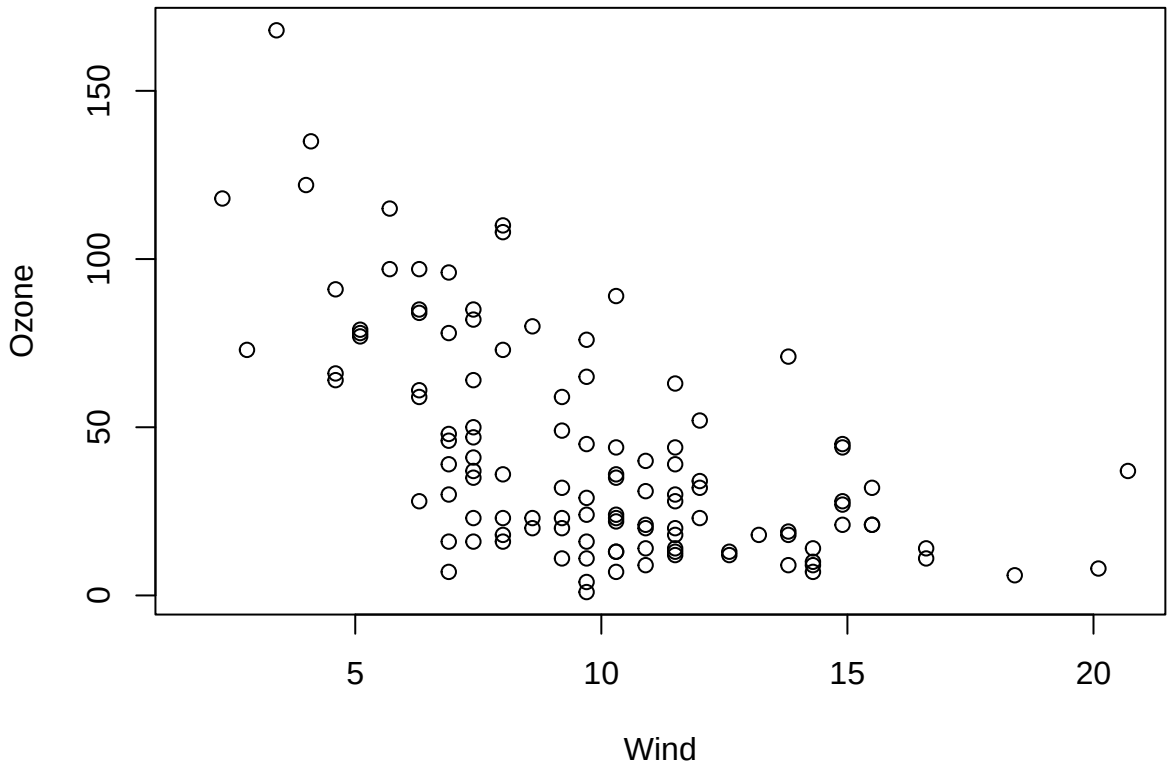
- ggplot2 combina conceptos de gráficos base y de celosía, pero utiliza una implementación independiente Los gráficos base se utilizan con mayor frecuencia y son un sistema muy poderoso para crear gráficos 2-D.
- Hay dos *fases* para crear un diagrama base
  - Inicializando una nueva trama
  - Anotar (agregar a) una parcela existente
- Llamar a `plot (x, y)` o `hist (x)` lanzará un dispositivo gráfico (si uno no está ya abierto) y dibujará un nuevo gráfico en el dispositivo
- Si los argumentos de `plot` no son de alguna clase especial, entonces se llama al método *default* para`plot`; esta función tiene *muchos* argumentos, lo que le permite establecer el título, la etiqueta del eje x, la etiqueta del eje y, etc.
- El sistema de gráficos base tiene *muchos* parámetros que se pueden configurar y modificar; estos parámetros están documentados en `?par`; ¡No estaría de más intentar memorizar esta página de ayuda!

Histograma

```
library(datasets)
hist(airquality$Ozone)  ## Draw a new plot
```

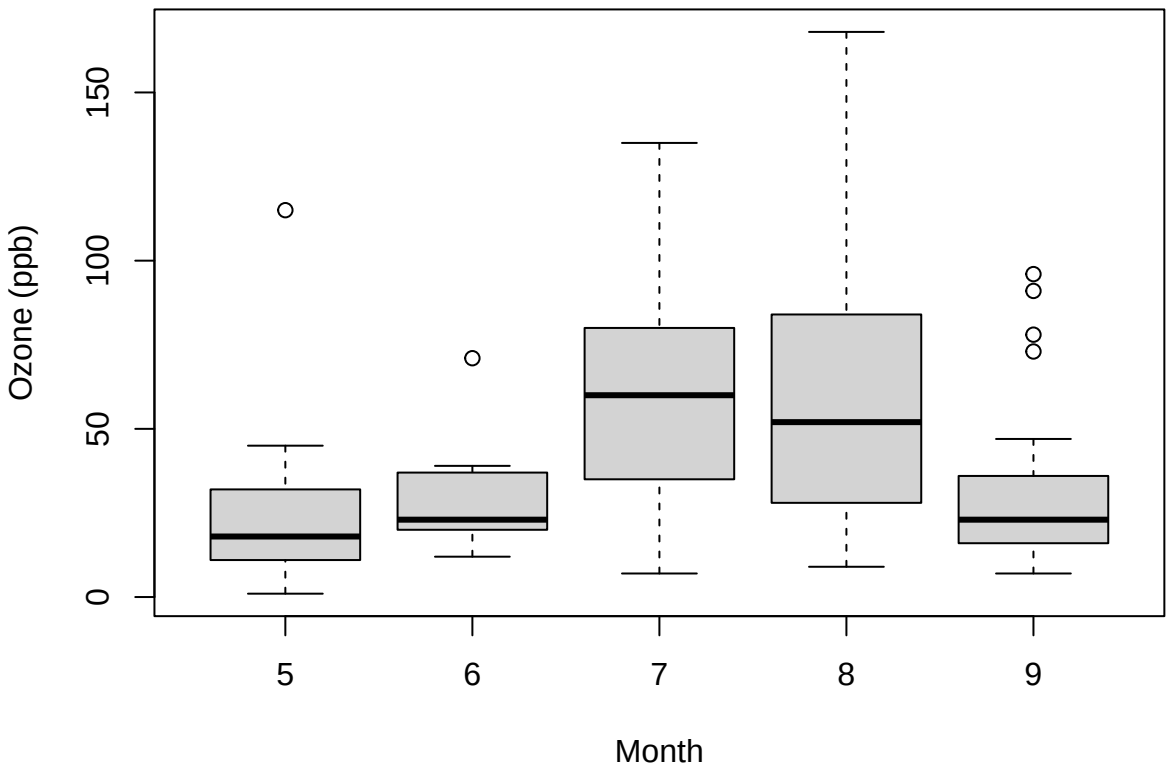
Scatterplot

```
with(airquality, plot(Wind, Ozone))
```



Boxplot

```
library(datasets)
airquality <- transform(airquality, Month = factor(Month))
boxplot(Ozone ~ Month, airquality, xlab = "Month", ylab = "Ozone (ppb)")
```



Algunos parámetros de gráficos básicos importantes

Muchas funciones de trazado base comparten un conjunto de parámetros. Aquí hay algunas claves:

- `pch`: el símbolo de trazado (el predeterminado es un círculo abierto)
- `lty`: el tipo de línea (el valor predeterminado es línea continua), puede ser discontinuo, punteado, etc.
- `lwd`: el ancho de la línea, especificado como un múltiplo entero
- `col`: el color de trazado, especificado como un número, cadena o código hexadecimal; la función `colors()` te da un vector de colores por nombre
- `xlab`: cadena de caracteres para la etiqueta del eje x
- `ylab`: cadena de caracteres para la etiqueta del eje y

La función `par()` se utiliza para especificar parámetros gráficos *globales* que afectan a todos los gráficos en una sesión R. Estos parámetros se pueden anular cuando se especifican como argumentos para funciones de trazado específicas.

- `las`: la orientación de las etiquetas de los ejes en el gráfico
- `bg`: el color de fondo
- `mar`: el tamaño del margen
- `oma`: el tamaño del margen exterior (el valor predeterminado es 0 para todos los lados)
- `mfrow`: número de parcelas por fila, columna (las parcelas se llenan por filas)
- `mfcol`: número de parcelas por fila, columna (las parcelas se llenan por columnas)

Valores predeterminados para parámetros gráficos globales

```
par("bg")
```

```
## [1] "transparent"
```

```
par("mar")
```

```
## [1] 5.1 4.1 4.1 2.1
```

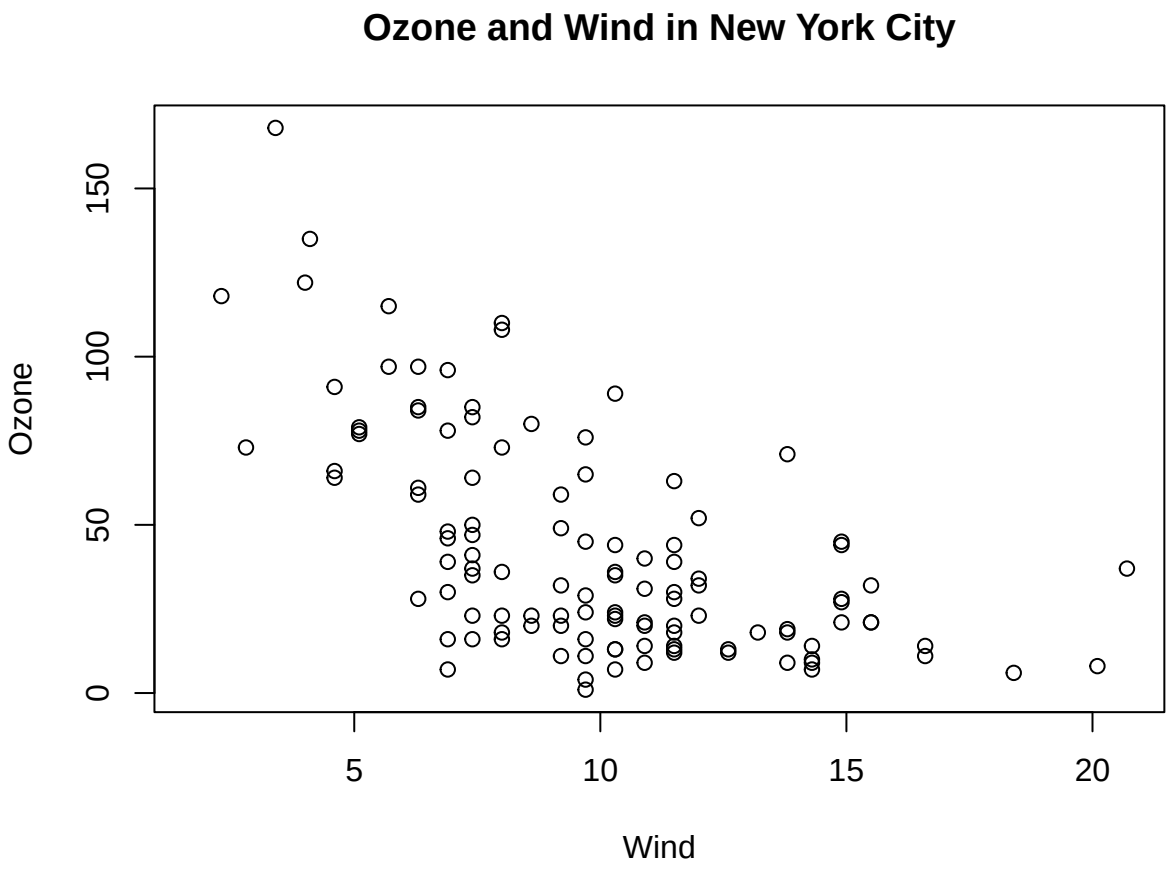
```
par("mfrow")
```

```
## [1] 1 1
```

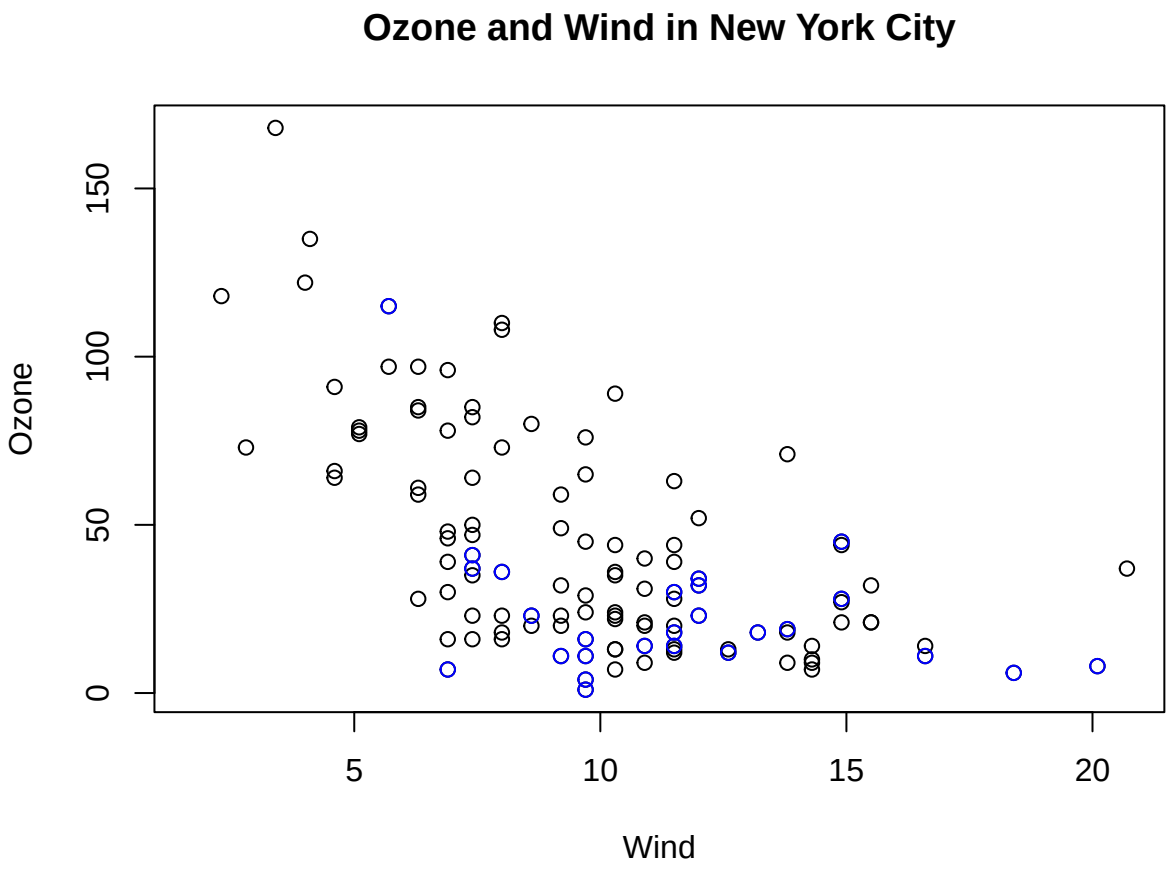
Funciones de trazado base

- `plot`: crea un diagrama de dispersión u otro tipo de diagrama dependiendo de la clase del objeto que se está trazando
- `lines`: añade líneas a un gráfico, dado un vector de valores x y un vector correspondiente de valores y (o una matriz de 2 columnas); esta función solo conecta los puntos
- `points`: agrega puntos a una gráfica
- `text`: agrega etiquetas de texto a un gráfico usando coordenadas x, y especificadas
- `title`: agrega anotaciones a las etiquetas del eje x, y, título, subtítulo, margen exterior
- `mtext`: agrega texto arbitrario a los márgenes (internos o externos) de la trama
- `axis`: agregando ticks / etiquetas de eje

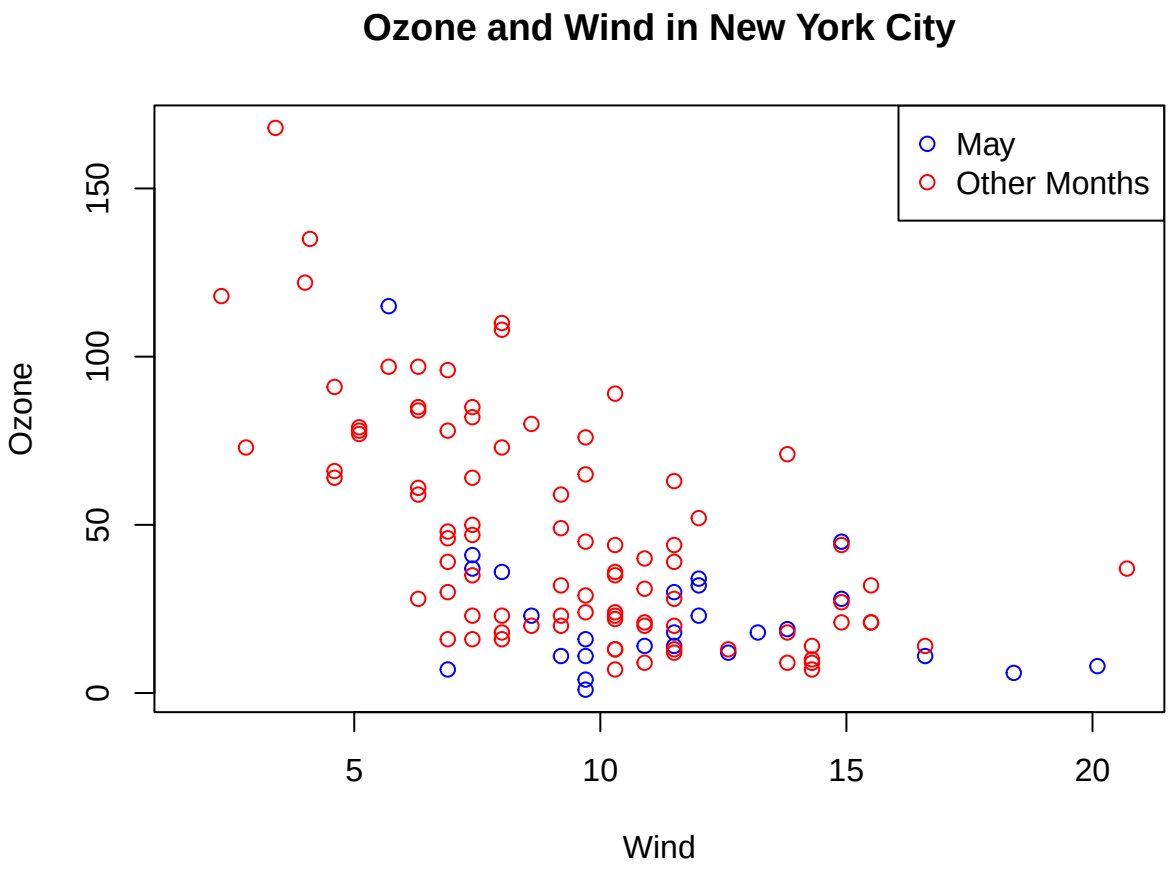
```
with(airquality, plot(Wind, Ozone))
title(main = "Ozone and Wind in New York City")  ## Add a title
```



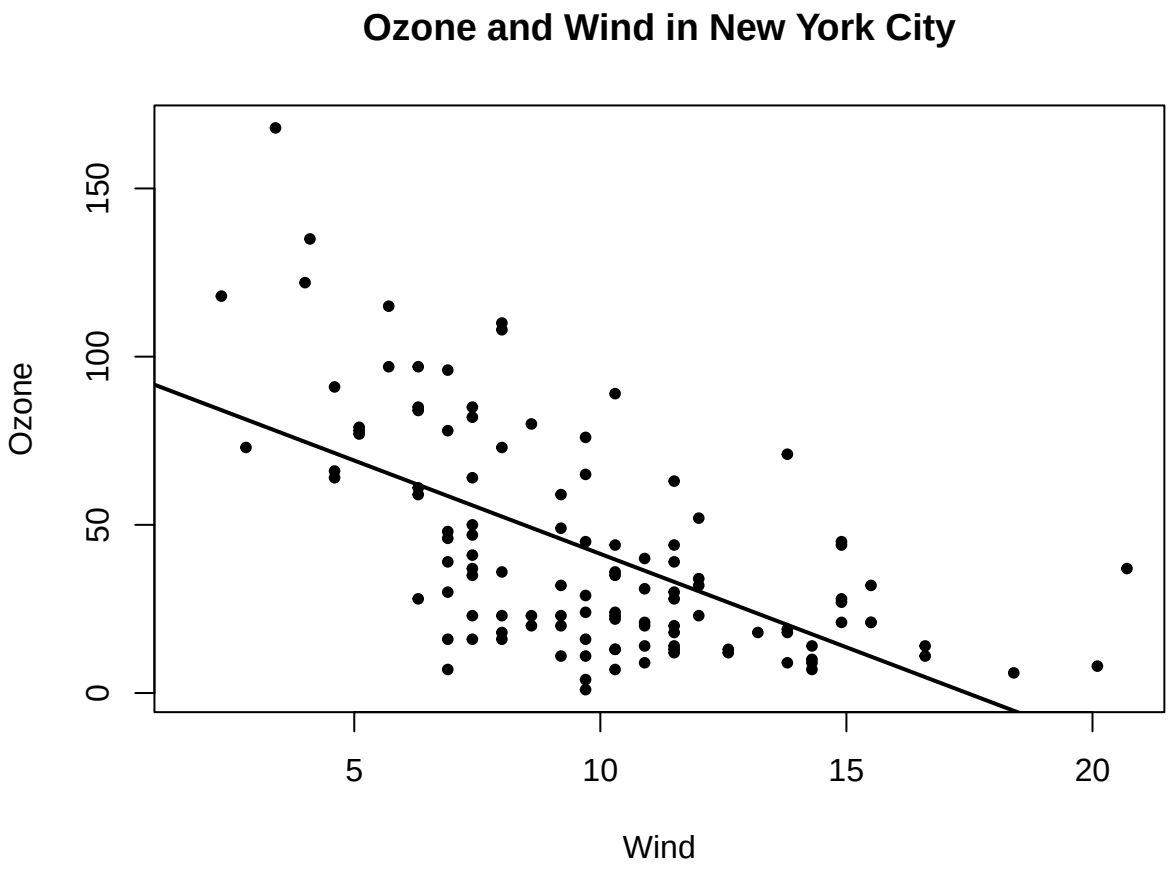

```
with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in New York City"))
with(subset(airquality, Month == 5), points(Wind, Ozone, col = "blue"))
```



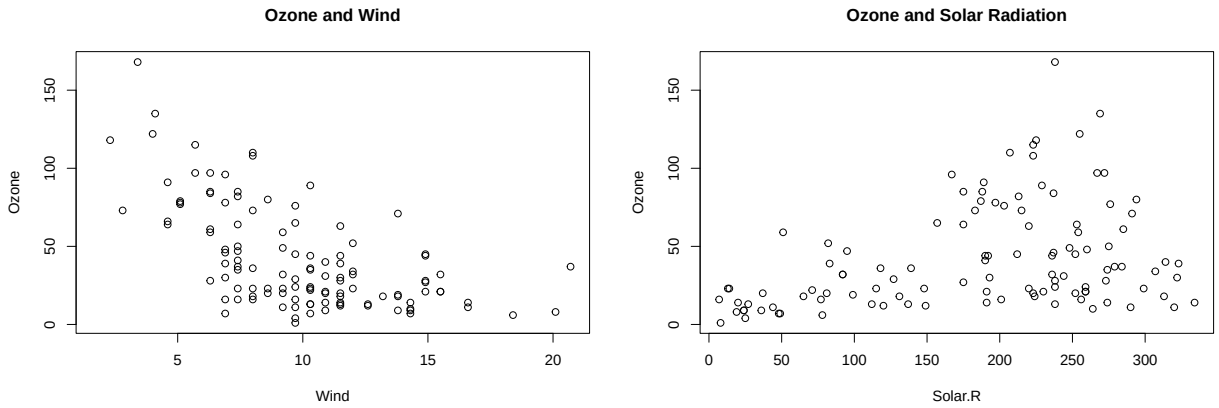

```
with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in New York City", type = "n"))
with(subset(airquality, Month == 5), points(Wind, Ozone, col = "blue"))
with(subset(airquality, Month != 5), points(Wind, Ozone, col = "red"))
legend("topright", pch = 1, col = c("blue", "red"), legend = c("May", "Other Months"))
```




```
with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in New York City", pch = 20))
model <- lm(Ozone ~ Wind, airquality)
abline(model, lwd = 2)
```

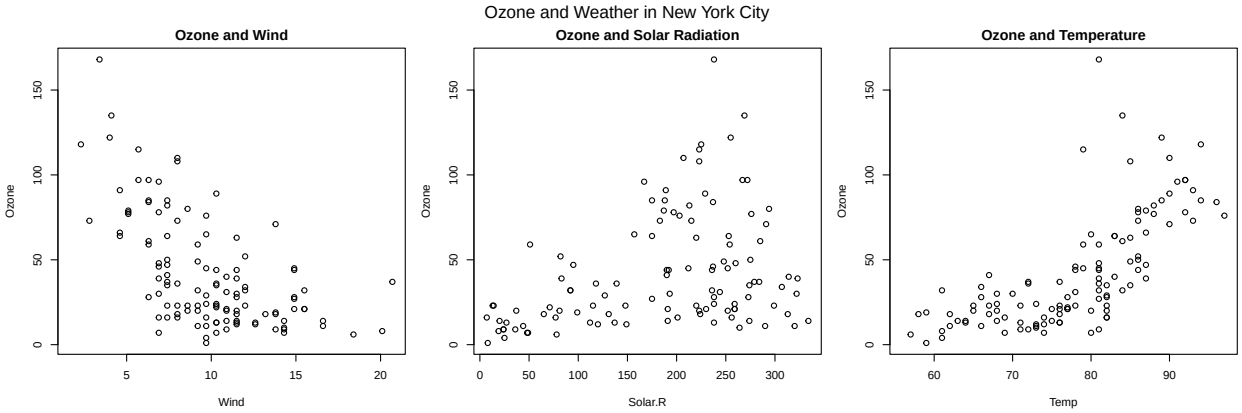



```
par(mfrow = c(1, 2))
with(airquality, {
    plot(Wind, Ozone, main = "Ozone and Wind")
    plot(Solar.R, Ozone, main = "Ozone and Solar Radiation")
})
```




```
par(mfrow = c(1, 3), mar = c(4, 4, 2, 1), oma = c(0, 0, 2, 0))
with(airquality, {
```

```
    plot(Wind, Ozone, main = "Ozone and Wind")
    plot(Solar.R, Ozone, main = "Ozone and Solar Radiation")
    plot(Temp, Ozone, main = "Ozone and Temperature")
    mtext("Ozone and Weather in New York City", outer = TRUE)
})
```



Dispositivos gráficos en R

¿Qué es un dispositivo gráfico?

- Un dispositivo gráfico es algo en lo que puede hacer que aparezca una trama.
  - Una ventana en su computadora (dispositivo de pantalla)
  - Un archivo PDF (dispositivo de archivo)
  - Un archivo PNG o JPEG (dispositivo de archivo)
  - Un archivo de gráficos vectoriales escalables (SVG) (dispositivo de archivo)
- Cuando crea un gráfico en R, debe "enviarse" a un dispositivo gráfico específico
- El lugar más común para "enviar" una trama es el *dispositivo de pantalla*
  - En una Mac, el dispositivo de pantalla se inicia con el `quart()`
  - En Windows, el dispositivo de pantalla se inicia con `windows()`
  - En Unix / Linux, el dispositivo de pantalla se inicia con `x11()`
- Al hacer una trama, debe considerar cómo se utilizará la trama para determinar a qué dispositivo se debe enviar la trama.
  - La lista de dispositivos se encuentra en "?devices"; También hay dispositivos creados por usuarios en CRAN.
- Para visualizaciones rápidas y análisis exploratorio, generalmente desea usar el dispositivo de pantalla
  - Funciones como `plot` en base,`xyplot` en lattice o `qplot` en ggplot2 enviarán por defecto un gráfico al dispositivo de pantalla
  - En una plataforma determinada (Mac, Windows, Unix / Linux) solo hay un dispositivo de pantalla
- Para los gráficos que pueden imprimirse o incorporarse a un documento (por ejemplo, artículos / informes, presentaciones de diapositivas), generalmente es más apropiado un *dispositivo de archivo*
  - Hay muchos dispositivos de archivo diferentes para elegir

- NOTA: No todos los dispositivos gráficos están disponibles en todas las plataformas (es decir, no puede iniciar `windows()` en una Mac)

¿Cómo se crea una parcela?

Hay dos enfoques básicos para trazar. El primero es el más común:

1. Llame a una función de trazado como `plot`,`xyplot` o `qplot`
2. El gráfico aparece en el dispositivo de pantalla.
3. Anote el gráfico si es necesario
4. Disfruta

El segundo enfoque para el trazado se usa más comúnmente para dispositivos de archivo:

1. Inicie explícitamente un dispositivo gráfico
2. Llame a una función de trazado para hacer un trazado (Nota: si está utilizando un archivo dispositivo, no aparecerá ningún gráfico en la pantalla)
3. Anote el gráfico si es necesario
4. Cierre explícitamente el dispositivo de gráficos con `dev.off()` (¡esto es muy importante!)

```
pdf(file = "myplot.pdf")  ## Open PDF device; create 'myplot.pdf' in my working directory
## Create plot and send to a file (no plot appears on screen)
with(faithful, plot(eruptions, waiting))
title(main = "Old Faithful Geyser data")  ## Annotate plot; still nothing on screen
dev.off()  ## Close the PDF file device
## Now you can view the file 'myplot.pdf' on your computer
```

el codigo anterior guarda la imagen en un archivo pdf

Dispositivos de archivos gráficos

Hay dos tipos básicos de dispositivos de archivo: *vector* y *mapa de bits* dispositivos

Formatos vectoriales:

- `pdf`: útil para gráficos de tipo de línea, cambia de tamaño bien, generalmente portátil, no es eficiente si una trama tiene muchos objetos / puntos
- `svg`: gráficos vectoriales escalables basados en XML; admite animación e interactividad, potencialmente útil para tramas basadas en la web
- `win.metafile`: formato de metarchivo de Windows (solo en Windows)
- `postscript`: formato antiguo, también cambia bien de tamaño, generalmente portátil, se puede usar para crear archivos postscript encapsulados; Los sistemas Windows a menudo no tienen un visor de postscript

Formatos de mapa de bits

- `png`: formato de mapa de bits, bueno para dibujos de líneas o imágenes con colores sólidos, usa compresión sin pérdida (como el antiguo formato GIF), la mayoría de los navegadores web pueden leer este formato de forma nativa, bueno para trazar muchos muchos puntos, no cambia de tamaño bien
- `jpeg`: bueno para fotografías o escenas naturales, usa compresión con pérdida, bueno para trazar muchos muchos puntos, no cambia de tamaño bien, puede ser leído por casi cualquier computadora y cualquier navegador web, no es bueno para dibujos lineales
- `tiff`: Crea archivos de mapa de bits en formato TIFF; admite compresión sin pérdidas
- `bmp`: un formato de mapa de bits nativo de Windows

Varios dispositivos gráficos abiertos

- Es posible abrir varios dispositivos gráficos (pantalla, archivo o ambos), por ejemplo, al ver varios gráficos a la vez
- El trazado solo puede ocurrir en un dispositivo gráfico a la vez
- El dispositivo gráfico *actualmente activo* se puede encontrar llamando a `dev.cur ()`
- A cada dispositivo gráfico abierto se le asigna un número entero ≥ 2.
- Puede cambiar el dispositivo gráfico activo con `dev.set (<integer>)` donde `<integer>` es el número asociado con el dispositivo gráfico al que desea cambiar

Copiar un gráfico en otro dispositivo puede ser útil porque algunos gráficos requieren una gran cantidad de código y puede ser complicado escribir todo eso nuevamente para un dispositivo diferente.

- `dev.copy`: copia un gráfico de un dispositivo a otro
- `dev.copy2pdf`: copia específicamente un gráfico a un archivo PDF

NOTA: Copiar un gráfico no es una operación exacta, por lo que el resultado puede no ser idéntico al original.

```
library(datasets)
with(faithful, plot(eruptions, waiting))  ## Create plot on screen device
title(main = "Old Faithful Geyser data")  ## Add a main title
dev.copy(png, file = "geyserplot.png")  ## Copy my plot to a PNG file
dev.off()  ## Don't forget to close the PNG device!
```

el codigo anterior copia una gráfica

graficacion de funciones matematicas

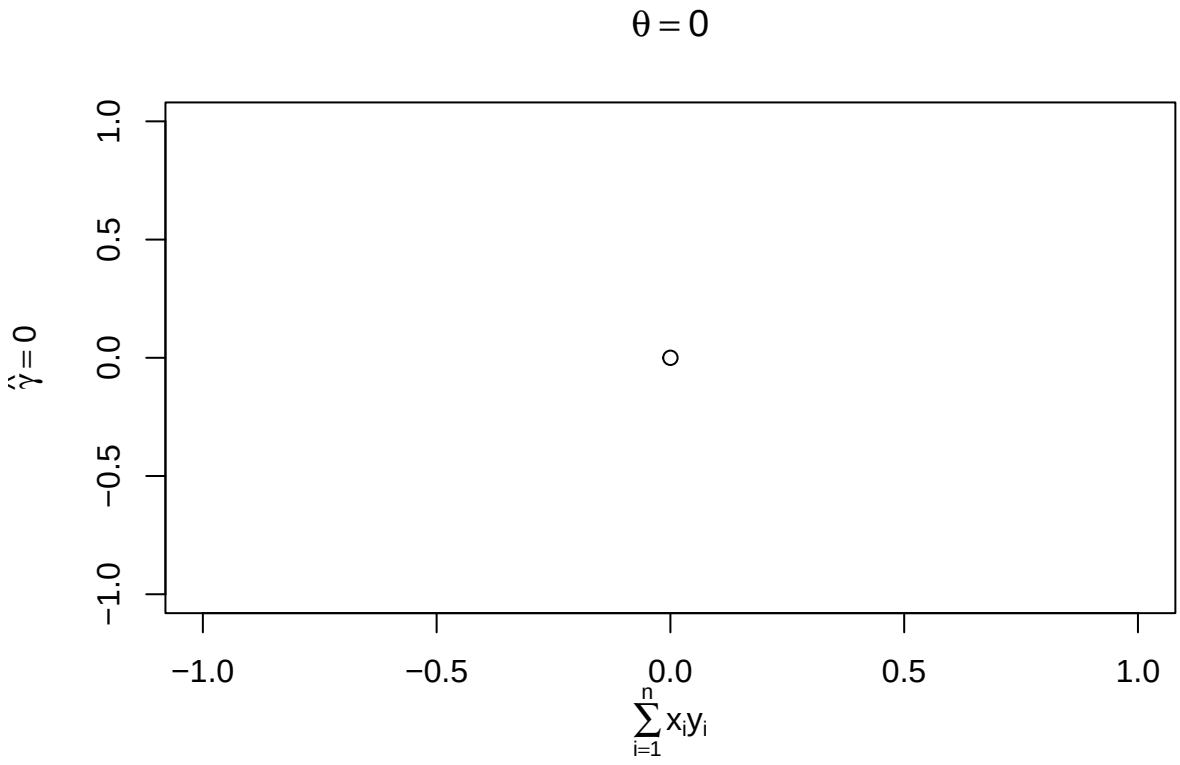
Anotación matemática

R puede producir símbolos similares a LaTeX en un gráfico para anotaciones matemáticas. Esto es muy útil y útil para burlarse de las personas que usan otros paquetes estadísticos.

- Los símbolos matemáticos son "expresiones" en R y deben incluirse en la función `expression`
- Hay una lista establecida de símbolos permitidos y esto está documentado en ?plotmath
- Las funciones de trazado que toman argumentos para texto generalmente permiten expresiones para símbolos matemáticos

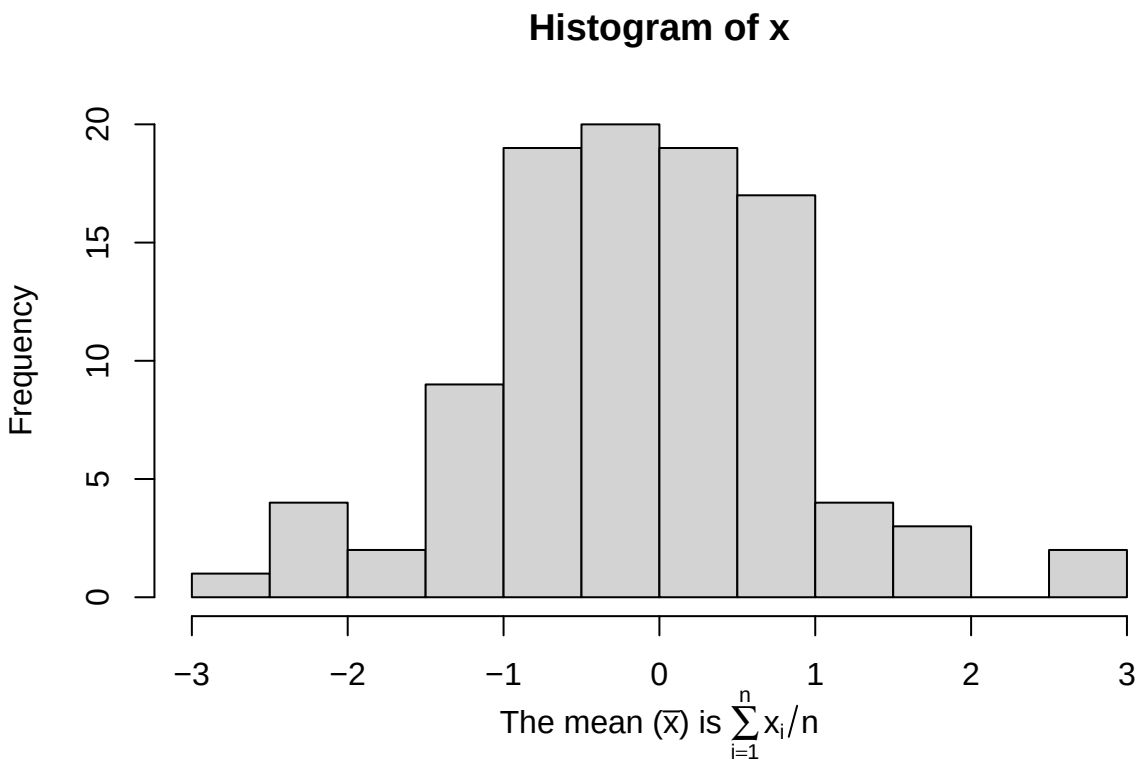
Algunos ejemplos:

```
plot(0, 0, main = expression(theta == 0),
     ylab = expression(hat(gamma) == 0),
     xlab = expression(sum(x[i] * y[i], i==1, n)))
```



Pegar cadenas

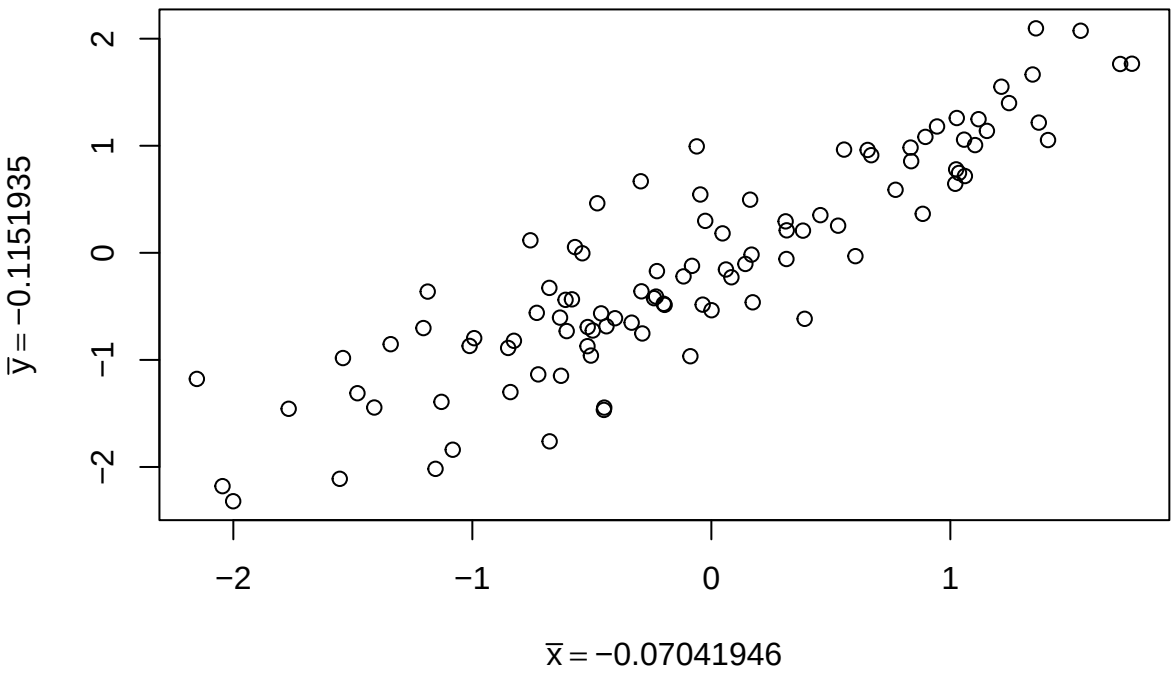
```
x <- rnorm(100)
hist(x,
     xlab=expression("The mean (" * bar(x) * ") is " *
                     sum(x[i]/n,i==1,n)))
```



Sustituyendo

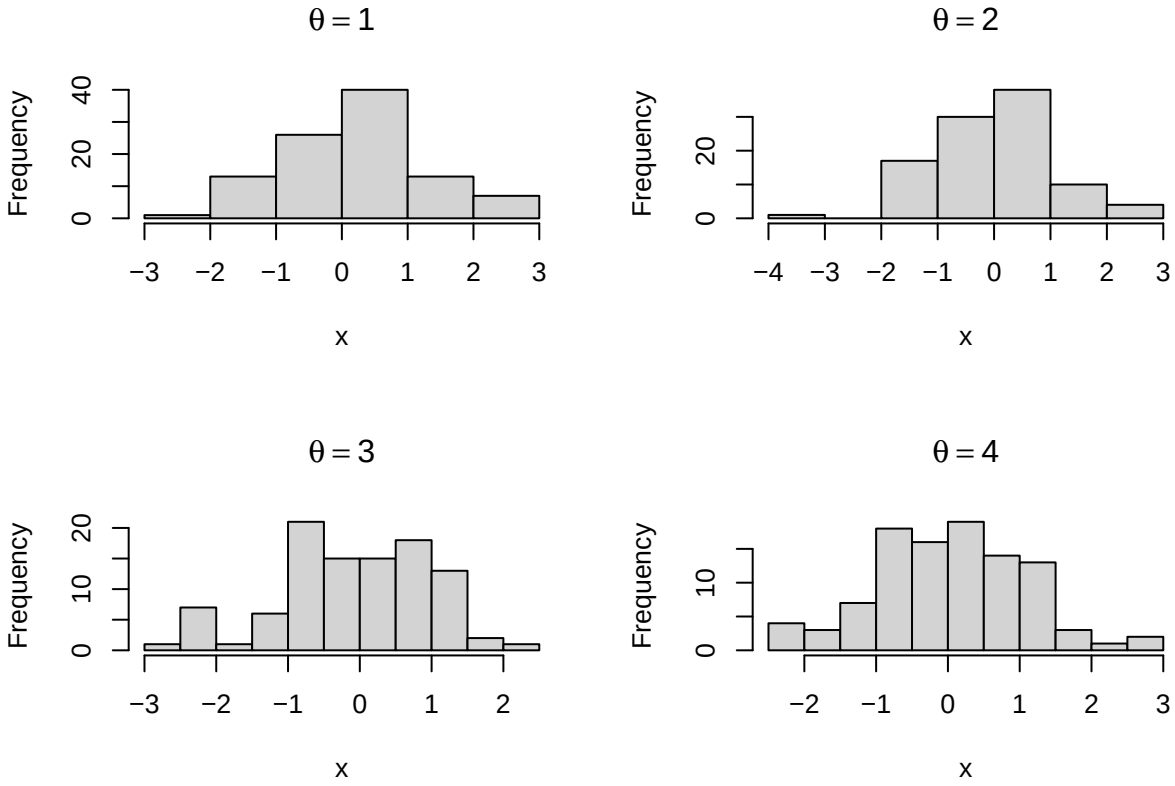
¿Qué sucede si desea utilizar un valor calculado en la anotación?

```
x <- rnorm(100)
y <- x + rnorm(100, sd = 0.5)
plot(x, y,
     xlab=substitute(bar(x) == k, list(k=mean(x))),
     ylab=substitute(bar(y) == k, list(k=mean(y)))
     )
```



o un bucle de graficas

```
par(mfrow = c(2, 2))
for(i in 1:4) {
        x <- rnorm(100)
        hist(x, main=substitute(theta==num,list(num=i)))
}
```


θ = 1
Frequency
0
20
40
−3
−2
−1
0
1
2
3
x
θ = 2
Frequency
0
20
−4
−3
−2
−1
0
1
2
3
x
θ = 3
Frequency
0
10
20
−3
−2
−1
0
1
2
x
θ = 4
Frequency
0
10
−2
−1
0
1
2
3
x