Boas práticas na construção de gráficos

Sara Mortara, Andrea Sánchez-Tapia, Diogo S. B. Rocha

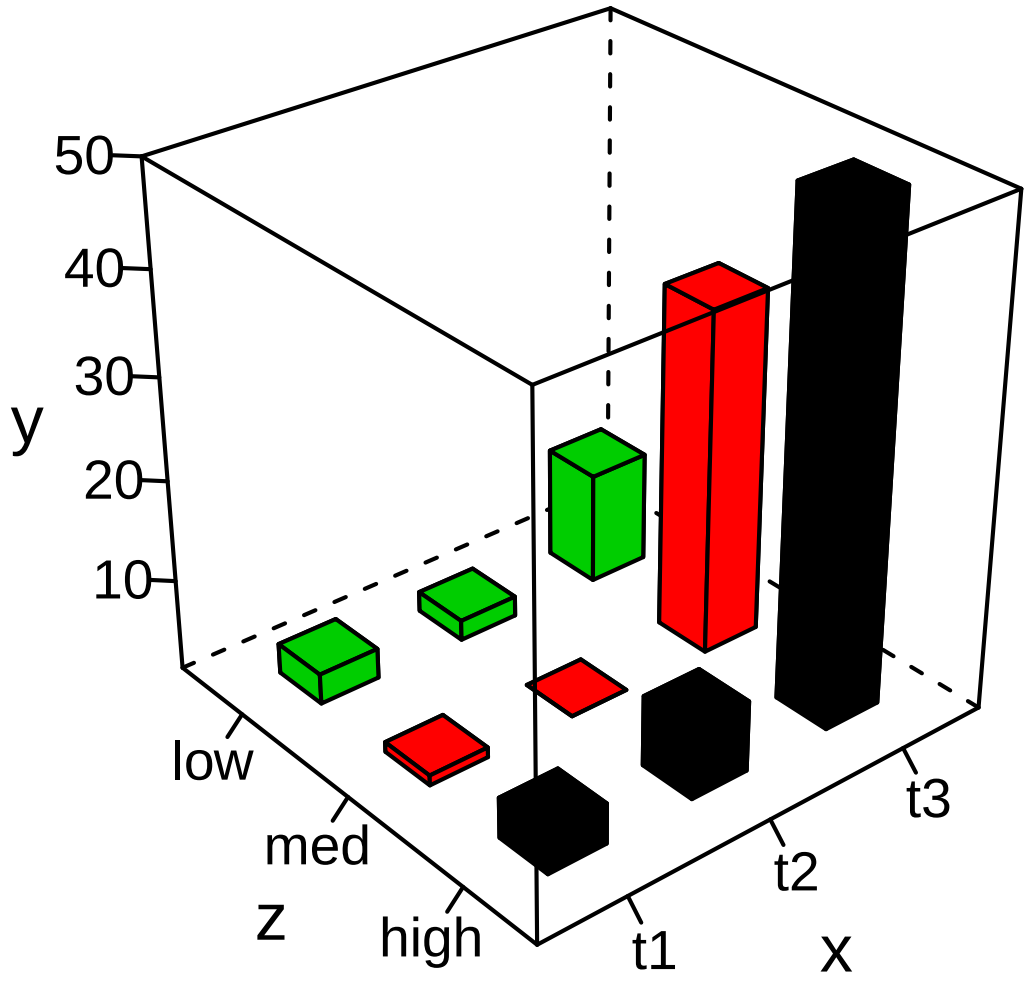
aula 07

sobre a aula

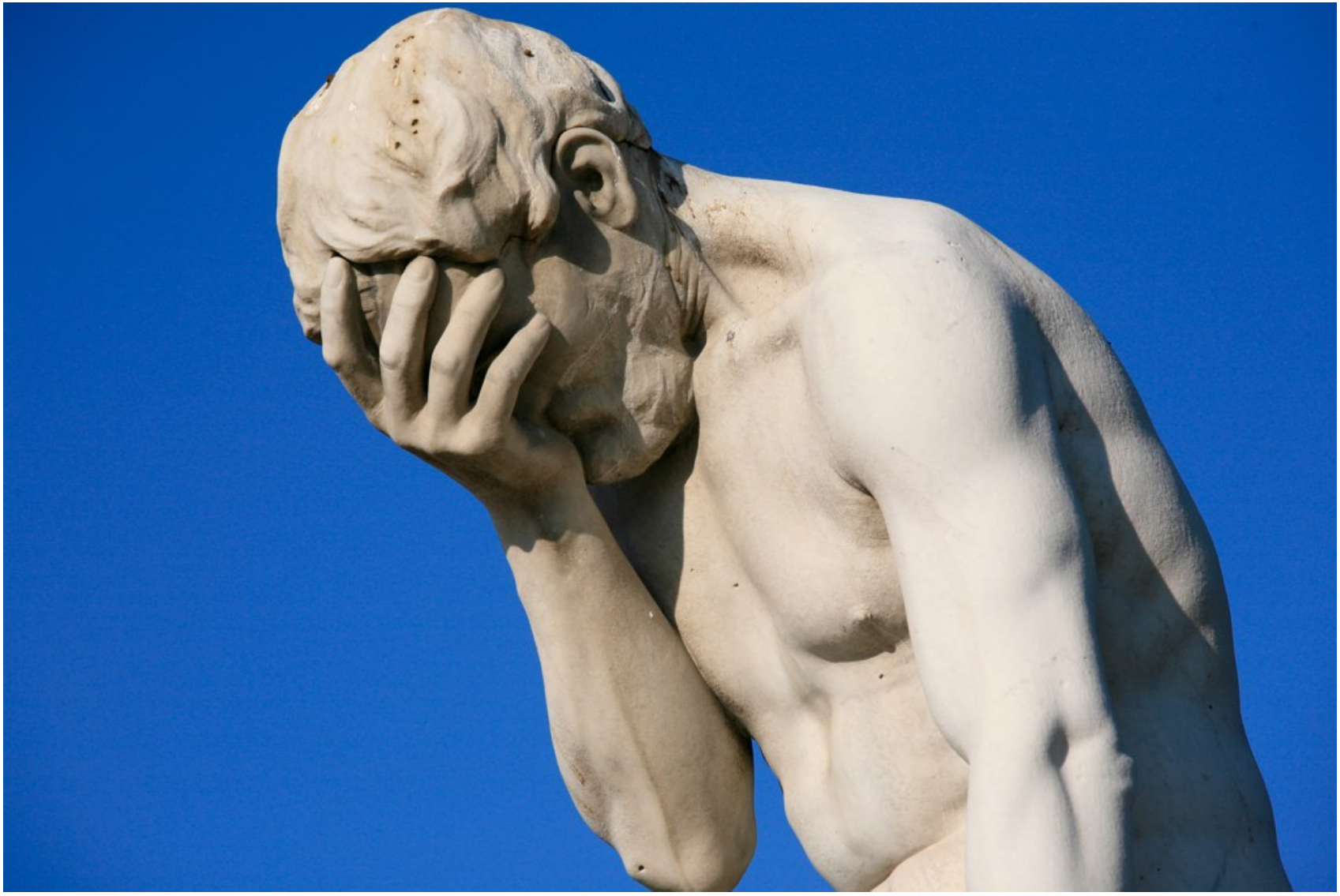
1. noções básicas de gráficos
2. editando gráficos básicos no R
3. perfumaria

1. noções básicas de gráficos

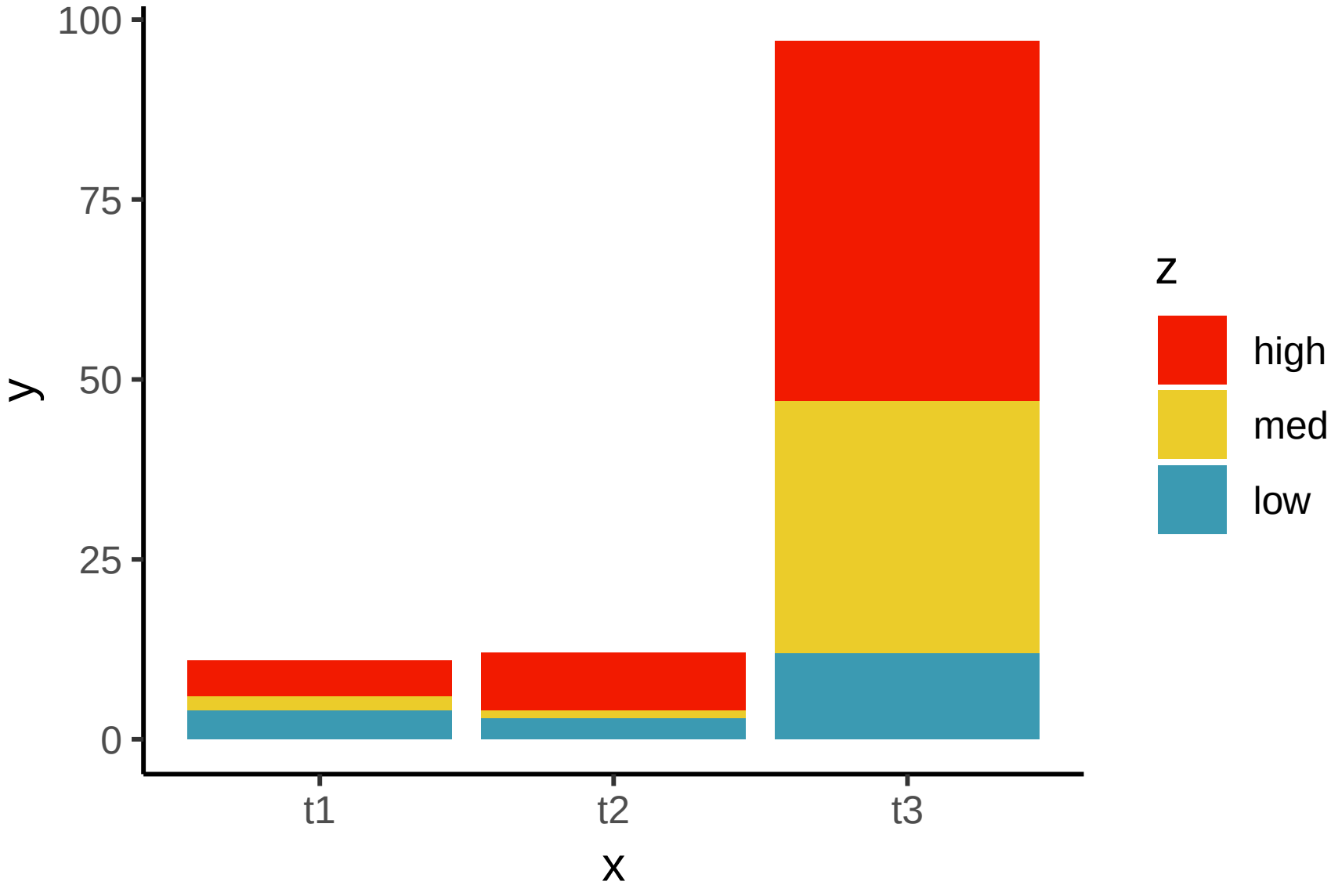
como [NÃO] fazer



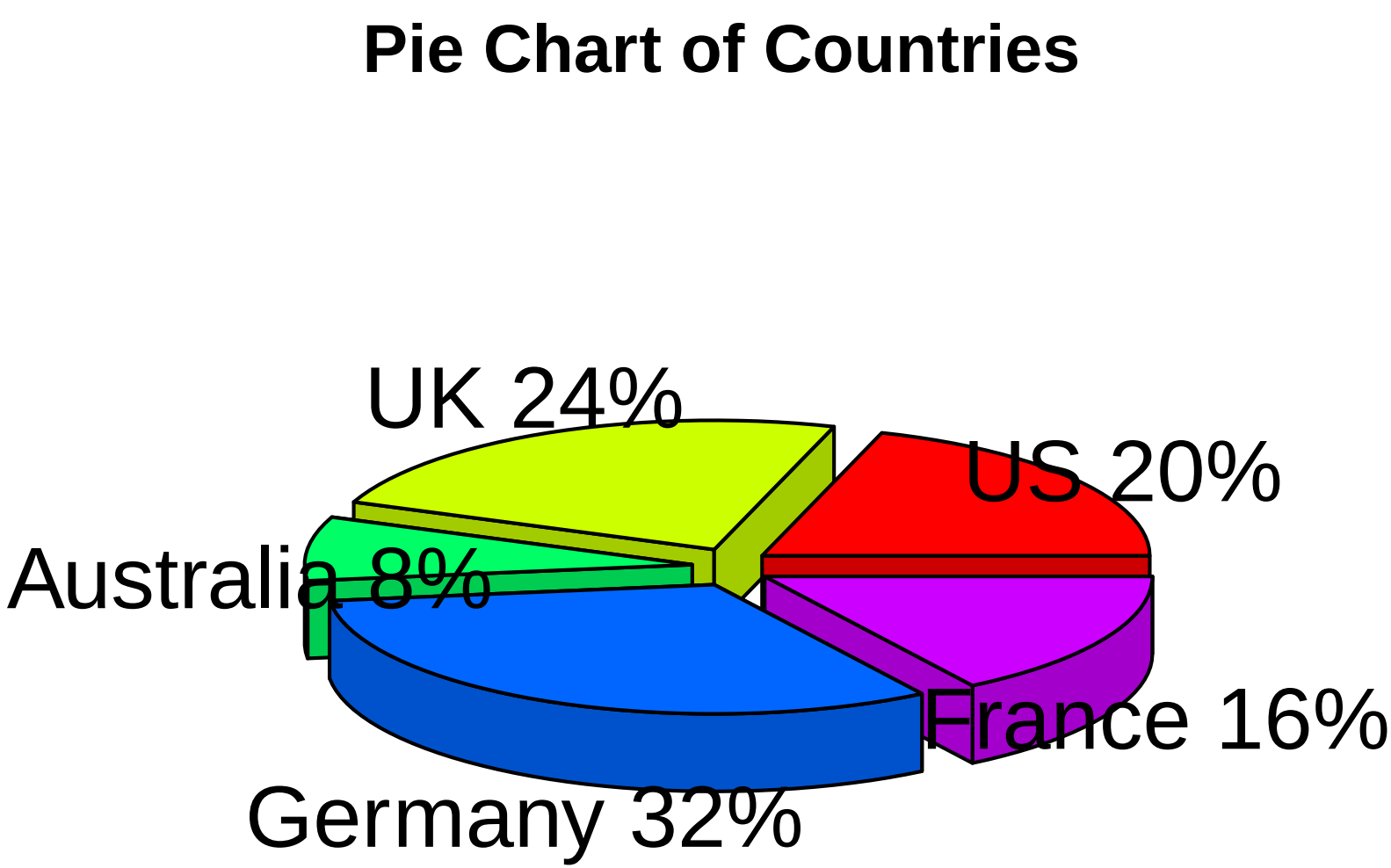
não mesmo



como fazer melhor

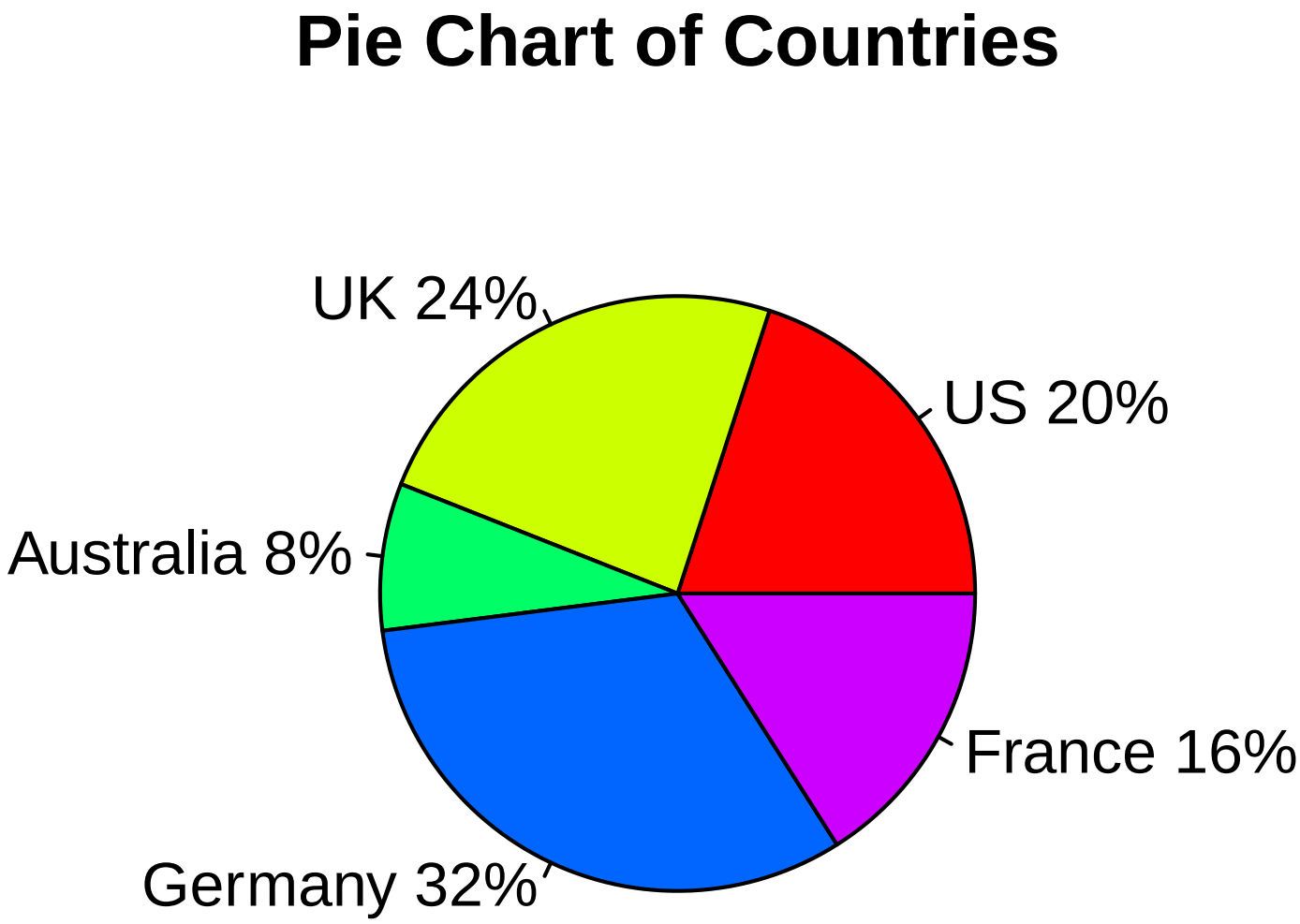


como [NUNCA] fazer

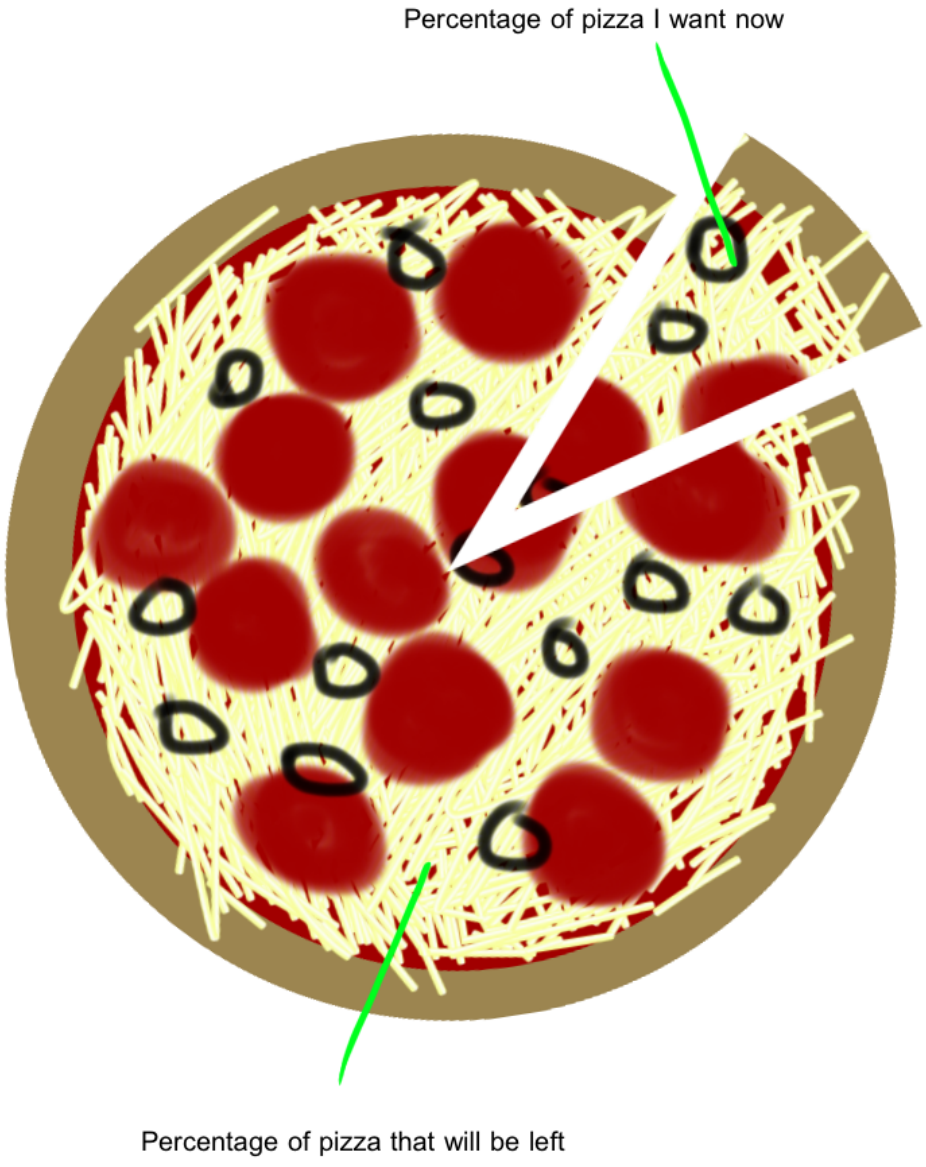


#nunca

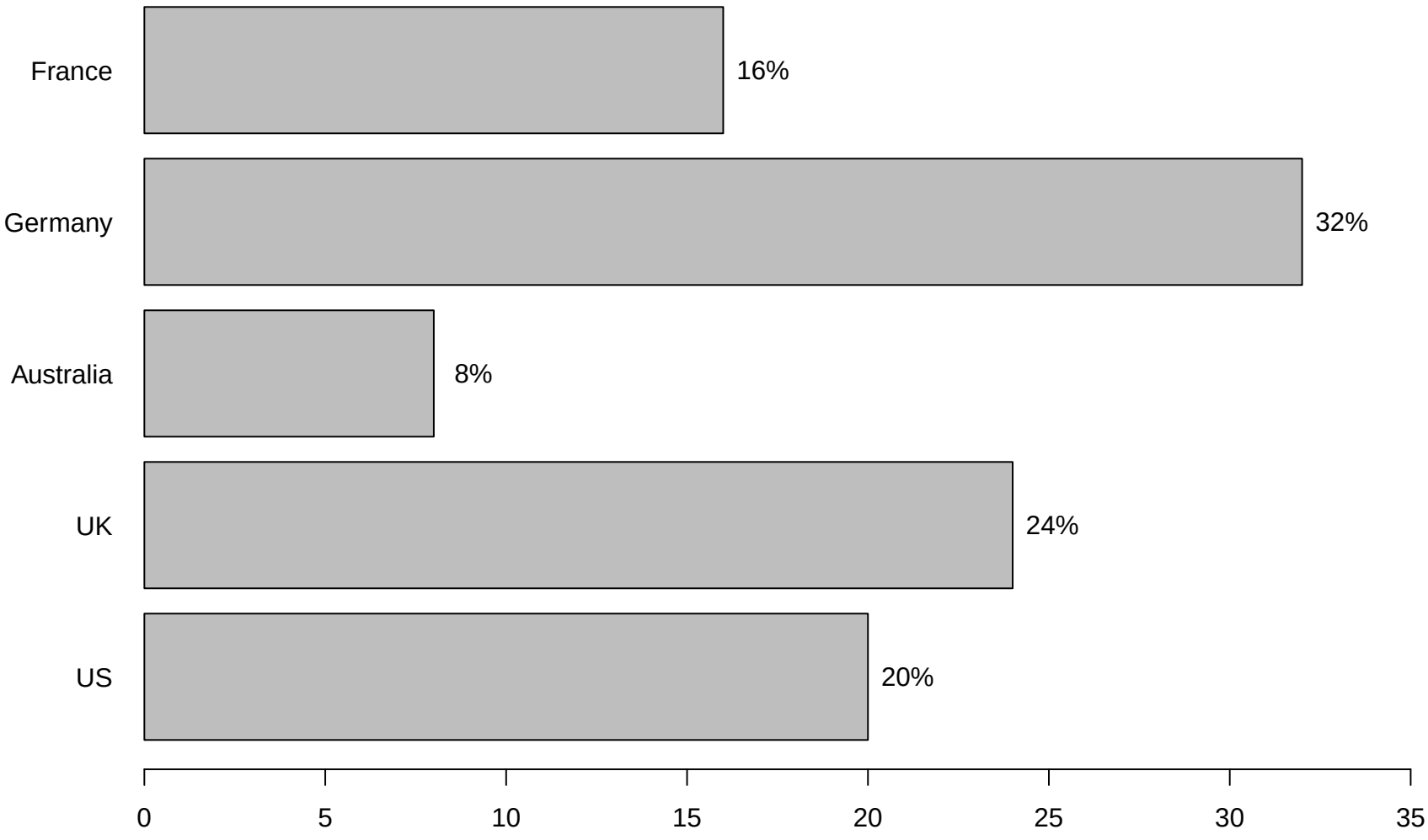
como [NÃO] fazer



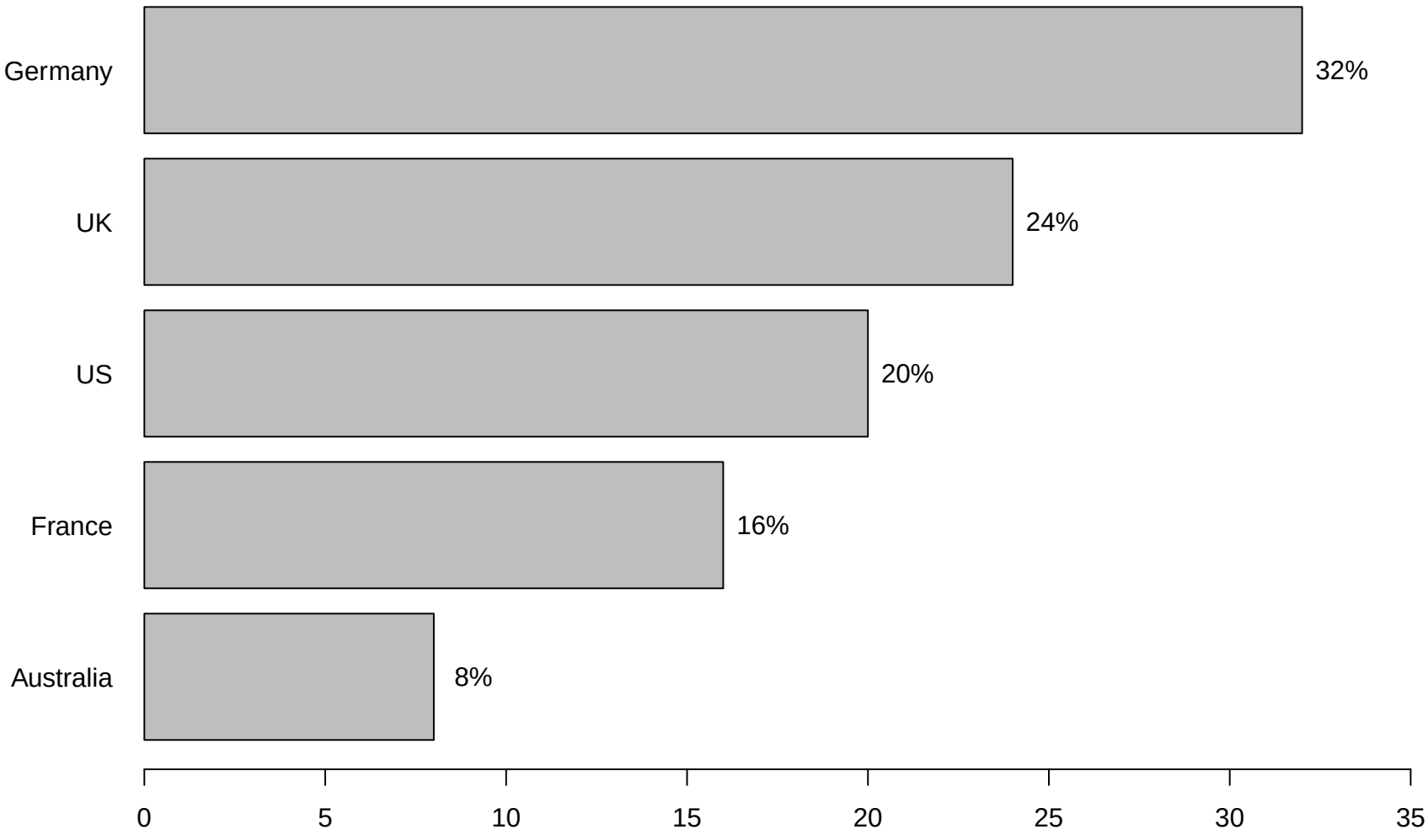
único gráfico de pizza possível



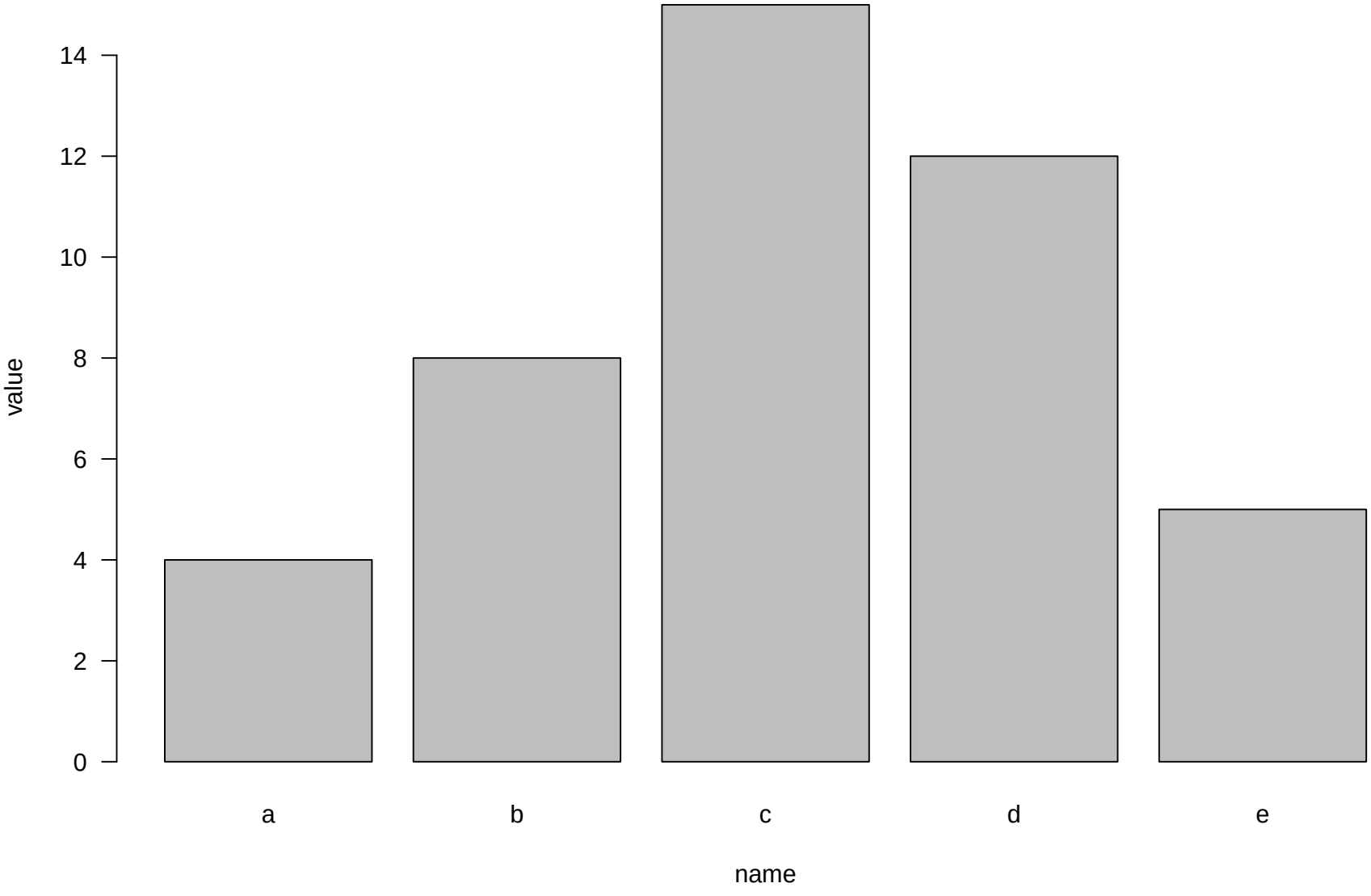
como (tentar) fazer melhor



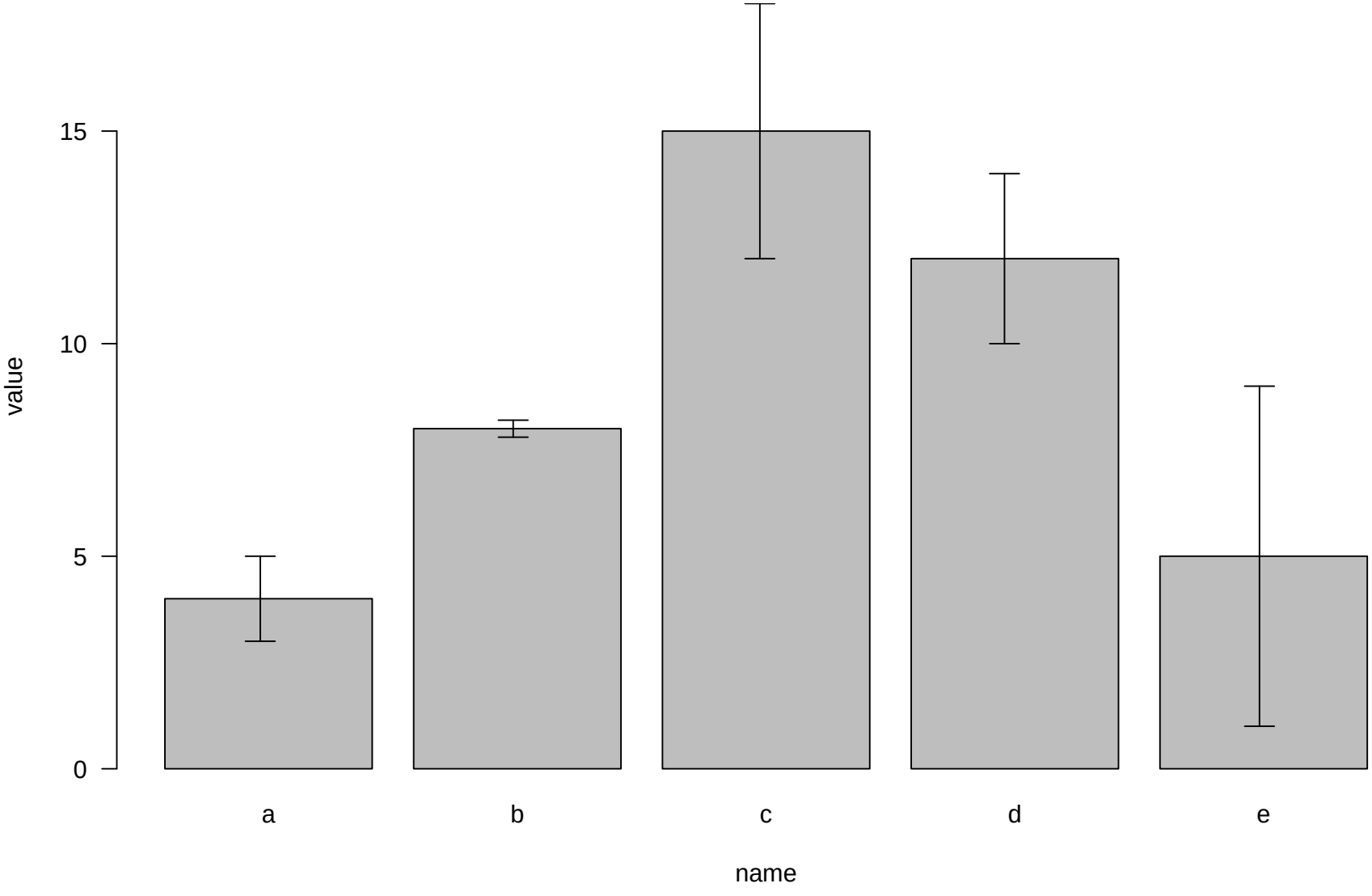
como fazer melhor



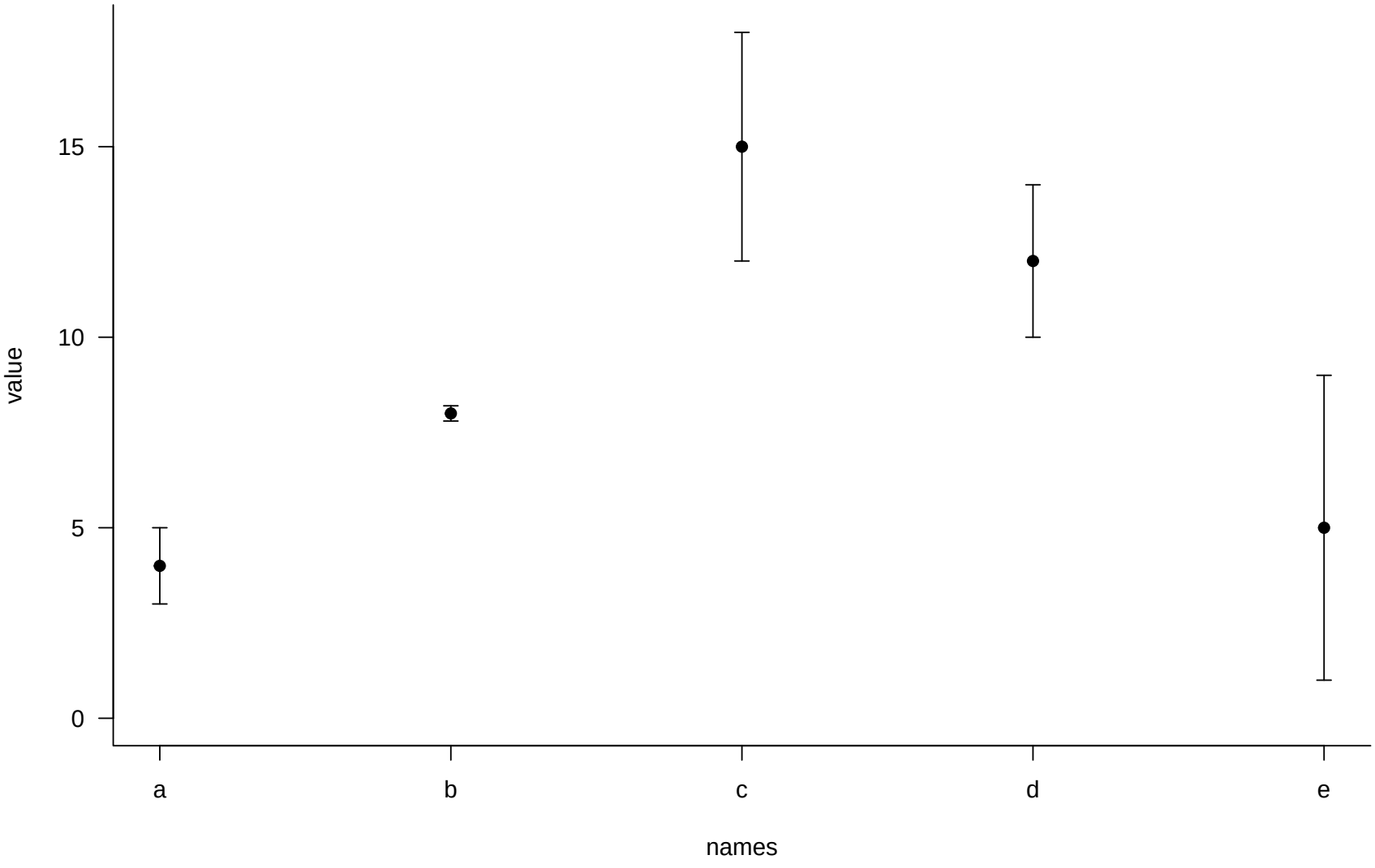
mas sempre pode gráfico de barras?



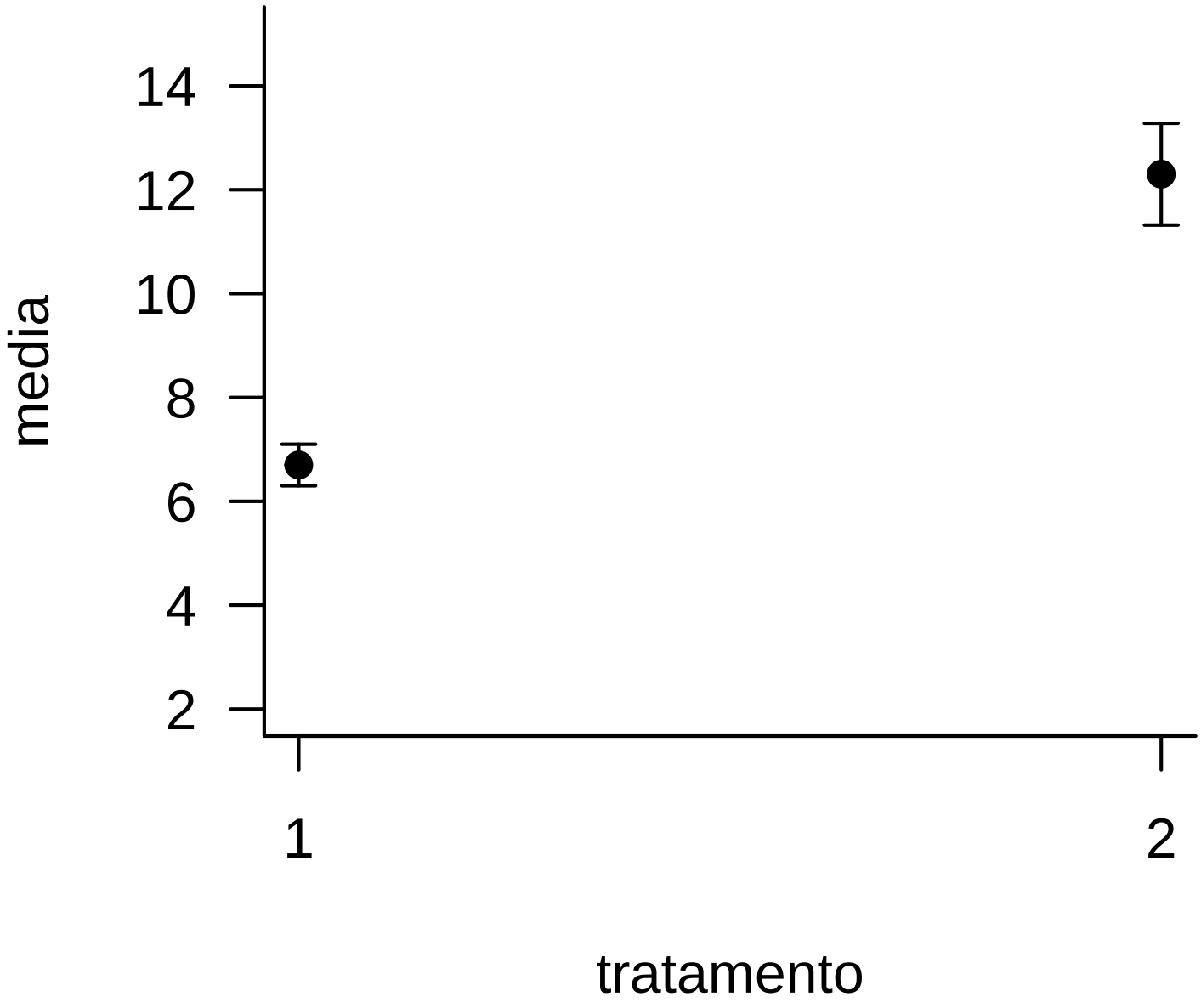
um gráfico de barras com medida de erro é melhor



melhor ainda é economizar tinta



mas também a tinta tem que valer a pena



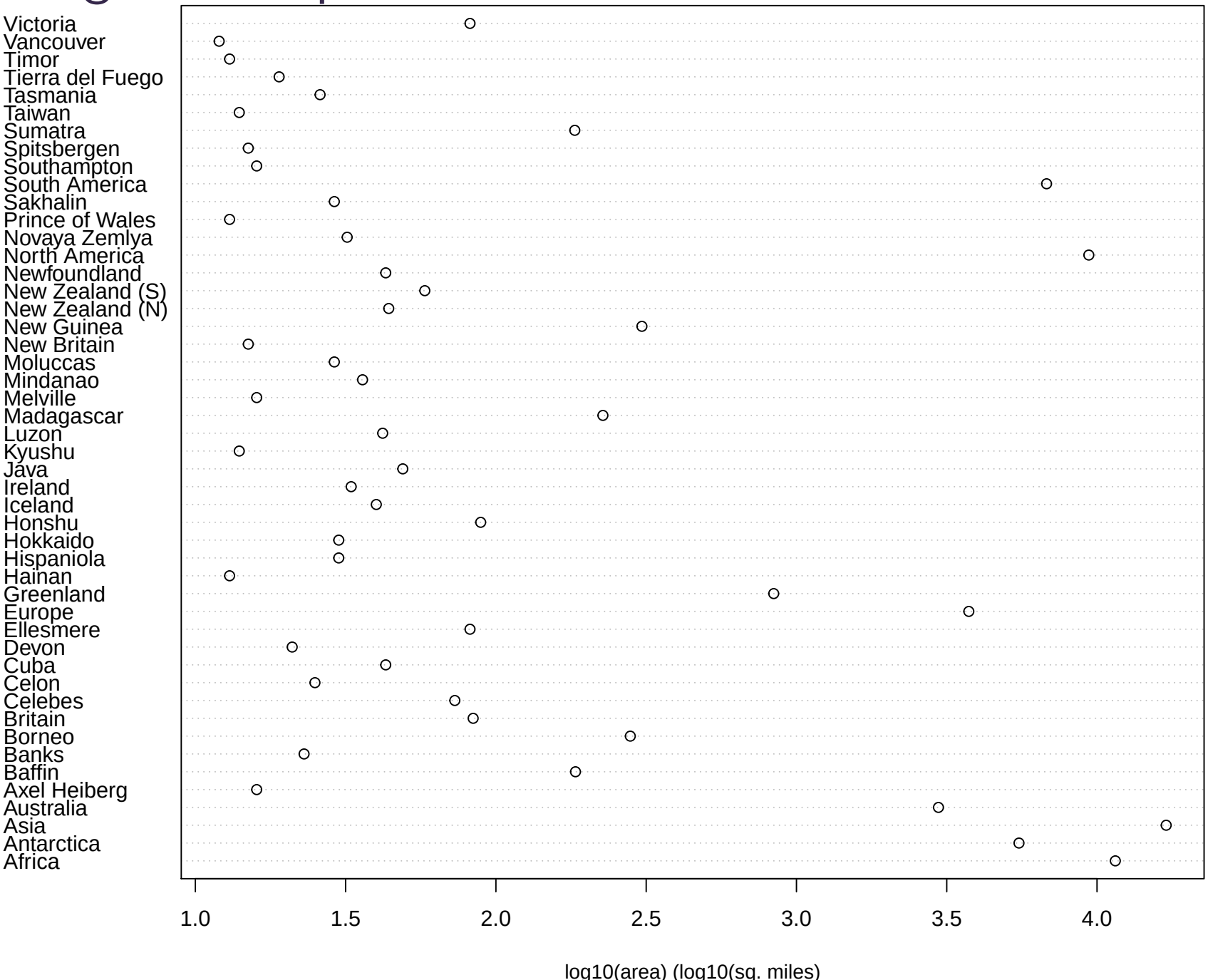
uma tabela?

Tratamento	Efeito
1	6.7 ± 0.4
2	12.3 ± 0.98

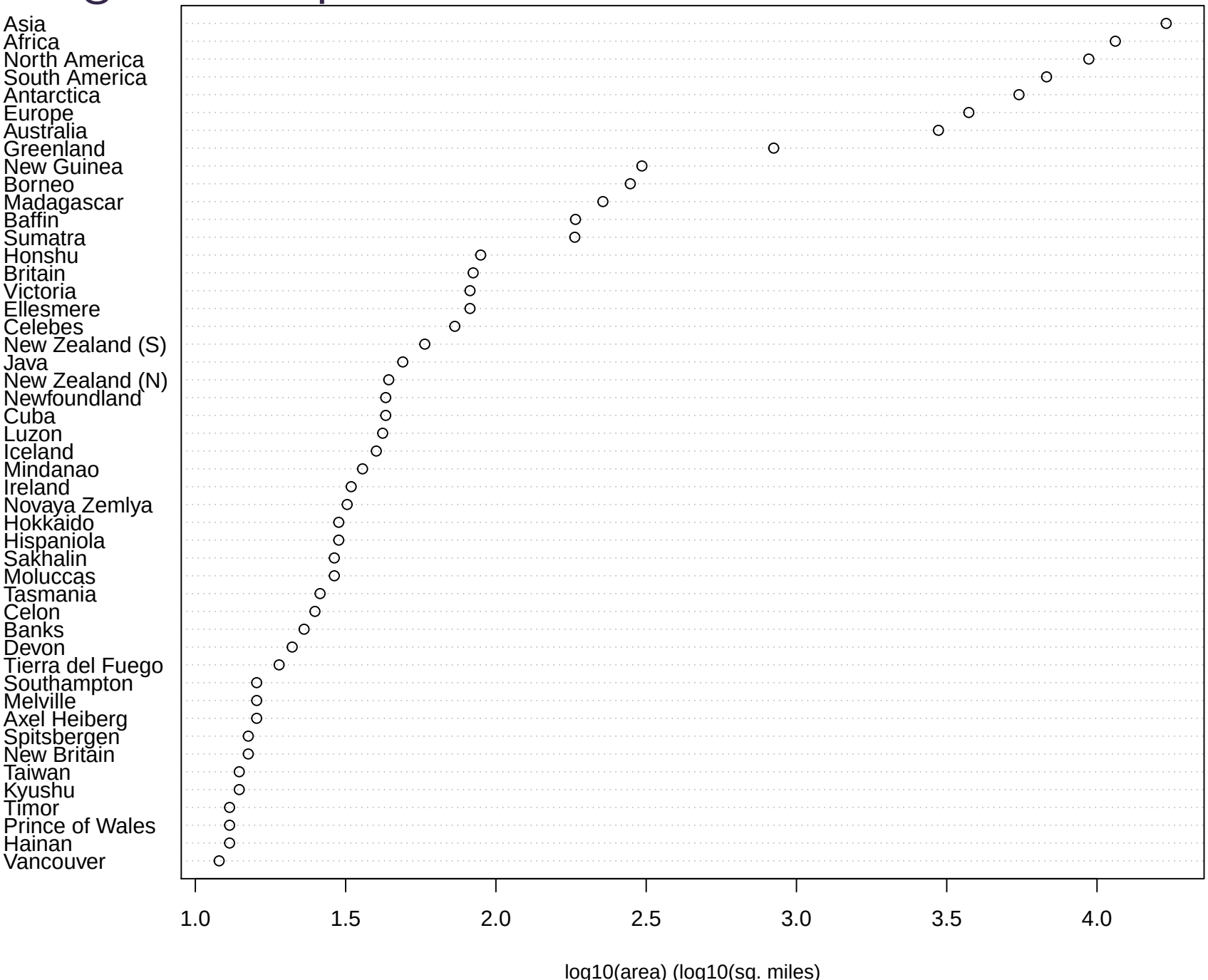
no corpo do texto

O efeito do tratamento 2 (12.3 ± 0.98) foi maior do que o tratamento 1 (6.7 ± 0.4).

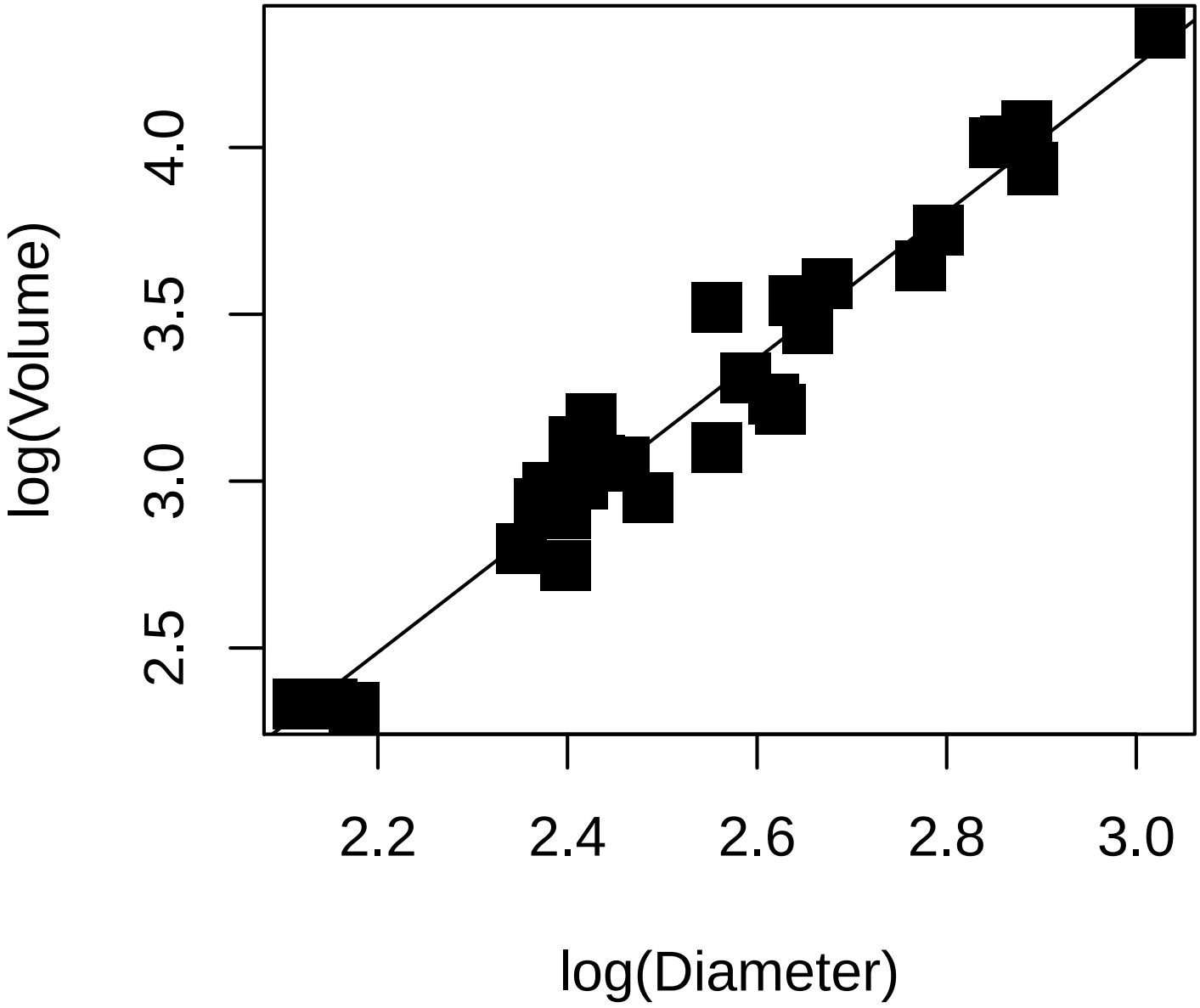
um diagrama de pontos



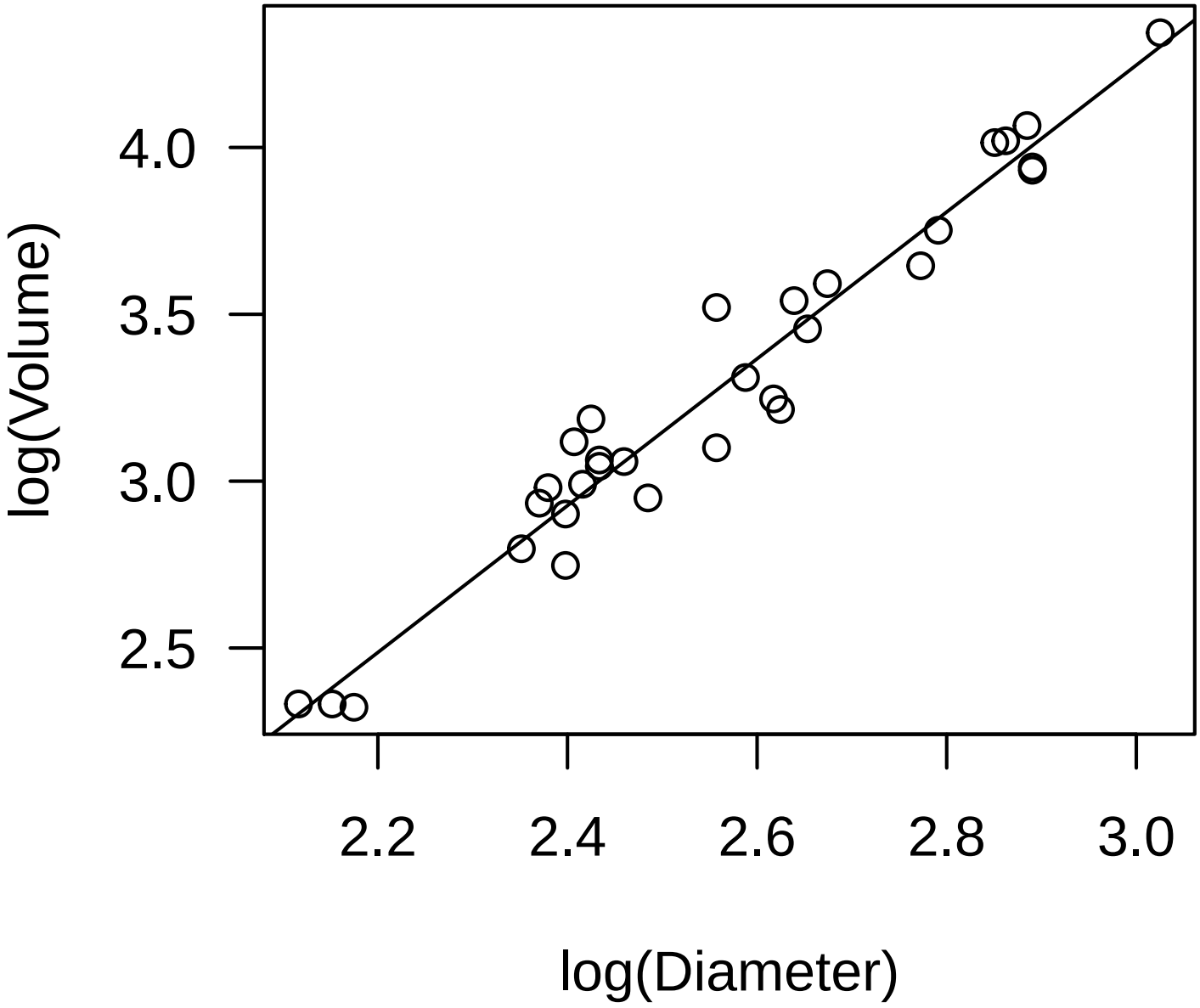
um diagrama de pontos melhor



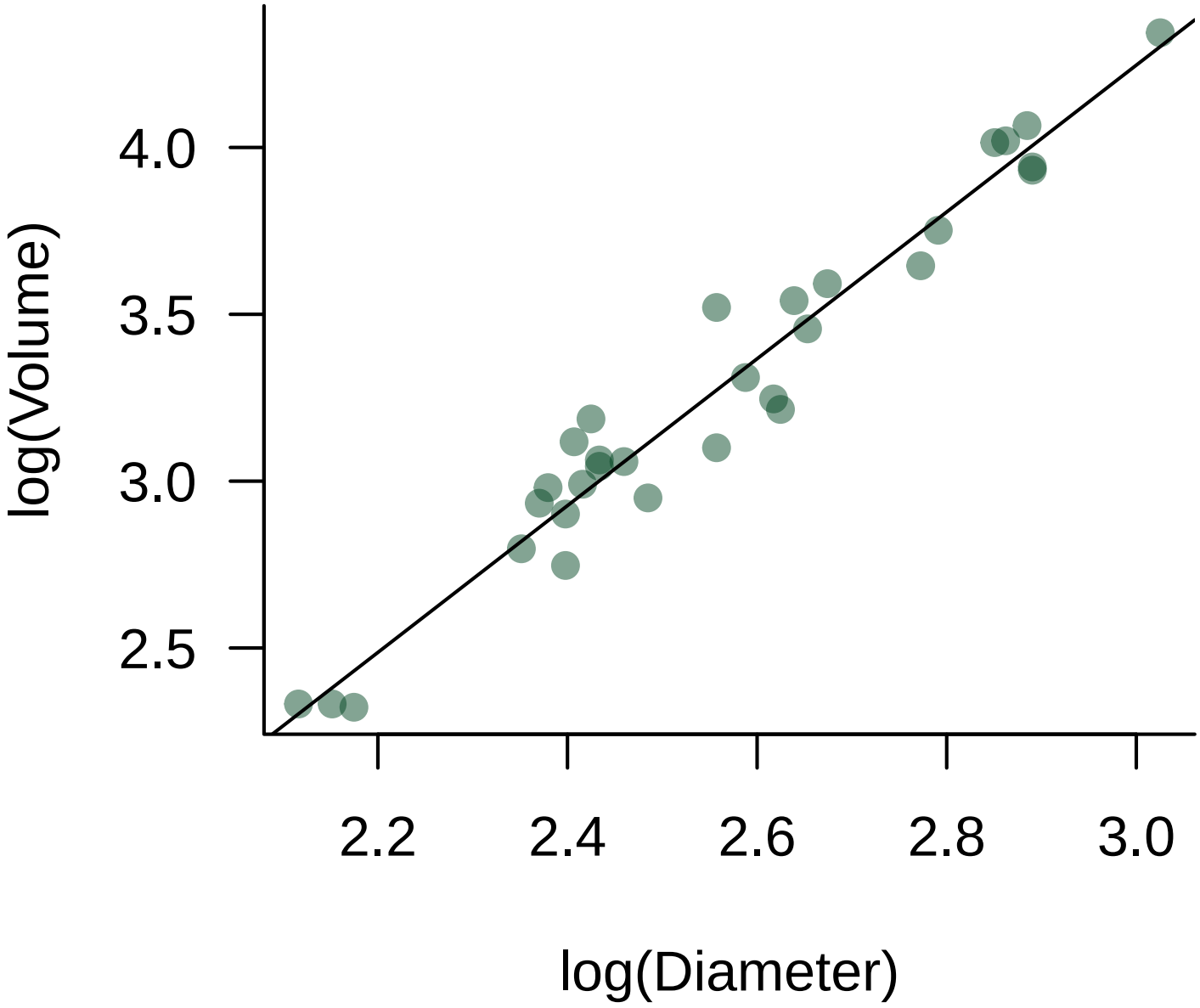
como [NÃO] fazer



como fazer melhor



como fazer melhor de outro jeito



boas práticas na construção de gráficos

boas práticas na construção de gráficos

1. só apresentar um gráfico quando necessário

boas práticas na construção de gráficos

1. só apresentar um gráfico quando necessário
2. não enganar o(a) leitor(a)

boas práticas na construção de gráficos

1. só apresentar um gráfico quando necessário
2. não enganar o(a) leitor(a)
3. moderar na quantidade de cores e de tinta

boas práticas na construção de gráficos

1. só apresentar um gráfico quando necessário
2. não enganar o(a) leitor(a)
3. moderar na quantidade de cores e de tinta
4. sempre que possível apresentar medida de erro

boas práticas na construção de gráficos

1. só apresentar um gráfico quando necessário
2. não enganar o(a) leitor(a)
3. moderar na quantidade de cores e de tinta
4. sempre que possível apresentar medida de erro
5. ordem dos elementos importa

boas práticas na construção de gráficos

1. só apresentar um gráfico quando necessário
2. não enganar o(a) leitor(a)
3. moderar na quantidade de cores e de tinta
4. sempre que possível apresentar medida de erro
5. ordem dos elementos importa
6. ‘las=1‘ importa muito mais

2. editando gráficos no R

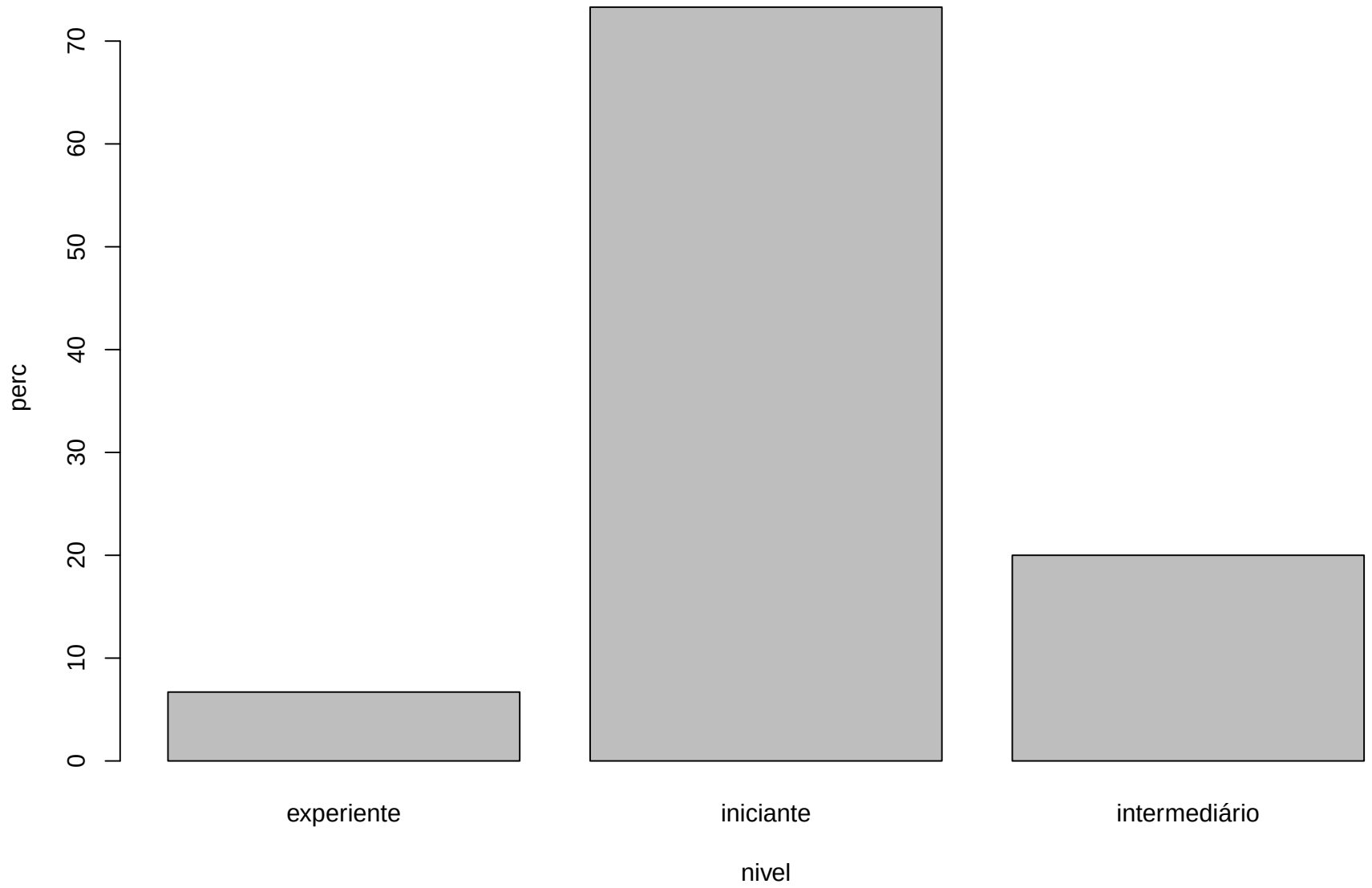
gráfico de barras

```r
#perfil dos alunxs da disciplina
nivel <- c("iniciante", "intermediário", "experiente")
perc <- c(73.3, 20, 6.7)
alunxs <- data.frame(nivel, perc)
alunxs
```

```
##           nivel perc
## 1     iniciante 73.3
## 2 intermediário 20.0
## 3    experiente  6.7
```

gráfico de barras padrão

```
barplot(perc ~ nivel, data=alunxs)
```



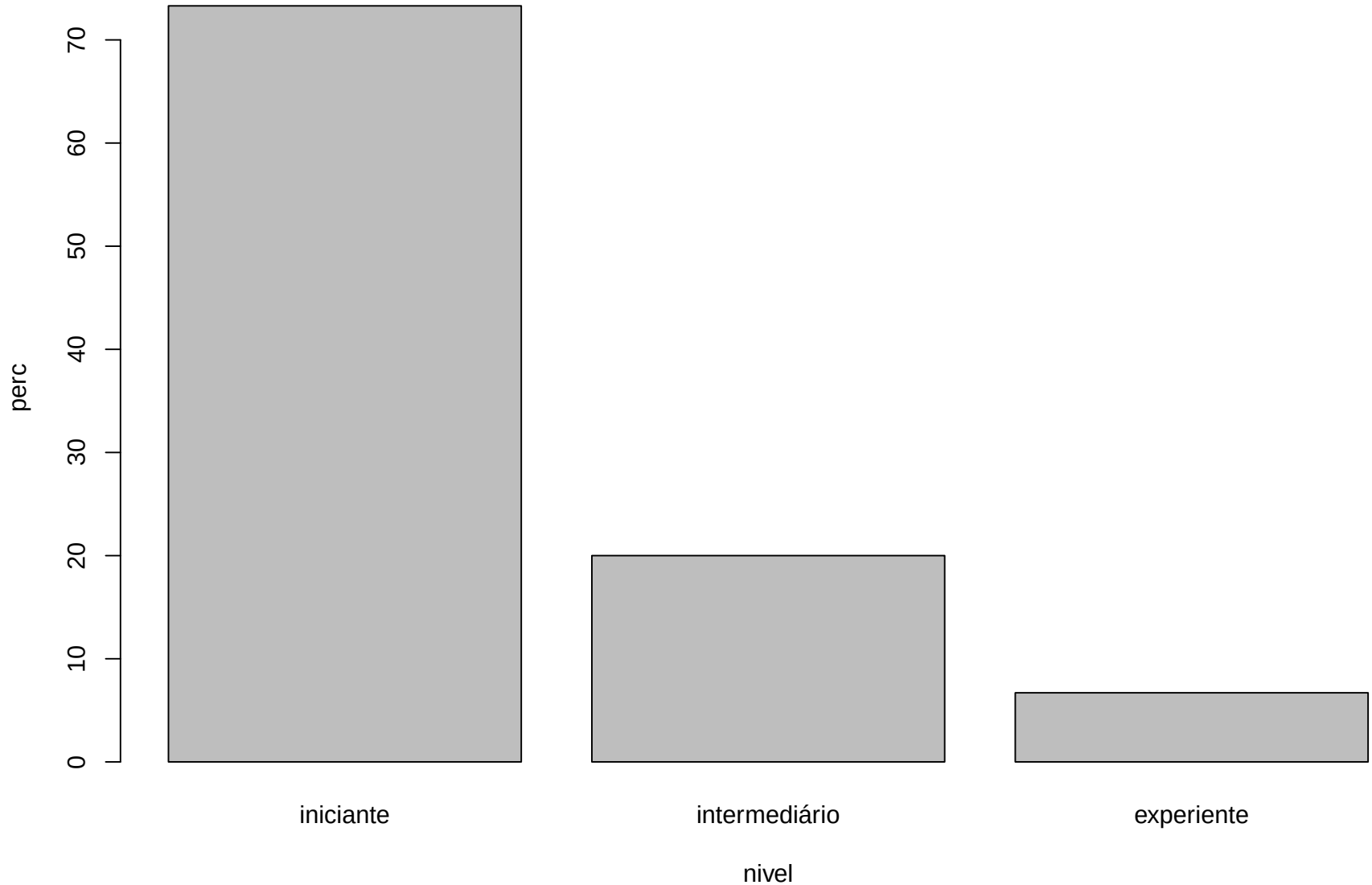
como mudar a ordem das barras?

```
alunxs$nivel <- factor(alunxs$nivel,
                       levels= c("iniciante",
                                 "intermediário",
                                 "experiente"))
alunxs$nivel
```

```
## [1] iniciante     intermediário experiente
## Levels: iniciante intermediário experiente
```

gráfico de barras ordenado

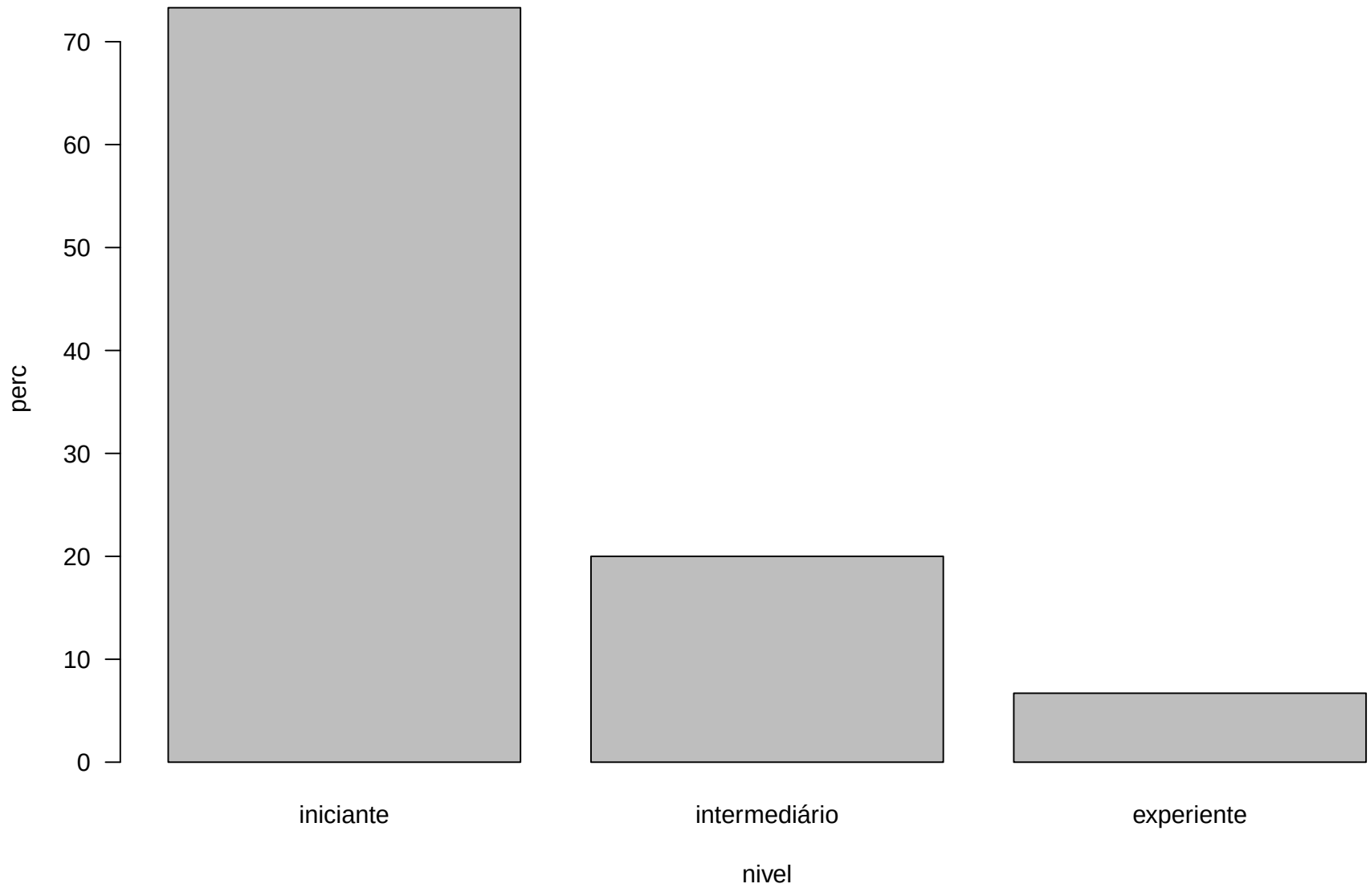
```r
# agora vamos refazer o gráfico
barplot(perc ~ nivel, data=alunxs)
```



deixando o rótulo do eixo y na vertical

```
# las=1
barplot(perc ~ nivel, data=alunxs,
        las=1)
```

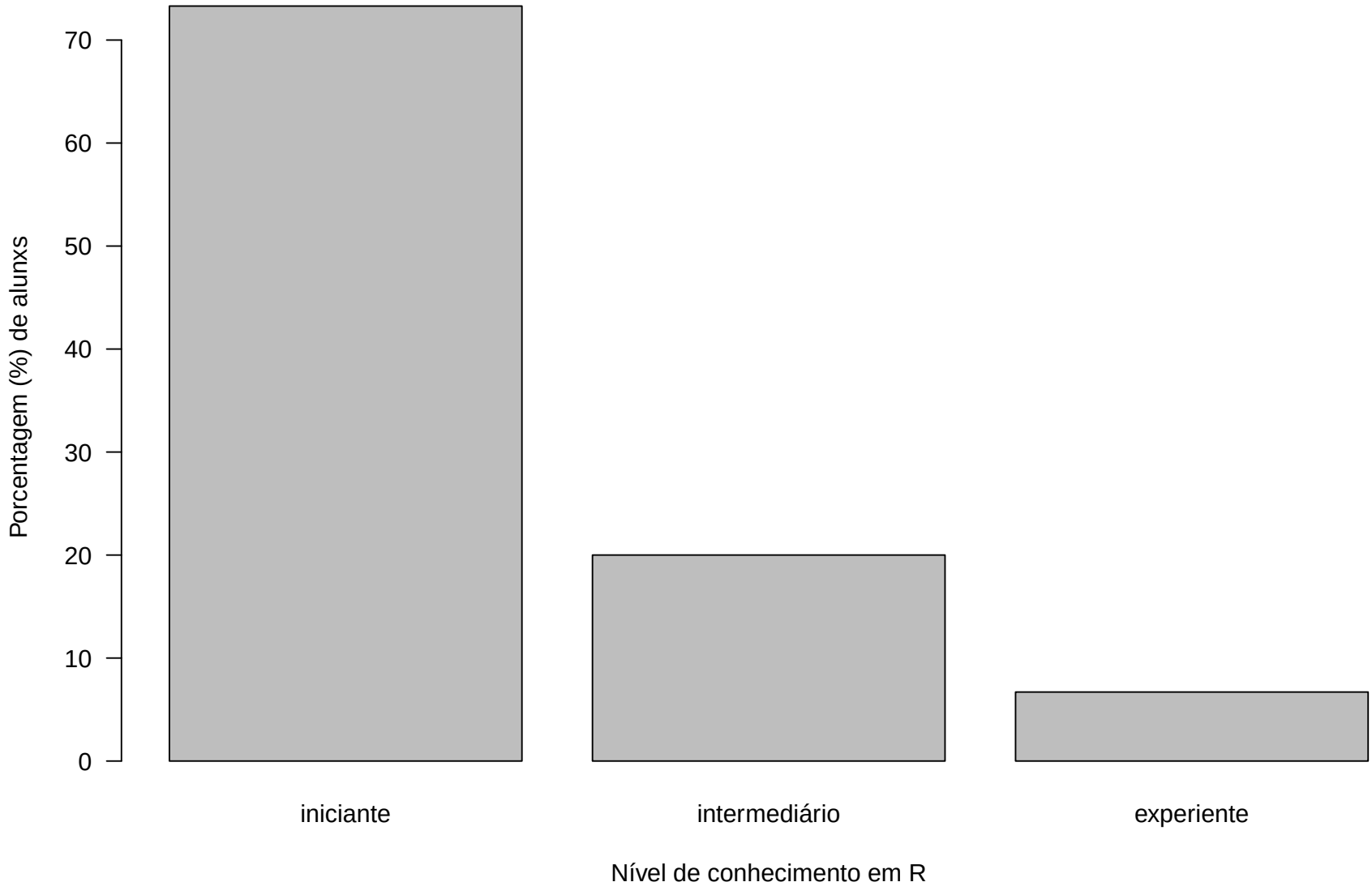
deixando o rótulo do eixo y na vertical



mudando o rótulo dos eixos

```r
# usando xlab e ylab
barplot(perc ~ nivel, data=alunxs,
        las=1,
        xlab="Nível de conhecimento em R",
        ylab="Porcentagem (%) de alunxs")
```

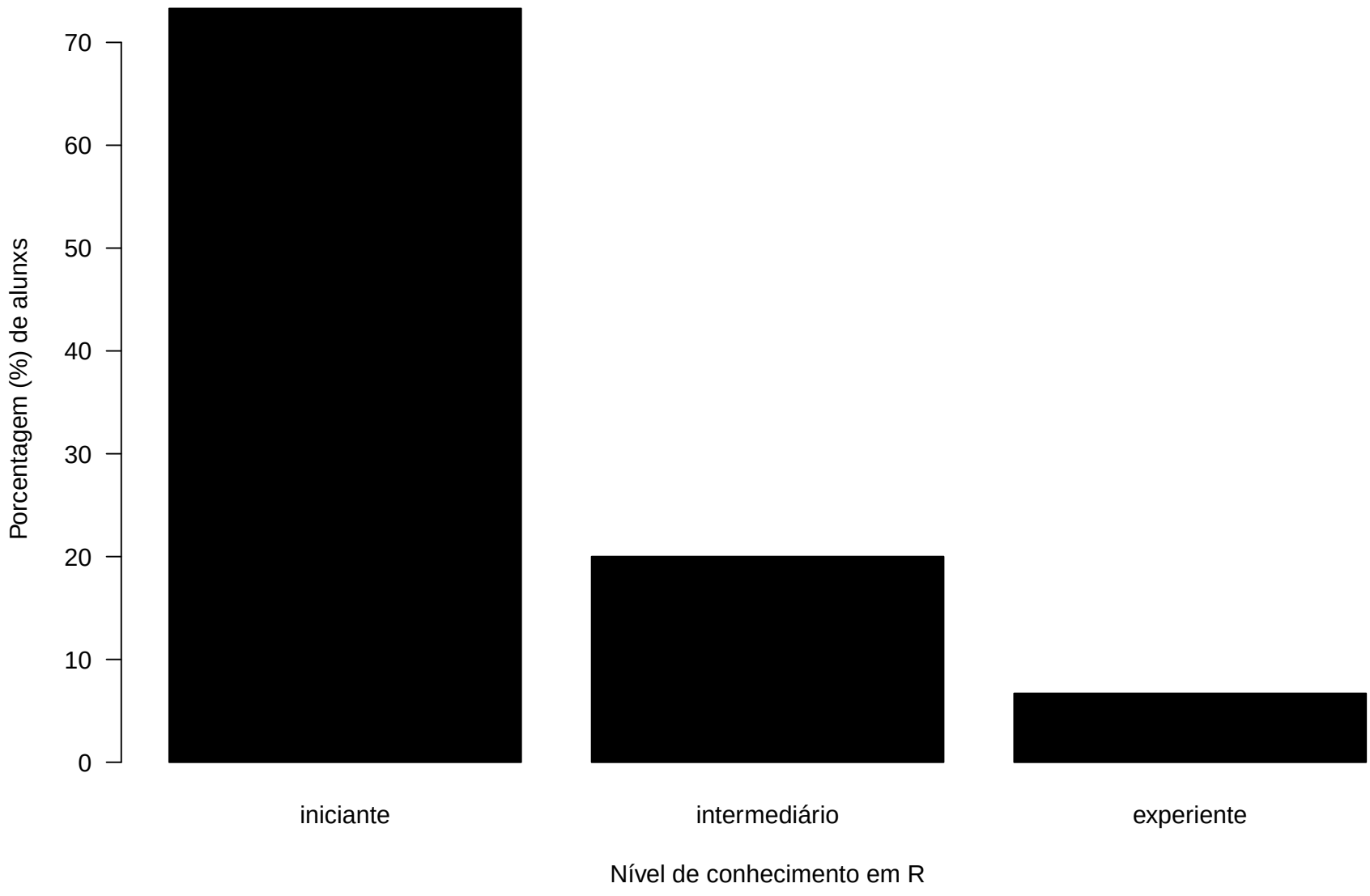
mudando o rótulo dos eixos



incluindo cor

```r
# argumento col
barplot(perc ~ nivel, data=alunxs,
        las=1,
        xlab="Nível de conhecimento em R",
        ylab="Porcentagem (%) de alunxs",
        col=1)
```

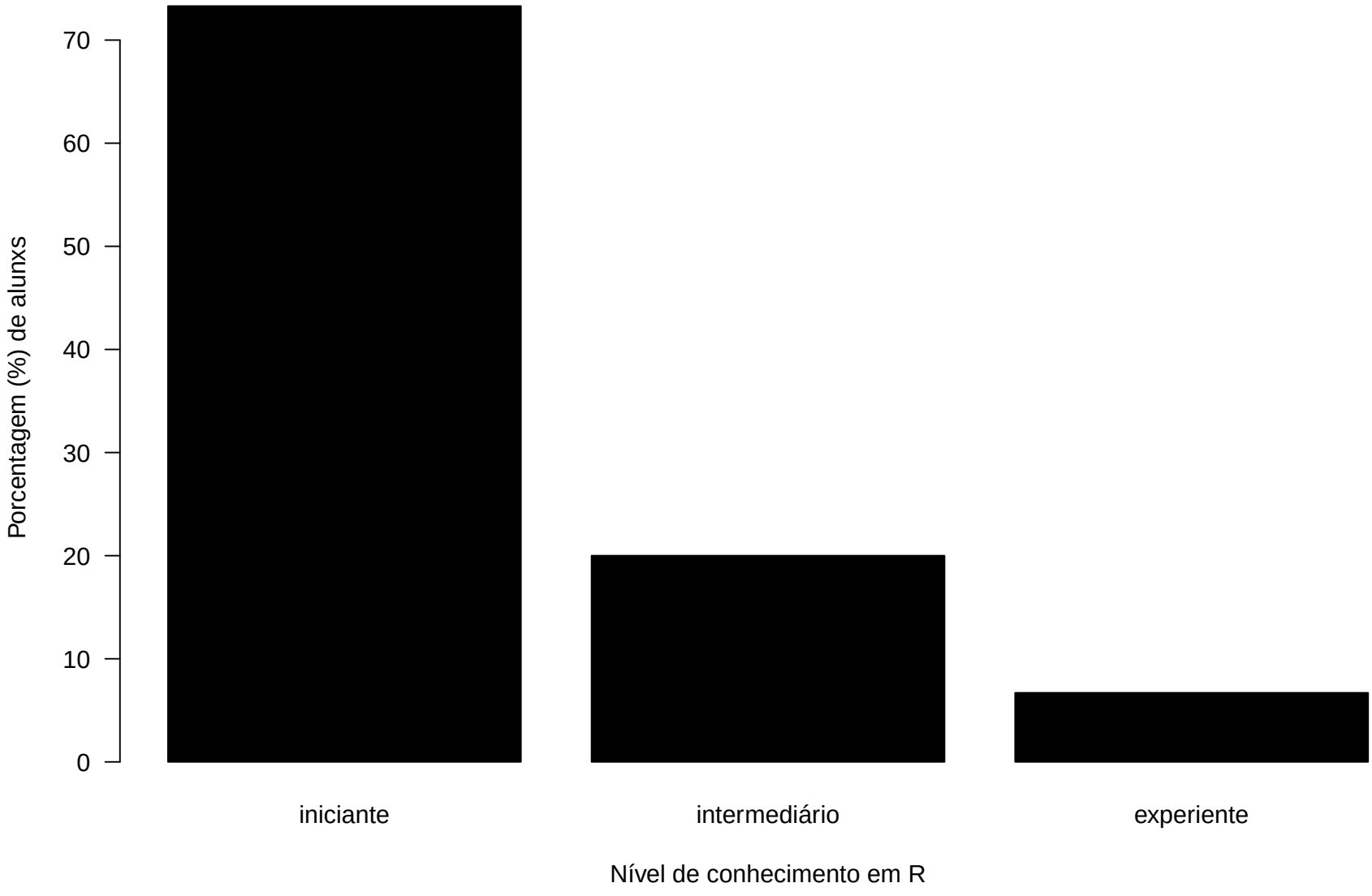
incluindo cor



incluindo cor pelo nome

```r
# argumento col
barplot(perc ~ nivel, data=alunxs,
        las=1,
        xlab="Nível de conhecimento em R",
        ylab="Porcentagem (%) de alunxs",
        col='black')
```

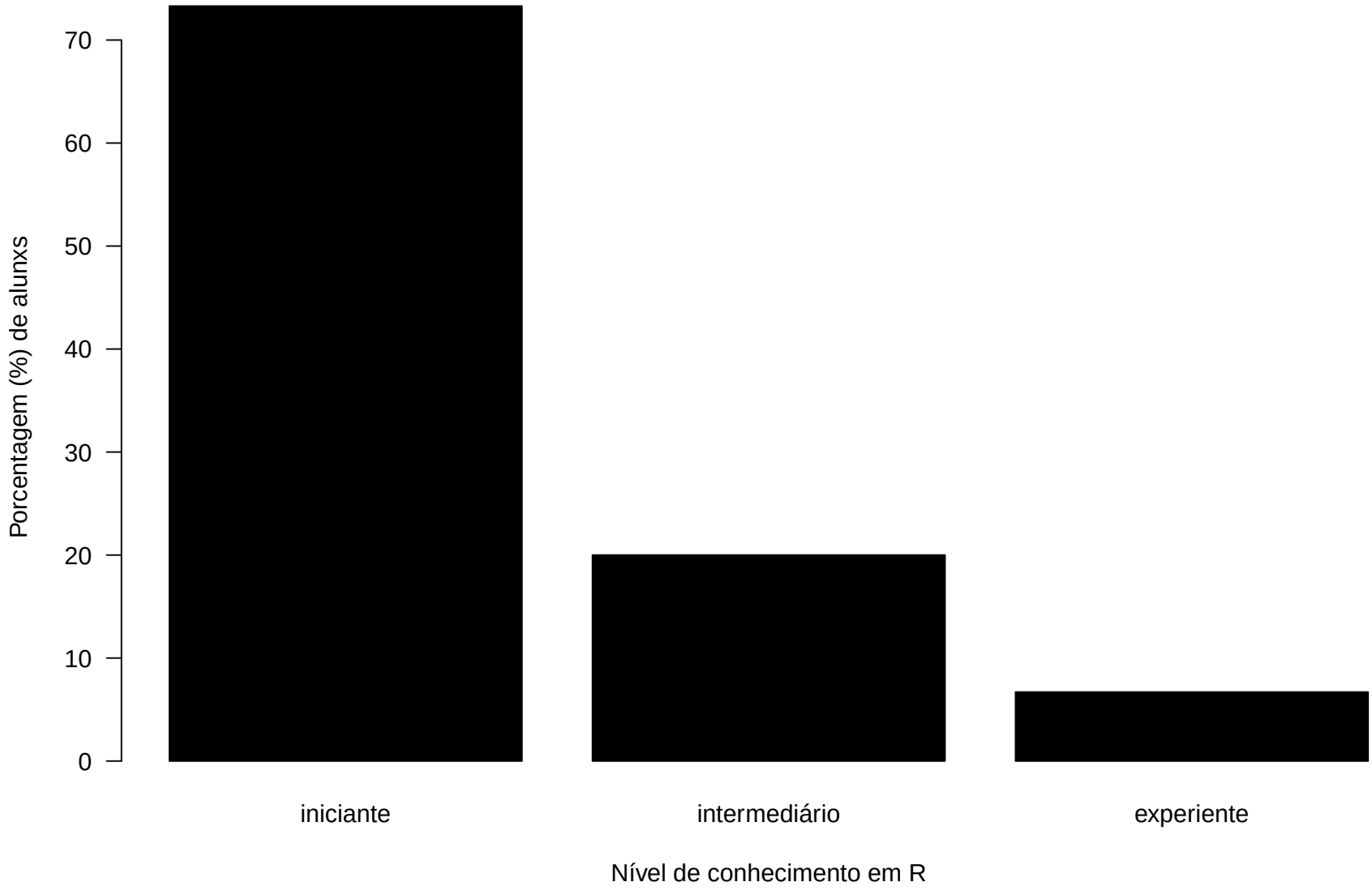
incluindo cor pelo nome



incluindo cor pelo código hexadecimal

```r
# argumento col
barplot(perc ~ nivel, data=alunxs,
        las=1,
        xlab="Nível de conhecimento em R",
        ylab="Porcentagem (%) de alunxs",
        col="#000000")
```

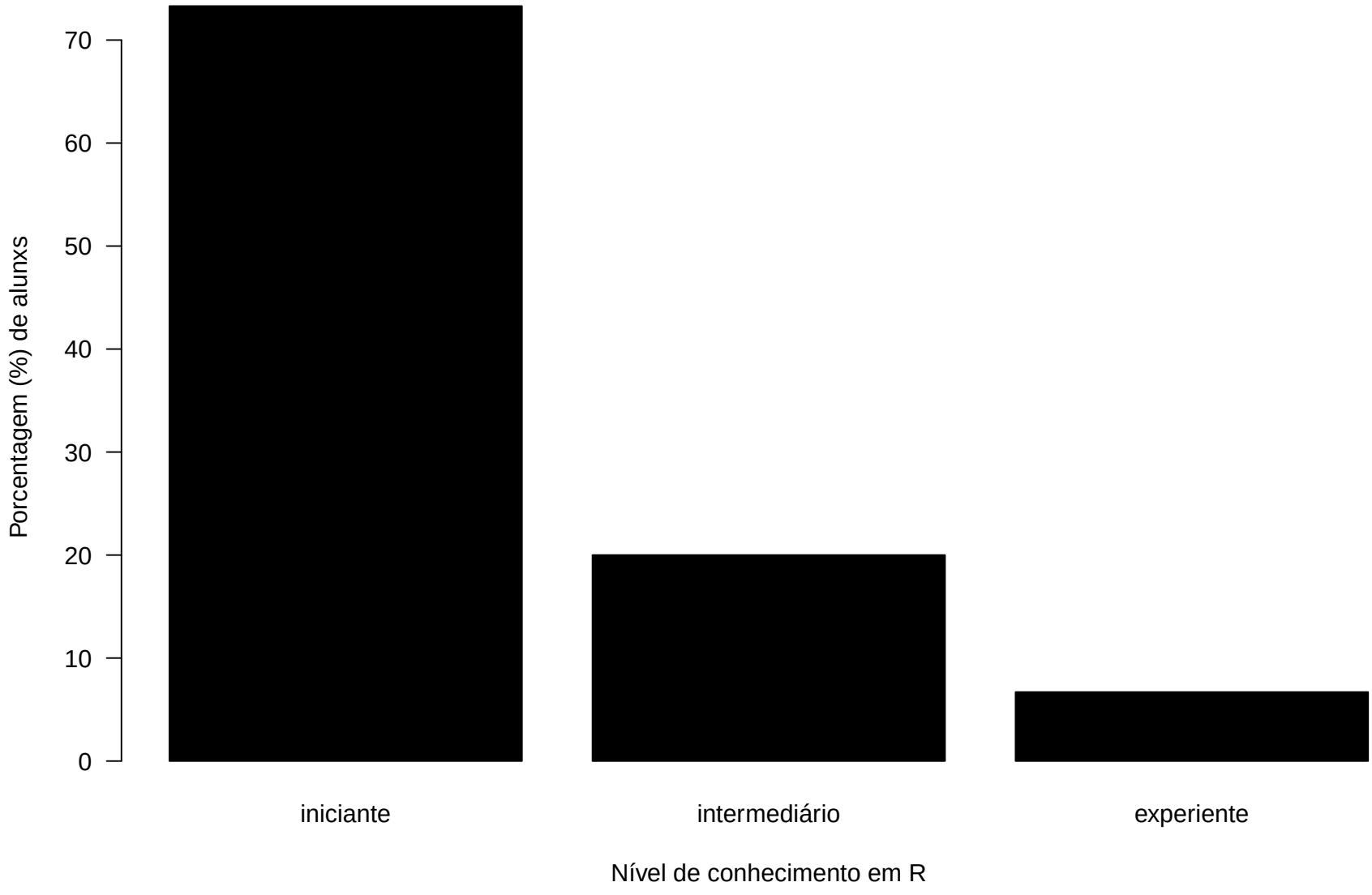
incluindo cor pelo código hexadecimal



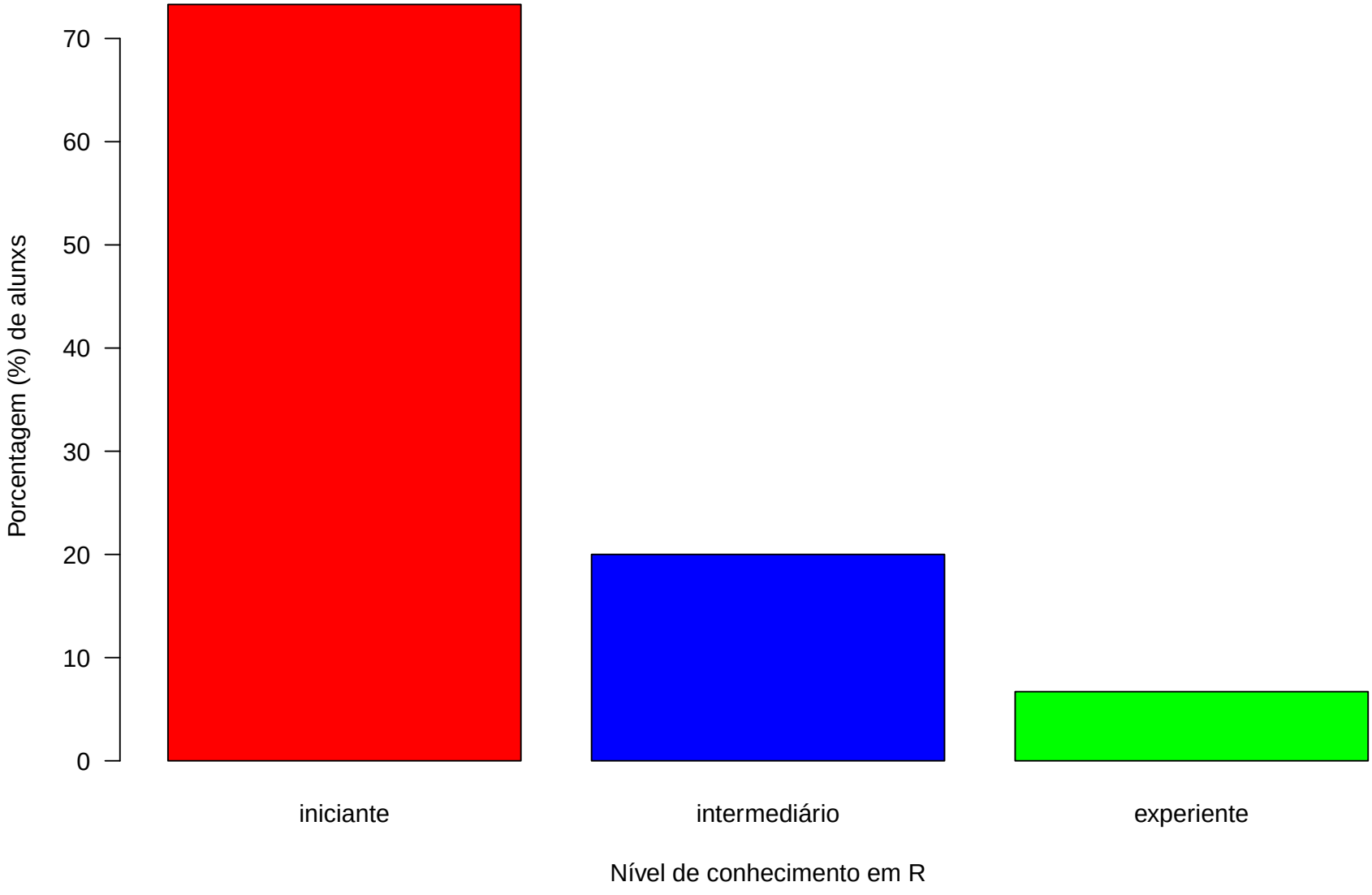
incluindo cor pelo código RGB

```
# funcao rgb
barplot(perc ~ nivel, data=alunxs,
        las=1,
        xlab="Nível de conhecimento em R",
        ylab="Porcentagem (%) de alunxs",
        col=rgb(0, 0, 0, maxColorValue=255))
```

incluindo cor pelo código RGB



precisa de cor?



precisa de cor?

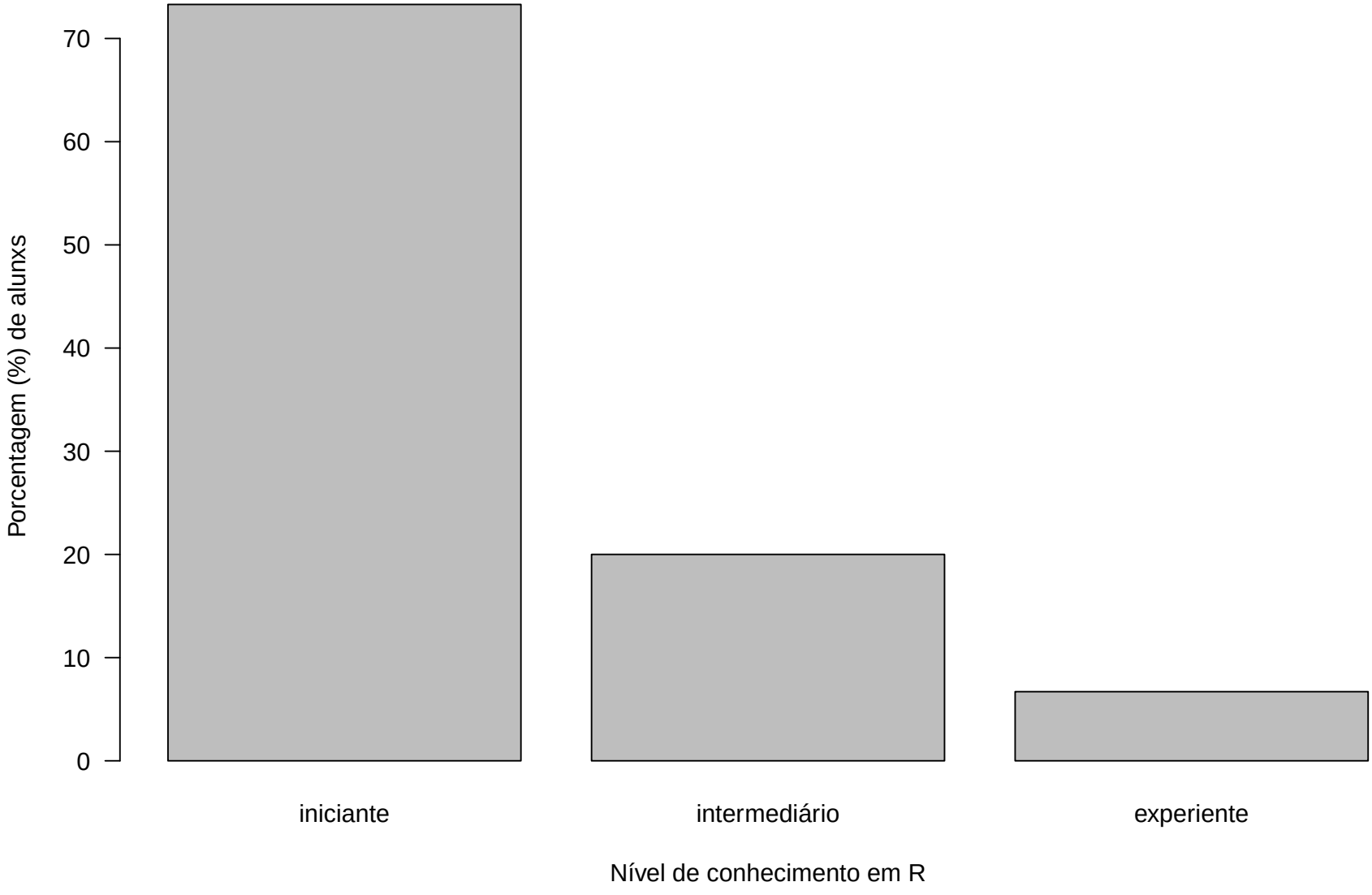


gráfico de barras com subcategorias

```
d.Titanic <- as.data.frame(Titanic)
head(d.Titanic)
```

```
##    Class    Sex   Age Survived Freq
## 1    1st   Male Child       No    0
## 2    2nd   Male Child       No    0
## 3    3rd   Male Child       No   35
## 4   Crew   Male Child       No    0
## 5    1st Female Child       No    0
## 6    2nd Female Child       No    0
```

```
barplot(Freq ~ Class + Survived, data = d.Titanic,
        subset = Age == "Adult" & Sex == "Male",
        ylab = "N of Passengers", legend = TRUE)
```

gráfico de barras com subcategorias

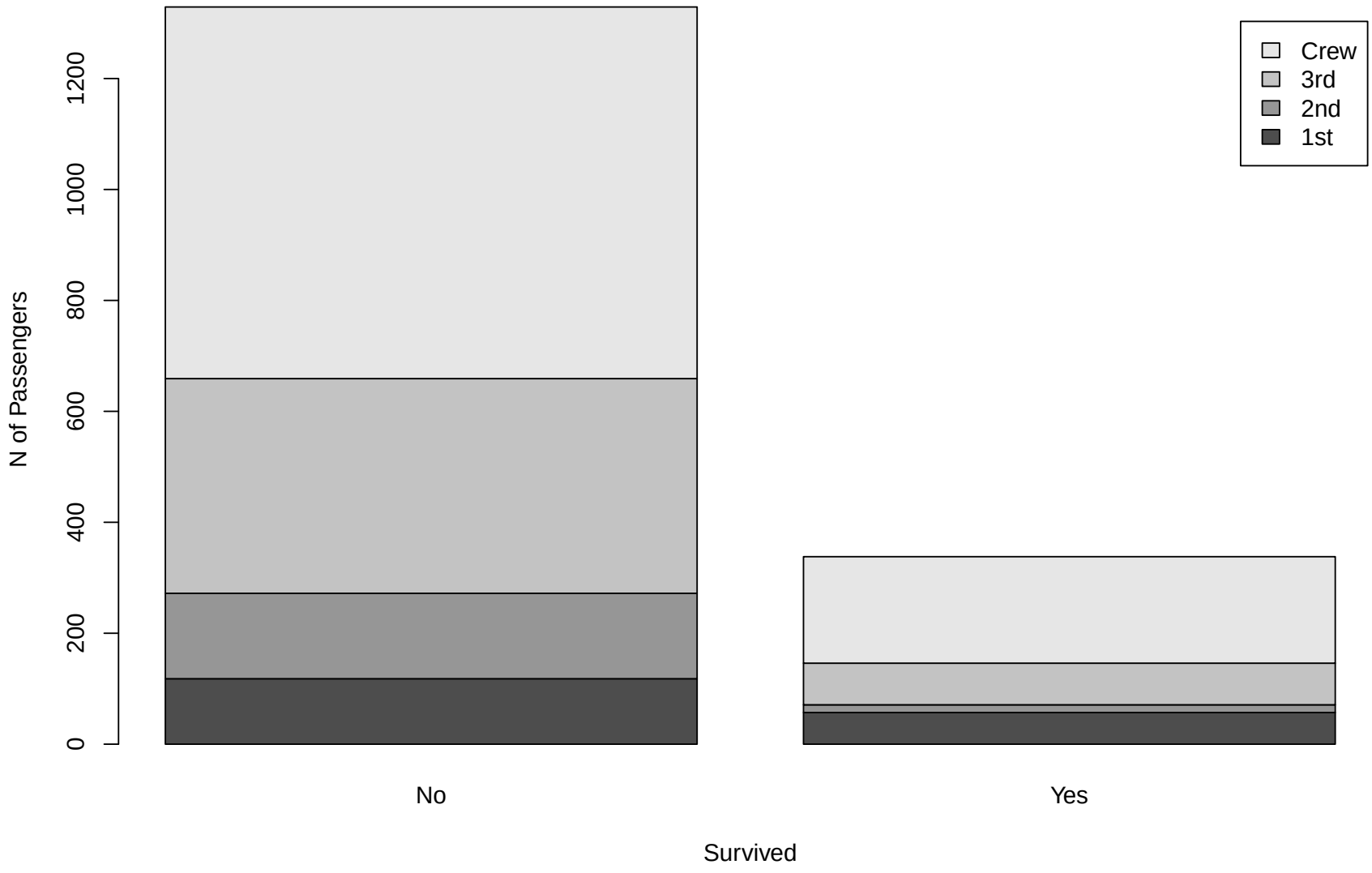


gráfico de barras em “mosaico”

```
# Corresponding table :
xt <- xtabs(Freq ~ Survived + Class + Sex,
            d.Titanic,
            subset = Age=="Adult")
# Alternatively, a mosaic plot :
mosaicplot(xt[,,"Male"], main="", color=TRUE)
```

gráfico de barras em “mosaico”

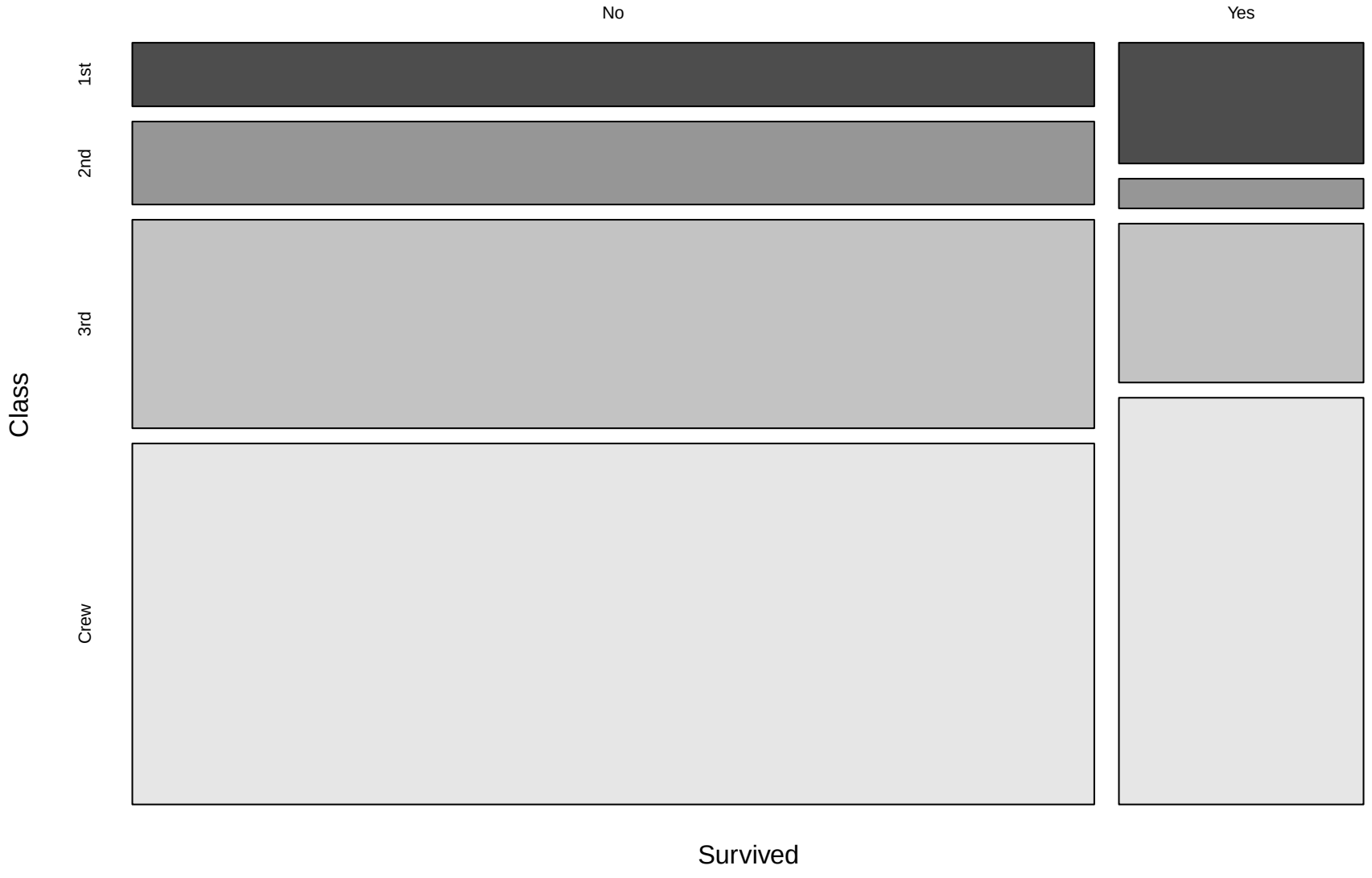


gráfico de dispersão

```
m1 <- lm(log(dist) ~ log(speed), data = cars)
plot(log(dist) ~ log(speed), data = cars,
     las=1, bty='l', pch=19)
abline(m1)
```

gráfico de dispersão

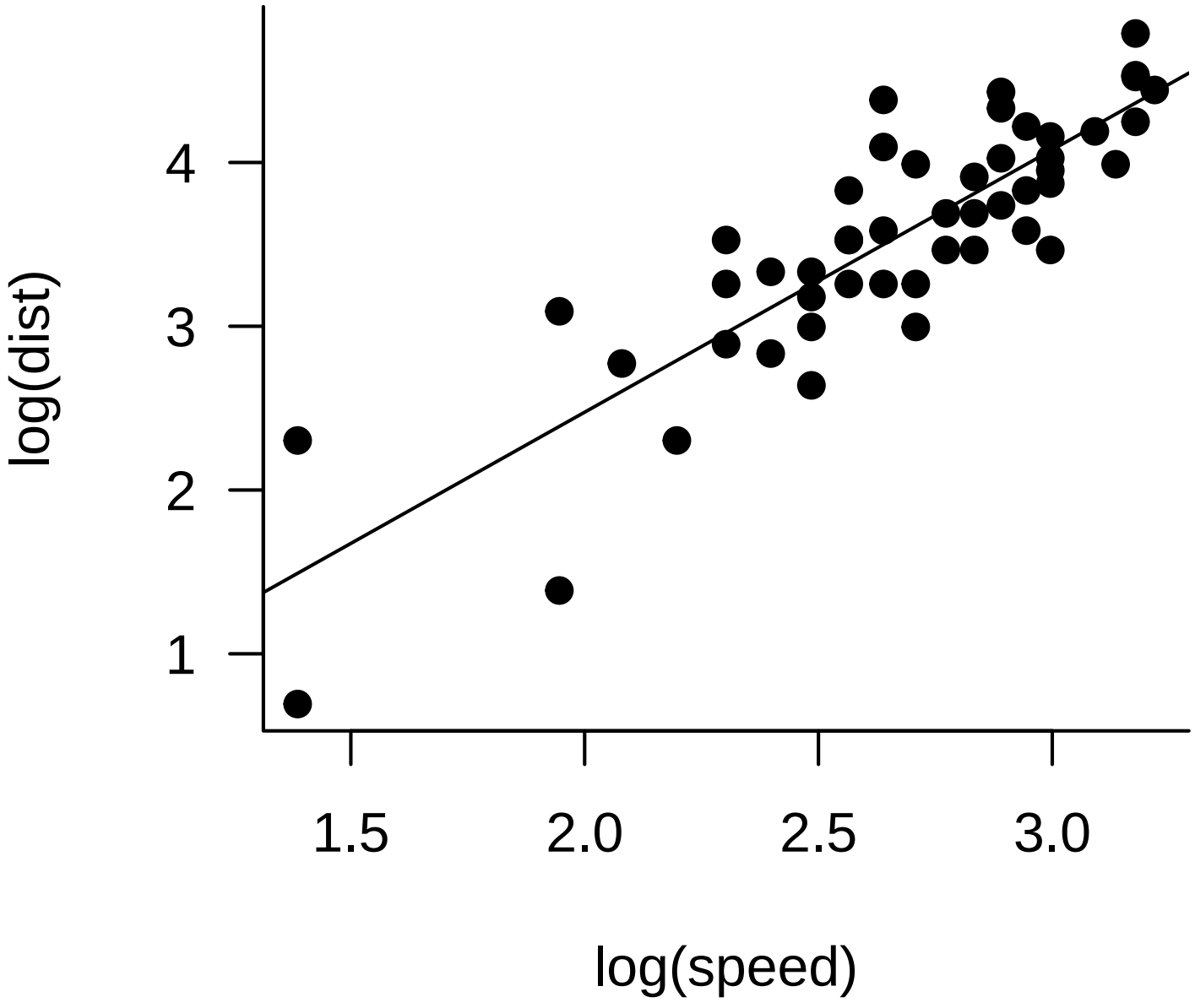
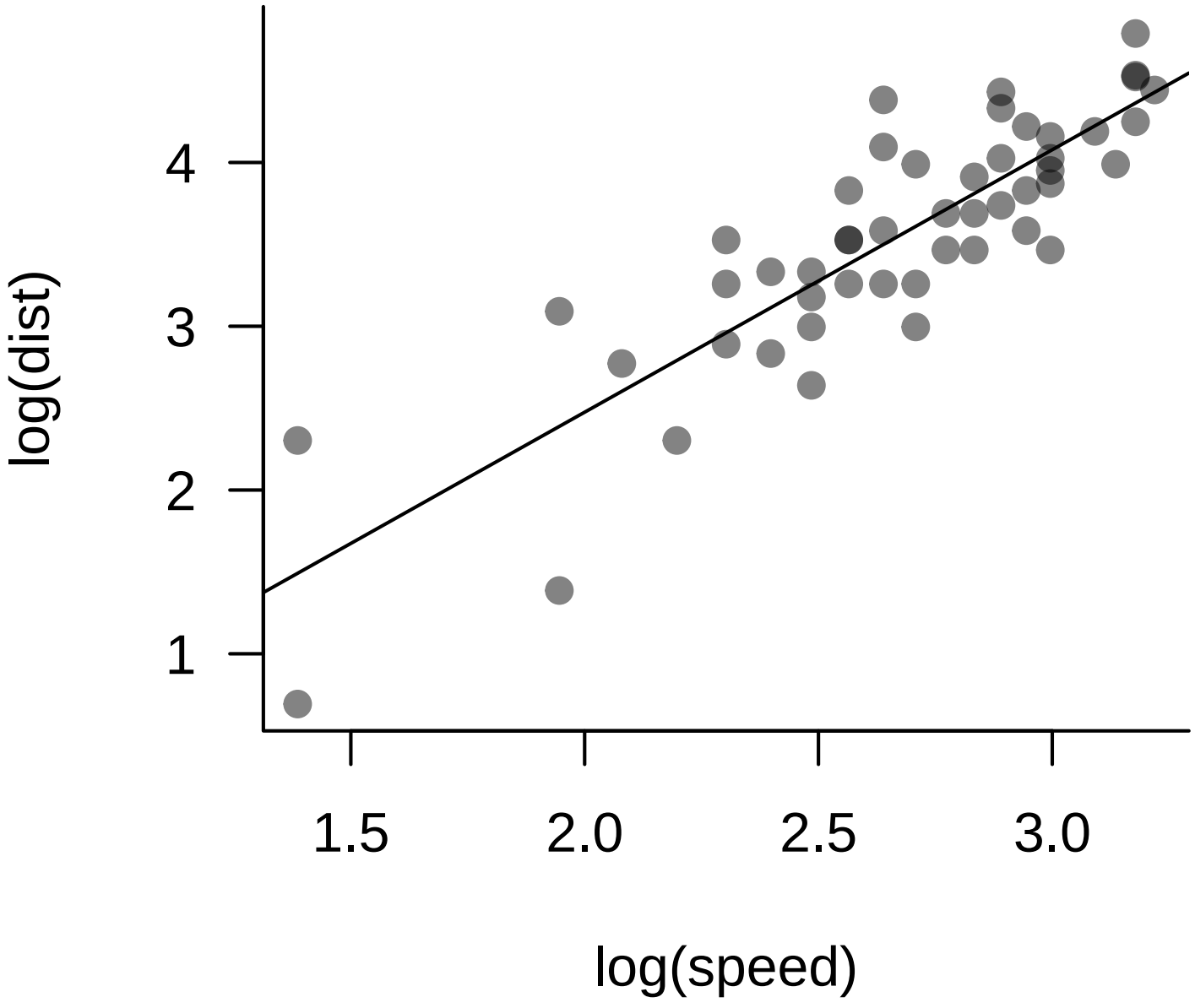


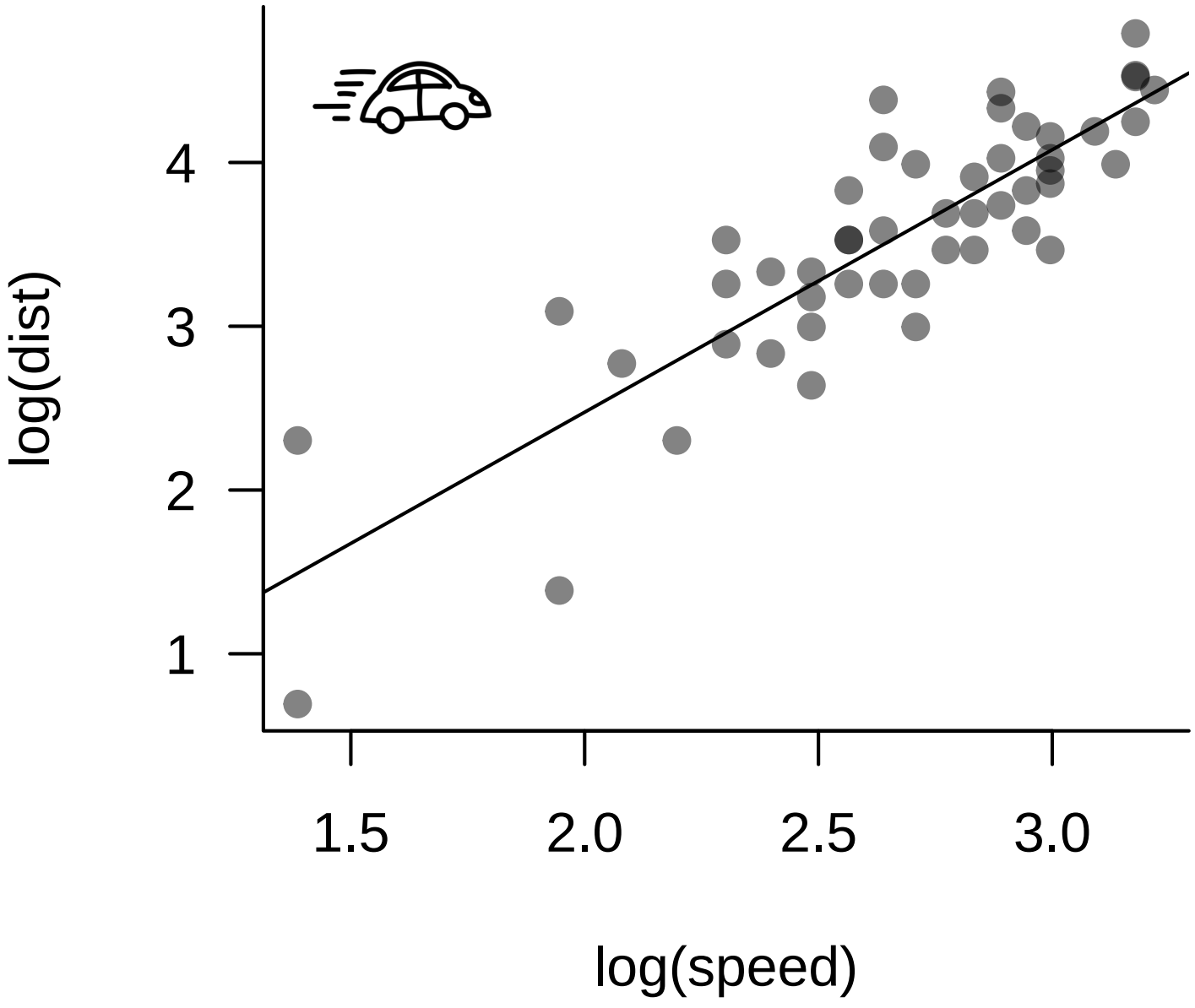
gráfico de dispersão com cor transparente



adicionando uma imagem ao gráfico

```
library(png)
library(grid)
pic <- readPNG("figs/car.png")
plot(log(dist) ~ log(speed), data = cars,
     las=1, bty='l', pch=19,
     col=mycol)
abline(m1)
grid.raster(pic, .25, .83, width=.2)
```

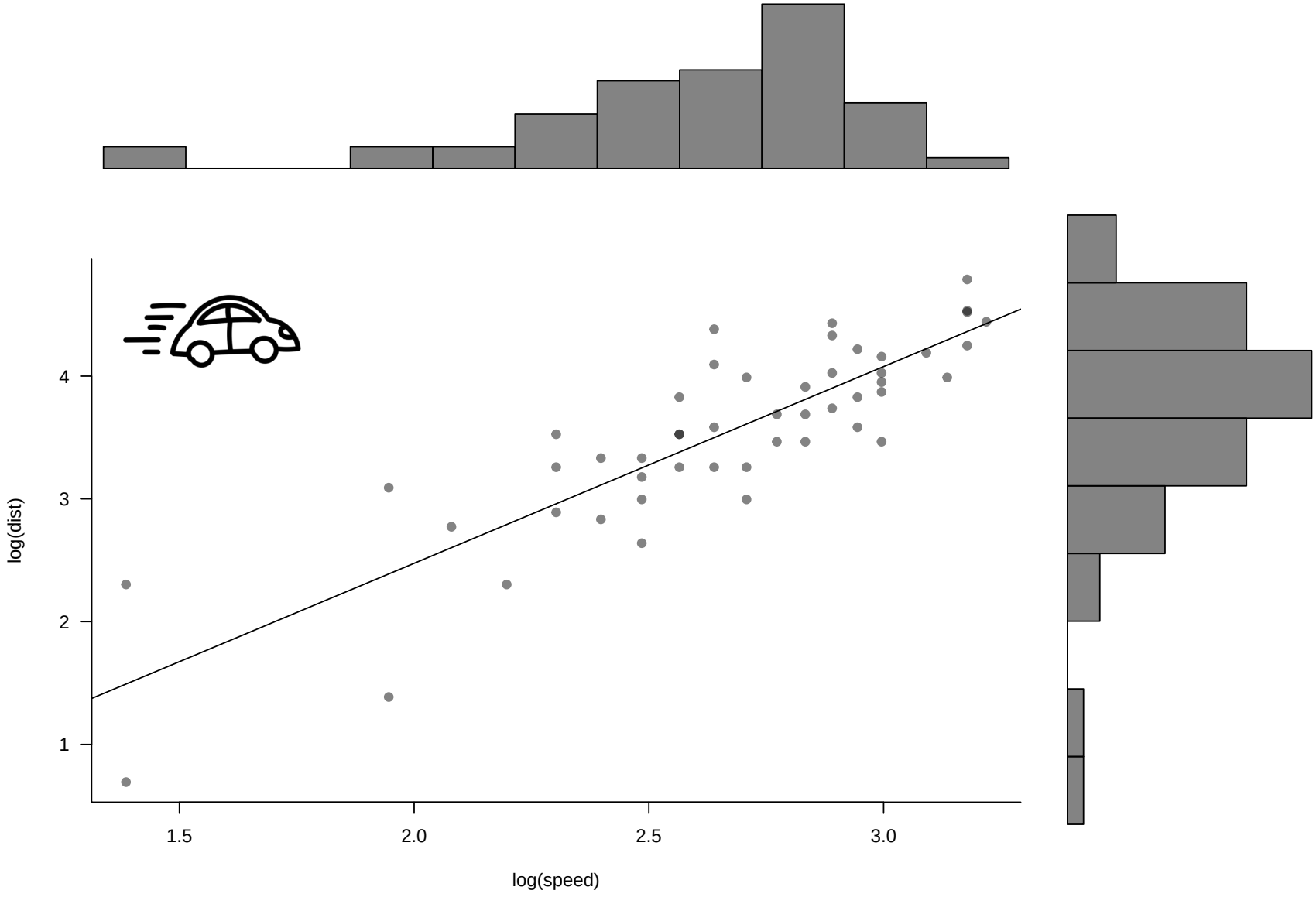
adicionando uma imagem ao gráfico



gráficos com múltiplas janelas

```
x <- log(cars$speed)
y <- log(cars$dist)
zones <- matrix(c(2,0,1,3), ncol=2, byrow=TRUE)
layout(zones, widths=c(4/5, 1/5), heights=c(1/5, 4/5))
xhist <- hist(x, plot=FALSE)
yhist <- hist(y, plot=FALSE)
top <- max(c(xhist$counts, yhist$counts))
plot(x,y, las=1, xlab="log(speed)", ylab="log(dist)",
     col=mycol, bty='l', pch=19)
abline(m1)
grid.raster(pic, .16, .63, width=.15)
par(mar=c(0,3,1,1))
barplot(xhist$counts, axes=FALSE, ylim=c(0, top),
        space=0, col=mycol)
par(mar=c(3,0,1,1))
barplot(yhist$counts, axes=FALSE, xlim=c(0, top),
        space=0, horiz=TRUE, col=mycol)
```

gráficos com múltiplas janelas



mais gráficos com múltiplas janelas

```
sal <- read.csv("data/salarios.csv")
head(sal)
```

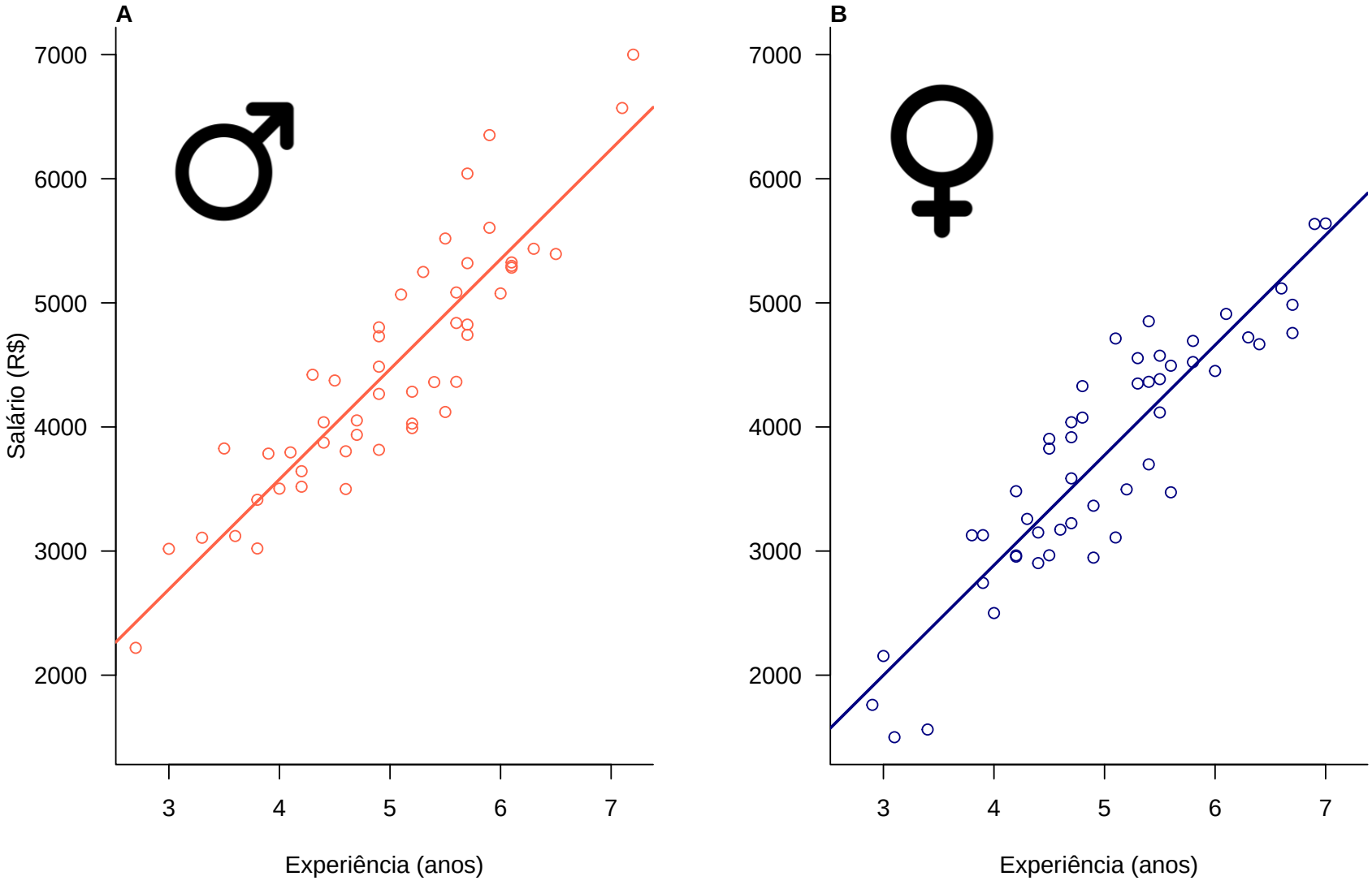
```
##   salario experiencia sexo
## 1 3936.68          4.7    H
## 2 3502.86          4.0    H
## 3 4837.77          5.6    H
## 4 5083.71          5.6    H
## 5 7000.00          7.2    H
## 6 4283.94          5.2    H
```

```
# criando modelos lineares
mh <- lm(salario ~ experiencia, data=sal[sal$sexo=="H",])
mm <- lm(salario ~ experiencia, data=sal[sal$sexo=="M",])
coefh <- coef(mh)
coefm <- coef(mm)
# definindo os limites dos eixos
limy <- c(min(sal$salario),max(sal$salario))
```

criando o gráfico com janelas A e B

```
# define parametros graficos
par(mfrow=c(1,2), las=1, bty="l")
# plot dos valores de salario dos homens
plot(salario ~ experiencia, data=sal[sal$sexo=="H",],
     col="tomato",
     ylim=limy, xlim=limx,
     ylab=laby, xlab=labx)
# linha do previsto pelo modelo a + bx
abline(a=coefh[1], b=coefh[2],
       col='tomato', lwd=2)
mtext("A", 3, adj=0, font=2)
grid.raster(readPNG("figs/male.png"), .16, .75, width=.10)
## plot do salario das mulheres
plot(salario ~ experiencia, data=sal[sal$sexo=="M",],
     col="navy",
     ylim=limy, xlim=limx,
     ylab="", xlab=labx)
mtext("B", 3, adj=0, font=2)
grid.raster(readPNG("figs/female.png"), .66, .75, width=.08)
abline(a=coefm[1], b=coefm[2],
       col='navy', lwd=2)
```

criando o gráfico com janelas A e B



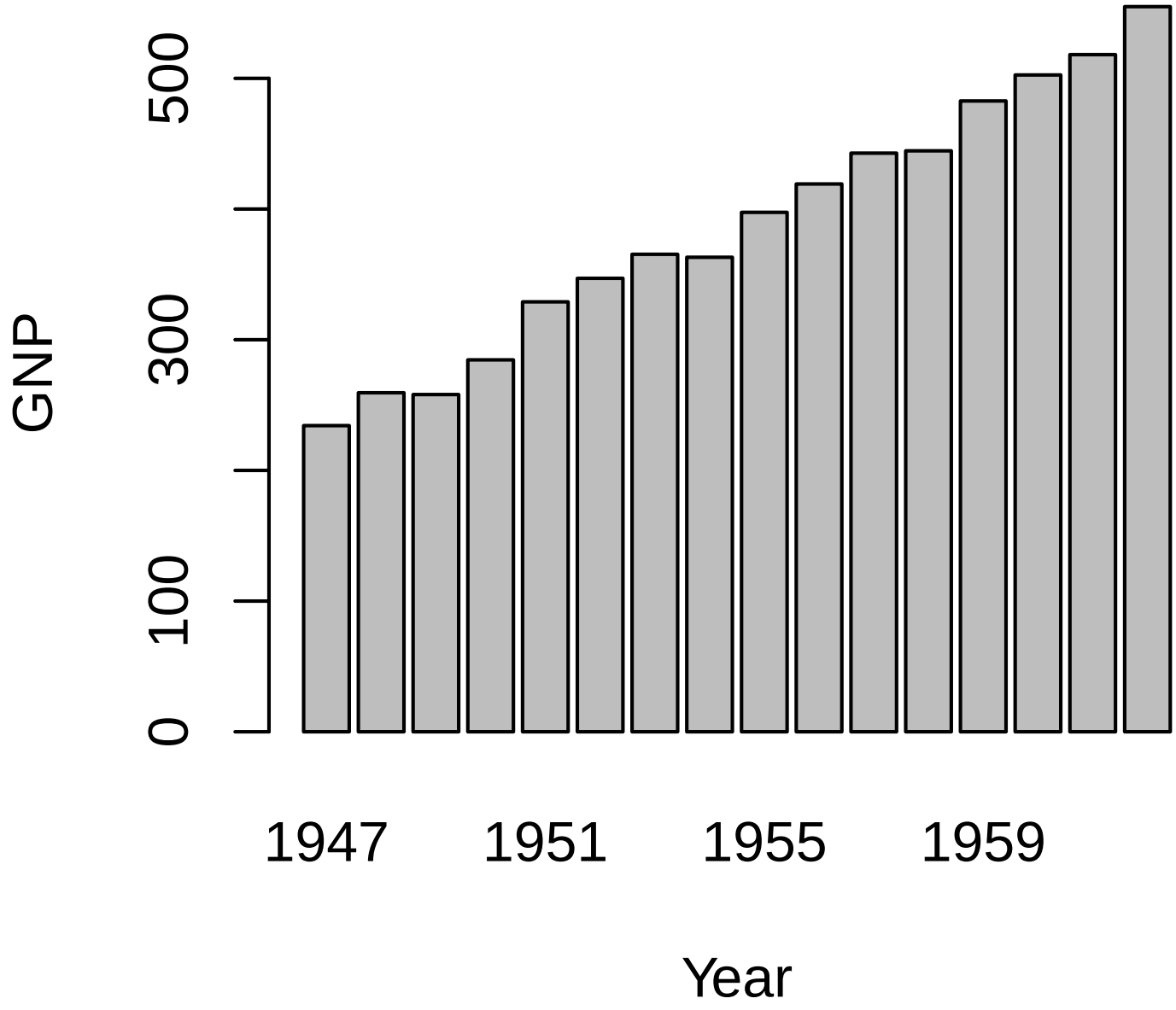
séries temporais

```
data(longley)
head(longley)
```

```
##      GNP.deflator     GNP Unemployed Armed.Forces Population Year Em
## 1947         83.0 234.289      235.6        159.0    107.608 1947
## 1948         88.5 259.426      232.5        145.6    108.632 1948
## 1949         88.2 258.054      368.2        161.6    109.773 1949
## 1950         89.5 284.599      335.1        165.0    110.929 1950
## 1951         96.2 328.975      209.9        309.9    112.075 1951
## 1952         98.1 346.999      193.2        359.4    113.270 1952
```

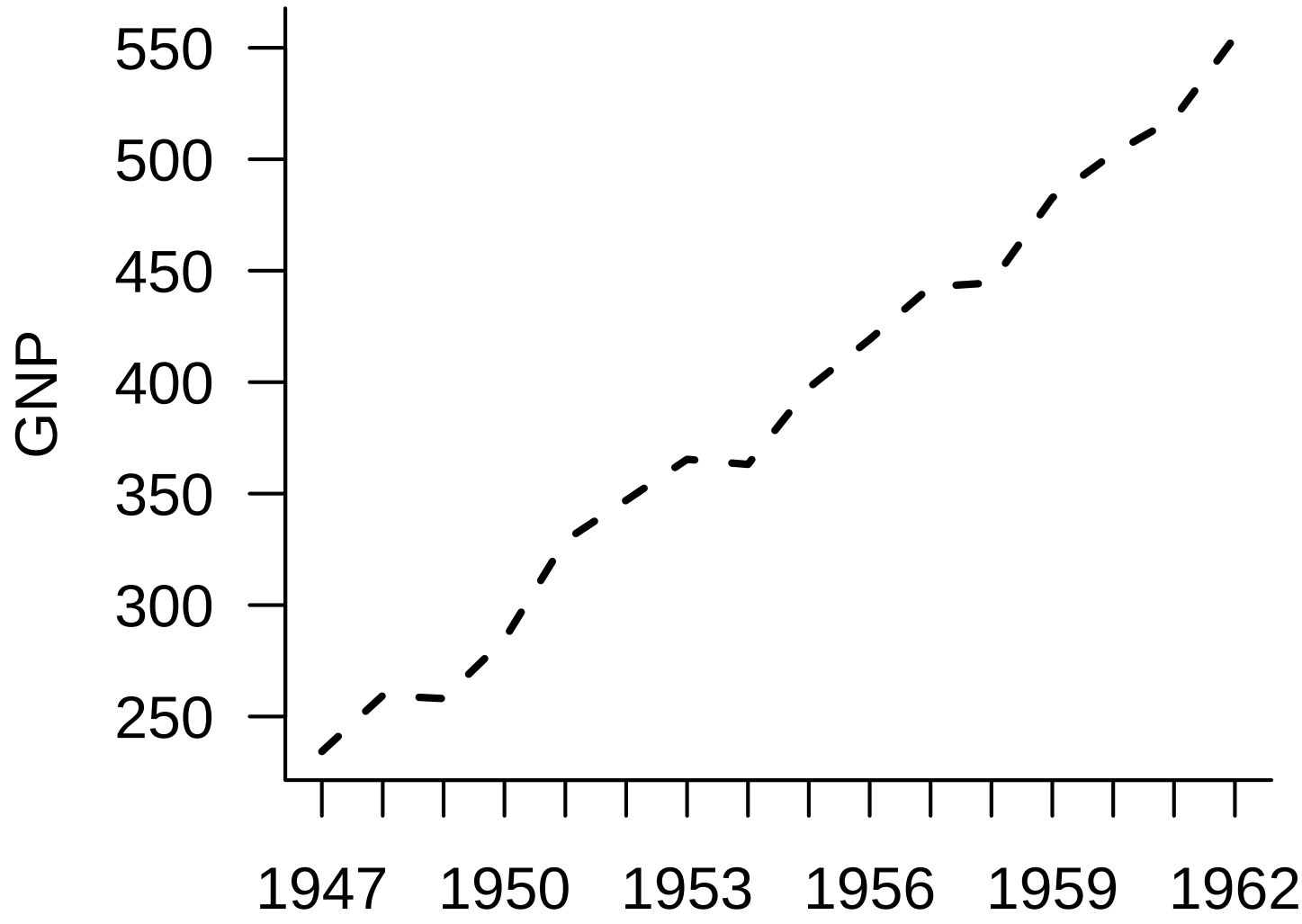
séries temporais

```
barplot(GNP ~ Year, data = longley)
```



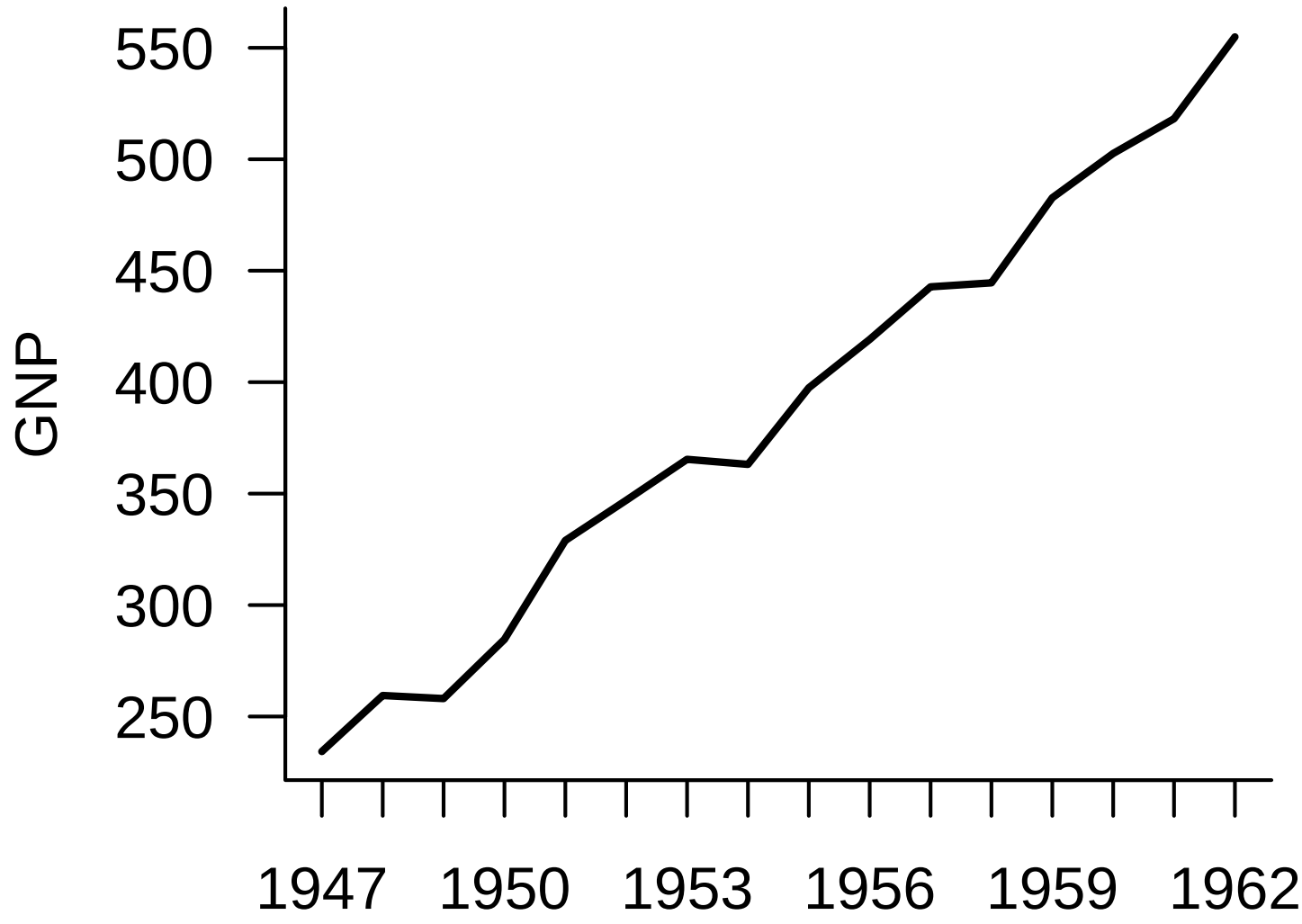
séries temporais de outra forma

```
plot(GNP ~ Year, data = longley, type='l',
     xaxt='n', las=1, bty='l', lty=2, lwd=2)
axis(1, at=longley$Year)
```



séries temporais de uma melhor forma

```
plot(GNP ~ Year, data = longley, type='l',
     xaxt='n', las=1, bty='l', lty=1, lwd=2)
axis(1, at=longley$Year)
```



3. perfumaria

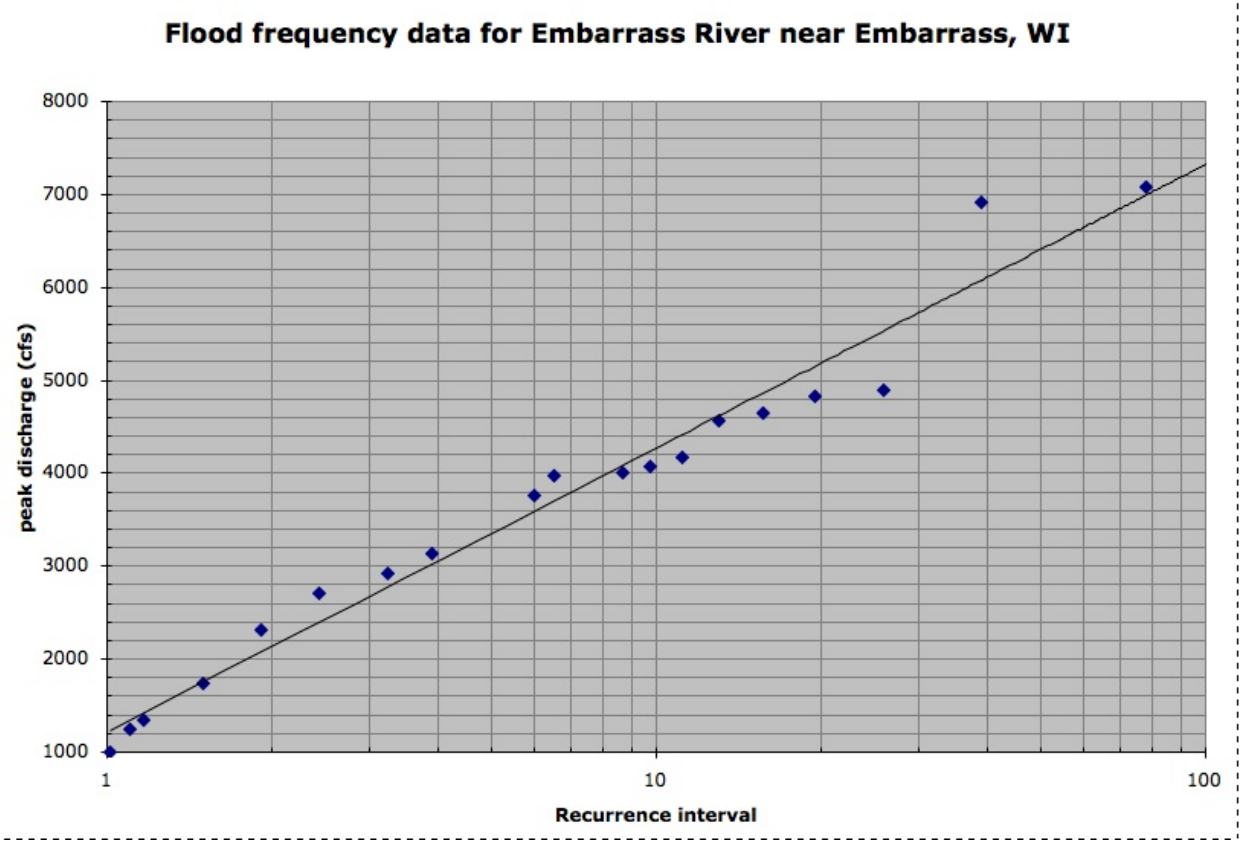
recursos para apresentações

- paletas de cores:
  - Color Brewer
  - Adobe Color
  - COLOR lovers
- ícones para adicionar em gráficos:
  - The noun project
- pacotes de R:
  - `wesanderson`
  - `RColorBrewer`
  - `swatches`
  - `colourlovers`

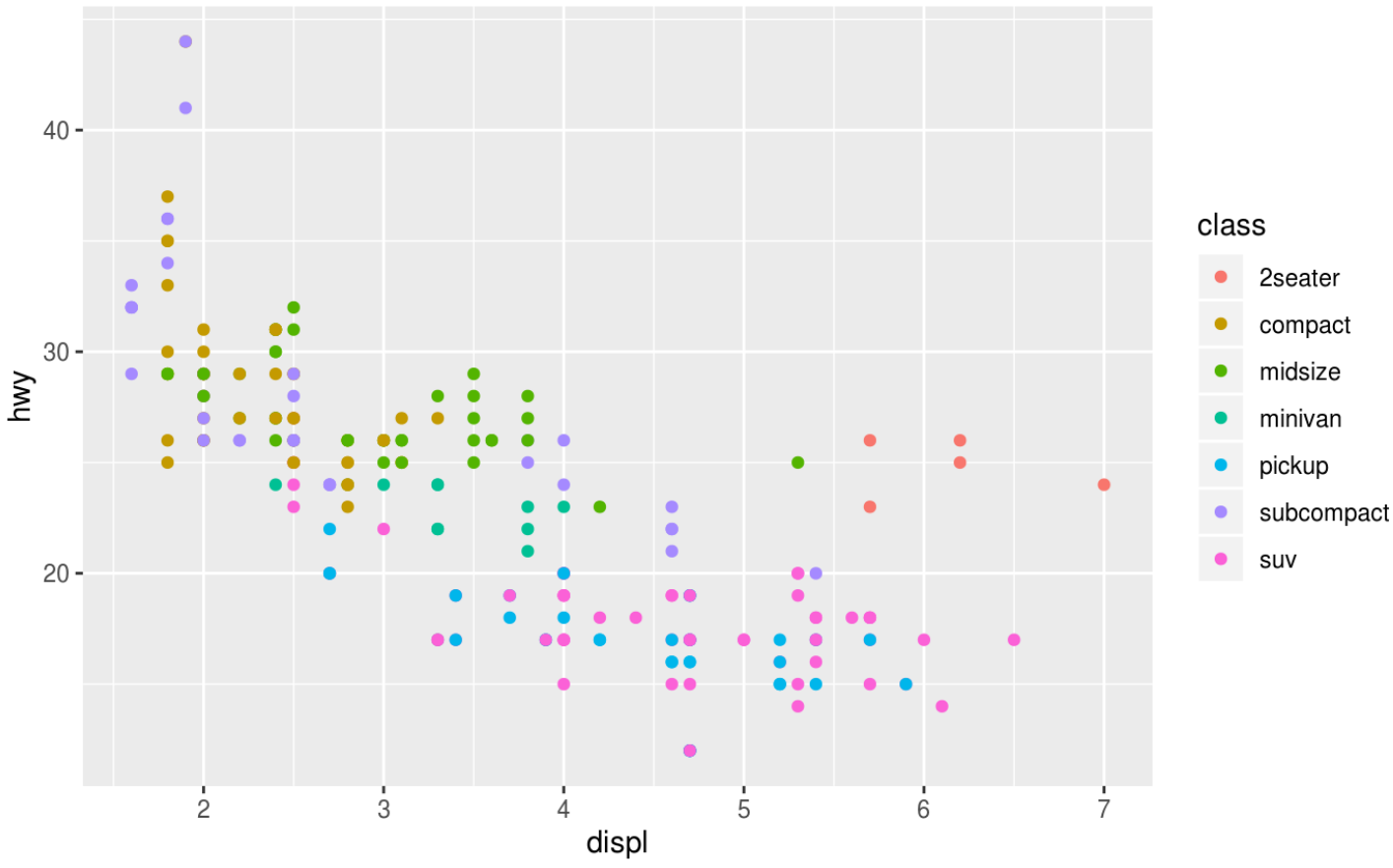
temos que falar de:

- `ggplot2`
- `gganimate`

por favor não vamos voltar no tempo



por favor não vamos voltar no tempo



$ %## extra %Vamos brincar um pouco com as cores, o R aceita tanto as especificação de cor usando um nome já existente no R como usando o código RGB ou hexadecimal de definição de cores em bytes. Se for usar nomes, tem nomes como “black”, “darkgreen”, “tomato” e você encontra uma vasta lista no link (http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf). Se for usar a especificação em RGB precisamos usar a função RGB, ou de forma mais direta, o R aceita o nome da cor em hexadecimal diretamente. A especificação de cores em código é bastante útil quando você quer criar sua própria paleta de cores. $