Manipulación y Gráficas en R

Luis Mauricio Camargo Badillo

23 de octuble de 2023

Introducción a R

Ejercicio 0

Sumar 2+2:

```
2+2
```

```
## [1] 4
```

Raíz cuadrada de 10:

```
sqrt(10)
```

```
## [1] 3.162278
```

Ejercicio 1

Página de ayuda de `read_csv()`:

```
library(tidyverse)
?read_csv()
```

Leer la información de los nombres de bebés y asignarla a `baby_names`:

```
baby_names <- read_csv("Datos/babyNames.csv")
```

Guardar `baby_names` como un data set de R en un archivo `babynames.rds`:

```
write_rds(baby_names, file = "Datos/babynames.rds")
```

Ejercicio 2.1

Usar `filter()` para extraer los datos de mi nombre (Mauricio):

```
baby_names_mauricio <- filter(baby_names, Name == "Mauricio")
```

Ordenar los datos del paso anterior según `Count`:

```
arrange(baby_names_mauricio, desc(Count))
```

```
## # A tibble: 50 x 4
##    Name     Sex   Count  Year
##    <chr>    <chr> <dbl> <dbl>
##  1 Mauricio Boys    829  2008
##  2 Mauricio Boys    809  2006
##  3 Mauricio Boys    807  2005
```

```
##  4 Mauricio Boys    792  2007
##  5 Mauricio Boys    775  2002
##  6 Mauricio Boys    755  2001
##  7 Mauricio Boys    755  2004
##  8 Mauricio Boys    684  2003
##  9 Mauricio Boys    605  2000
## 10 Mauricio Boys    485  1999
## # i 40 more rows
```

Filtrar los datos, extrayendo solo la fila que contiene el nombre más popular para niños en 1999:

```
baby_names_boys <- filter(baby_names, Sex == "Boys" & Year == 1999)
filter(baby_names_boys, Count == max(Count))
```

```
## # A tibble: 1 x 4
##   Name  Sex   Count  Year
##   <chr> <chr> <dbl> <dbl>
## 1 Jacob Boys  35361  1999
```

Ejercicio 2.2

```
baby_names %>% filter(Name == "Mauricio") %>% arrange(desc(Count))
```

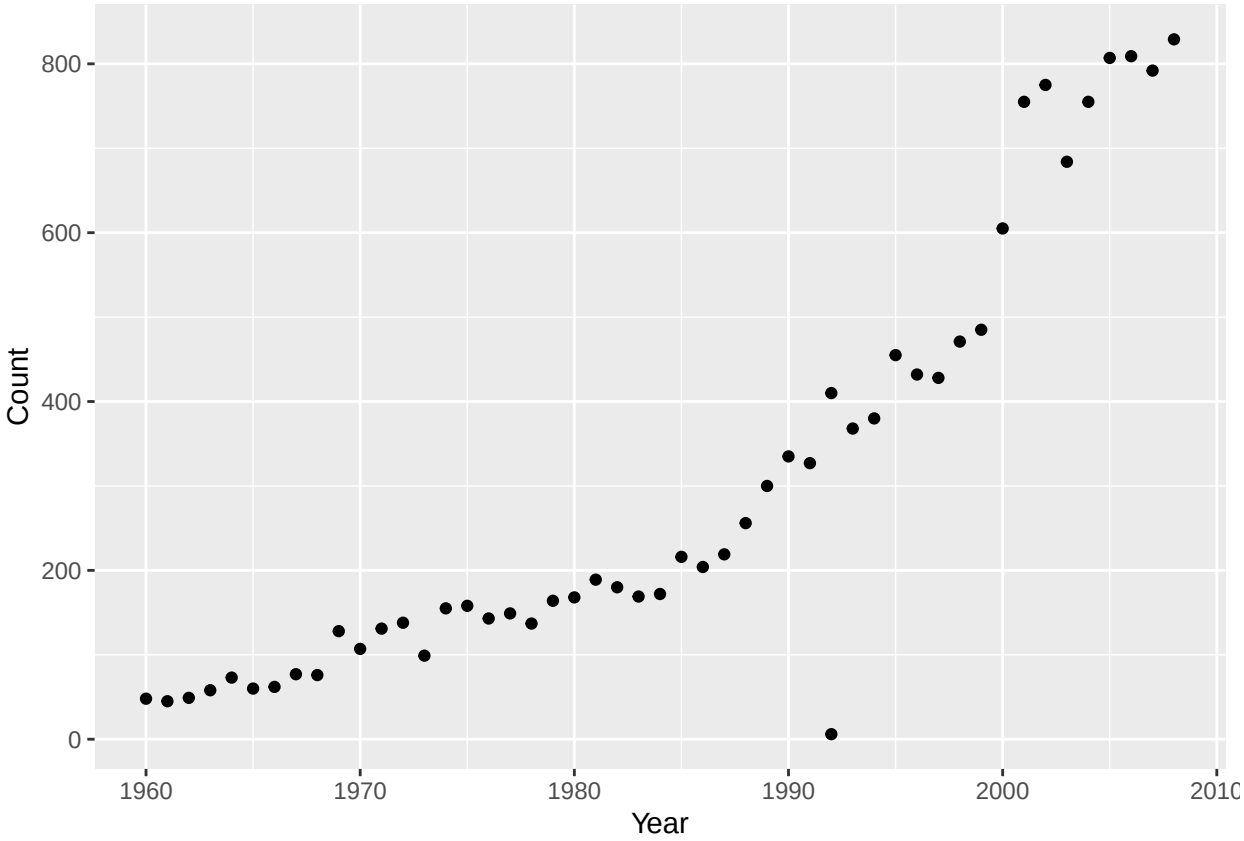
```
## # A tibble: 50 x 4
##    Name     Sex   Count  Year
##    <chr>    <chr> <dbl> <dbl>
##  1 Mauricio Boys    829  2008
##  2 Mauricio Boys    809  2006
##  3 Mauricio Boys    807  2005
##  4 Mauricio Boys    792  2007
##  5 Mauricio Boys    775  2002
##  6 Mauricio Boys    755  2001
##  7 Mauricio Boys    755  2004
##  8 Mauricio Boys    684  2003
##  9 Mauricio Boys    605  2000
## 10 Mauricio Boys    485  1999
## # i 40 more rows
```

Ejercicio 3

No hace falta volver a extraer los datos correspondientes a mi nombre.

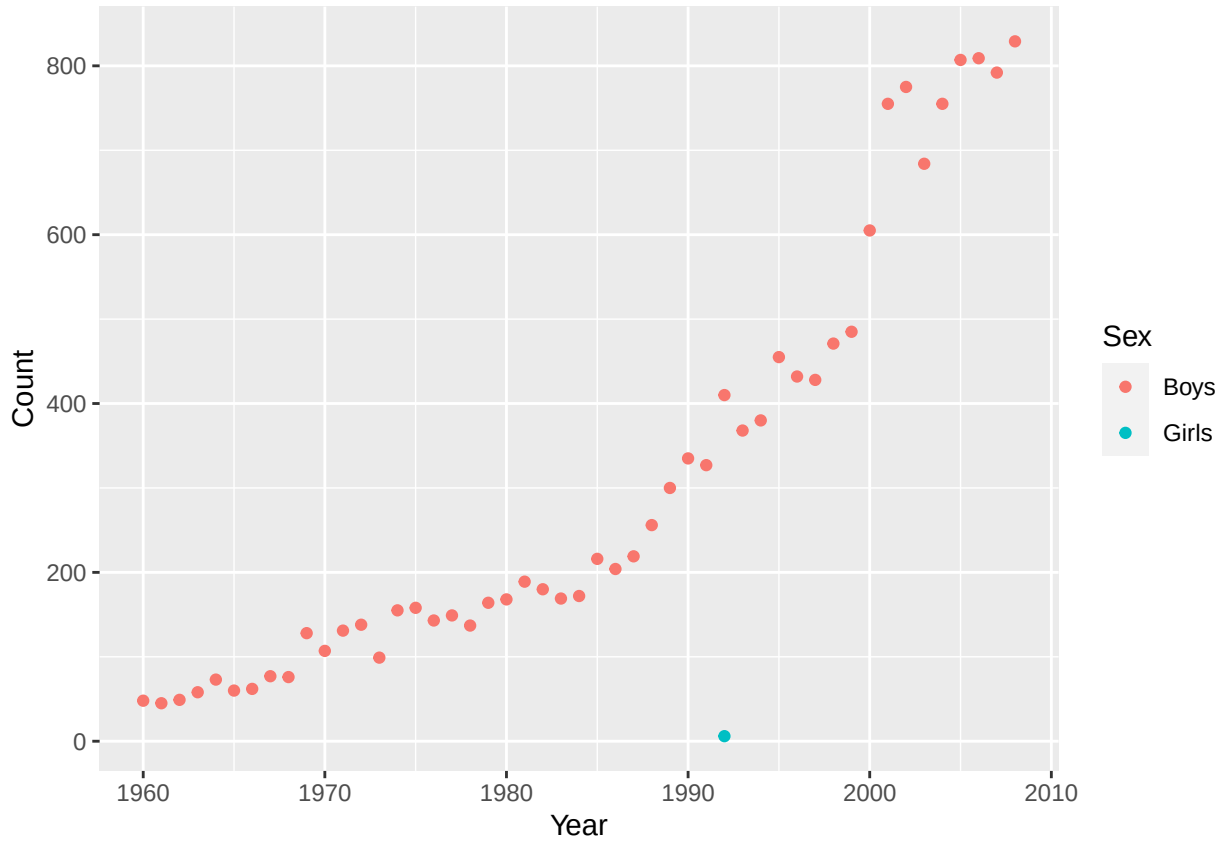
Graficar los datos correspondientes a mi nombre, con `Year` en las abscisas y `Count` en las ordenadas:

```
qplot(x = Year, y = Count, data = baby_names_mauricio)
```



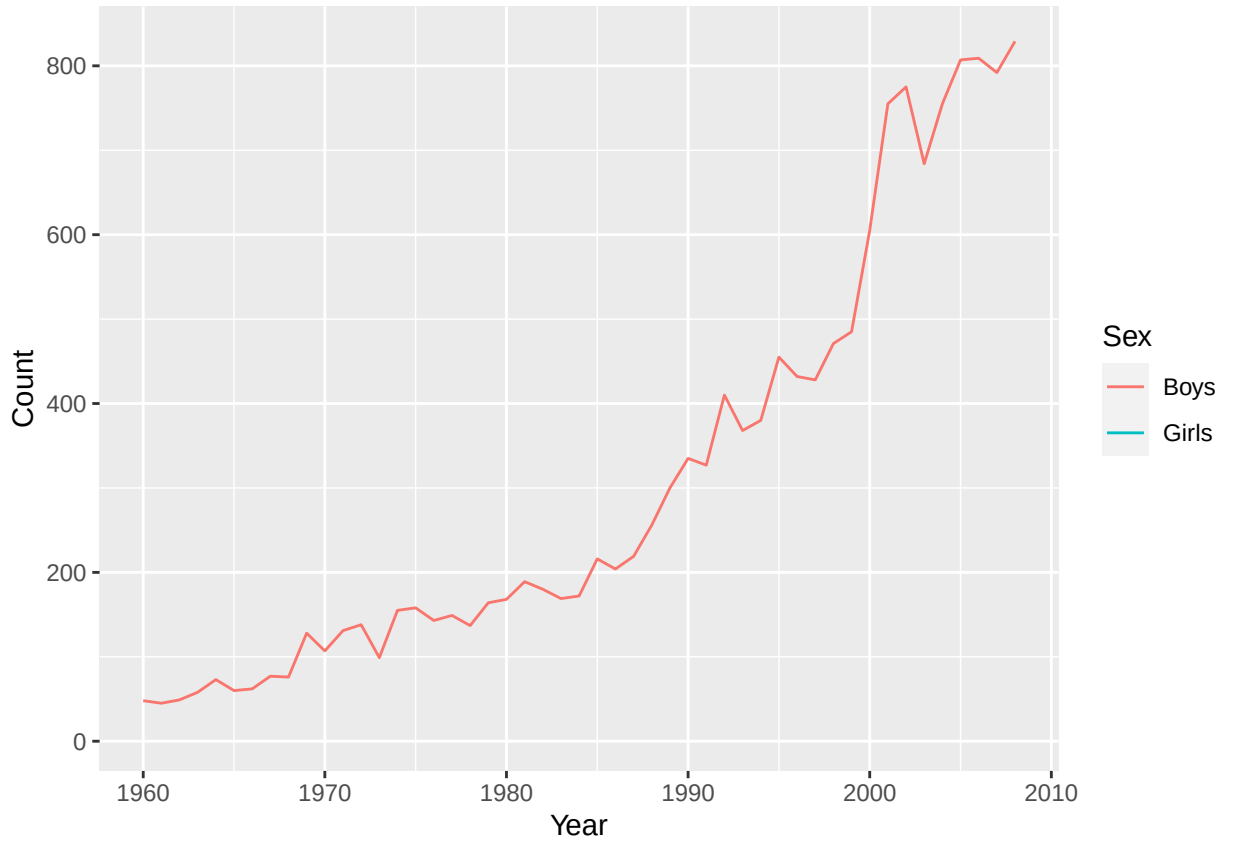
Ajustando la gráfica para mostrar a niños y niñas con diferentes colores:

```
qplot(x = Year, y = Count, colour = Sex, data = baby_names_mauricio)
```



En la gráfica, utilizar líneas en lugar de puntos:

```
qplot(x = Year, y = Count, colour = Sex,
    data = baby_names_mauricio, geom = "line")
```



Ejercicio 4

Utilizar `mutate()` y `group_by()` para crear una columna `Proportion` donde `Proportion = Count/sum(Count)` para cada grupo `Year X Sex`:

```
baby_names <-
    baby_names %>% group_by(Year, Sex) %>%
    mutate(Proportion = Count/sum(Count)) %>%
    ungroup()

baby_names
```

```
## # A tibble: 1,048,575 x 5
##    Name     Sex   Count  Year Proportion
##    <chr>    <chr> <dbl> <dbl>      <dbl>
##  1 Mary     Girls 51474  1960     0.0255
##  2 Susan    Girls 39200  1960     0.0194
##  3 Linda    Girls 37314  1960     0.0185
##  4 Karen    Girls 36376  1960     0.0180
##  5 Donna    Girls 34133  1960     0.0169
##  6 Lisa     Girls 33702  1960     0.0167
##  7 Patricia Girls 32102  1960     0.0159
##  8 Debra    Girls 26737  1960     0.0132
##  9 Cynthia  Girls 26725  1960     0.0132
## 10 Deborah  Girls 25264  1960     0.0125
## # i 1,048,565 more rows
```

Utilizar `mutate()` y `group_by()` para crear una columna llamada `Rank` donde `Rank = rank(desc(Count))` para cada grupo `Year X Sex`:

```
baby_names <-
    baby_names %>% group_by(Year, Sex) %>%
    mutate(Rank = rank(desc(Count))) %>%
    ungroup()

baby_names
```

```
## # A tibble: 1,048,575 x 6
##    Name     Sex   Count  Year Proportion  Rank
##    <chr>    <chr> <dbl> <dbl>      <dbl> <dbl>
##  1 Mary     Girls 51474  1960     0.0255     1
##  2 Susan    Girls 39200  1960     0.0194     2
##  3 Linda    Girls 37314  1960     0.0185     3
##  4 Karen    Girls 36376  1960     0.0180     4
##  5 Donna    Girls 34133  1960     0.0169     5
##  6 Lisa     Girls 33702  1960     0.0167     6
##  7 Patricia Girls 32102  1960     0.0159     7
##  8 Debra    Girls 26737  1960     0.0132     8
##  9 Cynthia  Girls 26725  1960     0.0132     9
## 10 Deborah  Girls 25264  1960     0.0125    10
## # i 1,048,565 more rows
```

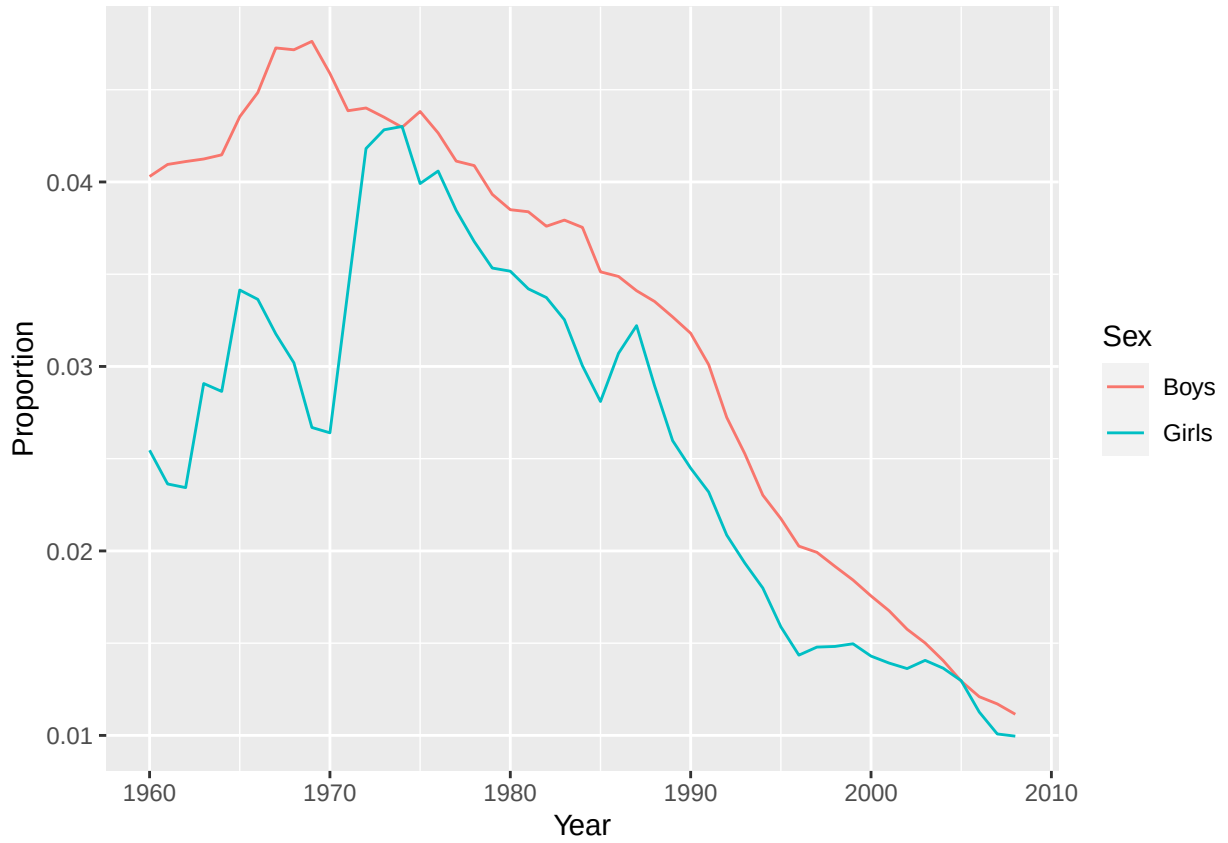
Filtrar los datos, msotrando solo el nombre más popular para cada grupo `Year X Sex`, manteniendo solo las columnas `Year`, `Name`, `Sex` y 'Proportion:

```
baby_names_popular <-
    baby_names %>%
    filter(Rank == 1) %>%
    select(Year, Name, Sex, Proportion)

baby_names_popular
```

```
## # A tibble: 98 x 4
##     Year Name    Sex   Proportion
##    <dbl> <chr>   <chr>      <dbl>
##  1  1960 Mary    Girls     0.0255
##  2  1960 David   Boys      0.0403
##  3  1961 Mary    Girls     0.0236
##  4  1961 Michael Boys      0.0409
##  5  1962 Lisa    Girls     0.0234
##  6  1962 Michael Boys      0.0411
##  7  1963 Lisa    Girls     0.0291
##  8  1963 Michael Boys      0.0412
##  9  1964 Lisa    Girls     0.0286
## 10  1964 Michael Boys      0.0415
## # i 88 more rows
```

Graficar los datos del paso anterior, poniendo a `Year` en las abscisas y `Proportion` en las ordenadas:

```
qplot(x = Year, y = Proportion, colour = Sex,
    data = baby_names_popular, geom = "line")
```



Ejercicio 5

Filtrar los datos `baby_names`, conservando únicamente los 10 nombres más populares para niñas y niños en cada año:

```
baby_names_top10 <-
    baby_names %>%
    group_by(Year, Sex) %>%
    filter(Rank <= 10)

baby_names_top10
```

```
## # A tibble: 980 x 6
## # Groups:   Year, Sex [98]
##    Name     Sex   Count  Year Proportion  Rank
##    <chr>    <chr> <dbl> <dbl>      <dbl> <dbl>
##  1 Mary     Girls 51474  1960     0.0255     1
##  2 Susan    Girls 39200  1960     0.0194     2
##  3 Linda    Girls 37314  1960     0.0185     3
##  4 Karen    Girls 36376  1960     0.0180     4
##  5 Donna    Girls 34133  1960     0.0169     5
##  6 Lisa     Girls 33702  1960     0.0167     6
##  7 Patricia Girls 32102  1960     0.0159     7
##  8 Debra    Girls 26737  1960     0.0132     8
##  9 Cynthia  Girls 26725  1960     0.0132     9
## 10 Deborah  Girls 25264  1960     0.0125    10
```

```
## # i 970 more rows
```

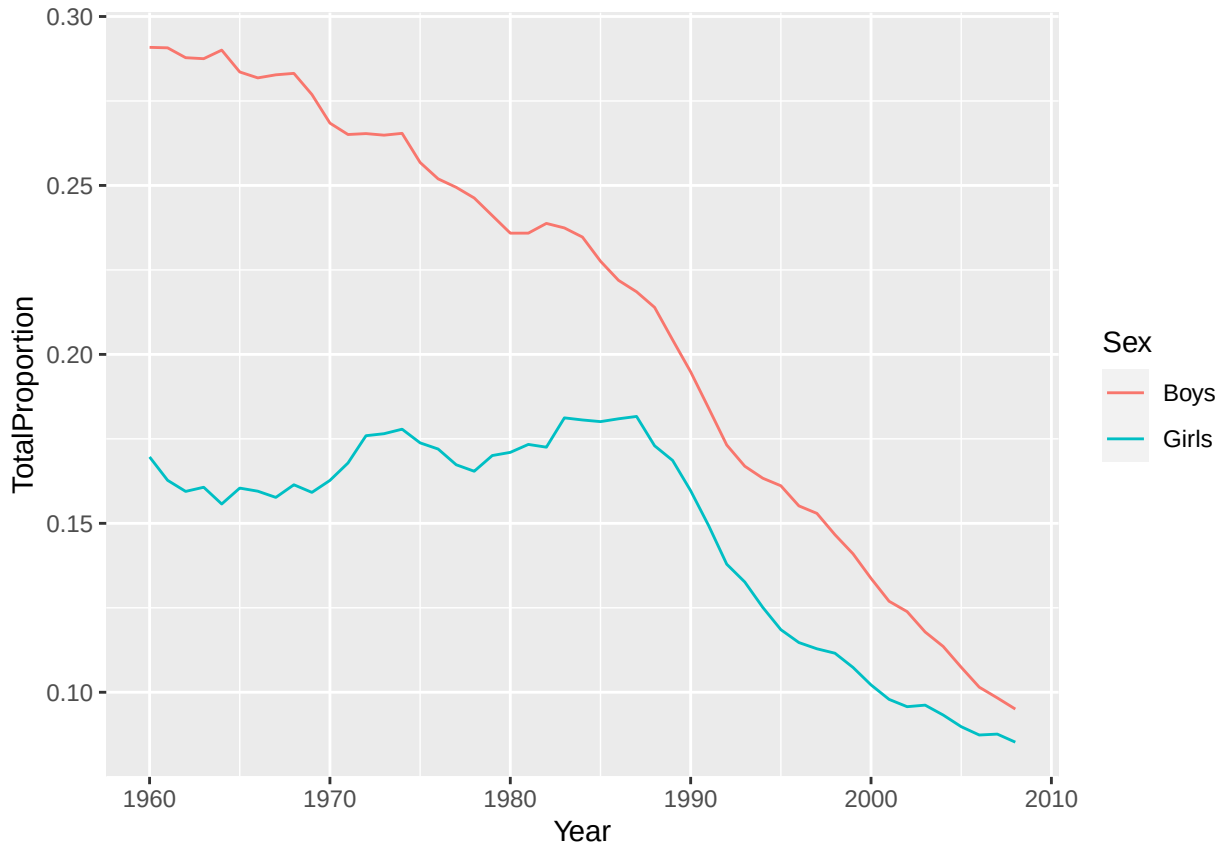
Resumir los datos del paso anterior para calcular la proporción total `TotalProportion` de niñas y niños con nombres en el top 10:

```
baby_names_top10_proportion <-
    baby_names_top10 %>%
    summarise(TotalProportion = sum(Proportion))

baby_names_top10_proportion
```

```
## # A tibble: 98 x 3
## # Groups:   Year [49]
##     Year Sex   TotalProportion
##    <dbl> <chr>           <dbl>
##  1  1960 Boys            0.291
##  2  1960 Girls           0.170
##  3  1961 Boys            0.291
##  4  1961 Girls           0.163
##  5  1962 Boys            0.288
##  6  1962 Girls           0.159
##  7  1963 Boys            0.288
##  8  1963 Girls           0.161
##  9  1964 Boys            0.290
## 10  1964 Girls           0.156
## # i 88 more rows
```

Graficar los datos del paso anterior, con `Year` en las abscisas y `TotalProportion` en las ordenadas:

```
qplot(x = Year, y = TotalProportion, colour = Sex,
    data = baby_names_top10_proportion, geom = "line")
```



Gráficas en R

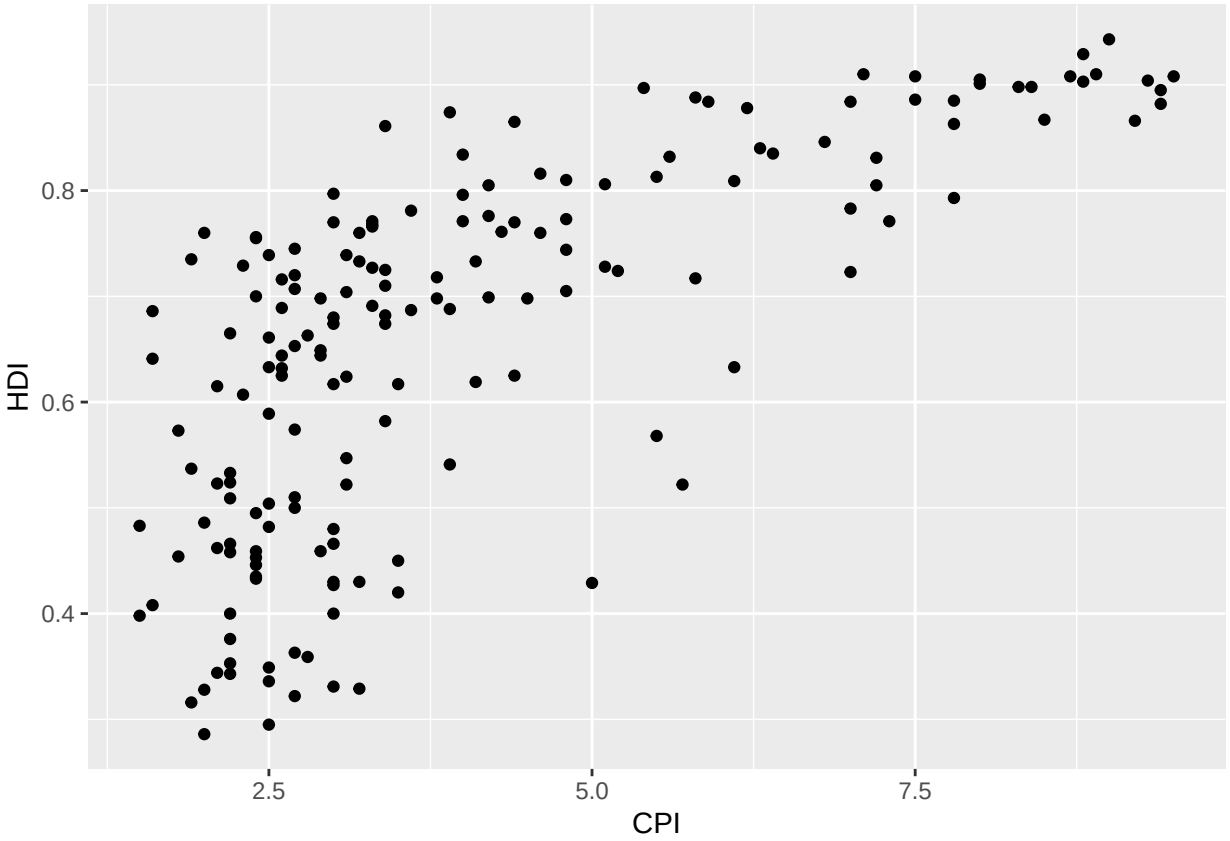
Ejercicio 0

Almacenar los datos en `dat`:

```
dat <- read_csv("Datos/EconomistData.csv")
```

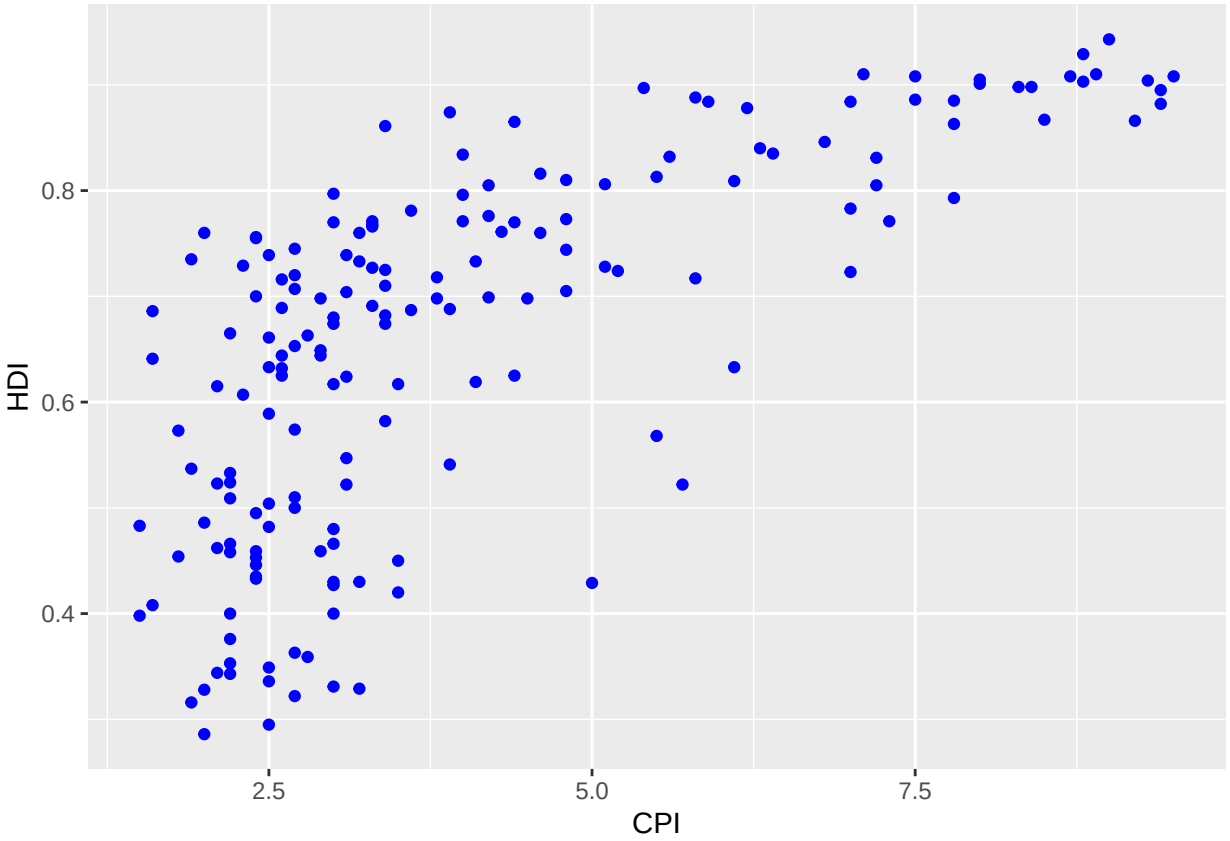
Crear una gráfica de dispersión con `CPI` en las abscisas y `HDI` en las ordenadas:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) + geom_point()
```



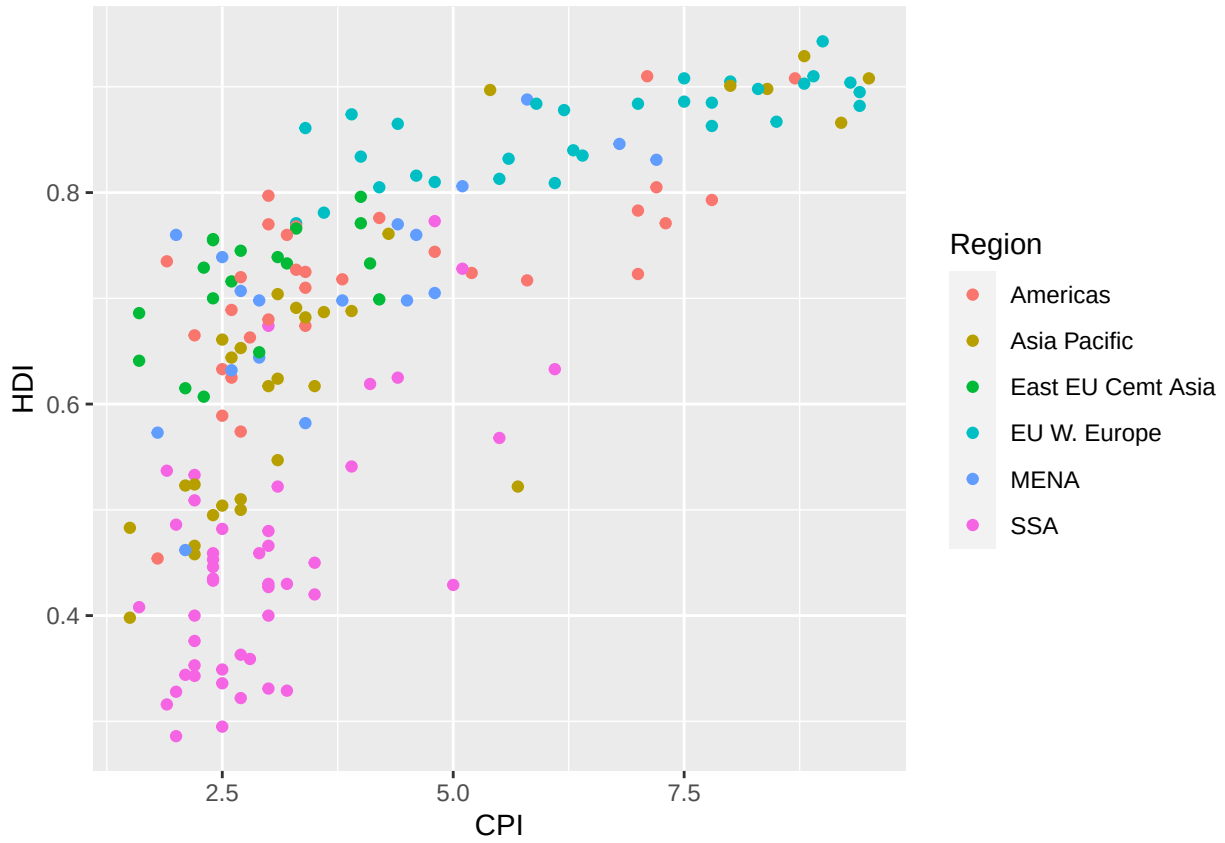
Colorear en azul los puntos de la gráfica anterior:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point(colour = "blue")
```



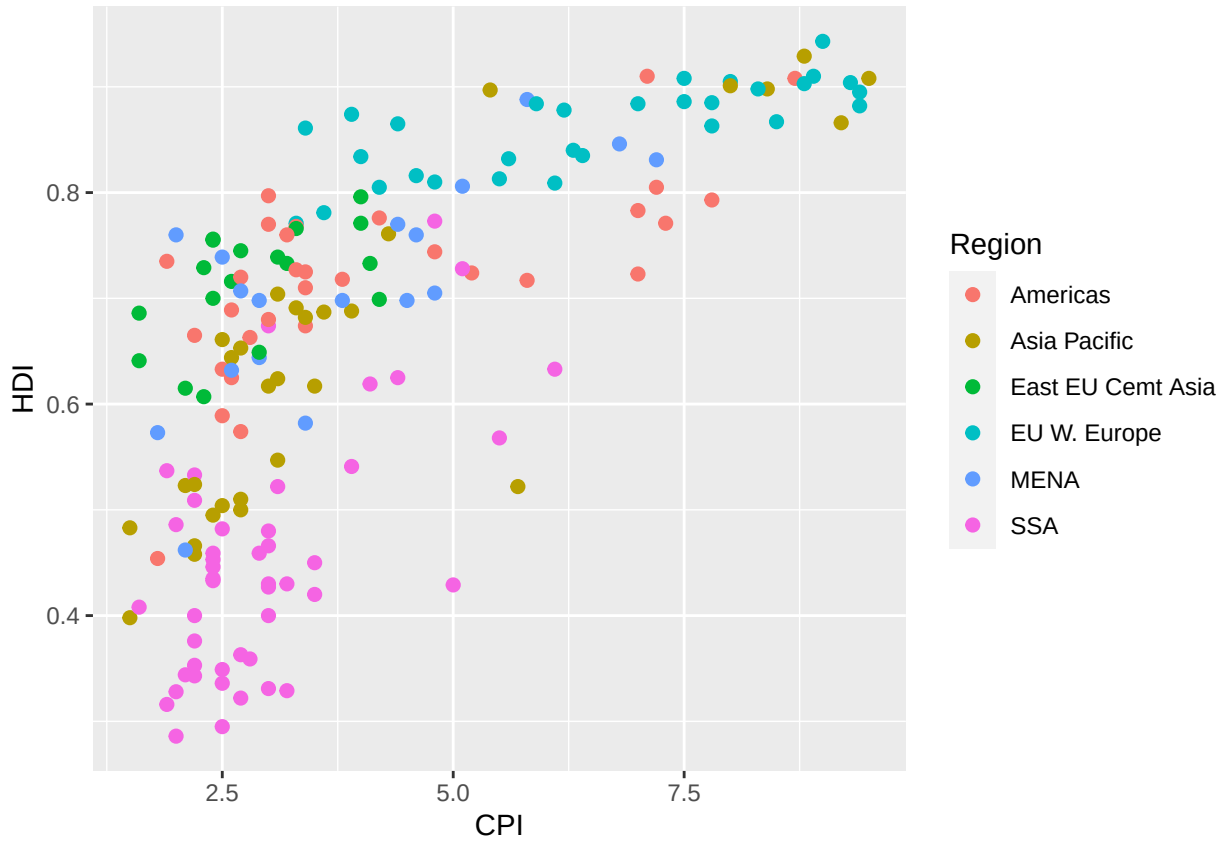
Asignar un color a los puntos según `Region`:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point(aes(colour = Region))
```



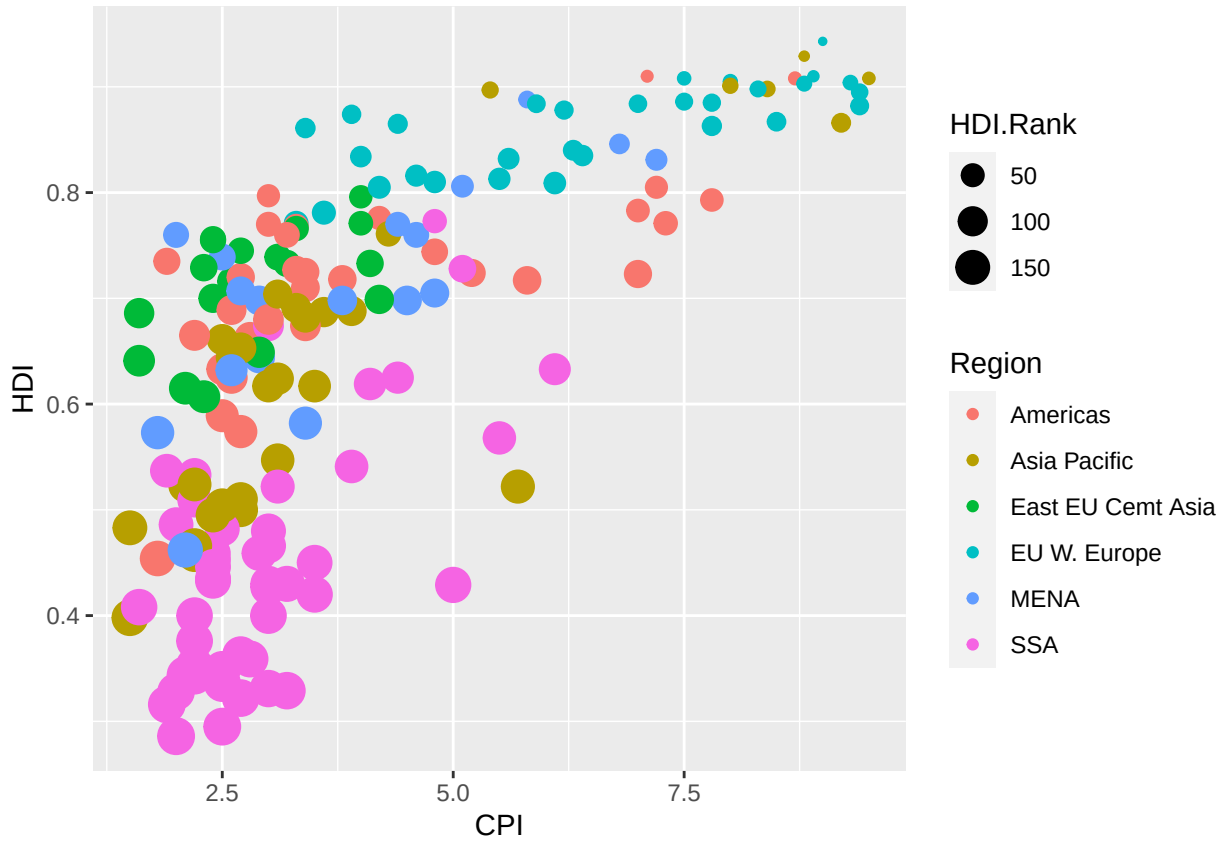
Manteniendo el color según `Region`, agranadr los puntos asignándoles un tamaño de 2:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point(aes(colour = Region), size = 2)
```



Manteniendo el color según `Region`, asignar un tamaño según `HDI.Rank`:

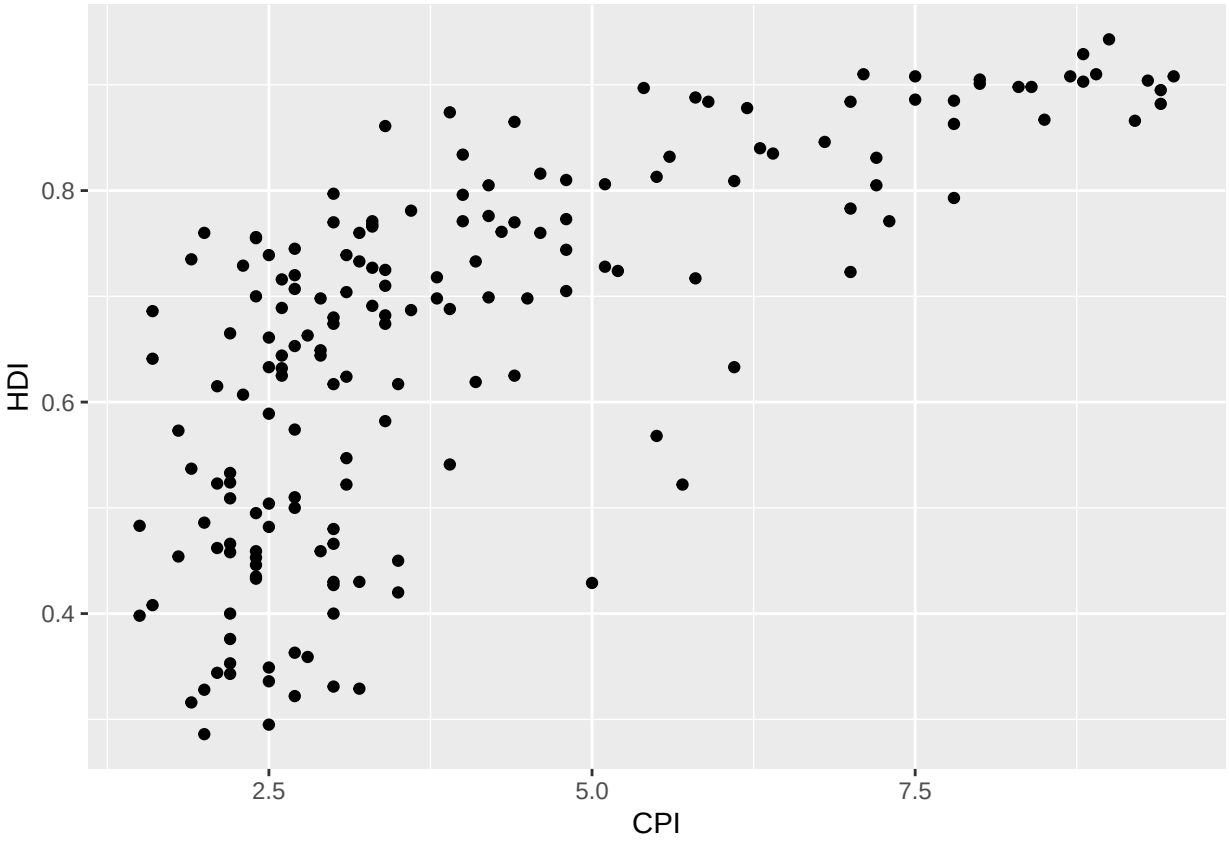
```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point(aes(colour = Region, size = HDI.Rank))
```



Ejercicio 1

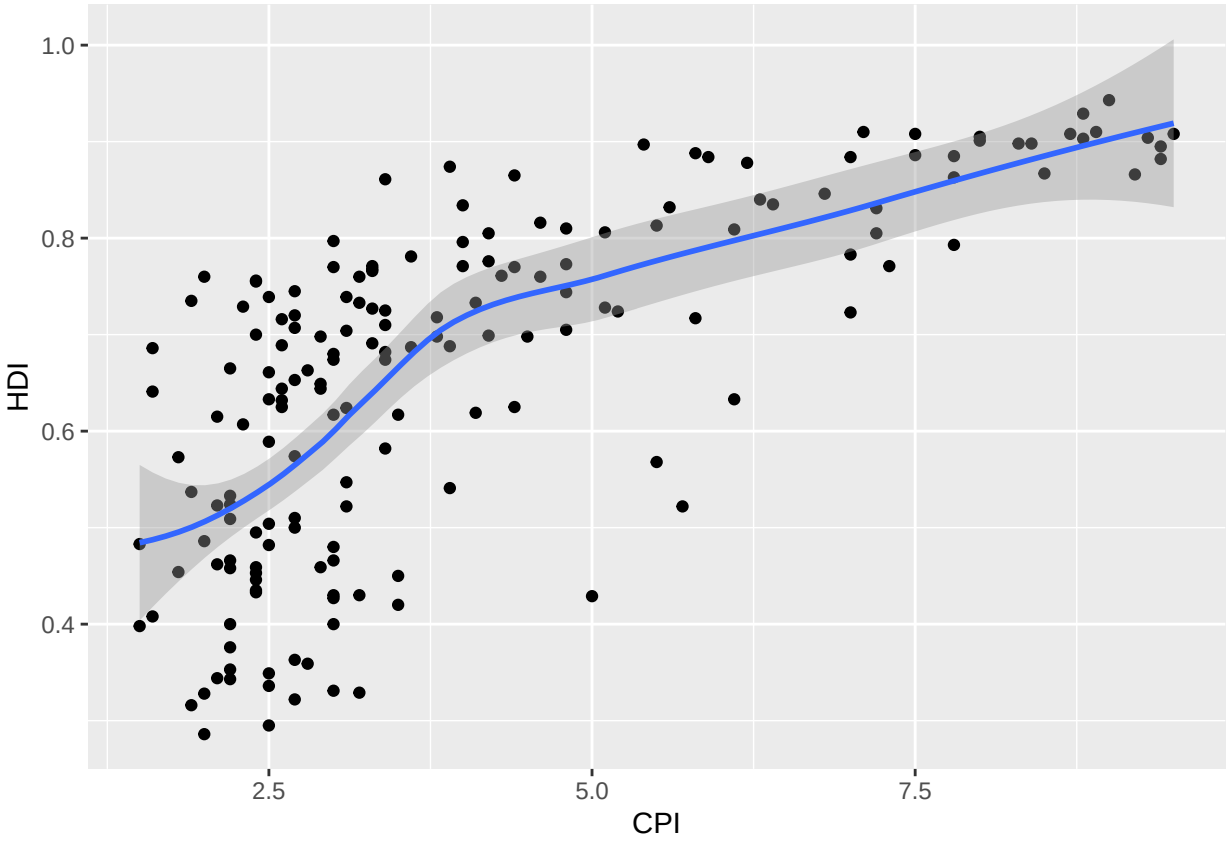
Recrear la gráfica de dispersión con `CPI` en las abscisas y `HDI` en las ordenadas, tal como en el ejercicio anterior:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point()
```



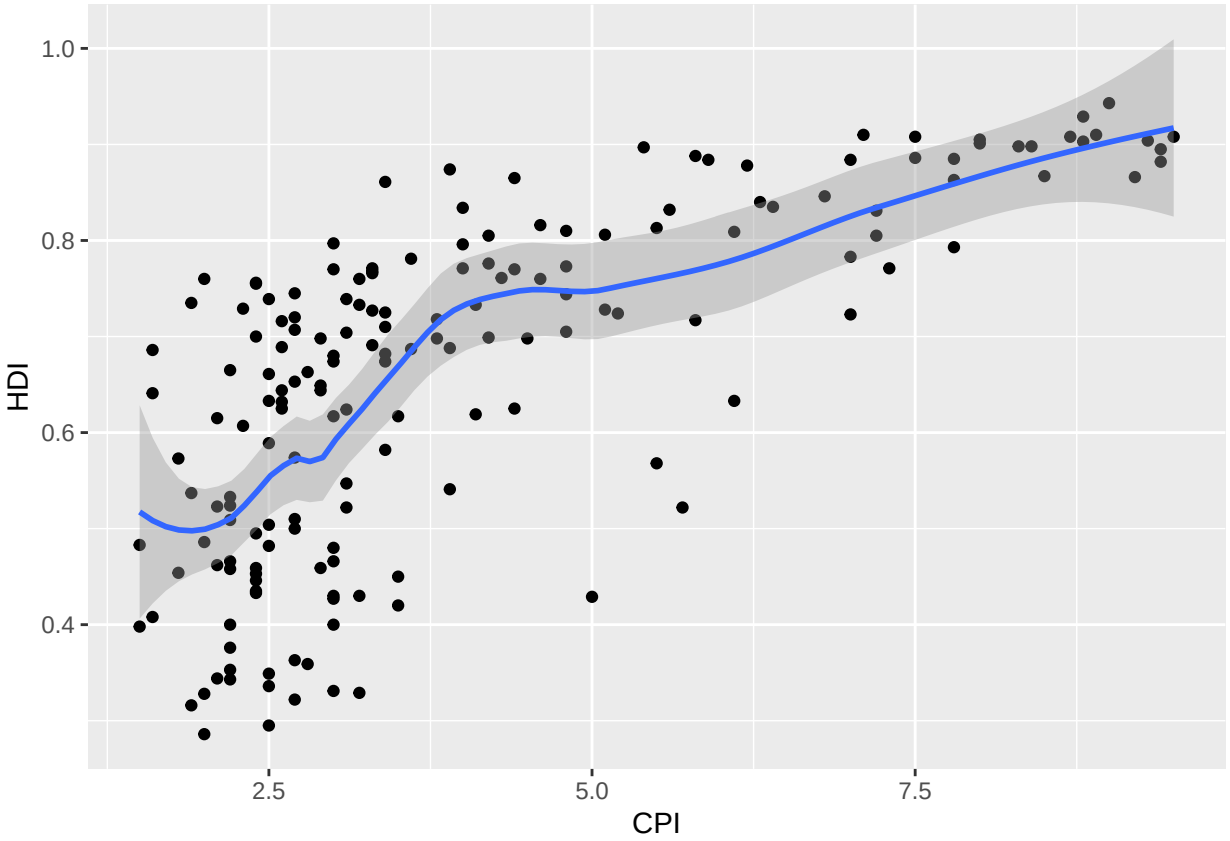
Superponer una línea de suavizado sobre la gráfica de dispersión, utilizando `geom_smooth()`:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point() + geom_smooth()
```



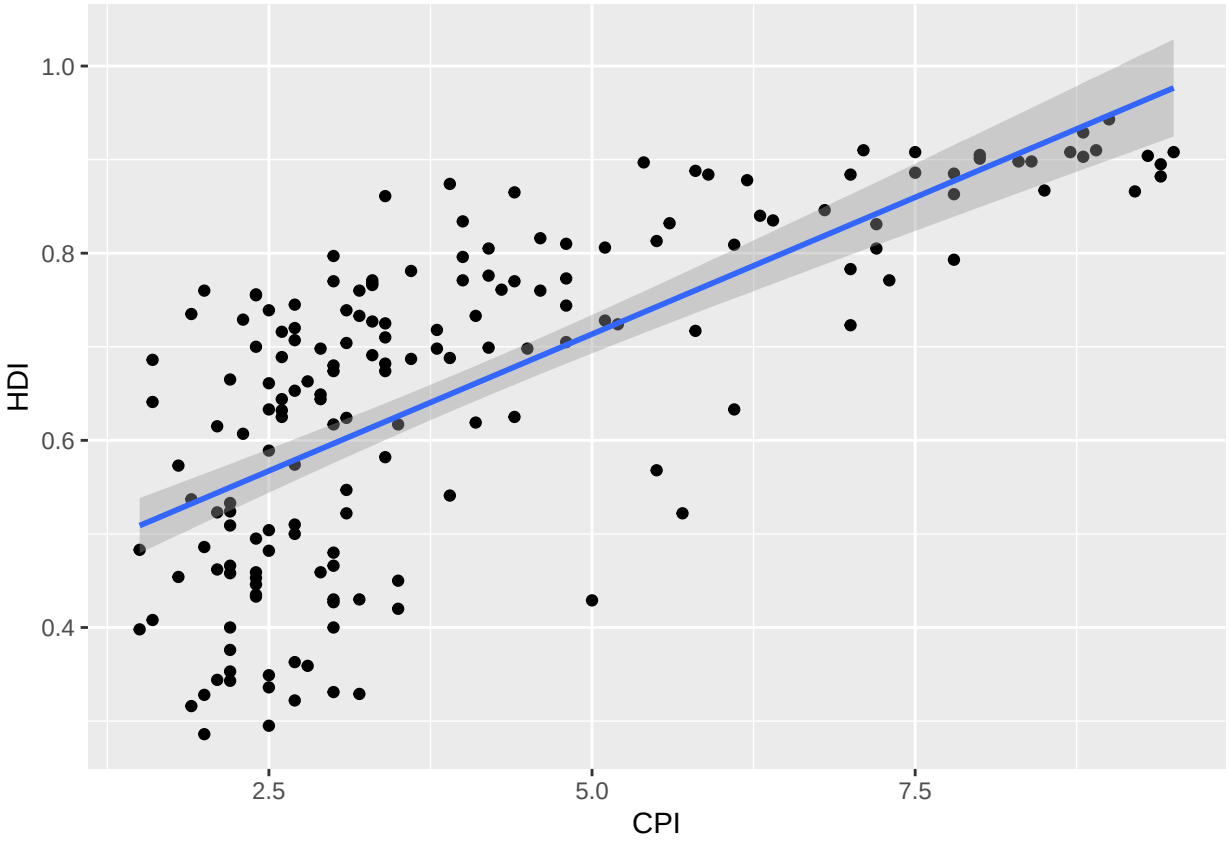
Hacer la línea de suavizado menos suave:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point() + geom_smooth(span = 0.5)
```



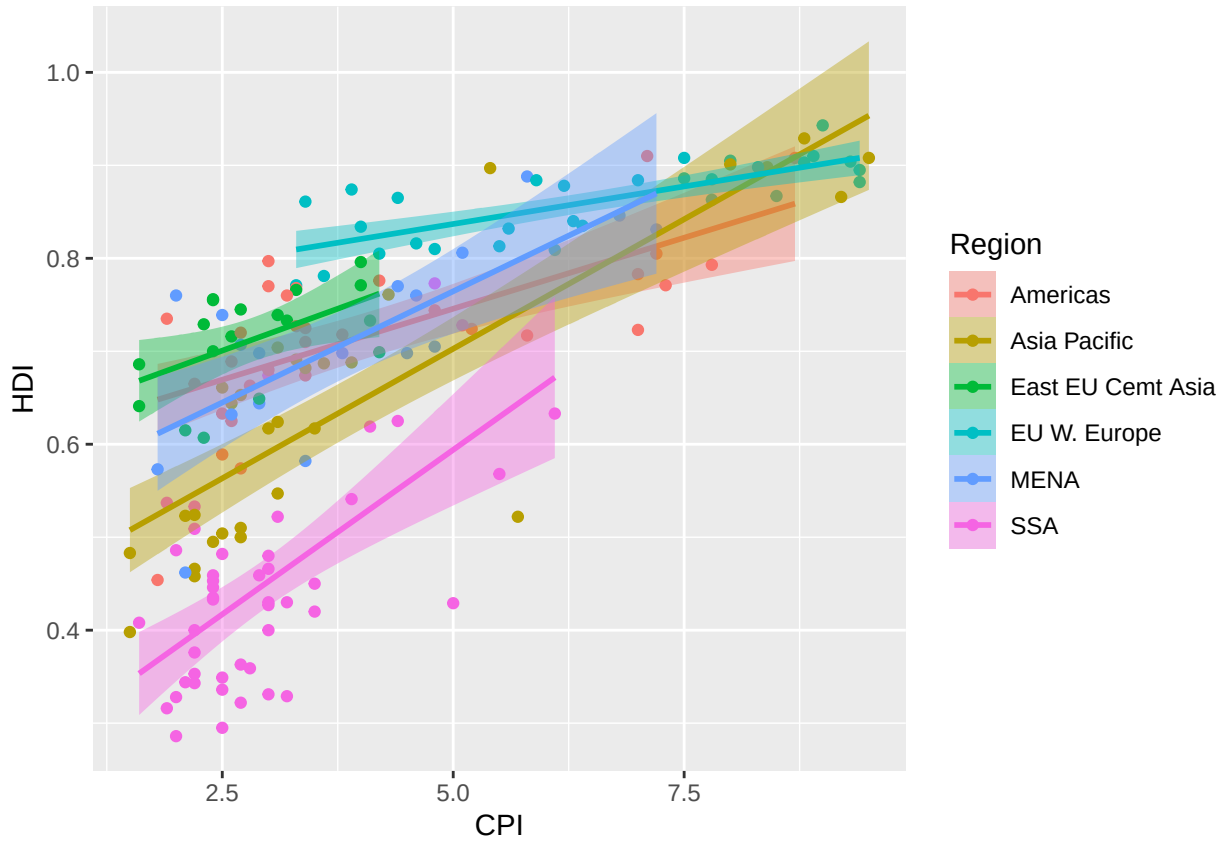
Cambiar la línea de suavizado para que utilize un modelo linear para las predicciones:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point() + geom_smooth(span = 0.5, method = "lm")
```



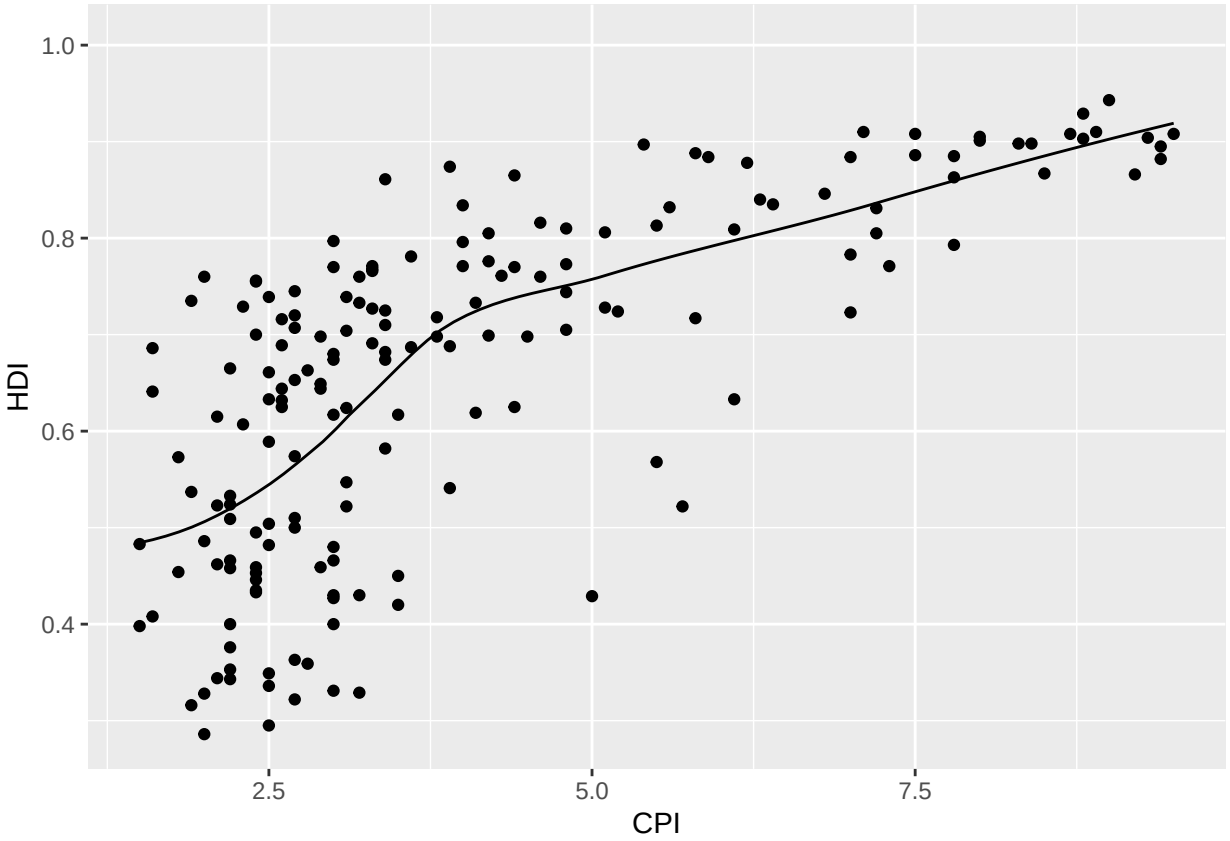
Permitir que la línea de suavizado del paso anterior varíe a través de los niveles de `Region`:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI,
    colour = Region, fill = Region)) + geom_point() +
    geom_smooth(span = 0.5, method = "lm")
```



Sobreponer una línea de suavizado LOESS sobre la gráfica de dispersión utilizando `geom_line()`:

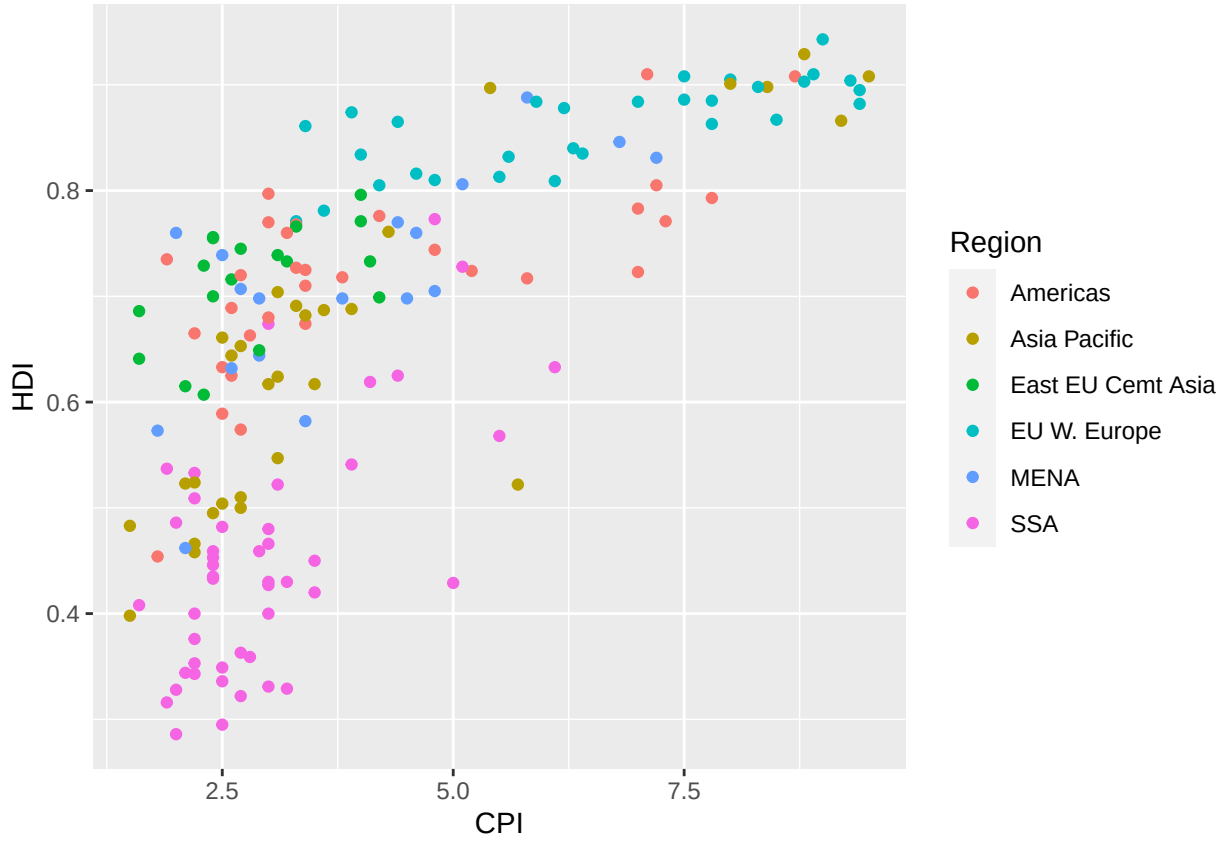
```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI)) +
    geom_point() + geom_line(stat = "smooth", method = "loess")
```



Ejercicio 2

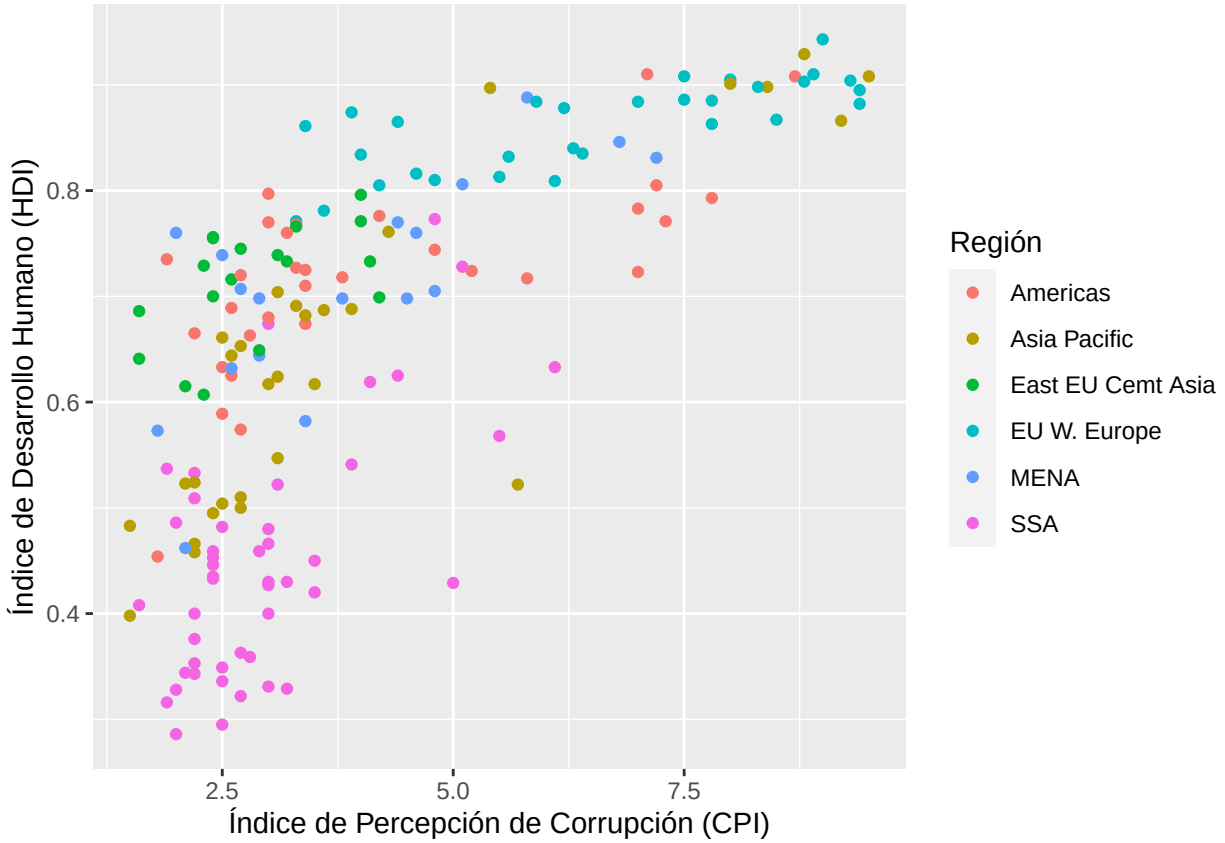
Crear una gráfica de dispersión con `CPI` en las abscisas y `HDI` en las ordenadas. Colorear los puntos para indicar `Region`:

```
ggplot(data = dat, mapping = aes(x = CPI,
    y = HDI, colour = Region)) +
    geom_point()
```



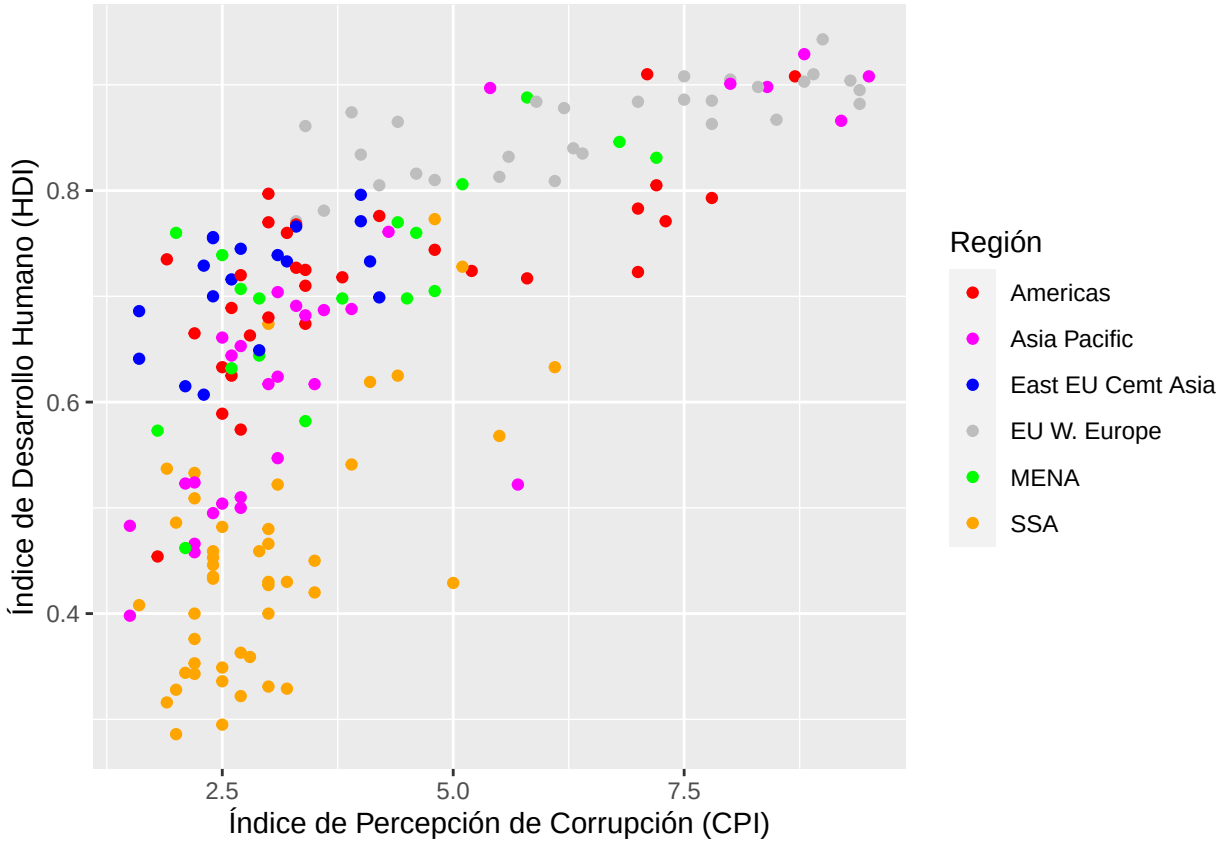
Modificar las escalas de las abscisas, ordenadas y los colores para que tengan nombres más entendibles:

```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI, colour = Region)) +
    geom_point() +
    scale_x_continuous(name = "Índice de Percepción de Corrupción (CPI)") +
    scale_y_continuous(name = "Índice de Desarrollo Humano (HDI)") +
    scale_colour_discrete(name = "Región")
```



Modificar la escala de colores para que utilice valores específicos de mi elección:

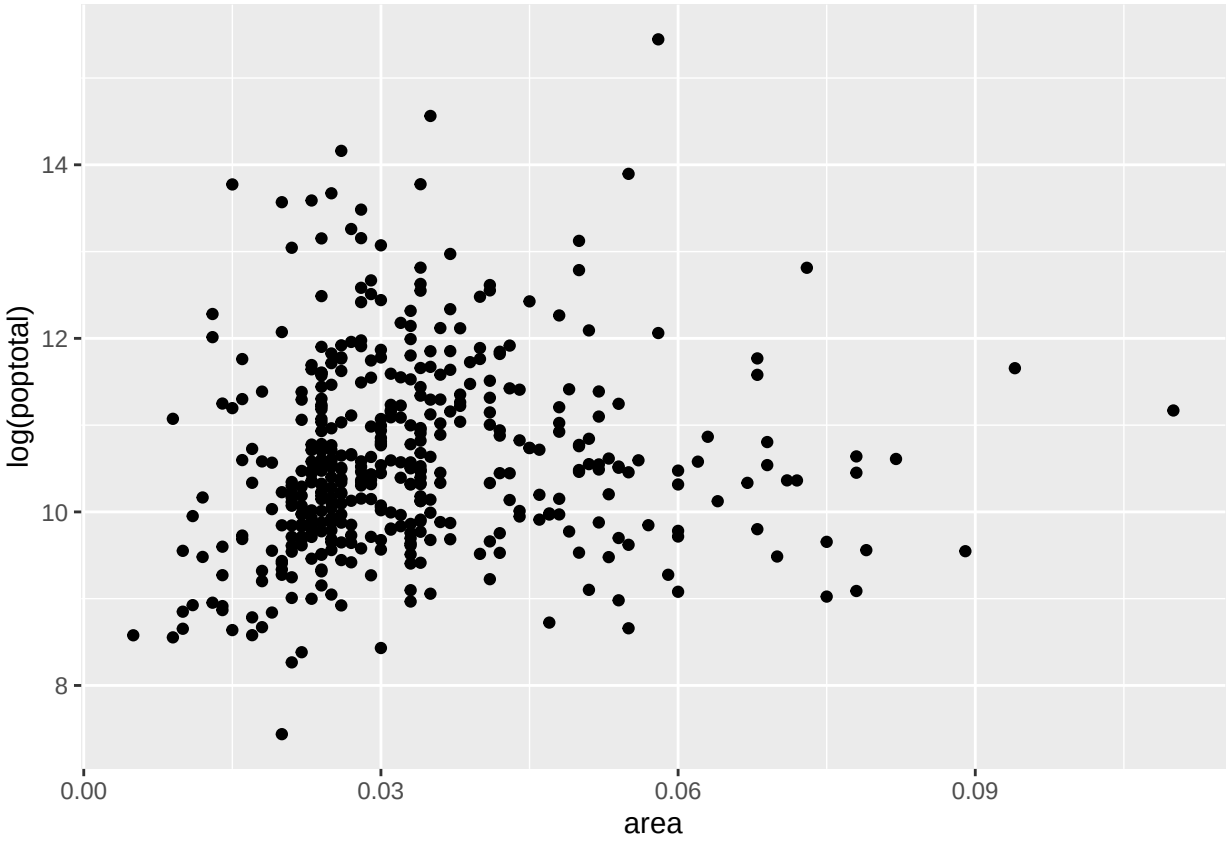
```
ggplot(data = dat, mapping = aes(x = CPI, y = HDI, colour = Region)) +
    geom_point() +
    scale_x_continuous(name = "Índice de Percepción de Corrupción (CPI)") +
    scale_y_continuous(name = "Índice de Desarrollo Humano (HDI)") +
    scale_colour_manual(name = "Región",
        values = c("red", "magenta", "blue",
            "grey", "green", "orange"))
```



Ejercicio 3

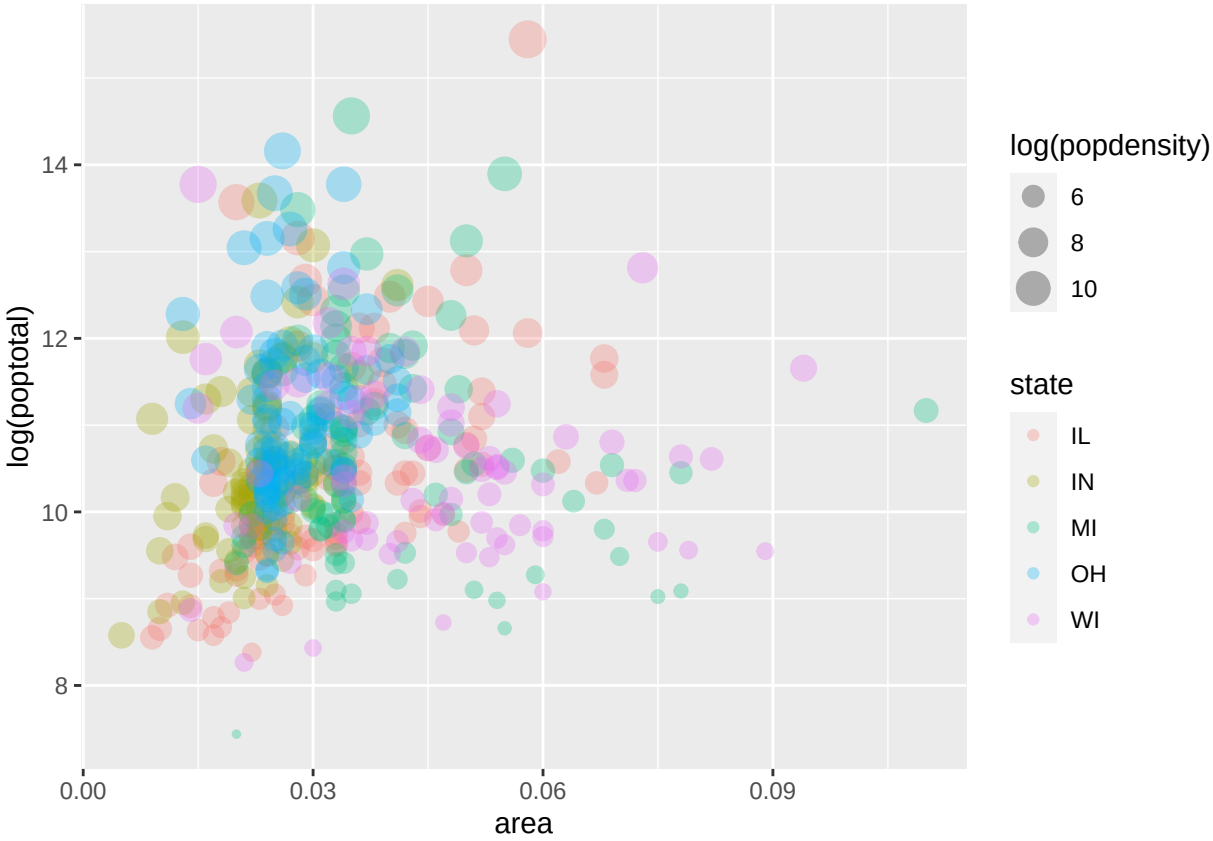
Crear una gráfica de dispersión con `area` en las abscisas y el logaritmo de `poptotal` en las ordenadas:

```
ggplot(data = midwest, mapping = aes(x = area, y = log(poptotal))) +
    geom_point()
```



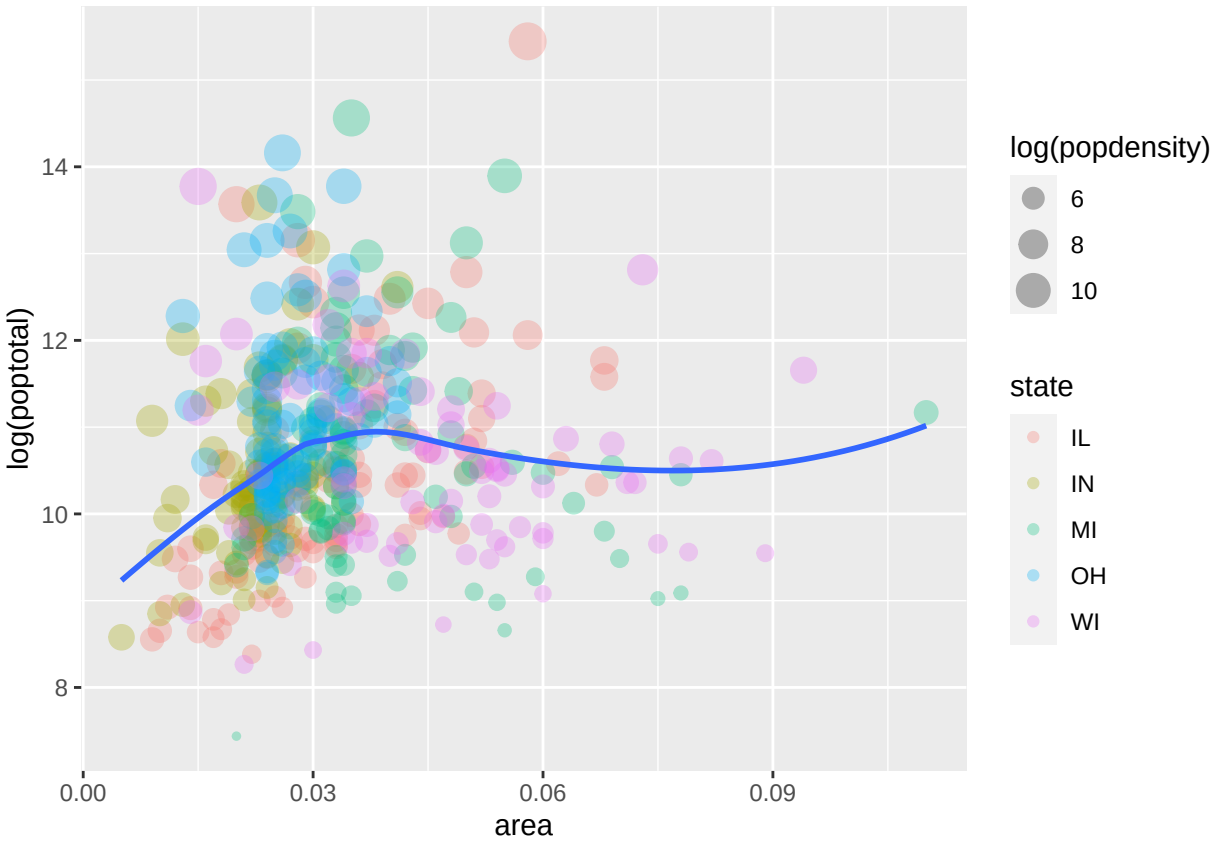
Dentro de la llamada a `geom_call()`, asignar color según `state`, asignar tamaño según el logaritmo de `popdensity` y fijar la transparencia (`alpha`) en 0.3:

```
scatter <- ggplot(data = midwest, mapping = aes(x = area, y = log(poptotal))) +
    geom_point(aes(colour = state, size = log(popdensity)), alpha = 0.3)

scatter
```



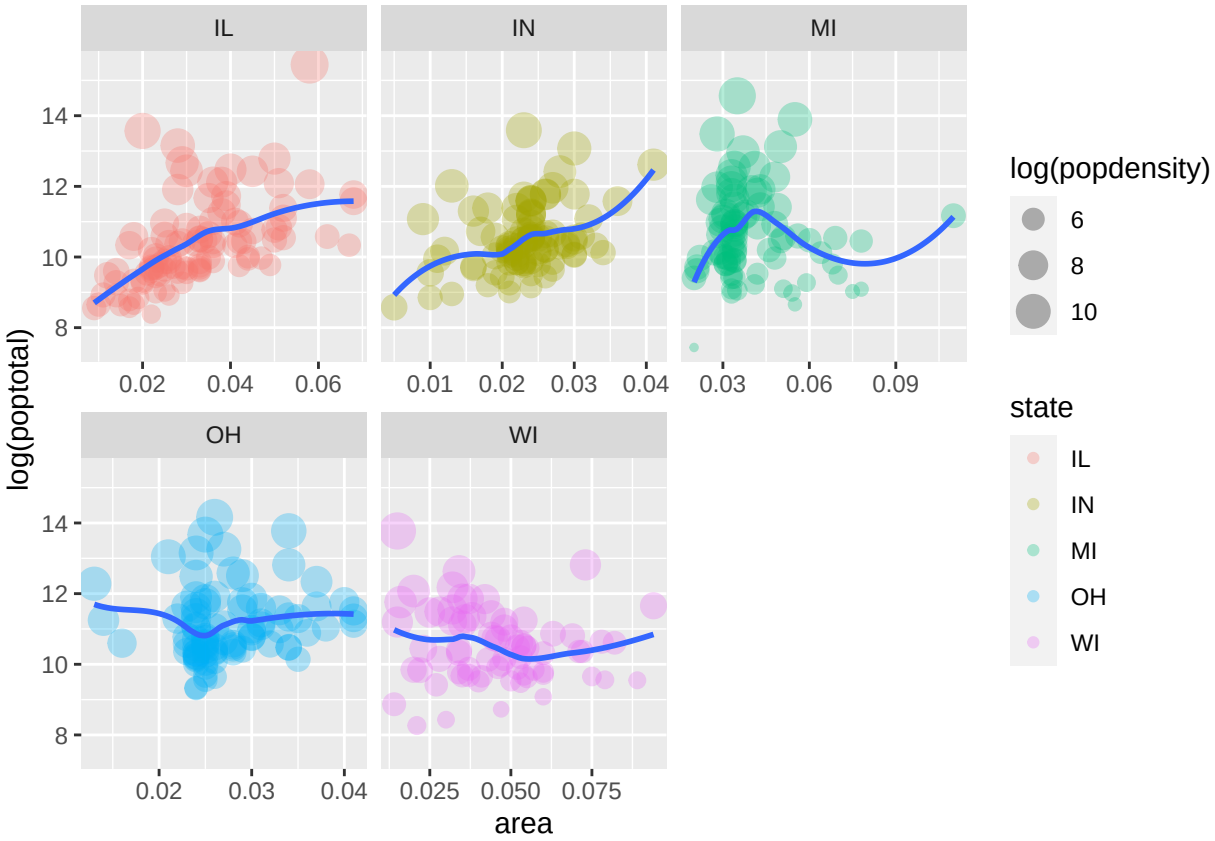
Agregar una línea de suavizado y desactivar la graficación del intervalo de confianza:

```
scatter_smoothed <- scatter + geom_smooth(se = FALSE)

scatter_smoothed
```



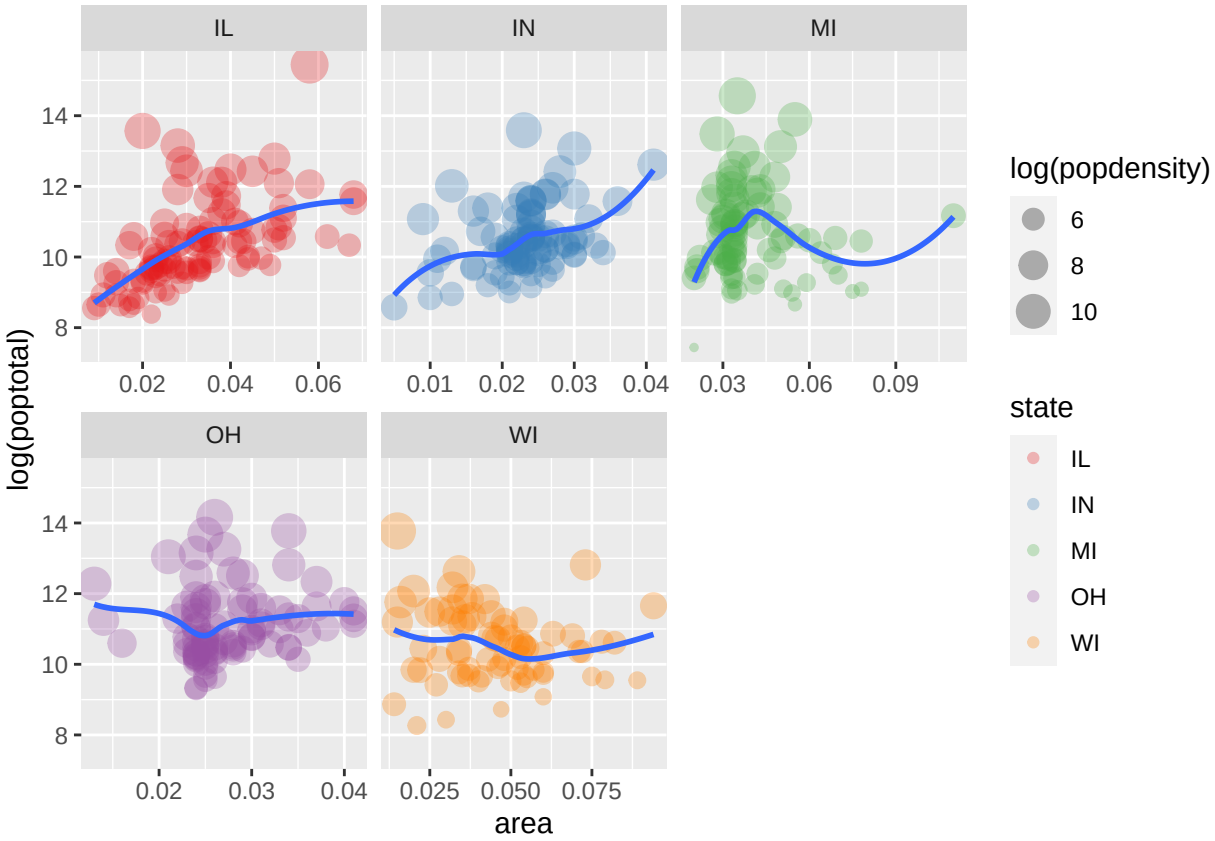
Facetar la gráfica según `state`. Poner `facet_wrap()` en el argumento `scales` para permitir rangos separados en las abscisas:

```
scatter_faceted <- scatter_smoothed + facet_wrap(~ state, scales = "free_x")

scatter_faceted
```



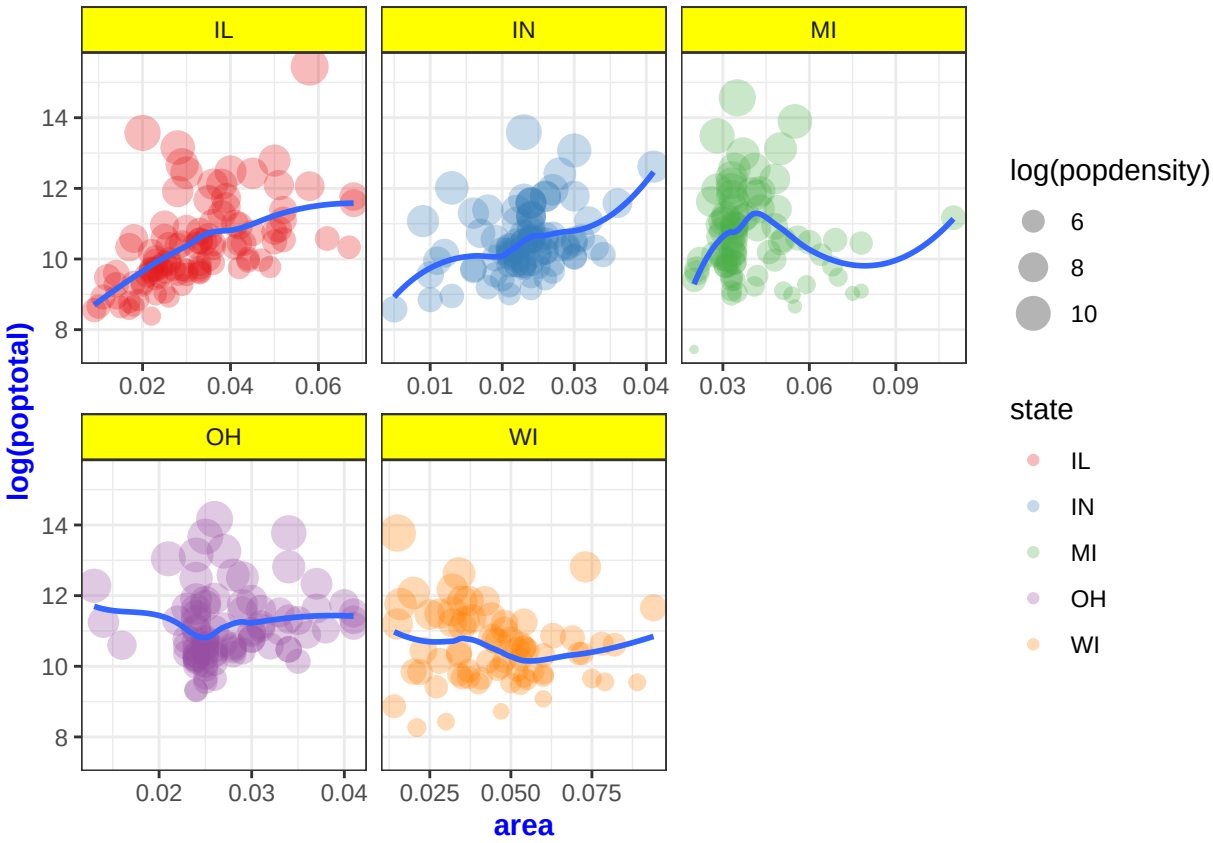
Cambiar la escala de colores por defecto para utilizar la paleta discreta `RColorBrewer` llamada `Set1`:

```
scatter_faceted_colour <- scatter_faceted +
    scale_colour_brewer(palette = "Set1")

scatter_faceted_colour
```



Cambiar el tema por defecto a `theme_bw()` y mofificarlo para que el título del eje sea azul en negritas y el fondo del nombre de la faceta sea amarillo.

```
scatter_faceted_themed <- scatter_faceted_colour + theme_bw() +
    theme(axis.title = element_text(colour = "blue", face = "bold"),
        strip.background = element_rect(fill = "yellow"))

scatter_faceted_themed
```


IL
IN
MI
OH
WI
log(poptotal)
area
14
12
10
8
0.02
0.04
0.06
0.01
0.02
0.03
0.04
0.03
0.06
0.09
0.02
0.03
0.04
0.025
0.050
0.075
log(popdensity)
6
8
10
state
IL
IN
MI
OH
WI