

Aula 7: Análise Exploratória de Dados

Profa. Yana Borges

Março de 2022

7.1 Introdução

Este tópico mostrará como usar a visualização e a transformação para explorar seus dados de maneira sistemática, uma tarefa que os estatísticos chamam de análise exploratória de dados, ou AED. A AED é um ciclo iterativo. Você:

1. Gere perguntas sobre seus dados.
2. Procure respostas visualizando, transformando e modelando seus dados.
3. Use o que você aprendeu para refinar suas perguntas e/ou gerar novas perguntas.

A AED não é um processo formal com um conjunto estrito de regras. Mais do que tudo, AED é um estado de espírito. Durante as fases iniciais da AED, você deve se sentir à vontade para investigar cada ideia que lhe ocorrer. Algumas dessas ideias vão dar certo e outras serão becos sem saída. À medida que sua exploração continuar, você se concentrará em algumas áreas particularmente produtivas que eventualmente escreverá e comunicará a outras pessoas.

A AED é uma parte importante de qualquer análise de dados, mesmo que as perguntas sejam entregues de bandeja, porque você sempre precisa investigar a qualidade de seus dados. A limpeza de dados é apenas uma aplicação da AED: você faz perguntas sobre se seus dados atendem às suas expectativas ou não. Para fazer a limpeza de dados, você precisará implantar todas as ferramentas de AED: visualização, transformação e modelagem.

7.2 Perguntas

“Não há questões estatísticas de rotina, apenas rotinas estatísticas questionáveis.” — Sir David Cox

“É muito melhor uma resposta aproximada para a pergunta certa, que muitas vezes é vaga, do que uma resposta exata para a pergunta errada, que sempre pode ser precisa.” — John Tukey

Seu objetivo durante a AED é desenvolver uma compreensão de seus dados. A maneira mais fácil de fazer isso é usar perguntas como ferramentas para orientar sua investigação. Quando você faz uma pergunta, a pergunta concentra sua atenção em uma parte específica do seu conjunto de dados e ajuda você a decidir quais gráficos, modelos ou transformações fazer.

A AED é fundamentalmente um processo criativo. E como a maioria dos processos criativos, a chave para fazer perguntas de qualidade é gerar uma grande quantidade de perguntas. É difícil fazer perguntas reveladoras no início de sua análise porque você não sabe quais insights estão contidos em seu conjunto de dados. Por outro lado, cada nova pergunta que você fizer o exporá a um novo aspecto de seus dados e aumentará sua chance de fazer uma descoberta. Você pode detalhar rapidamente as partes mais interessantes de seus dados - e desenvolver um conjunto de perguntas instigantes - se você acompanhar cada pergunta com uma nova pergunta com base no que encontrar.

Não existe uma regra sobre quais perguntas você deve fazer para orientar sua pesquisa. No entanto, dois tipos de perguntas sempre serão úteis para fazer descobertas em seus dados. Você pode escrever livremente

essas perguntas como:

1. Que tipo de variação ocorre dentro das minhas variáveis?
2. Que tipo de covariância ocorre entre minhas variáveis?

Para facilitar a discussão, vamos definir alguns termos:

- Uma **variável** é uma quantidade, qualidade ou propriedade que você pode medir.
- Um **valor** é o estado de uma variável quando você a mede. O valor de uma variável pode mudar de medição para medição.
- Uma **observação** é um conjunto de medições feitas em condições semelhantes (geralmente você faz todas as medições em uma observação ao mesmo tempo e no mesmo objeto). Uma observação conterá vários valores, cada um associado a uma variável diferente. Às vezes, me refiro a uma observação como um ponto de dados.
- **Dados tabulares** são um conjunto de valores, cada um associado a uma variável e uma observação. Os dados tabulares são organizados se cada valor for colocado em sua própria “célula”, cada variável em sua própria coluna e cada observação em sua própria linha.

Até agora, todos os dados que você viu foram organizados. Na vida real, a maioria dos dados não é organizada.

7.3 Variação

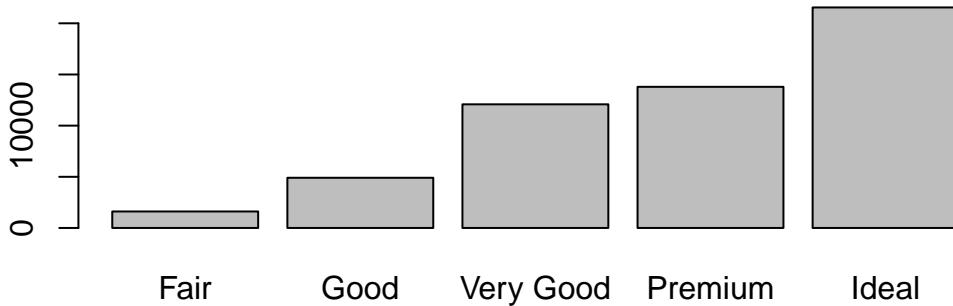
A **variação** é a tendência dos valores de uma variável mudarem de medição para medição. Você pode ver variações facilmente na vida real; se você medir qualquer variável contínua duas vezes, obterá dois resultados diferentes. Isso é verdade mesmo se você medir quantidades constantes, como a velocidade da luz. Cada uma de suas medições incluirá uma pequena quantidade de erro que varia de medição para medição. As variáveis categóricas também podem variar se você medir diferentes assuntos (por exemplo, as cores dos olhos de diferentes pessoas) ou diferentes tempos (por exemplo, os níveis de energia de um elétron em momentos diferentes). Cada variável tem seu próprio padrão de variação, o que pode revelar informações interessantes. A melhor maneira de entender esse padrão é visualizar a distribuição dos valores da variável.

7.3.1 Visualizando distribuições

Como você visualiza a distribuição de uma variável dependerá se a variável é categórica ou contínua. Uma variável é categórica se só pode receber um de um pequeno conjunto de valores. Em R, as variáveis categóricas geralmente são salvadas como fatores ou vetores de caracteres. Para examinar a distribuição de uma variável categórica, use um gráfico de barras

Vejamos o conjunto de dados `diamonds`, do pacote `ggplot2` (voltaremos a falar desse pacote):

```
library(ggplot2)
plot(diamonds$cut)
```



```
# graf2 <- ggplot(data = diamonds) +
  # geom_bar(mapping = aes(x = cut))

# grid.arrange(graf2,graf1,nrow=1)
```

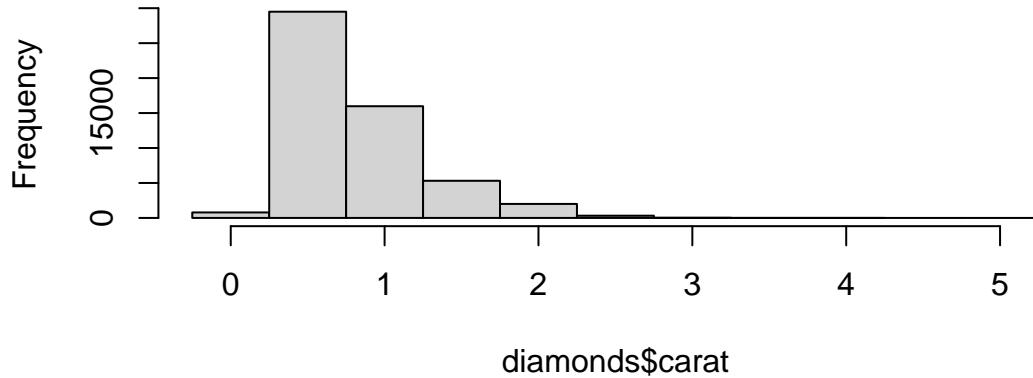
A altura das barras mostra quantas observações ocorreram com cada valor `x`. Você pode calcular esses valores manualmente:

```
table(diamonds$cut)

##
##      Fair      Good  Very Good   Premium     Ideal
##      1610      4906     12082     13791     21551
```

Uma variável é contínua se pode receber qualquer valor de um conjunto infinito de valores ordenados. Datas e horas são dois exemplos de variáveis contínuas. Para examinar a distribuição de uma variável contínua, use um histograma:

```
hist(diamonds$carat,main= NULL,
  breaks = seq(-0.25,5.25,0.5))
```



Você pode calcular isso manualmente:

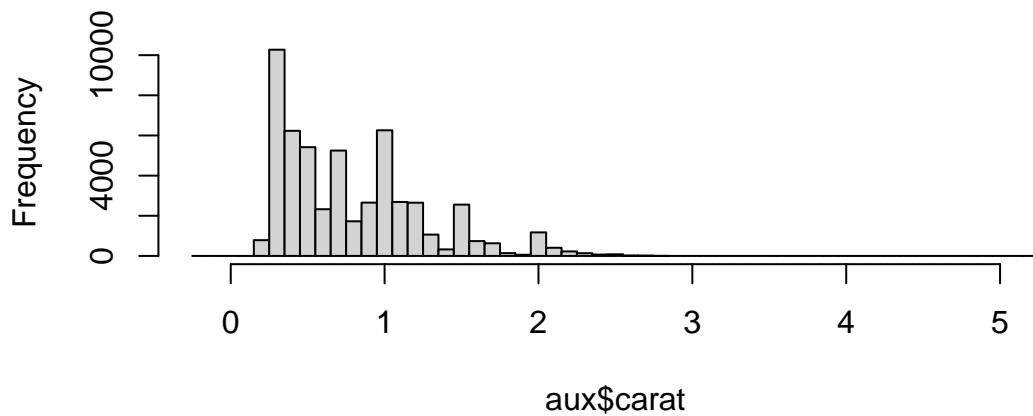
```
classes <- table(cut_width(diamonds$carat, 0.5))
cbind(classes)

##          classes
## [-0.25,0.25]    785
## (0.25,0.75]   29498
## (0.75,1.25]   15977
## (1.25,1.75]   5313
## (1.75,2.25]   2002
## (2.25,2.75]   322
## (2.75,3.25]    32
## (3.25,3.75]     5
## (3.75,4.25]     4
## (4.25,4.75]     1
## (4.75,5.25]     1
```

Um histograma divide o eixo `x` em compartimentos igualmente espaçados e, em seguida, usa a altura de uma barra para exibir o número de observações que caem em cada compartimento. No gráfico acima, a barra mais alta mostra que quase 30.000 observações têm um `carat` valor entre 0,25 e 0,75, que são as bordas esquerda e direita da barra.

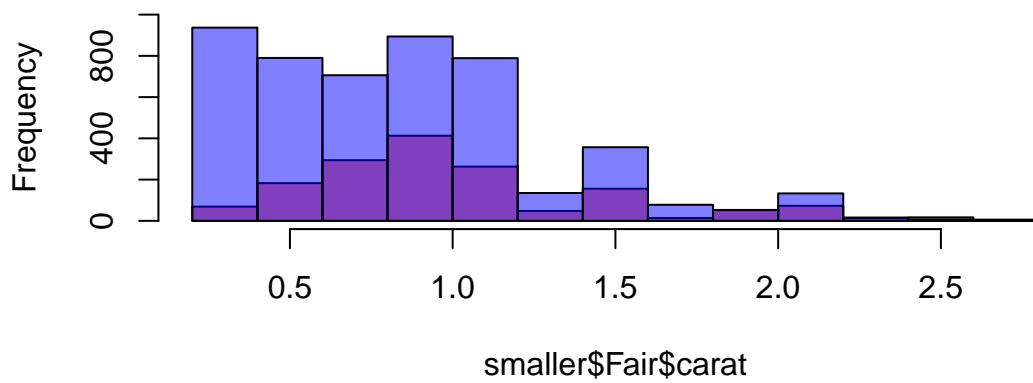
Você pode definir a largura dos intervalos em um histograma com o argumento `breaks`. Você deve sempre explorar uma variedade de `breaks` ao trabalhar com histogramas, pois diferentes `breaks` podem revelar padrões diferentes. Por exemplo, aqui está a aparência do gráfico acima quando ampliamos apenas os diamantes com um tamanho inferior a três quilates e escolhemos uma largura menor.

```
aux <- subset(diamonds, carat < 3)
hist(aux$carat, main= NULL,
     breaks = seq(-0.25, 5.25, 0.1))
```



Pode ser que você deseje sobrepor vários histogramas no mesmo gráfico,

```
aux <- subset(diamonds, carat < 3)
smaller <- split(aux, aux$cut)
hist(smaller$Fair$carat,
     main= NULL, ylim = c(0,1000),
     col=alpha("red", 0.5))
hist(smaller$Good$carat, col=alpha("blue", 0.5), add = TRUE)
```



Sugiro que, em vez de exibir as contagens com barras, use linhas. É muito mais fácil entender linhas sobrepostas do que barras.

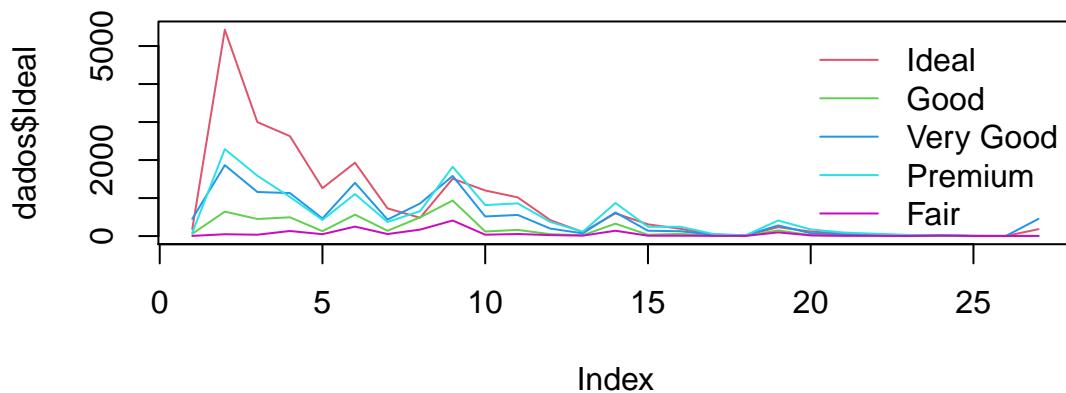
```
a <- table(cut_width(smaller$Ideal$carat, 0.1))
b <- table(cut_width(smaller$Good$carat, 0.1))
c <- table(cut_width(smaller$`Very Good`$carat, 0.1))
d <- table(cut_width(smaller$Premium$carat, 0.1))
e <- table(cut_width(smaller$Fair$carat, 0.1))
```

```

novo <- rbind.data.frame(a,b,c,d,e)
dados <- t(novo)
dados <- data.frame(dados, row.names = names(b))
names(dados) <- c("Ideal", "Good", "Very good", "Premium", "Fair")

plot(dados$Ideal, type ="l", col = 2)
lines(dados$Good,col = 3)
lines(dados$`Very good`, col = 4)
lines(dados$Premium, col = 5)
lines(dados$Fair, col = 6)
legend("topright", bty = "n",
       col = c(2:6), lty=1,
       legend = c("Ideal", "Good",
                 "Very Good", "Premium",
                 "Fair"),
       )

```



Agora que você pode visualizar a variação, o que você deve procurar em seus gráficos? E que tipo de perguntas de acompanhamento você deve fazer? Reuni uma lista abaixo dos tipos mais úteis de informações que você encontrará em seus gráficos, juntamente com algumas perguntas de acompanhamento para cada tipo de informação. A chave para fazer boas perguntas de acompanhamento será confiar em sua curiosidade (sobre o que você quer aprender mais?) bem como em seu ceticismo (como isso pode ser enganoso?).

7.3.2 Valores típicos

Tanto nos gráficos de barras quanto nos histogramas, as barras altas mostram os valores comuns de uma variável e as barras mais curtas mostram os valores menos comuns. Locais sem barras revelam valores que não foram vistos em seus dados. Para transformar essas informações em perguntas úteis, procure algo inesperado:

- Quais valores são os mais comuns? Por quê?
- Quais valores são raros? Por quê? Isso corresponde às suas expectativas?
- Você consegue ver algum padrão incomum? O que poderia explicá-lo?

Agrupamentos de valores semelhantes sugerem que existem subgrupos em seus dados. Para entender os subgrupos, pergunte:

Como as observações dentro de cada **cluster** são semelhantes entre si?

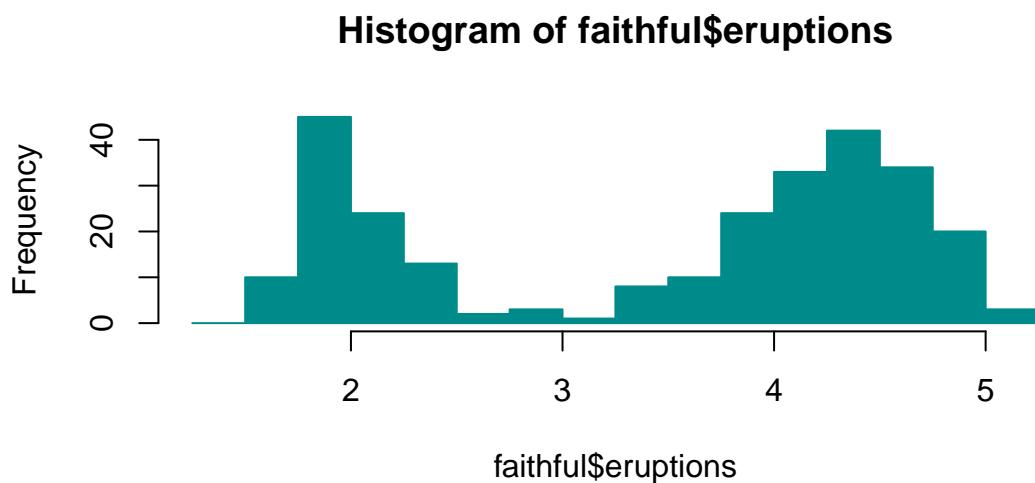
Como as observações em grupos separados são diferentes umas das outras?

Como você pode explicar ou descrever os clusters?

Por que a aparência de clusters pode ser enganosa?

O histograma abaixo mostra a duração (em minutos) de 272 erupções do Old Faithful Geyser no Parque Nacional de Yellowstone. Os tempos de erupção parecem estar agrupados em dois grupos: há erupções curtas (de cerca de 2 minutos) e erupções longas (4-5 minutos), mas pouco entre elas.

```
hist(faithful$erruptions,
      breaks = seq(1.25,5.25,.25),
      col="darkcyan", border = "darkcyan")
```

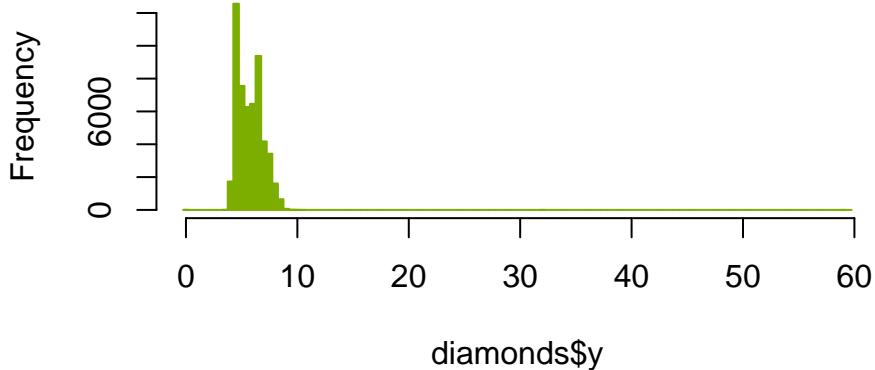


Muitas das perguntas acima o levarão a explorar um relacionamento entre variáveis, por exemplo, para ver se os valores de uma variável podem explicar o comportamento de outra variável.

7.3.3 Valores incomuns

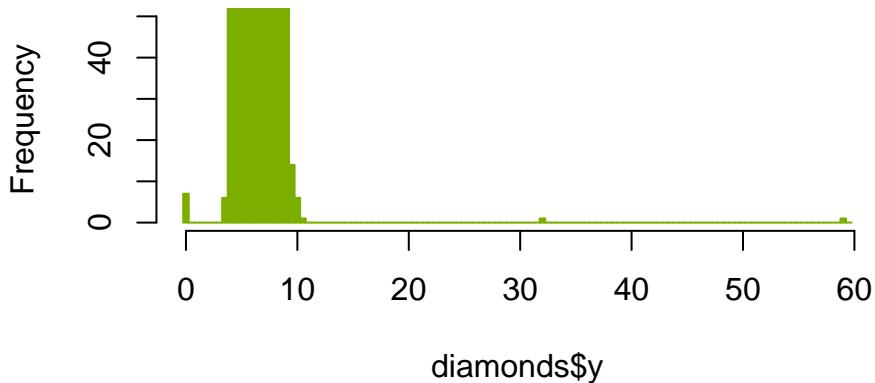
Outliers são observações incomuns; pontos de dados que não parecem se encaixar no padrão. Às vezes, os outliers são erros de entrada de dados; outras vezes, os valores discrepantes sugerem uma nova ciência importante. Quando você tem muitos dados, às vezes é difícil ver os valores discrepantes em um histograma. Por exemplo, pegue a distribuição da variável `y` do conjunto de dados de diamantes. A única evidência de outliers são os limites incomumente largos no eixo x.

```
hist(diamonds$y,main = "",
      breaks = seq(-0.25,60,0.5),
      col="#7CAE00", border = "#7CAE00")
```



Há tantas observações nas colunas com observações frequentes e colunas com pucas observações com colunas tão curtas que você não podevê-las (embora talvez se você olhar fixamente para 0 você verá algo). Para facilitar a visualização dos valores incomuns, precisamos ampliar os valores do eixo `y` com `ylim()`:

```
hist(diamonds$y, main = "",
      breaks = seq(-0.25, 60, 0.5),
      ylim = c(0, 50),
      col="#7CAE00", border = "#7CAE00")
```



Isso nos permite ver que existem três valores incomuns: 0, ~30 e ~60. Nós os selecionamos usando `filter()` e `select()` do pacote `dplyr`

```
library(dplyr)
incomun <- filter(diamonds, diamonds$y < 3 | diamonds$y > 20)
incomuns <- select(incomun, price, x,y,z)
incomuns

## # A tibble: 9 x 4
```

```

##   price      x      y      z
##   <int> <dbl> <dbl> <dbl>
## 1  5139     0      0      0
## 2  6381     0      0      0
## 3 12210    8.09   58.9   8.06
## 4 12800     0      0      0
## 5 15686     0      0      0
## 6 18034     0      0      0
## 7 2075     5.15   31.8   5.12
## 8 2130     0      0      0
## 9 2130     0      0      0

```

A variável `y` mede uma das três dimensões desses diamantes, em mm. Sabemos que os diamantes não podem ter largura de 0 mm, portanto, esses valores devem estar incorretos. Também podemos suspeitar que as medidas de 32 mm e 59 mm são implausíveis: esses diamantes têm mais de uma polegada de comprimento (2,54 cm), mas não custam centenas de milhares de dólares!

É uma boa prática repetir sua análise com e sem os valores discrepantes. Se eles tiverem um efeito mínimo nos resultados e você não conseguir descobrir por que eles estão lá, é razoável substituí-los por **valores ausentes** e seguir em frente. No entanto, se eles tiverem um efeito substancial em seus resultados, você não deve abandoná-los sem justificativa. Você precisará descobrir o que os causou (por exemplo, um erro de entrada de dados) e divulgar que os removeu.

7.4 Valores ausentes

Se você encontrou valores incomuns em seu conjunto de dados e simplesmente deseja passar para o restante de sua análise, você tem duas opções.

1. Retire as linhas inteiras com os valores estranhos:

```
diamonds2 <- filter(diamonds, between(y, 3, 20))
```

Eu não recomendo esta opção porque só porque uma medida é inválida, não significa que todas as medidas são. Além disso, se você tiver dados de baixa qualidade, quando tiver aplicado essa abordagem a todas as variáveis, poderá descobrir que não tem mais dados!

2. Em vez disso, recomendo substituir os valores incomuns por valores ausentes. A maneira mais fácil de fazer isso é usar `mutate()` para substituir a variável por uma cópia modificada. Você pode usar a função `if (condition) { }` para substituir valores incomuns por `NA`:

```

diamonds3 <- diamonds
for (i in 1:nrow(diamonds3)) {
  if (diamonds3$y[i] < 3 | diamonds3$y[i] > 20) {
    diamonds3$y[i] <- NA
  }
}
mean(diamonds3$y)

```

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

Existem várias alternativas para esse processo e vai depender do nível de conhecimento do usuário e/ou preferência para decidir o mais adequado.

Veja o resultado usando pipes:

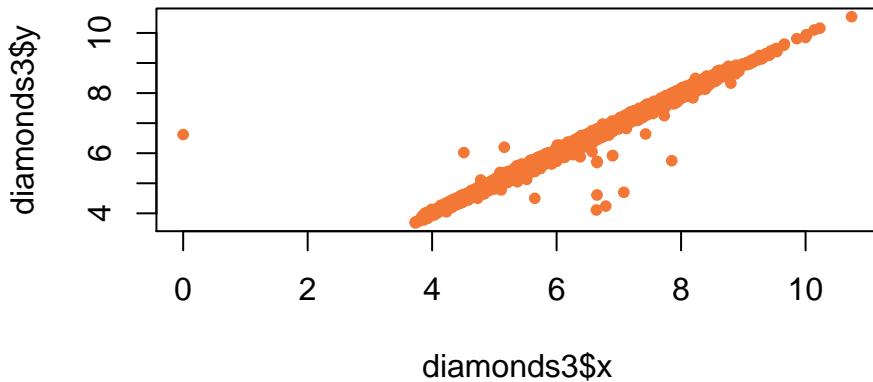
```

diamonds2 <- diamonds %>%
  mutate(y = ifelse(y < 3 | y > 20, NA, y))

```

Ao plotar o gráfico o R não inclui os valores ausentes. Se você não conhecer seu conjunto de dados e nele existirem valores ausentes, não será possível perceber somente observando o gráfico construído pelo comando `plot()`.

```
plot(diamonds3$x, diamonds3$y,  
      pch = 20, col = "#f37735")
```

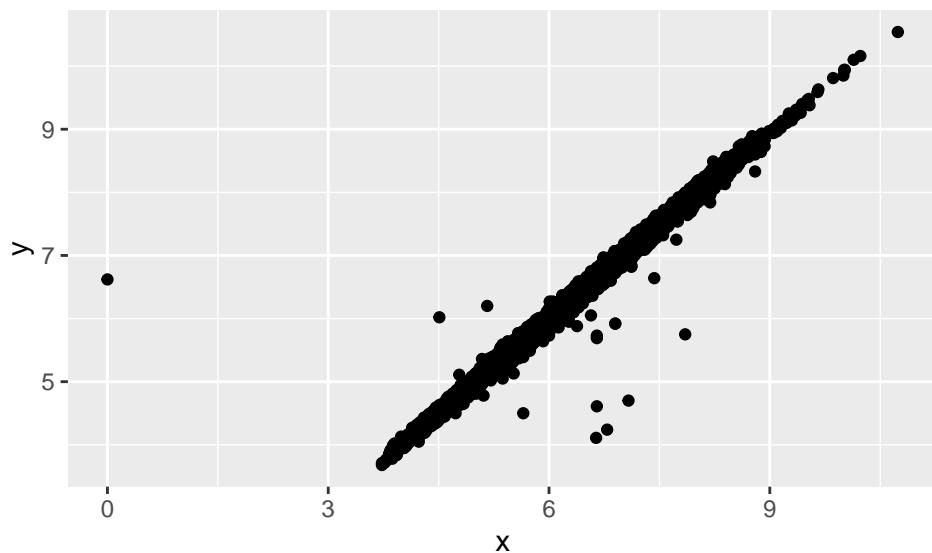


O pacote `ggplot2` é voltado para a construção de gráficos elegantes e com muitas opções de apresentação.

Assim como o R, o `ggplot2` segue a filosofia de que valores ausentes nunca devem desaparecer silenciosamente. Não é óbvio onde você deve plotar os valores ausentes, então `ggplot2` não os inclui no gráfico, mas avisa que eles foram removidos:

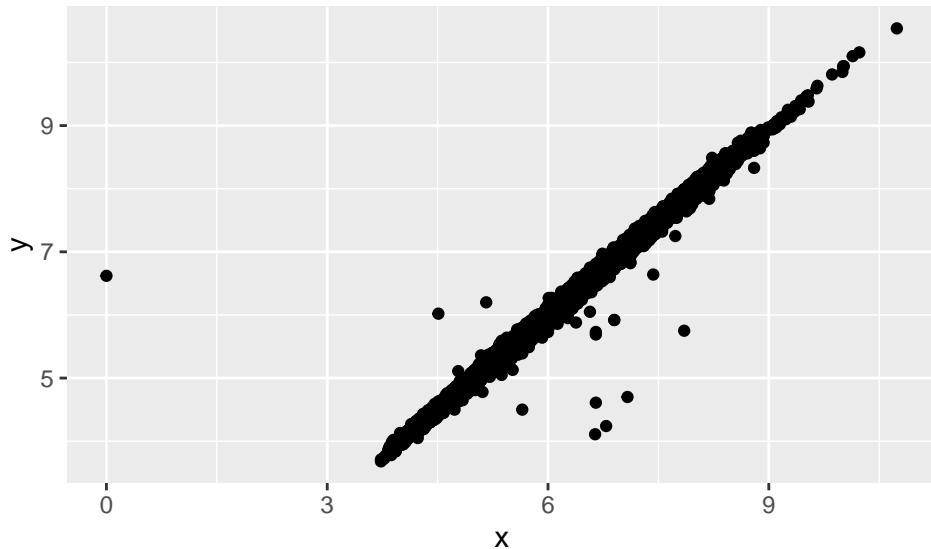
```
ggplot(data = diamonds3, mapping = aes(x = x, y = y)) +  
  geom_point()
```

```
## Warning: Removed 9 rows containing missing values (geom_point).
```



Para suprimir esse aviso, defina `na.rm = TRUE`:

```
ggplot(data = diamonds2, mapping = aes(x = x, y = y)) +  
  geom_point(na.rm = TRUE)
```



7.5 O conjunto de dados Anscombe

Frank Anscombe(1918-2001) afirmava que “um computador deve fazer cálculos e gráficos”, e ilustrou a importância de representar graficamente dados com quatro conjuntos de dados agora conhecidos como quarteto de Anscombe

Frank foi um estatístico que dentre suas muitas contribuições para a ciência advogou em prol do uso de gráficos para conhecer os dados.

Os dados já existem dentro do R, por isso você pode carregá-los usando a função `data`.

Primeiro, vamos checar os dados com comnados visto em aulas anteriores:

```
data("anscombe") # carrega os dados  
dim(anscombe) # dimensao dos dados  
head(anscombe) # seis primeiras linhas dos dados  
class(anscombe) # classe do objeto  
str(anscombe) # estrutura do objeto
```

Portanto, o conjunto de dados `anscombe` é do tipo dataframe, possui 11 linhas, 8 variáveis do tipo numérico.

7.5.1 Medidas descritivas

7.5.1.1 Média

As medidas descritivas precisam ser analisadas. Para se obter a média de cada coluna com título “x” podemos fazer:

```
mean(anscombe$x1)  
  
## [1] 9  
mean(anscombe$x2)  
  
## [1] 9
```

```
mean(anscombe$x3)
```

```
## [1] 9
```

```
mean(anscombe$x4)
```

```
## [1] 9
```

Função apply

A função `apply` aplica uma função a um vetor ou matriz a partir de uma margem especificada

Podemos realizar a mesma tarefa mas agora com apenas uma linha de comando:

```
## media de todas as variaveis
```

```
apply(anscombe, 2, mean)
```

```
##      x1      x2      x3      x4      y1      y2      y3      y4
## 9.000000 9.000000 9.000000 9.000000 7.500909 7.500909 7.500000 7.500909
```

```
## media de todos os vetores x
```

```
apply(anscombe[,1:4], 2, mean)
```

```
## x1 x2 x3 x4
## 9 9 9 9
```

```
## media de todos os vetores y
```

```
apply(anscombe[,5:8], 2, mean)
```

```
##      y1      y2      y3      y4
## 7.500909 7.500909 7.500000 7.500909
```

Outra função bastante útil para uma primeira análise dos dados é a função `summary`.

```
summary(anscombe)
```

```
##      x1          x2          x3          x4          y1
## Min.   : 4.0   Min.   : 4.0   Min.   : 4.0   Min.   : 8   Min.   : 4.260
## 1st Qu.: 6.5   1st Qu.: 6.5   1st Qu.: 6.5   1st Qu.: 8   1st Qu.: 6.315
## Median : 9.0   Median : 9.0   Median : 9.0   Median : 8   Median : 7.580
## Mean    : 9.0   Mean   : 9.0   Mean   : 9.0   Mean   : 9   Mean   : 7.501
## 3rd Qu.:11.5   3rd Qu.:11.5   3rd Qu.:11.5   3rd Qu.: 8   3rd Qu.: 8.570
## Max.   :14.0   Max.   :14.0   Max.   :14.0   Max.   :19   Max.   :10.840
##           y2          y3          y4
## Min.   :3.100   Min.   : 5.39   Min.   : 5.250
## 1st Qu.:6.695   1st Qu.: 6.25   1st Qu.: 6.170
## Median :8.140   Median : 7.11   Median : 7.040
## Mean   :7.501   Mean   : 7.50   Mean   : 7.501
## 3rd Qu.:8.950   3rd Qu.: 7.98   3rd Qu.: 8.190
## Max.   :9.260   Max.   :12.74   Max.   :12.500
```

Aplicaremos a mesma função para obter as demais medidas.

7.5.1.2 Variância

```
# variância dos dados
```

```
apply(anscombe, 2, var) # aplica a função var a todas as linhas do objeto
```

```
##      x1          x2          x3          x4          y1          y2          y3          y4
## 11.000000 11.000000 11.000000 11.000000 4.127269 4.127629 4.122620 4.123249
```

7.5.1.3 Correlação e regressão

Entendendo a correlação e coeficiente de regressão dos conjuntos x e y.

Podemos usar a função `cor` para calcular a correlação entre as variáveis x e y.

```
# função cor para correlação  
cor(anscombe$x1, anscombe$y1)
```

```
## [1] 0.8164205
```

```
cor(anscombe$x2, anscombe$y2)
```

```
## [1] 0.8162365
```

```
cor(anscombe$x3, anscombe$y3)
```

```
## [1] 0.8162867
```

```
cor(anscombe$x4, anscombe$y4)
```

```
## [1] 0.8165214
```

Podemos obter os coeficientes de regressão de cada análise de regressão ou podemos usar um método de repetição sem loop, usando `lapply`

- Função `lapply`

Similar ao `apply`, a diferença é que poderá receber uma lista, um vetor ou um `dataframe` como entrada.

Primeiro criamos objetos com as regressões dos quatro conjuntos

```
# coeficiente de regressão  
m1 <- lm(anscombe$y1 ~ anscombe$x1)  
m2 <- lm(anscombe$y2 ~ anscombe$x2)  
m3 <- lm(anscombe$y3 ~ anscombe$x3)  
m4 <- lm(anscombe$y4 ~ anscombe$x4)  
## vamos criar agora uma lista com todos os modelos para facilitar o trabalho  
mlist <- list(m1, m2, m3, m4)  
## agora sim podemos calcular de forma menos repetitiva os coeficientes de regressao  
lapply(mlist, coef)
```

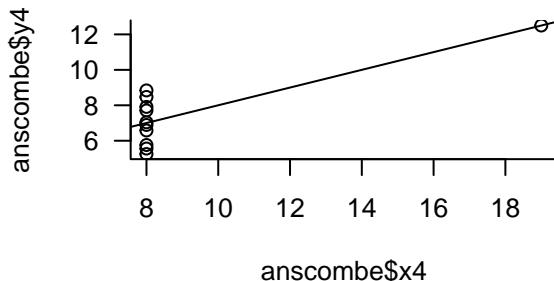
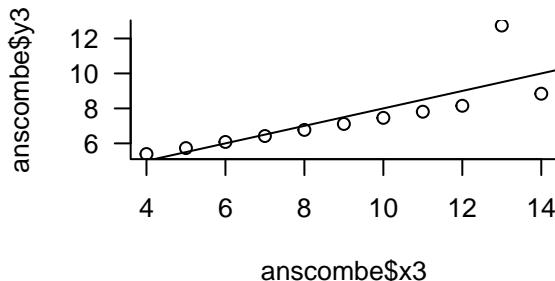
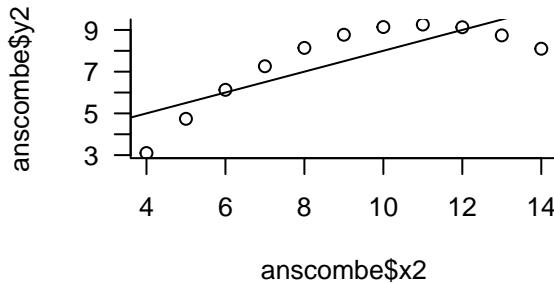
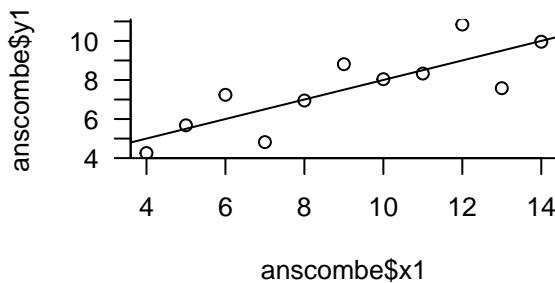
```
## [[1]]  
## (Intercept) anscombe$x1  
##     3.0000909   0.5000909  
##  
## [[2]]  
## (Intercept) anscombe$x2  
##     3.000909    0.5000000  
##  
## [[3]]  
## (Intercept) anscombe$x3  
##     3.0024545   0.4997273  
##  
## [[4]]  
## (Intercept) anscombe$x4  
##     3.0017273   0.4999091
```

Os dados têm mesma média, mesma variância, mesma correlação e mesmo valores dos coeficientes (intercepto e inclinação do modelo linear). Em que os dados são diferentes?

Os valores parecem diferentes. Mas quão diferentes?

7.5.1.4 Análise gráfica

```
par(mfrow=c(2, 2), #abre uma janela gráfica com 2 linhas e 2 colunas
    las=1, # deixa as legendas dos eixos na vertical
    bty="l") # tipo do box do grafico em L
plot(anscombe$y1 ~ anscombe$x1) #plot das variaveis
abline(mlist[[1]])
plot(anscombe$y2 ~ anscombe$x2)
abline(mlist[[2]])
plot(anscombe$y3 ~ anscombe$x3)
abline(mlist[[3]])
plot(anscombe$y4 ~ anscombe$x4)
abline(mlist[[4]])
```



E agora? Conseguimos perceber que os dados se comportam de maneira diferente?

7.6 O conjunto de dados Iris

O conjunto de dados iris foi coletado por Edgar Anderson e ficou famoso pelo trabalho de Ronald E. Fisher. Vamos carregar os dados no R.

7.6.1 Inspeções iniciais

Vamos então começar com as inspeções básicas do arquivo.

```

head(iris)

##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa

summary(iris)

##   Sepal.Length     Sepal.Width    Petal.Length    Petal.Width
##   Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##   1st Qu.:5.100  1st Qu.:2.800  1st Qu.:1.600  1st Qu.:0.300
##   Median  :5.800  Median  :3.000  Median  :4.350  Median  :1.300
##   Mean    :5.843  Mean    :3.057  Mean    :3.758  Mean    :1.199
##   3rd Qu.:6.400  3rd Qu.:3.300  3rd Qu.:5.100  3rd Qu.:1.800
##   Max.    :7.900  Max.    :4.400  Max.    :6.900  Max.    :2.500
##   Species
##   setosa    :50
##   versicolor:50
##   virginica :50
##
##
```

Quantas informações por espécie?

```

table(iris$Species)

##
```

	setosa	versicolor	virginica
50	50	50	

Qual a média das variáveis por espécie? Vamos usar as funções aggregate e tapply. As duas funções são semelhantes, o que muda são os argumentos e o formato de saída de cada uma delas.

- Função `tapply`

A função `tapply` aplica uma função a um subconjunto de um vetor que é construído a partir de um outro vetor (normalmente, um fator).

```

# media do comprimento de sepala por especie
tapply(X = iris$Sepal.Length, INDEX = list(iris$Species), FUN = mean)

##      setosa versicolor virginica
##      5.006    5.936    6.588

```

A mesma tarefa pode ser executada pela função `aggregate`, que terá outros argumentos e outra saída:

```

aggregate(x = iris$Sepal.Length, by = list(iris$Species), FUN = mean)

##   Group.1     x
## 1  setosa 5.006
## 2 versicolor 5.936
## 3 virginica 6.588

```

```
# ainda a mesma tarefa, com a mesma função mas em uma notação diferente
aggregate(Sepal.Length ~ Species, data = iris, mean)
```

```
##      Species Sepal.Length
## 1      setosa      5.006
## 2 versicolor      5.936
## 3 virginica      6.588
```

A função `tapply` retorna um objeto do tipo `array`, enquanto `aggregate` retorna um objeto do tipo `data.frame`, mas ambos com os mesmos resultados.

Podemos fazer o mesmo para as outras variáveis.

```
aggregate(Sepal.Length ~ Species, data = iris, mean)
```

```
##      Species Sepal.Length
## 1      setosa      5.006
## 2 versicolor      5.936
## 3 virginica      6.588
```

```
aggregate(Sepal.Width ~ Species, data = iris, mean)
```

```
##      Species Sepal.Width
## 1      setosa      3.428
## 2 versicolor      2.770
## 3 virginica      2.974
```

```
aggregate(Petal.Length ~ Species, data = iris, mean)
```

```
##      Species Petal.Length
## 1      setosa      1.462
## 2 versicolor      4.260
## 3 virginica      5.552
```

E agora vamos calcular o desvio padrão das variáveis.

```
tapply(X = iris$Sepal.Length, INDEX = list(iris$Species), FUN = sd)
```

```
##      setosa versicolor virginica
## 0.3524897 0.5161711 0.6358796
```

```
tapply(X = iris$Sepal.Width, INDEX = list(iris$Species), FUN = sd)
```

```
##      setosa versicolor virginica
## 0.3790644 0.3137983 0.3224966
```

```
tapply(X = iris$Petal.Length, INDEX = list(iris$Species), FUN = sd)
```

```
##      setosa versicolor virginica
## 0.1736640 0.4699110 0.5518947
```

```
tapply(X = iris$Petal.Width, INDEX = list(iris$Species), FUN = sd)
```

```
##      setosa versicolor virginica
## 0.1053856 0.1977527 0.2746501
```

Sempre que você está copiando e colando um comando, pense que existe uma maneira melhor de executar a sequência de tarefas. Afinal, se você tivesse 99 variáveis, copiar e colar 99x um comando não parece uma boa ideia. Veja abaixo uma solução de como calcular a média por espécie de todas as variáveis. Para isso, vamos usar o comando `for` e executar todas as tarefas em um mesmo ciclo.

```

# criando matriz de 3 colunas - uma para cada especie - e 4 linhas - uma para cada metrica
medias <- matrix(NA, ncol = 3, nrow = 4)
# definindo o nome das colunas e das linhas da matriz
colnames(medias) <- unique(iris$Species)
rownames(medias) <- names(iris)[-5]
for (i in 1:4){
  medias[i,] <- tapply(iris[,i], iris$Species, mean)
}
medias

##           setosa versicolor virginica
## Sepal.Length 5.006      5.936     6.588
## Sepal.Width  3.428      2.770     2.974
## Petal.Length 1.462      4.260     5.552
## Petal.Width   0.246      1.326     2.026

```

- **Função unique**

A função `unique` retorna um vetor, quadro de dados ou array como x, mas com elementos/linhas duplicados removidos.

7.6.2 Estatísticas descritivas

A tabela abaixo contém as medidas descritivas que usaremos.

Parâmetro	Descrição	Função
Média	média aritmética	<code>mean()</code>
Mediana	valor central	<code>median()</code>
Moda	valor mais frequente	<code>sort(table(), decreasing=TRUE)[1]</code>
Desvio-padrão	variação em torno da média	<code>sd()</code>
Quantis	corte que divide uma distribuição de probabilidades	<code>quantile()</code>

7.6.2.1 Medidas de tendência central

- Média

$$\bar{x} = \frac{\sum_{i=1}^n x_i}{n}$$

```

vars <- iris[, -5]
apply(vars, 2, mean)

```

```

## Sepal.Length  Sepal.Width Petal.Length  Petal.Width
##      5.843333    3.057333    3.758000    1.199333

```

- Mediana: 50º quantil, de forma que divide os dados em duas metades

```

apply(vars, 2, median)

```

```

## Sepal.Length  Sepal.Width Petal.Length  Petal.Width
##      5.80          3.00        4.35         1.30

```

- Moda: valor mais frequente na amostra

```

freq_sl <- sort(table(iris$Sepal.Length), decreasing = TRUE)
freq_sl[1]

```

```
## 5
## 10
```

7.6.2.2 Medidas de dispersão

- Variância: desvio da média

$$s^2 = \sum_{i=1}^n \frac{(x_i - \bar{x})^2}{n-1}$$

```
apply(vars, 2, var)
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width
## 0.6856935   0.1899794   3.1162779   0.5810063
```

- Desvio padrão: raiz quadrada da variância

$$S = \sqrt{\sum_{i=1}^n \frac{(x_i - \bar{x})^2}{n-1}}$$

```
sd01 <- apply(vars, 2, sd)
# outra forma:
sd02 <- apply(vars, 2, function(x) sqrt(var(x)))
sd01
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width
```

```
## 0.8280661   0.4358663   1.7652982   0.7622377
```

```
sd02
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width
```

```
## 0.8280661   0.4358663   1.7652982   0.7622377
```

```
sd01 == sd02
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width
##      TRUE        TRUE        TRUE        TRUE
```

- Coeficiente de variação: medida relativa de desvio padrão

$$CV = \frac{s}{\bar{x}} \times 100$$

Não existe no R base uma função para calcular o coeficiente de variação. Isto não é um problema. Vamos formalmente criar nossa primeira função de R. Para isso, usamos a função `function`

```
cv <- function(x){
  sd(x)/mean(x)*100
}
apply(vars, 2, cv)
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width
## 14.17113    14.25642    46.97441    63.55511
```

- Quantis ou percentis

É o valor que corta a enésima porcentagem de valores dos dados quando ordenados de forma ascendente. Por padrão, a função `quantile` retorna o mínimo, o 25º percentil, a mediana, o 50º percentil, o 75º percentil e o máximo, também conhecidos pelo sumário de cinco números proposto por Tukey (que também é o retorno da função `summary` de um vetor numérico). Os cinco números dividem os dados em quatro quantis, que, por isso são chamados de quartis. Os quartis são uma métrica bastante útil para descrever os dados pois possuem uma interpretação simples e não são afetados pela distribuição dos dados. É possível modificar os percentis desejados com o argumento `probs`.

```

# sumario de 5 numeros
apply(vars, 2, quantile)

##      Sepal.Length Sepal.Width Petal.Length Petal.Width
## 0%          4.3        2.0     1.00       0.1
## 25%         5.1        2.8     1.60       0.3
## 50%         5.8        3.0     4.35       1.3
## 75%         6.4        3.3     5.10       1.8
## 100%        7.9        4.4     6.90       2.5

# 5%, 50% e 95%
apply(vars, 2, quantile, probs=c(0.05, 0.5, 0.95))

```

```

##      Sepal.Length Sepal.Width Petal.Length Petal.Width
## 5%        4.600     2.345     1.30       0.2
## 50%        5.800     3.000     4.35       1.3
## 95%        7.255     3.800     6.10       2.3

```

- Amplitude total (range)

A amplitude total é a diferença entre o maior e o menor valor de determinada variável.

```

# a funcao range nos retorna os valores minimo e maximo
apply(vars, 2, range)

```

```

##      Sepal.Length Sepal.Width Petal.Length Petal.Width
## [1,]        4.3        2.0      1.0       0.1
## [2,]        7.9        4.4      6.9       2.5

# aplicando a funcao diff ao resultado do range, temos o valor desejado
my_range <- function(x){
  diff(range(x))
}
apply(vars, 2, my_range)

```

```

## Sepal.Length Sepal.Width Petal.Length Petal.Width
##          3.6        2.4        5.9       2.4

```

- Intervalo interquartílico

O intervalo interquartílico é a diferença entre o quartil superior (75 e o quartil inferior (25.

```

apply(vars, 2, IQR)

```

```

## Sepal.Length Sepal.Width Petal.Length Petal.Width
##          1.3        0.5        3.5       1.5

```

- Correlação

Uma matriz de correlação é uma tabela com os valores de correlação entre cada uma das variáveis par a par. As variáveis podem ser correlacionadas positivamente (valores positivos) ou negativamente (valores negativos). O que são variáveis altamente correlacionadas? Uma boa “regra de dedão” é que qualquer correlação ≥ 0.7 é considerada uma alta correlação.

```

cor(vars)

```

```

##      Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000 -0.1175698   0.8717538   0.8179411
## Sepal.Width     -0.1175698  1.0000000  -0.4284401  -0.3661259
## Petal.Length    0.8717538  -0.4284401   1.0000000   0.9628654
## Petal.Width     0.8179411  -0.3661259   0.9628654   1.0000000

```

7.6.3 Análise gráfica

Olhar para a distribuição dos dados é essencial (Anscombe já dizia). As distribuições de frequência são úteis porque nos ajudam a visualizar o centro e a distribuição dos dados. Estamos muito acostumados com a distribuição normal, mas os dados biológicos podem assumir diferentes distribuições de probabilidades contínuas ou discretas.

Uma breve descrição dos métodos gráficos:

Gráfico	Descrição
Barras	cada barra representa a frequência de observações de um grupo
Histograma	cada barra representa a frequência de observações em dado conjunto de valores
Densidade	estimativa da distribuição estatística baseada nos dados
Quantil-quantil	comparação dos dados em relação a uma distribuição normal
Box-plot	representação visual da média, quartis, simetria, outliers
Dispersão	display gráfico de variáveis contínuas nos eixos x,y

O pacote `ggplot2` é uma das melhores ferramentas para visualização gráfica (se não a melhor!) e possui a sintaxe ligeiramente diferente do R base, mas não tão trivial.

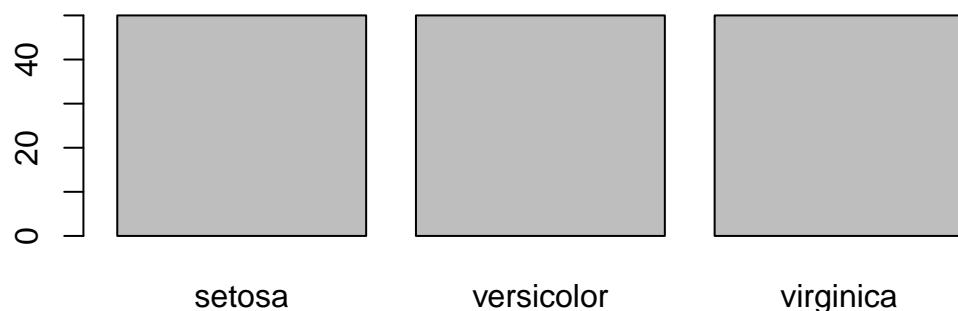
Funções gráficas no R base e no `ggplot2`:

Gráfico	R base	<code>ggplot2()</code>
Barras	<code>barplot()</code>	<code>geom_bar()</code>
Histograma	<code>histogram()</code>	<code>geom_histogram()</code>
Densidade	<code>plot(density())</code>	<code>geom_density()</code>
Quantil-quantil	<code>qqnorm()</code>	<code>geom_qq()</code>
Box-plot	<code>boxplot()</code>	<code>geom_boxplot()</code>
Dispersão	<code>plot()</code>	<code>geom_point()</code>

7.6.3.1 Gráficos de barras

Todas as espécies têm o mesmo número de observações.

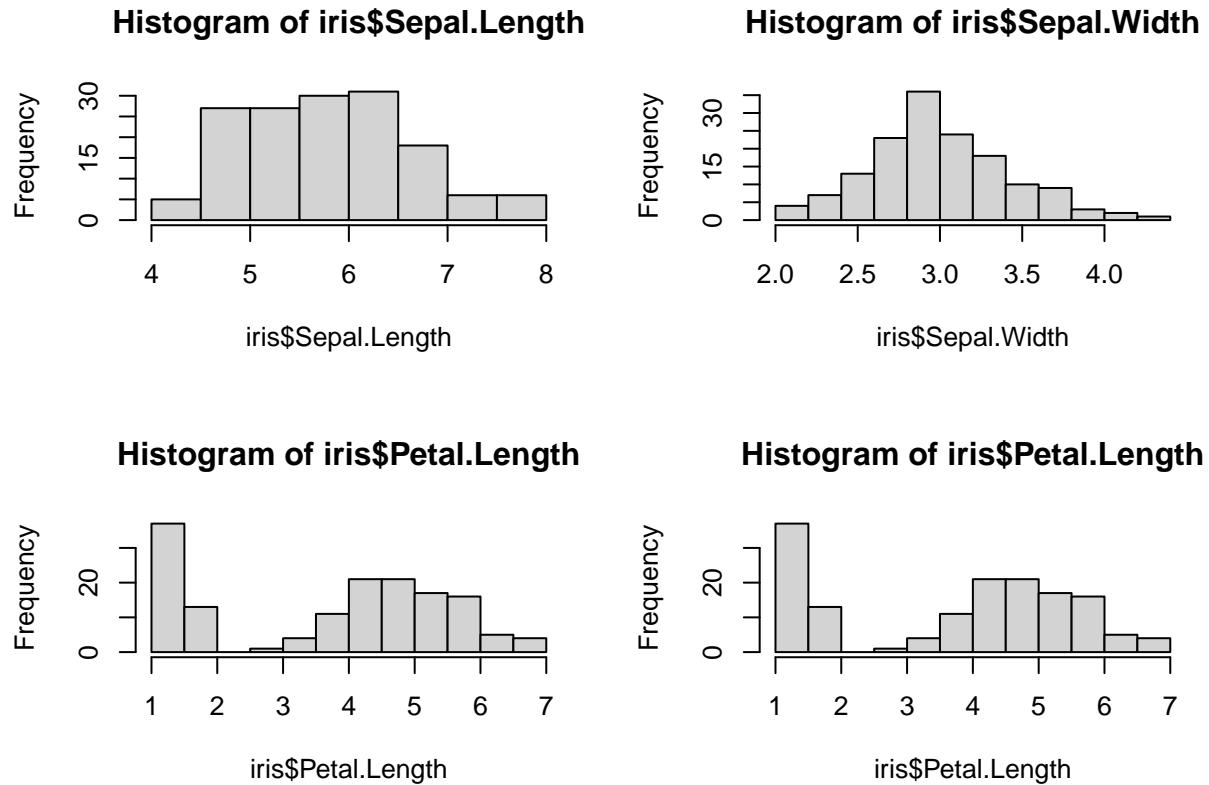
```
barplot(table(iris$Species))
```



7.6.3.2 Histograma

Para a variável `Sepal.Width`, uma distribuição normal parece ser adequada para descrevê-la, mas o mesmo não ocorre com as demais.

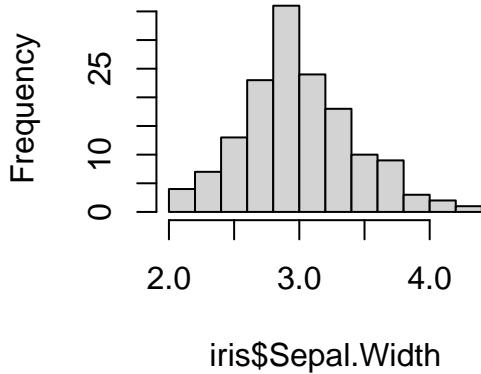
```
par(mfrow=c(2, 2))
hist(iris$Sepal.Length)
hist(iris$Sepal.Width)
hist(iris$Petal.Length)
hist(iris$Petal.Length)
```



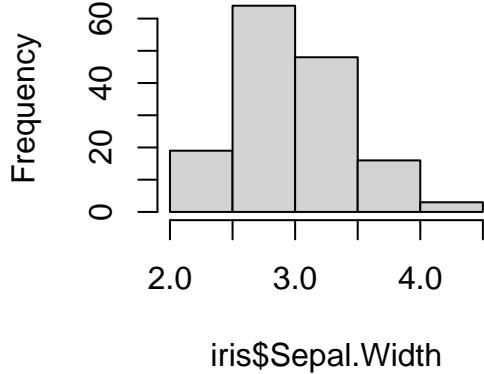
Agora para o comprimento da sépala das espécies de Iris, vamos ver o efeito do número de intervalos no histograma com o argumento `breaks`.

```
par(mfrow=c(1, 2))
hist(iris$Sepal.Width)
hist(iris$Sepal.Width, breaks = 4)
```

Histogram of iris\$Sepal.Width



Histogram of iris\$Sepal.Width

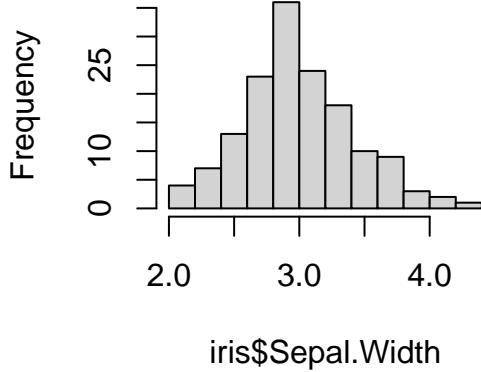


7.6.3.3 Curva de densidade

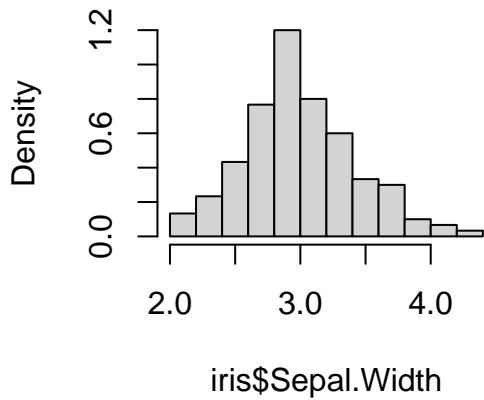
A curva de densidade mostra a probabilidade de observar determinado valor. Em comparação ao histograma, no eixo y, ao invés de termos a frequência, temos a densidade probabilística. Gráficos de densidade são utilizados para se conhecer a forma da distribuição dos dados.

```
par(mfrow=c(1, 2))
hist(iris$Sepal.Width)
hist(iris$Sepal.Width, freq = FALSE)
```

Histogram of iris\$Sepal.Width



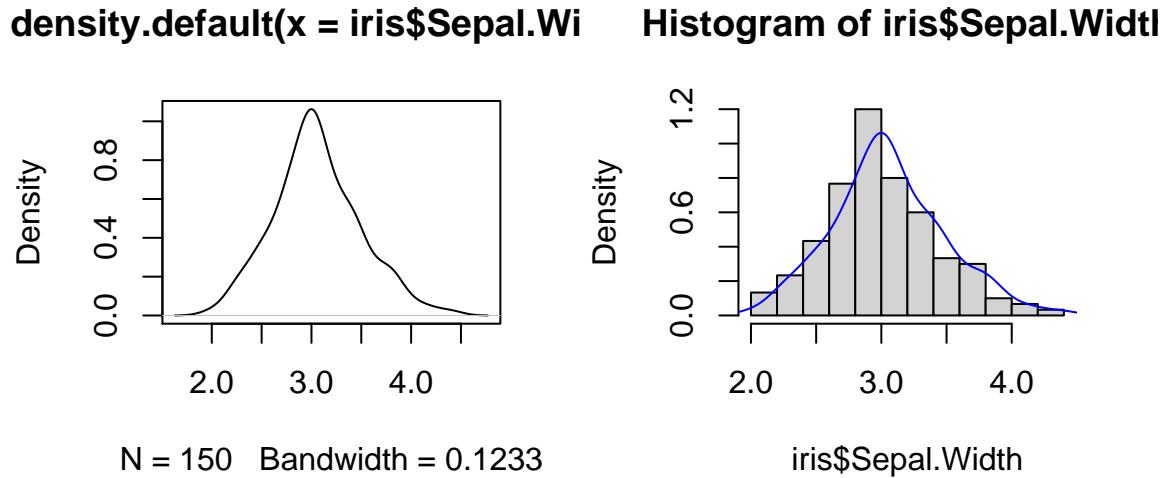
Histogram of iris\$Sepal.Width



No R podemos ver a curva de densidade a usando a função por meio do plot da função `density`.

```
par(mfrow=c(1, 2))
# plot da curva de densidade
plot(density(iris$Sepal.Width))
# plot da curva de densidade sobre o histograma de densidade
hist(iris$Sepal.Width, freq = FALSE)
```

```
lines(density(iris$Sepal.Width), col="blue")
```



7.6.3.4 Box-plot

O boxplot ou diagrama de caixa é uma ferramenta gráfica que permite visualizar a distribuição e valores discrepantes (outliers) dos dados, fornecendo assim um meio complementar para desenvolver uma perspectiva sobre o caráter dos dados. Além disso, o boxplot também é uma disposição gráfica comparativa.

As medidas de estatísticas descritivas como o mínimo, máximo, primeiro quartil, segundo quartil ou mediana e o terceiro quartil formam o boxplot.

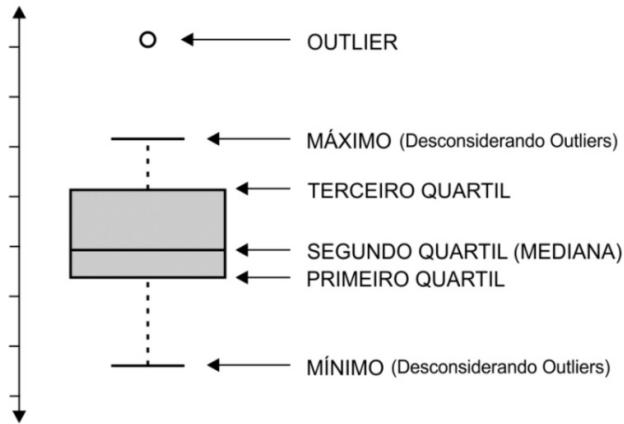


Figure 1: A nice image.

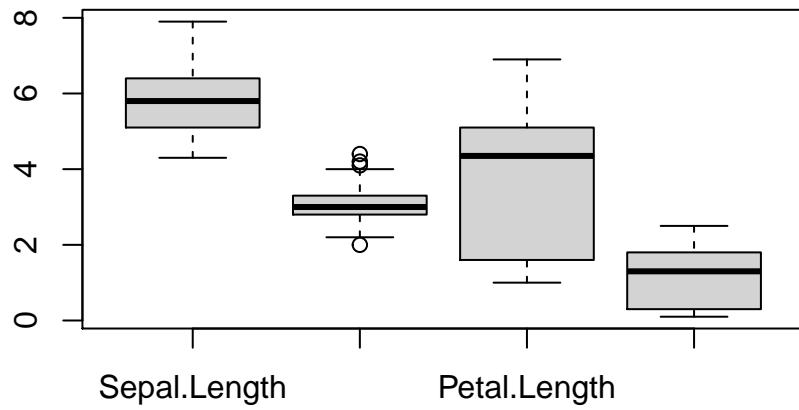
Veja mais em operdata.com.br.

- **Outlier**

Outliers não são um erro (necessariamente, mas podem ser). Outliers representam valores extremos comparados ao restante dos dados. É sempre importante avaliar os outliers para garantir que sejam medidas corretas.

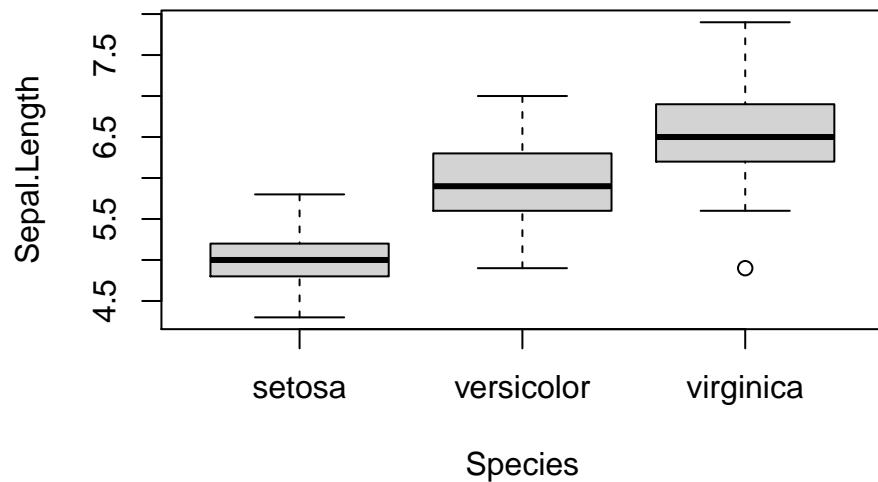
Vamos agora fazer os box-plots das variáveis contidas no objeto iris. Vamos começar com as variáveis gerais.

```
boxplot(iris[-5]) #não inclui coluna 5
```

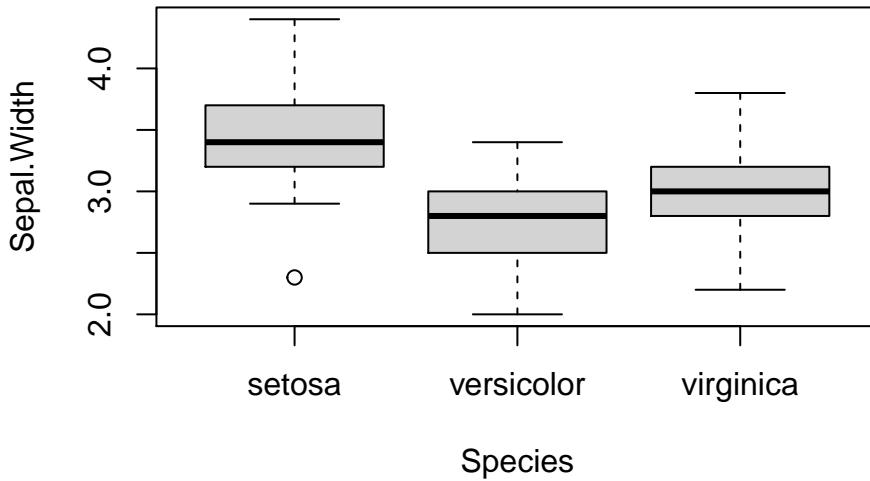


Agora vamos olhar para os valores por espécie.

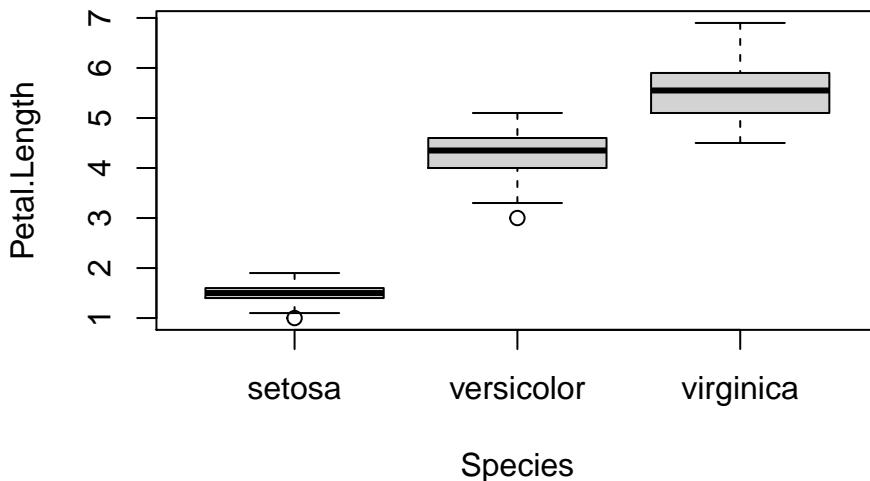
```
boxplot(Sepal.Length ~ Species, data =iris)
```



```
boxplot(Sepal.Width ~ Species, data =iris)
```



```
boxplot(Petal.Length ~ Species, data =iris)
boxplot(Petal.Length ~ Species, data =iris)
```



Você identifica outliers no conjunto de dados? Como podemos checar os outliers? Vamos usar a própria função `boxplot` para identificar os outliers.

```
my_boxplot <- boxplot(iris$Sepal.Width, plot=FALSE)
# o objeto é uma lista e os valores outliers estão guardados no elemento $out da lista
outliers <- my_boxplot$out
#qual a posicao dos outliers
which(iris$Sepal.Width %in% outliers)
```

```

## [1] 16 33 34 61
# vamos usar a posicao para indexar o objeto
iris[which(iris$Sepal.Width %in% outliers), c("Sepal.Width", "Species")]

```

```

##   Sepal.Width   Species
## 16       4.4   setosa
## 33       4.1   setosa
## 34       4.2   setosa
## 61       2.0 versicolor

```

No caso anterior consideramos outliers em relação à distribuição da variável para todas as espécies juntas. É razoável assumir que cada espécie tenha um padrão morfométrico distinto de modo que poderíamos identificar outliers de maneira espécie específica.

```

my_boxplot2 <- boxplot(Sepal.Width ~ Species, data=iris, plot=FALSE)
# o objeto é uma lista e os valores outliers estão guardados no elemento $out da lista
outliers2 <- my_boxplot2$out
# neste caso, queremos apenas os outliers da especie setosa
# vamos usar a posicao para indexar o objeto
iris[iris$Sepal.Width %in% outliers2 &
      iris$Species == "setosa",
      c("Sepal.Width", "Species")]

##   Sepal.Width Species
## 42       2.3   setosa

```

7.7 Entendendo a distribuição dos dados

Para muitas análises estatísticas espera-se que os dados assumam uma distribuição normal. Isto nem sempre é o padrão em dados biológicos (ou na maioria dos dados). O tipo de distribuição do dado vai depender da natureza do dado. Alguns exemplos de distribuições de probabilidade são:

Distribuição	Tipo	$E(X)$	$\sigma^2(X)$	Uso
Normal	contínua	μ	σ^2	curva simétrica
Binomial	discreta	np	$np(-p)$	nº de sucessos em n tentativas
Poisson	discreta	λ	λ	contagem de eventos raros
Log-normal	contínua	$\log(\mu)$	$\log(\sigma)$	curva simétrica

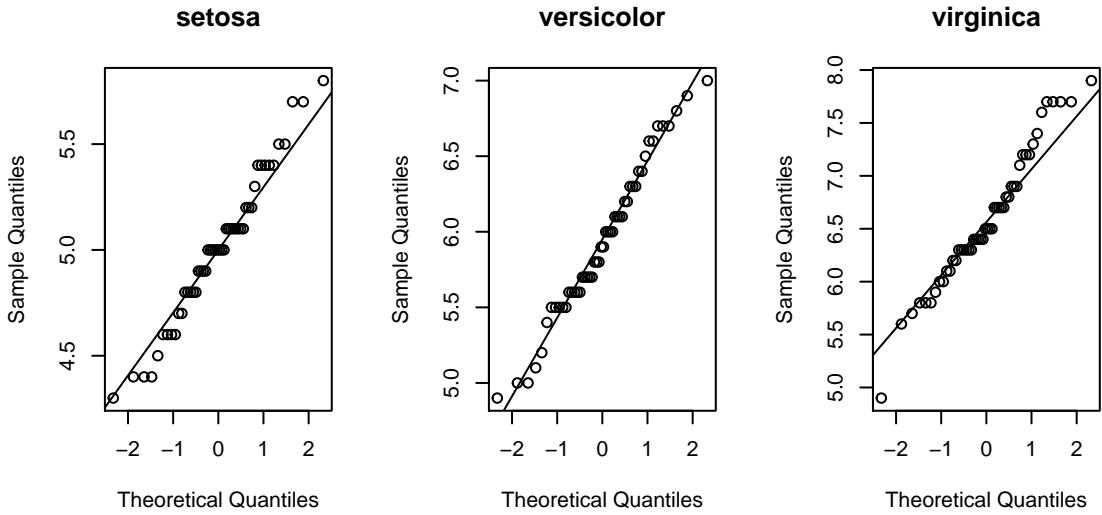
Se os seus dados não seguem uma distribuição normal, isso pode ser tanto porque a natureza dos dados não é normal (gaussiana) ou você não tem um tamanho amostral suficiente. Se os dados não são normais, uma abordagem pode ser transformá-los ou encontrar uma análise apropriada ao tipo de distribuição dos dados (por exemplo, se está fazendo uma regressão linear, pode fazer um modelo linear generalizado adequado à sua distribuição).

Vamos olhar para os dados morfométricos das espécies de Iris e comparar com uma distribuição normal. No R, isto pode ser feito de forma visual com as funções `qqnorm` e `qqline`.

```

par(mfrow = c(1,3))
qqnorm(iris$Sepal.Length[iris$Species == "setosa"], main = "setosa")
qqline(iris$Sepal.Length[iris$Species == "setosa"])
qqnorm(iris$Sepal.Length[iris$Species == "versicolor"], main = "versicolor")
qqline(iris$Sepal.Length[iris$Species == "versicolor"])
qqnorm(iris$Sepal.Length[iris$Species == "virginica"], main = "virginica")
qqline(iris$Sepal.Length[iris$Species == "virginica"])

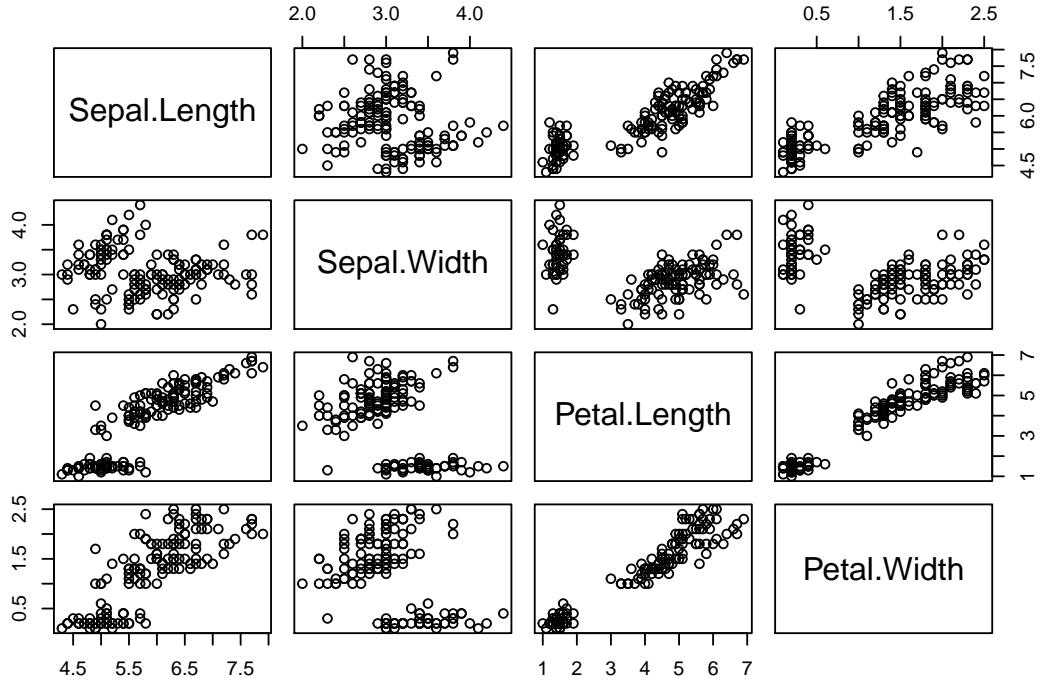
```



7.8 Relação entre variáveis

Uma função no R que nos ajuda a explorar a relação entre muitas variáveis é a `pairs`. O resultado é uma matriz com variáveis em linhas e colunas o gráfico que vemos é o gráfico de dispersão para cada par de variáveis. A diagonal da matriz contém os nomes das variáveis. Note que o gráfico é espelhado de modo que a relação entre tamanho e comprimento de sépala aparece tanto na linha 1 e coluna 2 como na linha 2 e coluna 1.

```
pairs(vars)
```



Referências:

- R for Data Science
- Tutorial de Análise Exploratória de Dados
- Introdução à Ciência de Dados
- R Markdown Cookbook
- Guia de Bolso: ggplot2 | Gráficos elegantes no R
- medium.com/@fernando.gama