Formação Cientista de Dados

Dia 01 - Tarde - Tipos de Dados

Vítor Wilher

Cientista de Dados | Mestre em Economia

Plano de Voo

- Dados Univariados
- Medidas de tendência central
- Medidas de dispersão e formatos de distribuição
- Variância e desvio-padrão
- Dados categóricos
- Dados Bivariados
- Amostras Independentes
- Gráficos de densidade
- Manipulação de Dados no **R**
- Dados Emparelhados
- Dados Multivariados
- Trabalhando com data frames
- Pesquisa de dados

Dados Univariados

O **R** possui cinco classes básicas (ou *atômicas*) de objetos, a saber: *character*, *numeric*, *integer*, *complex* e *logical*. Os objetos no **R**, por sua vez, são divididos em algumas `estruturas de dados`, a saber: *vetores atômicos*, *matrizes*, *arrays*, *listas* e *data frames*. Cada uma dessas estruturas possui características próprias, servindo a determinados tipos de dados. Elas podem, entretanto, ser organizadas, como propõe Wickham [2014], pela *dimensionalidade* e pelo fato de poderem receber dados homogêneos (todos os elementos precisam ser do mesmo tipo) ou heterogêneos, da seguinte forma:

Dados Univariados

	Homogêneos	Heterogêneos
1d	Atomic Vector	List
2d	Matrix	Data frame
nd	Array	

Dados Univariados

O mais básico tipo de objeto no **R** é o `vetor`. Vetores podem ser de dois tipos: `vetores atômicos` ou `listas`. Eles possuem três propriedades comuns:

- Tipo, `typeof()`, o que é;
- Tamanho, `length()`, quantos elementos possui;
- Atributos, `attributes()`, metadados arbitrários adicionais.

Dados Univariados

A diferença entre eles, como visto acima, é que todos os elementos em um vetor atômico precisam ser da mesma classe, enquanto na lista, não necessariamente. Abaixo um exemplo de vetor numérico de tamanho 10 no **R**.

```
x = seq(1:10) # criando uma sequência de 1 a 10
vetor = rnorm(x, 34, 12) # geração aleatório de números
class(vetor) # verificando a classe do objeto
```

[1] "numeric"

```
length(vetor) # verificando o tamanho do objeto
```

[1] 10

```
typeof(vetor) # o tipo do objeto
```

[1] "double"

```
str(vetor) # A estrutura do objeto
```

num [1:10] 19.9 44.7 46.9 43.7 29.1 ...

Dados Univariados

Sobre um vetor, é possível fazer operações de *redução* ou de *vetorização*, como abaixo.

```
sum(vetor)/length(vetor) # Obtendo a média dos dados
```

```
[1] 32.55152
```

```
vetor - mean(vetor) # O quanto cada observação se distancia da média
```

```
[1] -12.638591 12.169858 14.309088 11.172608 -3.420345 -7.737849 [7] 7.566059 -17.709770 -27.107909 23.396851
```

No primeiro caso, estamos fazendo uma operação que irá resultar em único número. Já no segundo, estamos fazendo uma operação com cada elemento do vetor, o que naturalmente irá resultar em um novo vetor.

Dados Univariados

Vamos avançar um pouco no entendimento e manipulação dessa estrutura básica no **R** por meio de um exemplo mais real. Para isso, precisamos instalar o pacote `UsingR`, que acompanha o livro Verzani [2014], nossa referência bibliográfica básica nesse curso. O código abaixo ilustra a instalação e posterior carregamento do pacote.[1]

```
install.packages('UsingR')
require(UsingR)
```

[1] Caso tenha dificuldades nesse processo, dê uma olhada na primeira seção da apostila.

Dados Univariados

Feito isso, podemos acessar todos os `data sets` do pacote com o comando abaixo.

```
data(package='UsingR')
```

Dados Univariados

Escolhemos *brincar* com o *data set* `coldvermont`, que traz a temperatura mínima diária, em *Fahrenheit*, em Woodstock Vermont entre os anos de 1980 e 1985.

```
library(UsingR)
class(coldvermont)
```

```
[1] "ts"
```

Observe que a classe do objeto é `ts`, de série temporal. Podemos, para facilitar a interpretação, passar de graus Fahrenheit para graus Celsius, através da fórmula abaixo.

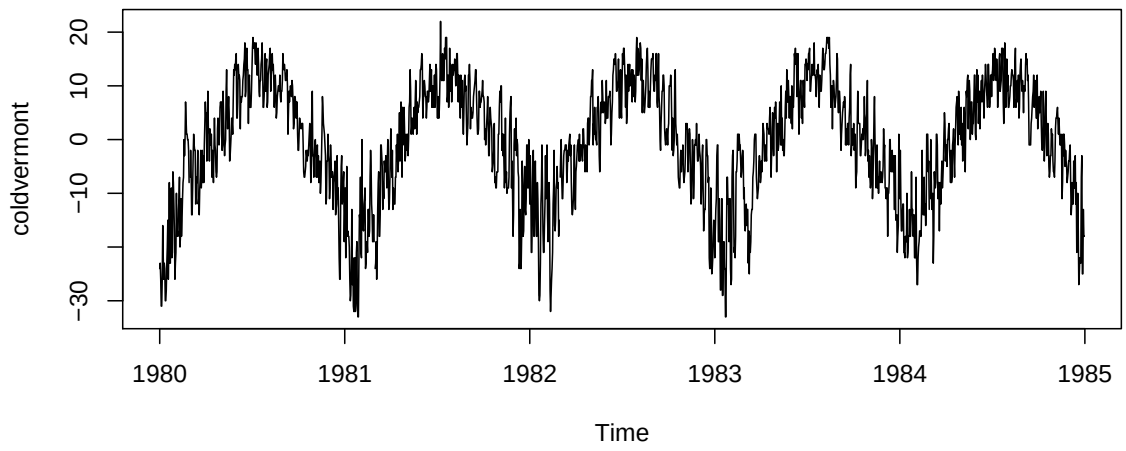
Dados Univariados

```
coldvermont = round((coldvermont-32)/1.8,0)
```

Dados Univariados

E abaixo, plotamos um gráfico rápido do nosso conjunto de dados.

```
plot(coldvermont)
```



Dados Univariados

Observe que as temperaturas mínimas apresentam uma *sazonalidade* ao longo do ano. Isto é, elas aumentam durante a primeira metade e diminuem na segunda metade do ano.

Medidas de tendência central

De modo a compreender melhor o nosso conjunto de dados `coldvermont`, podemos agora fazer uso de algumas estatísticas descritivas, tais como as de **medida central**, **amplitude** e **forma**. Para começar, vamos verificar a média e a mediana do nosso conjunto de dados.[2]

```
mean(coldvermont, na.rm=TRUE)
```

```
## [1] -0.641955
```

```
median(coldvermont, na.rm=TRUE)
```

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

[2] Observe que estamos utilizando o argumento *na.rm*, de modo a ignorar os valores faltantes (*missing values*) dentro da nossa amostra.

Medidas de tendência central

Enquanto a média do nosso conjunto de dados é -0.64, a mediana é 1. A mediana, por suposto, divide a amostra em dois, isto é, valores menores do que ela e valores maiores do que ela. Em termos gerais, podemos dividir a amostra em *quantis*, de forma a estabelecer *medidas de posição*, tais como:

```
quantile(coldvermont, seq(0,1,0.25), na.rm=TRUE)
```

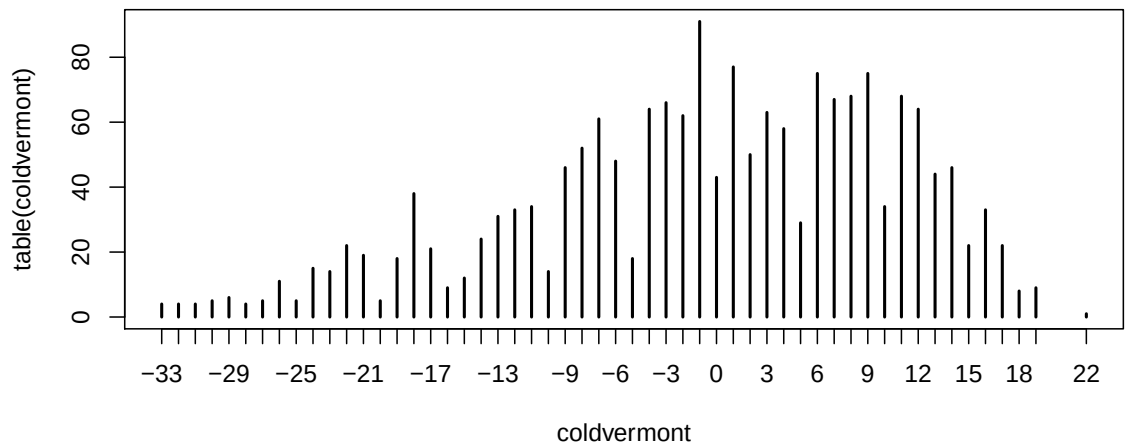
```
##   0%  25%  50%  75% 100%
##  -33   -7    1    8   22
```

Medidas de tendência central

Observe que o primeiro quartil da nossa amostra vai até a temperatura -7 graus Celsius, enquanto o terceiro quartil vai até 8 graus Celsius. Uma outra medida importante para analisarmos o nosso conjunto de dados é a **moda**, isto é, o valor mais frequente que podemos encontrar. Não há uma função direta no R que faz isso, infelizmente. Mas podemos brincar um pouco para descobrir. Podemos usar a função `table` de modo a tabular o número de vezes que cada temperatura ocorre na nossa amostra. Fazemos isso com o código abaixo. Para calculá-lo no R, podemos utilizar o código abaixo.

Medidas de tendência central

```
plot(table(coldvermont))
```



Medidas de tendência central

Fica fácil visualizar que o número -1 é a temperatura mais comum no nosso conjunto de dados. De modo a obter esse número, podemos utilizar a linha de código abaixo.

```
as.numeric(names(which(table(coldvermont)==max(table(coldvermont)))))
```

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

Medidas de dispersão e formatos de distribuição

A amplitude ou variabilidade de um conjunto de dados é uma importante característica. Uma medida simples disso poderia ser o intervalo de um determinado conjunto de dados, que nada mais é do que a distância entre o mínimo e o máximo. Isto é,

```
c(min(coldvermont, na.rm=TRUE), max(coldvermont, na.rm=TRUE))
```

```
## [1] -33  22
```

Medidas de dispersão e formatos de distribuição

Nosso conjunto de temperaturas *varia* assim de -33 a 22, sendo o valor mais comum -1, a média -0.64 e a mediana 1. Pode ser chato, ademais, toda hora ter de aplicar o argumento *na.rm* nas funções. Nesse caso, podemos simplesmente retirar os valores faltantes do nosso conjunto de dados com o comando abaixo.

Medidas de dispersão e formatos de distribuição

```
coldvermont = coldvermont[complete.cases(coldvermont)]
```

Medidas de dispersão e formatos de distribuição

Os dados *sumarizados* acima, podem, ademais, ser obtidos de forma direta com a função *summary*, como abaixo.

```
summary(coldvermont)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -33.000  -7.000   1.000  -0.642   8.000  22.000
```

Medidas de dispersão e formatos de distribuição

A distância entre o menor e o maior valor de um conjunto de dados também pode ser visto de forma direta com a função `range`, como abaixo.

```
range(coldvermont)
```

```
## [1] -33  22
```

```
diff(range(coldvermont))
```

```
## [1] 55
```

Já a diferença entre o primeiro e o terceiro quartil, denominado como *Interquartile Range (IQR)*, pode ser obtido da seguinte forma.

Medidas de dispersão e formatos de distribuição

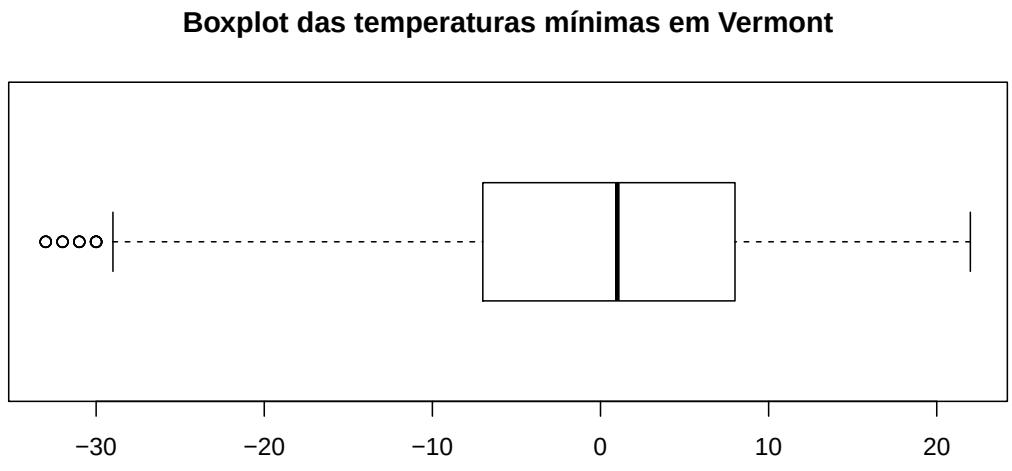
```
IQR(coldvermont)
```

```
## [1] 15
```

Esses cinco valores de um conjunto de dados, ademais, pode ser visualizado de forma direta por meio de *boxplots*. No R, está implementado na função de mesmo nome, como abaixo.

Medidas de dispersão e formatos de distribuição

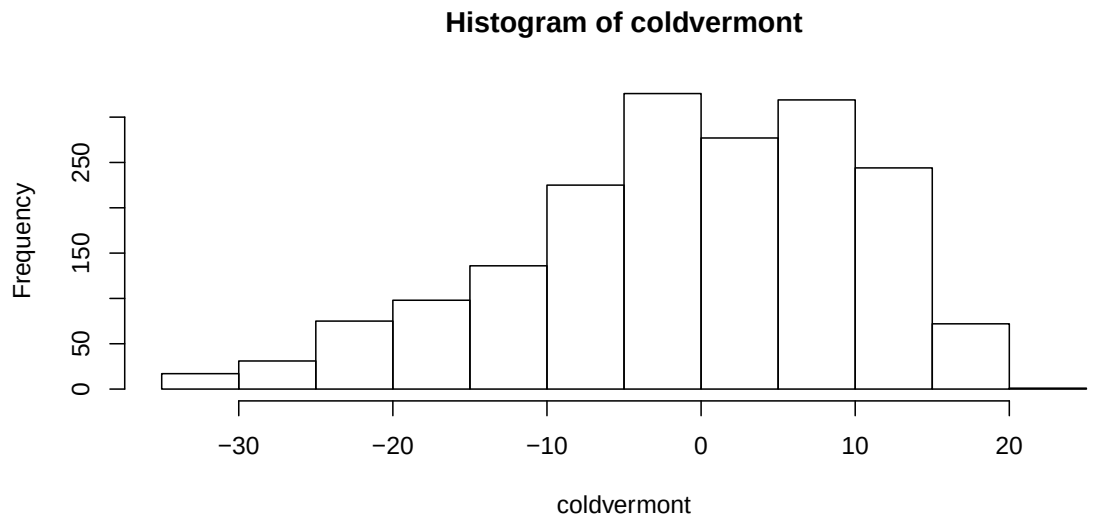
```
boxplot(coldvermont,
        main='Boxplot das temperaturas mínimas em Vermont',
        horizontal=TRUE)
```



Medidas de dispersão e formatos de distribuição

Uma outra forma de visualizar a amplitude do nosso conjunto de dados é através de *histogramas*, como abaixo.

```
hist(coldvermont)
```



Medidas de dispersão e formatos de distribuição

Através do boxplot e do histograma, a propósito, é possível ilustrar a **forma** como as observações de um determinado conjunto de dados se distribui. Elas podem apresentar, basicamente, três formatos: distribuição simétrica, uma assimetria à esquerda e uma assimetria à direita. No caso das temperaturas diárias mínimas de Vermont, vemos claramente uma assimetria à esquerda.[3] É o caso típico quando a média é menor do que a mediana.[4]

[3] Alguns livros se referem a isso como **assimetria negativa**.

[4] Quando a média é maior do que a mediana, ocorre uma *assimetria à direita*. Já quando a média é igual a mediana, ocorre uma simetria.

Variância e desvio-padrão

Por fim, de modo a identificar melhor a variabilidade de um determinado conjunto de dados, podemos calcular a distância entre uma determinada observação e a média do mesmo. De forma a considerar esses desvios independentes do sinal, devemos somar os desvios ao quadrado, controlando pelo tamanho da amostra. Isto é,

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

De modo que, quanto mais espalhado for o conjunto de dados, maior será a sua **variância**. Podemos, ademais, padronizar a unidade de medida através da raiz quadrada da variância, o que dará o **desvio-padrão**. No R, isso é facilmente implementado conforme o código abaixo.

Variância e desvio-padrão

```
var(coldvermont)
```

```
## [1] 122.0113
```

```
sd(coldvermont)
```

```
## [1] 11.04587
```

Quanto maiores forem os valores para a variância, mais amplo será o espalhamento dos dados em relação à média. Já o desvio-padrão indica, em média, quanto cada valor se diferencia da média.

Dados categóricos

Sumarizar dados categóricos é algo bastante direto. A ferramenta básica é **tabular** os valores e apresentá-los, seja de forma gráfica ou descritiva. Para ilustrar, vamos considerar a variável `smoke` no conjunto de dados `babies`, que representa a condição de fumante entre mães.[5]

```
x = babies$smoke
x = factor(x, labels=c('nunca', 'fuma agora', 'até ficar grávida',
                       'já, hoje não', 'desconhecido'))
table(x)
```

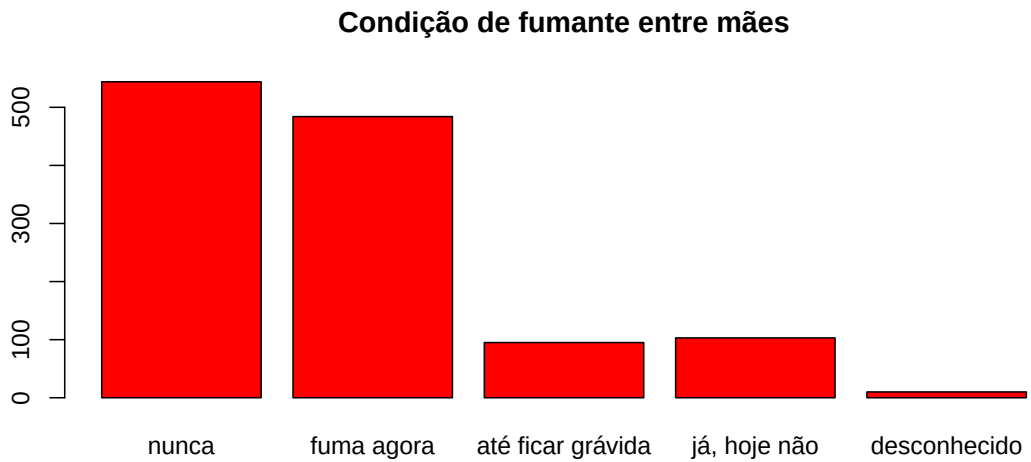
```
## x
##              nunca        fuma agora até ficar grávida      já, hoje não
##                544                484                 95                103
##      desconhecido
##                 10
```

[5] Os códigos utilizados estão disponíveis na ajuda do conjunto de dados.

Dados categóricos

Como podemos ver, a função `table` relaciona cada categoria à frequência encontrada. Podemos visualizar isso através de um gráfico de barras, como abaixo.

```
barplot(table(x),
        col='red', main='Condição de fumante entre mães',
        horiz = FALSE)
```



Dados Bivariados

Nessa seção, veremos os conjuntos de dados bivariados, dados envolvendo duas variáveis. Quando estamos tratando desse tipo de conjunto, nós podemos indagar se há relação entre as variáveis analisadas. Para *amostras independentes*, nós podemos nos perguntar se diversas métricas, como medidas centrais, de variabilidade ou forma são similares ou diferentes. Já para *dados emparelhados*, podemos verificar a existência ou ausência de relacionamento entre as variáveis.

Amostras Independentes

Um experimento estatístico bastante comum consiste em separar uma determinada coorte em grupos de controle e de tratamento. Enquanto o grupo de tratamento recebe algum tipo de tratamento, o de controle não recebe nada. Depois, medidas para cada um dos grupos são tomadas. De modo a controlar o viés ao separar a coorte, uma alocação aleatória costuma ser utilizada. Esse tipo de procedimento leva a independência - onde o conhecimento sobre a performance de um determinado participante não sugere nada sobre outro. Ademais, para controlar o *efeito placebo*, o grupo de controle geralmente recebe algum tratamento, mas que não é esperado ter qualquer efeito.

Amostras Independentes

Por exemplo, suponha que estamos investigando quais alimentos podem ter impacto sobre a performance esportiva. O grupo de tratamento foi alimentado com $\frac{3}{2}$ copos de beterraba 75 minutos antes de uma atividade física, enquanto o grupo de controle não foi. A atividade física foi uma corrida cronometrada. O pesquisador acredita que o nitrato encontrado na beterraba ampliaria os vasos sanguíneos do indivíduo, aumentando o fluxo sanguíneo nos músculos exercicitados, diminuindo o tempo de duração da atividade. Suponha que as medidas em minutos das atividades sejam dadas por:

`Beterrabas`	41	40	41	42	44	35	41	36	47	45
`Sem Beterrabas`	51	51	50	42	40	31	43	45		

Amostras Independentes

O que podemos dizer sobre esses dados? Os três maiores tempos estão no grupo *sem beterrabas*, mas este também possui o menor tempo. Algumas questões pertinentes:

- Os dados parecem ter sido extraídos da mesma população, os dois possuem a mesma forma?
- Os dados parecem ter as mesmas medidas centrais e de variabilidade?

Amostras Independentes

Vamos para o **R** responder essas questões...

```
beets = c(41,40,41,42,44,35,41,36,47,45)
nobeets = c(51,51,50,42,40,31,43,45)

c(media_beets=mean(beets), media_nobeets=mean(nobeets),
  sd_beets=sd(beets), sd_nobeets=sd(nobeets))
```

```
##   media_beets media_nobeets      sd_beets    sd_nobeets
##     41.200000     44.125000      3.705851      6.812541
```

Amostras Independentes

O grupo que recebeu as beterrabas possui uma média de 3 minutos a menos do que o grupo de controle, com uma variabilidade menor. Se essa diferença se dá por termos corredores mais rápidos no grupo de tratamento ou se é resultado da ingestão das beterrabas, nós não sabemos.

Amostras Independentes

Assim como na análise univariada, pode ser importante visualizar os dados.

```
library(aplpack)
stem.leaf.backback(beets, nobeets, rule.line='Sturges')
```

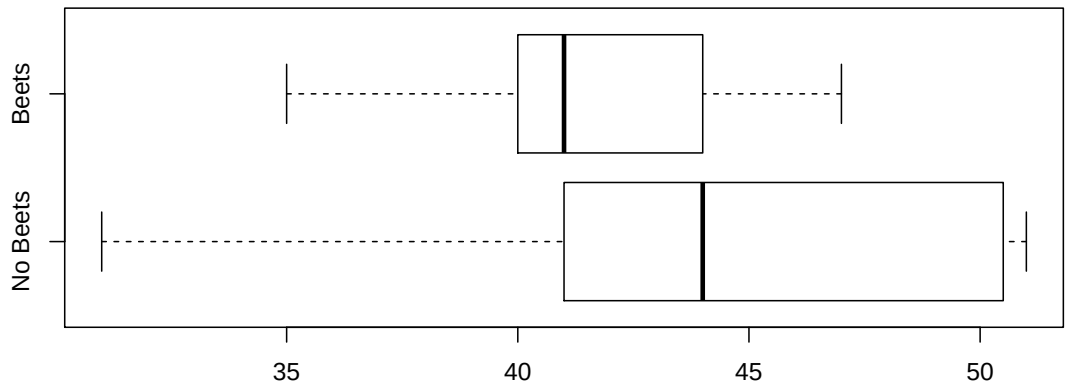
```
## ------------------------------
##   1 | 2: represents 12, leaf unit: 1
##         beets      nobeets
## ------------------------------
##               | 3* |1        1
##    2       65| 3. |
##   (6)  421110| 4* |023      4
##    2       75| 4. |5       (1)
##               | 5* |011      3
##               | 5. |
##               | 6* |
## ------------------------------
## n:         10      8
## ------------------------------
```

Amostras Independentes

Os lados mais à esquerda e à direita registram a posição da observação no conjunto de dados, as hastes ficam no meio. Após focarmos no meio, percebe-se que o grupo de controle (sem beterrabas) possui maior variabilidade, mas a diferença no centro não é tão óbvia. De forma a complementar a análise, colocamos abaixo um `boxplot`. Observamos medianas diferentes, mas ambas estão dentro da amplitude da outra. A ideia de olhar as diferenças no centro de uma escala determinada pela amplitude é a chave para a inferência estatística.

Amostras Independentes

```
boxplot(nobeets, beets, names=c('No Beets', 'Beets'),
        horizontal = TRUE)
```

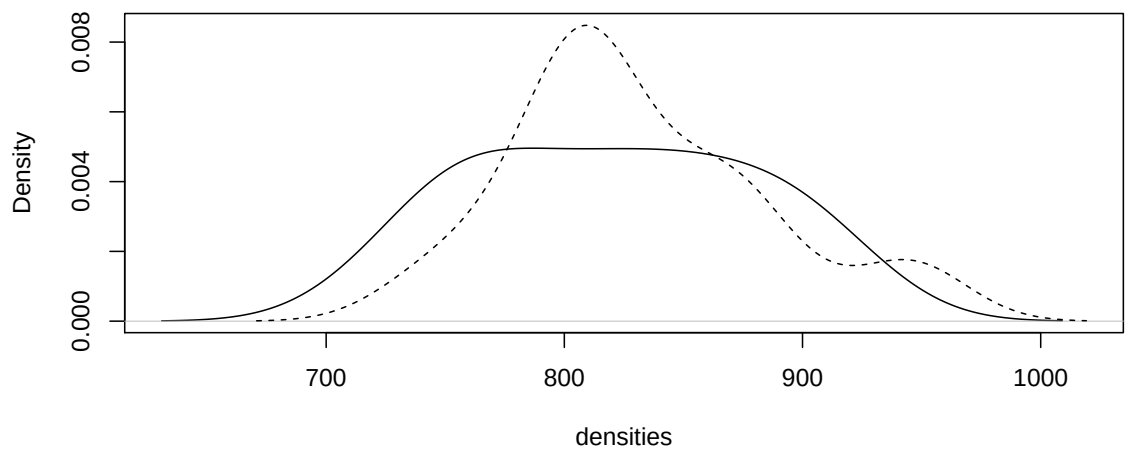


Gráficos de densidade

Gráficos de densidade são semelhantes aos `histogramas` que vimos na seção anterior, mas difere por permitir que mostremos dois ou mais conjuntos de dados. Vamos considerar o conjunto de dados `michelson` do pacote `MASS`, que contém medidas de velocidade da luz no ar produzidas por Michelson em 1879. O conjunto de dados contém cinco experimentos. Nós comparamos o quarto e quinto abaixo.

Gráficos de densidade

```
library(MASS)
speed <- michelson$Speed; expt <- michelson$Expt
fourth <- speed[expt==4]
fifth <- speed[expt==5]
d_4 <- density(fourth)
d_5 <- density(fifth)
xrange <- range(c(d_4$x, d_5$x))
yrange <- range(c(d_4$y, d_5$y))
plot(d_4, xlim=xrange, ylim=yrange, xlab='densities', main='')
lines(d_5, lty=2)
```



Manipulação de Dados no **R**

O **R** possui duas estruturas que podem simplificar o trabalho com dados bivariados: listas e data frames. As listas têm uma vantagem sobre vetores atômicos, como vimos, por permitir a alocação de dados com categorias diferentes. Podemos construí-las com o código abaixo.

```
beets = c(41,40,41,42,44,35,41,36,47,45)
nobeets = c(51,51,50,42,40,31,43,45)

b = list(beets=beets, 'no beets'=nobeets)
b
```

```
$beets [1] 41 40 41 42 44 35 41 36 47 45
$no beets [1] 51 51 50 42 40 31 43 45
```

Manipulação de Dados no **R**

As listas, a propósito, podem ser subdivididas.

```
b$beets
```

```
##  [1] 41 40 41 42 44 35 41 36 47 45
```

```
b$`no beets`
```

```
## [1] 51 51 50 42 40 31 43 45
```

```
b[1]
```

```
## $beets
##  [1] 41 40 41 42 44 35 41 36 47 45
```

```
b[2]
```

```
## $`no beets`
## [1] 51 51 50 42 40 31 43 45
```

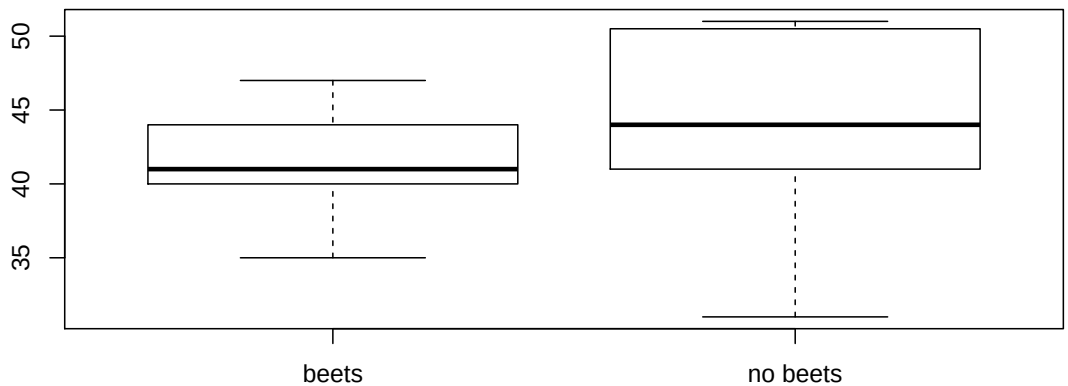
```
b[[1]][1]
```

```
## [1] 41
```

Manipulação de Dados no **R**

E facilitam a construção do *boxplot*.

```
boxplot(b)
```



Manipulação de Dados no **R**

`Data frames`, por sua vez, são uma forma muito utilizada para armazenar variáveis, de forma que as colunas representam variáveis e as observações ficam nas linhas. Abaixo um exemplo.

```
student = c('Alice', 'Bob')
grade = c('A', 'B')
attendance = c('awesome', 'bad')
data = data.frame(student, grade, attendance)
data
```

```
##   student grade attendance
## 1   Alice     A    awesome
## 2     Bob     B        bad
```

Dados Emparelhados

Considere o conjunto de dados `fat` do pacote `UsingR` que contém medidas de diferentes dimensões do corpo de uma coorte de 252 homens. O objetivo desse conjunto de dados foi o de verificar se alguma previsão do índice de gordura corporal pode ser feita a partir de algumas variáveis que podem ser obtidas a partir de uma fita métrica. Para esse exemplo particular, estamos interessados apenas nas relações entre as variáveis.

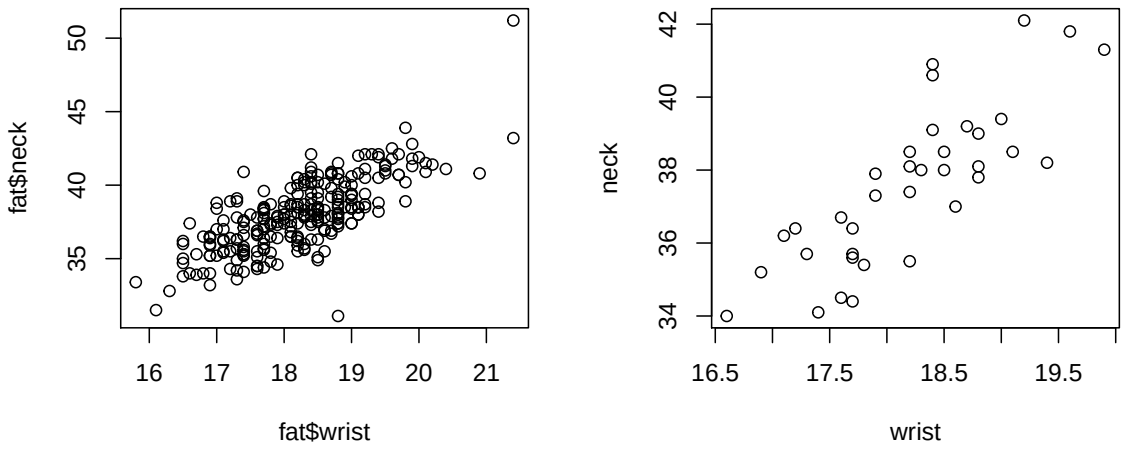
```
library(UsingR)
names(fat)
```

```
##  [1] "case"          "body.fat"      "body.fat.siri" "density"
##  [5] "age"           "weight"        "height"        "BMI"
##  [9] "ffweight"      "neck"          "chest"         "abdomen"
## [13] "hip"           "thigh"         "knee"          "ankle"
## [17] "bicep"         "forearm"       "wrist"
```

Dados Emparelhados

E abaixo um primeiro `gráfico de correlação` entre as medidas do pulso (*wrist*) e do pescoço (*neck*).

```
par(mfrow=c(1,2))
plot(fat$wrist, fat$neck)
plot(neck~wrist, data=fat, subset= 20 <= age & age < 30)
```



Dados Emparelhados

A correlação é um sumário numérico que diz o quão perto estão relacionadas as medidas de duas variáveis numéricas quando elas estão uma relação linear. O *coeficiente de correlação de Pearson* pode ser definido como

$$cor(x, y) = \frac{1}{n-1} \sum \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right) \tag{2}$$

de modo que o número permaneça no intervalo entre 1 e -1, representando, respectivamente, perfeita correlação positiva e negativa. Se o número for 0, por outro lado, isso significa que as variáveis são perfeitamente não correlacionadas.

Dados Multivariados

Um processo de análise de dados típico consiste em algumas etapas: tratamento dos dados, transformação dos dados, visualização exploratória, sumários numéricos e modelagem. Aprender de modo eficiente a como trabalhar no R com `data frames` torna essas etapas muito mais diretas e gerenciáveis. Nessa seção, aprenderemos sobre como trabalhar com `data frames`.

Trabalhando com data frames

`Data frames` se situam na fronteira entre matrizes e listas de modo que nós temos duas interfaces para trabalhar dentro da mesma estrutura. Ademais, é possível utilizar determinadas funções para tornar o gerenciamento dessa estrutura de dados ainda mais conveniente. Vamos organizar nossa discussão em alguns aspectos, como se segue.

Trabalhando com data frames

Data frames são listas de variáveis. Eles podem ser construídos como se segue.

```
df = data.frame(nm1=vec1, nm2=vec2)
```

Trabalhando com data frames

A combinação `key=value` equivale às colunas, enquanto as linhas são criadas automaticamente. Observe, por suposto, que os vetores precisam ter o mesmo tamanho para estarem no mesmo data frame. Com a função `as.data.frame` é possível ainda forçar uma estrutura a ser um data frame.

Pesquisa de dados

É possível acesssar as variáveis de um data frame através da notação $, da seguinte forma:

```
data(mtcars)
mtcars$mpg
```

```
##  [1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2
## [15] 10.4 10.4 14.7 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4
## [29] 15.8 19.7 15.0 21.4
```

Pesquisa de dados

É possível ainda pesquisar dados como em matrizes, da seguinte forma:

```
mtcars[,1]
```

```
##  [1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2
## [15] 10.4 10.4 14.7 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4
## [29] 15.8 19.7 15.0 21.4
```

Pesquisa de dados

Ou fazer operações dentro do mesmo data frame com a função `with`:

```
with(mtcars, mean(mpg) - sd(mpg))
```

```
## [1] 14.06368
```

J. Verzani. *Using R for Introductory Statistics*. CRC Press, 2014.

H. Wickham. *Advanced R*. CRC Press, 2014.