



# Introdução à Análise de Dados com Linguagem R

## Módulo 1

Analista Ambiental Robson Cruz

## Contents

<b>1</b>	<b>Introdução</b>	<b>3</b>
<b>2</b>	<b>Instalação do R</b>	<b>3</b>
<b>3</b>	<b>Instalação do RStudio</b>	<b>3</b>
3.1	Conhecendo a Interface Gráfica do RStudio . . . . .	4
3.2	Obter e Configurar o ambiente de trabalho . . . . .	4
<b>4</b>	<b>Operadores Aritméticos e de Atribuição em R</b>	<b>5</b>
4.1	Exercício Prático - Operadores Aritméticos e de Atribuição . . . . .	5
<b>5</b>	<b>Operadores de Comparação em R</b>	<b>5</b>
<b>6</b>	<b>Objeto em R</b>	<b>5</b>
6.1	Conferindo o conteúdo de um objeto . . . . .	6
6.2	Descobrindo o tipo de dados armazenado em um objeto R . . . . .	6
<b>7</b>	<b>Estrutura de Dados em R</b>	<b>6</b>
7.1	Vetores . . . . .	6
7.2	Funções Estatísticas Aplicadas a Vetores . . . . .	9
<b>8</b>	<b>Como Saber se Há Valores Ausentes (NA) nos Dados?</b>	<b>12</b>
<b>9</b>	<b>Exercício Prático - Vetores e Operadores de Comparação</b>	<b>12</b>
<b>10</b>	<b>Testes Lógicos com Vetores</b>	<b>13</b>
10.1	Exercício Prático - Índice de Vetor . . . . .	13
10.2	Visualização Gráfica de Vetores . . . . .	13
10.3	Modificar a Aparência dos Gráficos . . . . .	17
<b>11</b>	<b>Matriz</b>	<b>28</b>
11.1	Sumar linhas e Colunas de uma Matriz . . . . .	28
11.2	Sumar os Elementos da Diagonal de uma Matriz . . . . .	29
11.3	Sentido de Preenchimento dos Dados em uma Matriz . . . . .	29
11.4	Atribuir Nomes as Linhas e Colunas de uma Matriz . . . . .	29
11.5	Obter os nomes das Linhas e Colunas de uma Matriz . . . . .	30

11.6 Acessar Linhas e Colunas da Matriz . . . . .	30
11.7 Acessar Elementos da Matriz . . . . .	30
11.8 Alterar os Elementos de uma Matriz . . . . .	30
11.9 Operações com Matrizes . . . . .	31
11.10 Combinar Vetores em Matriz . . . . .	32
11.11 Matriz de Correlação . . . . .	33
11.12 Atribuir um Atributo a uma Matriz . . . . .	34
11.13 Gerar Gráficos a partir dos Dados de uma Matriz . . . . .	35
<b>12 Array</b>	<b>36</b>
12.1 Acessar Elementos do Array . . . . .	37
12.2 Operações com Arrays . . . . .	37
12.3 Atribuir Nomes as Dimensões do Array . . . . .	37
12.4 Inserir Atributo em um Array . . . . .	38
<b>13 Data Frames</b>	<b>38</b>
13.1 Criar um data frame . . . . .	38
13.2 Carregar um data frame a partir de um arquivo . . . . .	39
13.3 Leitura de Arquivos .csv e .txt . . . . .	40
13.4 Obter os nomes das colunas de um Dataframe . . . . .	41
13.5 Obter número de linhas e colunas de uma Dataframe . . . . .	41
13.6 Obter a Estrutura dos Dados de um DataFrame . . . . .	42
13.7 Resumo dos Dados de um Dataframe . . . . .	42
13.8 Acessar colunas do data frame . . . . .	43
13.9 Filtrar Linhas e/ou Colunas de uma Data Frame . . . . .	43
13.10 Filtrar dados com a função <code>subset</code> . . . . .	50
13.11 Criar uma nova coluna . . . . .	50
13.12 Excluir Linhas e Colunas de um Data Frame . . . . .	51
13.13 Ler Dados de Arquivo Demilitado por Tabulação . . . . .	52
13.14 Ler Dados de Arquivo Delimitado por Espaço . . . . .	52
13.15 Ler Dados de Arquivo Hospedado na Internet . . . . .	52
13.16 Leitura de Dados em Formatos de Arquivos de Softwares Proprietários . . . . .	52
13.17 Ler Dados de Arquivos .xls e .xlsx . . . . .	52
<b>14 Salvar Data Frame</b>	<b>53</b>
14.1 .csv . . . . .	53
14.2 .txt . . . . .	54
<b>15 2. Fatores</b>	<b>54</b>
15.1 Converter colunas de uma data frame para <code>factor</code> . . . . .	56
15.2 Atribuir Rótulos ao Fator . . . . .	56
15.3 Fatores - Lista Espécies Fauna Ameaçada de Extinção . . . . .	58
<b>16 Listas</b>	<b>59</b>
16.1 Acessando elementos em uma lista . . . . .	60
16.2 Nomear os elementos de uma lista . . . . .	61
16.3 Funções Aplicadas à lista . . . . .	61
16.4 Converter Vetor para Lista . . . . .	62
16.5 Conhecendo a Versatilidade de uma lista . . . . .	64
<b>17 Aplicação do uso de lista em R</b>	<b>66</b>

# **1 Introdução**

R é uma linguagem de programação gratuita de código aberto usada principalmente para computação estatística e gráficos. R é uma linguagem interpretada semelhante a Python, onde você não precisa compilar primeiro para executar seu programa. Depois de criar seu programa, você pode executá-lo em uma ampla variedade de plataformas UNIX, Windows e MacOS.

R é uma linguagem específica de domínio de código aberto, explicitamente projetada para ciência de dados. Muito popular em finanças e academia, R é uma linguagem perfeita para manipulação, processamento e visualização de dados, bem como computação estatística e aprendizado de máquina.

Como qualquer outra linguagem de programação, R também suporta extensão na forma de pacotes, portanto, os desenvolvedores podem criar seus próprios pacotes e reutilizá-los quando necessário.

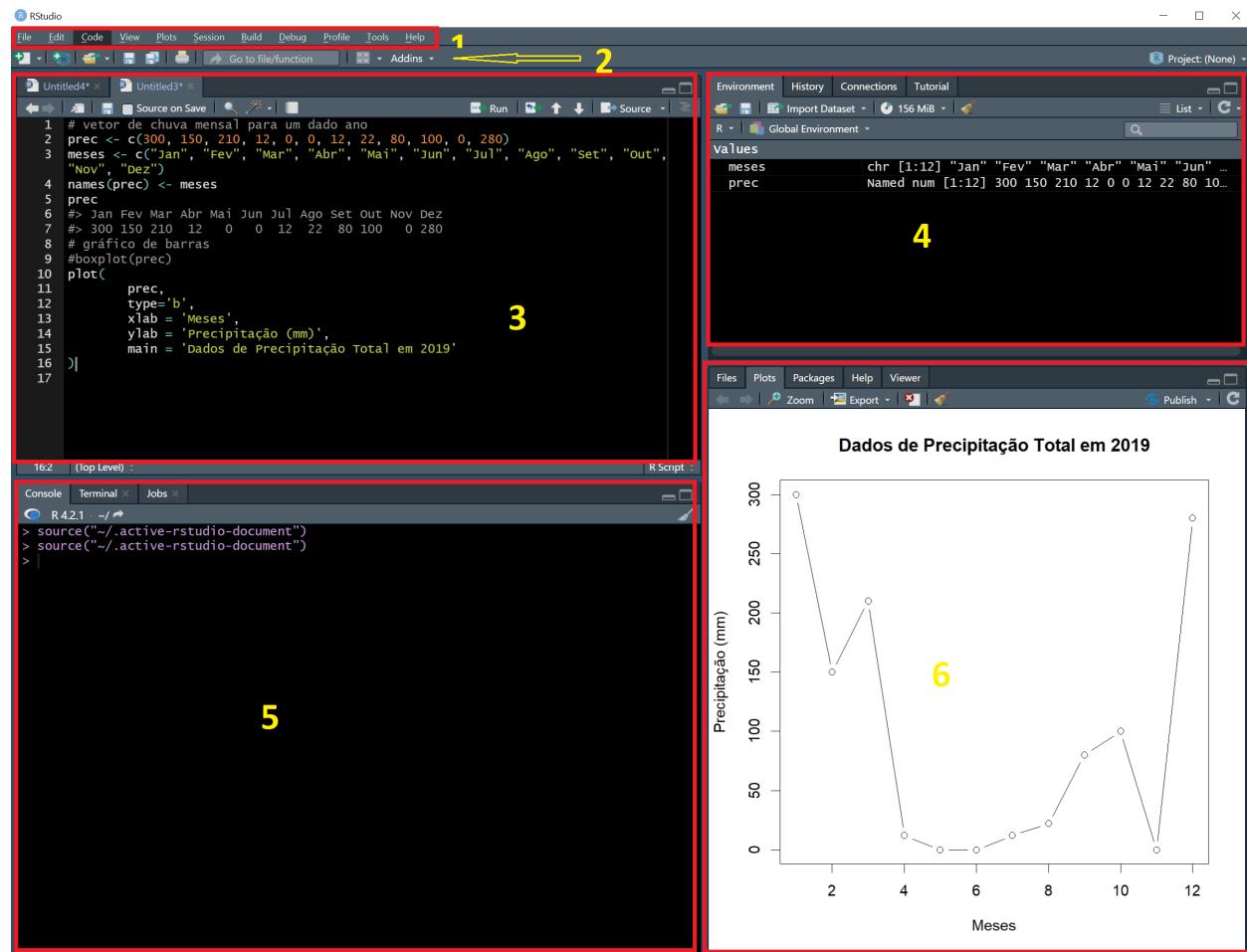
# **2 Instalação do R**

A instalação padrão da linguagem R é feita a partir do CRAN, uma rede formada de servidores espalhados pelo mundo que armazena versões atualizadas do código fonte e executável (Windows), assim como a documentação da linguagem R.

# **3 Instalação do RStudio**

RStudio é um ambiente de desenvolvimento integrado para linguagem R (IDE - integrated development environment), porém pode rodar scripts SQL, C, C++ e Python. A vantagem se é poder trabalhar com IDE é que ela disponibiliza ferramentas de apoio ao desenvolvimento de códigos em linguagem de programação. Para download do RStudio acesse o endereço <https://www.rstudio.com/>, e escolha a opção RStudio Desktop

### 3.1 Conhecendo a Interface Gráfica do RStudio



A interface do RStudio é dividida em 4 painéis, e duas barras: \* 1 - Barra de menu \* 2 - Barra de ferramentas \* 3 - Painel de scripts e arquivos \* 4 - Painel de variáveis de Ambiente/Histórico/Conexões/Tutorial \* 5 - Painel de Console/Terminal \* 6 - Painel de Árvore de Arquivos/Gráficos/Pacotes/Ajuda/Visualizador

### 3.2 Obter e Configurar o ambiente de trabalho

Para obter o ambiente de trabalho atualmente em uso pelo RStudio utilizamos a função `getwd()`; esta função vem do termo em inglês: Get Working Directory, traduzido para o português como “Obter Ambiente de Trabalho”. No sistema operacional Windows, por padrão, o RStudio configura o ambiente de trabalho em “C:/Usuários/Nome\_do\_Usuário/Documents”

Para configurar um ambiente de trabalho diferente do padrão utilizamos a função `setwd()`. Esta função tem nome bem sugestivo na língua inglesa, a saber: Set Working Directory, “Configurar Ambiente de Trabalho”.

#### 3.2.1 Exercício Prático - Ambiente de trabalho

**Instrução 1/2** \* Use a janela console do RStudio para obter o atual diretório de trabalho em uso no seu computador.

**Instrução 2/2** \* Configure seu ambiente de trabalho para 'C:/Users/Seu\_nome\_de\_usuario/Downloads'

*Atenção: Em seu nome\_de\_usuario insira seu nome de usuário.*

## 4 Operadores Aritméticos e de Atribuição em R

Operador	Função
+	Soma
-	Subtração
/	Divisão
*	Multiplicação
%%	Resto da divisão
%/%	Parte inteira divisão
^	Potenciação
**	Potenciação
<-	Atribuição
=	Atribuição

### 4.1 Exercício Prático - Operadores Aritméticos e de Atribuição

**Instrução 1/3** \* Obter o resto da divisão entre os números inteiros 10 e 3.

**Instrução 2/3** \* Obter a parte interira da divisão de 10 por 3.

**Instrução 3/3** \* Obter o quadrado de um número inteiro qualquer.

## 5 Operadores de Comparação em R

Operador	Significado
==	igual a
!=	diferente de
>	maior que
<	menor que
>=	maior ou igual a
<=	menor ou igual a

## 6 Objeto em R

Um objeto é simplesmente qualquer variável que armazena um caractere numérico, alfabético ou uma cadeia de caracteres (*string*). Em R temos objetos especiais para manipulação de grande volume de dados, a exemplo de vetores, listas e dataframes. Para criação de um objeto se utiliza o operador de atribuição <- . Nos exemplos a seguir são criados os objetos *x*, *y* e *z* para armazenar um número inteiro, uma letra e uma frase.

```
# definição de um objeto do tipo inteiro  
int <- 42
```

```
# definição de um objeto do tipo caractere
```

```

letra <- "R"

# # definição de um objeto do tipo string (cadeia de caracteres)
string <- "R é massa"

```

Em ambos os trechos de código acima se observa o uso do caractere `#`, em R e python este caractere é utilizado para fazer comentário no código, portanto, toda linha que contém este caractere é ignorada na execução do programa.

A seguir é mostrado como fazer comentário com várias linhas.

```

"
Este é um comentário em R utilizando
várias linhas para documentação de
códigos.
"

```

```
## [1] "\nEste é um comentário em R utilizando \nvárias linhas para documentação de\nncódigos.\n"
```

## 6.1 Conferindo o conteúdo de um objeto

Para conferir o conteúdo de um objeto em R, fazemos uma chamada diretamente pelo nome do objeto de interesse ou através do uso da função `print()`

```

int

## [1] 42
print(int)

## [1] 42
cat(int)

## 42
42

```

## 6.2 Descobrindo o tipo de dados armazenado em um objeto R

Para descobrir o tipo de dados armazenado em um objeto, podemos utilizar a função `class`

```

class(int)

## [1] "numeric"
class(letra)

## [1] "character"
class(string)

## [1] "character"

```

# 7 Estrutura de Dados em R

## 7.1 Vetores

Vetor em R é um objeto R que armazena um ou mais elementos de valores indexados, ou seja, cada elemento dentro do vetor possui uma posição específica. Para criação de um vetor basta colocar os valores dentro de

`c()`. Vetor é uma estrutura de dados especialmente importante em análise de dados. A seguir temos um exemplo de como criar um vetor de inteiros.

```
inteiros <- c(42, 33, 0, -1, 5)
```

Para acessar o primeiro elemento do vetor `inteiros` usamos o comando `vetor[x]`, onde *vetor* é nome atribuído ao vetor e *x* é o índice do elemento a ser buscado no vetor. Se quisermos mostrar apenas o elemento de índice 1 do vetor `inteiros`, ou seja: filtrar apenas o primeiro elemento, basta usar `inteiros[1]`

```
# acessar o primeiro elemento de um vetor  
inteiros[1]
```

```
## [1] 42
```

Ao usar um índice negativo para filtrar um vetor estamos pedindo que a saída dos daos do vetor não mostre o elemte relativo ao índice negativo. No código a seguir vamos mostrar todos os elementos do vetor `inteiros`, exceto o quarto elemento (elemento de índice 4).

```
# mostrar todos os elementos do vetor com exceção do elemento de índice 4  
inteiros[-4]
```

```
## [1] 42 33 0 5
```

### 7.1.1 Substituição de elementos de um vetor

```
# Substituir o primeiro elemento do vetor "inteiros" por 2  
inteiros[1] <- 2  
inteiros
```

```
## [1] 2 33 0 -1 5
```

### 7.1.2 Funções básicas aplicadas a vetores

- `length()` Retorna o tamanho de um vetor, ou seja, o número de elementos armazenados no vetor.

```
length(inteiros)
```

```
## [1] 5
```

```
# Defini um vetor de números inteiros chamado "dias_semana"  
dias_semana <- c(1:7)
```

```
# Mostrar na tela do console os dados do vetor  
dias_semana
```

#### 7.1.2.1 `names()` Retorna os nomes atribuídos a cada elemento de um vetor

```
## [1] 1 2 3 4 5 6 7
```

```
# Checar se o vetor possui nomes associados aos seus elementos  
names(dias_semana)
```

```
## NULL
```

```
# Atribuir nomes aos elementos do vetor utilizando a função names()  
names(dias_semana) <- c('Domingo', 'Segunda', 'Terça', 'Quarta',  
                         'Quinta', 'Sexta', 'Sábado')
```

```
# Checar se o vetor possui nomes associados aos seus elementos  
names(dias_semana)
```

```
## [1] "Domingo" "Segunda" "Terça"    "Quarta"   "Quinta"   "Sexta"    "Sábado"  
# Mostrar na tela do console os dados do vetor  
dias_semana
```

```
## Domingo Segunda Terça Quarta Quinta Sexta Sábado  
##      1       2       3       4       5       6       7
```

```
# Verificar se os vetores "inteiros" e "dias_semana"  
# possuem algum atributo  
attributes(inteiros)
```

**7.1.2.2 attributes()** Retorna uma lista com os atributos associados a um vetor.

```
## NULL  
attributes(dias_semana)
```

```
## $names  
## [1] "Domingo" "Segunda" "Terça"    "Quarta"   "Quinta"   "Sexta"    "Sábado"
```

```
# gerar uma sequência de 1 a 10, saltando 2 números  
sequencia <- seq(0, 10, 2)  
print(sequencia)
```

**7.1.2.3 seq()** Cria uma sequência dentro de um vetor

```
## [1] 0 2 4 6 8 10
```

```
# Gerar uma repetição de três vezes o vetor formado pelos número de 1 a 4  
rep(1:4, 3)
```

**7.1.2.4 rep()** Cria uma repetição de um vetor

```
## [1] 1 2 3 4 1 2 3 4 1 2 3 4
```

```
# Vetor com números inteiros duplicados  
vetor <- c(1, 2, 1, 3, 4, 5, 4)
```

```
# Mostrar os números duplicados  
duplicated(vetor)
```

**7.1.2.5 duplicated()** Mostra a localização de elementos duplicados.

```
## [1] FALSE FALSE TRUE FALSE FALSE FALSE TRUE
```

Por padrão a saída da função `duplicated()` é um vetor lógico. O código abaixo mostra com seria a saída em forma de vetorial para os dados acima.

```
# Mostrar apenas os números repetidos  
vetor[duplicated(vetor)]
```

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

**7.1.2.6 `unique()`** Retorna apenas os valores distintos. A seguir utilizaremos os dados dos vetores dap (diâmetro a altura do peito), categoria, altura e nomes\_científicos, os quais são parte de um inventário florestal realizado na Unidade de Manejo Florestal 4 da Floresta Nacional de Altamira.

```
# Leitura de vetores de um inventário florestal
load('./data/dados_modulo_1.rda')

# Mostrar os objetos atualmente disponíveis no ambiente R
ls()

## [1] "altura"           "categoria"        "dap"
## [4] "dias_semana"      "int"              "inteiros"
## [7] "letra"             "nomes_cientificos" "sequencia"
## [10] "string"            "vetor"
```

A seguir mostraremos quantas árvores foram inventariadas, ou seja, o tamanho do nosso vetor.

```
length(nomes_cientificos)
```

```
## [1] 18406
```

O saída código acima mostrar que foram inventariadas 18.406 árvores.

A seguir mostraremos a quantas espécies estas árvores pertencem, e para tal as funções `length()` e `unique()`.

```
# Vetor apenas com a relação distinta de espécies inventariadas
especies <- unique(nomes_cientificos)

# somar o vetor especies
length(especies)
```

```
## [1] 67
```

Portanto, temos a que para a área do inventário ocorrem 67 espécies de interesse comercial. O código acima poderia ser resumido em apenas uma linha, vejamos o exemplo a seguir:

```
length(unique(nomes_cientificos))
```

```
## [1] 67
```

## 7.2 Funções Estatísticas Aplicadas a Vetores

- `mean()` Retorna a média aritmética
- `median()` Retorna a mediana
- `min()` Retorna o menor valor
- `max()` Retorna o maior valor
- `sd()` Retorna o desvio padrão
- `summary()` Retorna a estatística descritiva.
- `cor()` Retorna a correlação entre dois vetores.

### 7.2.1 Calcular a média do vetor `dap` (diâmetro médio das árvores)

```
# média do vetor dap (diâmetro médio das árvores)
mean(dap)
```

```
## [1] 72.27282
```

### 7.2.2 Calcular a mediana do vetor *altura* (altura comercial das árvores)

```
### Calcular a mediana do vetor _altura_ (altura comercial das árvores)
median(altura)
```

```
## [1] 18
```

### 7.2.3 Mostrar os valores mínimo e máximo do vetor dap

```
# valor mínimo de dap
min(dap)
```

```
## [1] 40
```

```
# dap Máximo
max(dap)
```

```
## [1] 312.26
```

### 7.2.4 Calcular o desvio padrão do vetor dap

```
# desvio padrão para o vetor dap
sd(dap)
```

```
## [1] 24.02588
```

### 7.2.5 Mostrar a estatística Descritiva do Vetor altura

```
summary(altura)
```

```
##      Min.   1st Qu.   Median   Mean   3rd Qu.   Max. 
##      7.00    16.00    18.00   18.14    20.00   40.00
```

### 7.2.6 Calcular a correlação linear entre diâmetro e altura das árvores

```
cor(dap, altura)
```

```
## [1] 0.2731689
```

### 7.2.7 Função lapply() aplicada a vetores

A função lapply, parte do pacote base do R, no caso específico de vetores, recebe 2 argumentos como parâmetro: o vetor contendo de dados e uma função a ser aplicada aos elementos do vetor.

```
nomes <- c('MASSARANDUBA', 'IPÊ', 'GARAPEIRA', 'JATOBÁ')
nomes
```

```
## [1] "MASSARANDUBA" "IPÊ"          "GARAPEIRA"     "JATOBÁ"
```

```
lapply(nomes, tolower)
```

```
## [[1]]
## [1] "massaranduba"
##
## [[2]]
## [1] "ipê"
##
```

```

## [[3]]
## [1] "garapeira"
##
## [[4]]
## [1] "jatobá"

```

### 7.2.8 Função sapply()

```

sapply(nomes, tolower)

## MASSARANDUBA          IPÊ          GARAPEIRA        JATOBÁ
## "massaranduba"        "ipê"        "garapeira"      "jatobá"

```

### 7.2.9 Função mapply()

Versão multivariada das funções lapply e sapply, utilizada para iterar entre elementos de vetores ou listas.

```

# Definição dos Vetores a e b
a <- c(7, 12, 5, 2, 1)
b <- c(4, 2, 3, 5, 1)

# Nomes para os vetores
dias_semana <- c('Segunda', 'Terça', 'Quarta', 'Quinta', 'Sexta')

# Atribuir nome aos vetores
names(a) <- dias_semana
names(b) <- dias_semana

# Uso da função mapply() para retornar a soma
# entre os elementos dos vetores a e b
mapply(max, a, b)

```

```

## Segunda   Terça   Quarta   Quinta   Sexta
##       7       12       5       5       1

```

### 7.2.10 Função tapply()

Aplica uma função sobre um vetor com agrupamento em outro vetor categórico. Recebe como parâmetros: um vetor numérico, um vetor categórico e uma função. O código a seguir aplica a função média sobre o vetor volume agrupado ao vetor ut (unidades de trabalho)

```

# Calcular o volume médio por unidade de trabalho
tapply(dap, categoria, mean)

```

```

## Explorar Remanescente Substituta
## 76.88725     67.95807    71.95437

```

### 7.2.11 Valores Ausentes (NA)

Em R valores ausentes são conhecidos como NA, uma sigla em inglês que significa *Not Available*, ou seja, valores não disponíceis. Na literatura técnica e também em outras linguagens de programação esta sigla é definida como Nan (*Not available number*)

```

# Definição do Vetor "num" com elemento "NA"
num <- c(2, 11, 25, NA, 45)

```

Para o caso do vetor acima se utilizarmos a função `mean` para calcular a média aritmética do vetor `num`, teríamos que dizer a função para desconsiderar o valor `NA`, o que se faz definindo o parâmetro `na.rm = TRUE`, vejamos o exemplo a seguir:

```
# Uso da função mean() sem desconsiderar valor ausente (NA).
mean(num)

## [1] NA
# Desconsiderar valores NA
mean(num, na.rm = TRUE)

## [1] 20.75
```

## 8 Como Saber se Há Valores Ausentes (NA) nos Dados?

Para conferir a presença de valores `NA` nos dados utilizamos a função `is.na()`, a qual recebe como parâmetro de entrada apenas o vetor de dados.

- `is.na( )` Testa se o vetor contém valores ausentes (*Not Available*)

```
vetor <- c(NA, 2, 3, 6)
is.na(vetor)

## [1] TRUE FALSE FALSE FALSE
```

Vemos acima que o retorno da função `is.na()` retorna um vetor lógico, mostrando `TRUE` sempre que o elemento do vetor é do tipo `NA`. Podemos melhor a saída acima para uma forma tabular através do uso da função `summary`

```
summary(is.na(vetor))

##      Mode   FALSE    TRUE
## logical      3       1
```

Poderíamos ainda filtrar o vetor de forma a não mostrar valores ausentes, usando o operador de negação ou “diferente”, qual seja `!`. Este operador tem a mesma função da negação utilizada em lógica matemática porém nesta área utiliza-se os caracteres `¬` e `~`

```
vetor[!is.na(vetor)]
```

```
## [1] 2 3 6
```

## 9 Exercício Prático - Vetores e Operadores de Comparação

Para este exercício, considere o vetor `temperatura`. Esse vetor possui dados de temperatura média mensal da Estação Meteorológica Manual INMET 82861, localizada no município de Conceição do Araguaia.

```
temperatura <- c(26.38452, 26.90357, 27.04064, 27.42467,
                  28.53548, 28.90000, NA, 29.73818,
                  30.54667, 27.21652, 27.28800, 27.84000)
```

**Instrução 1/4 \*** Obter a temperatura média do vetor `temperatura`

**Instrução 2/4 \*** Obter as temperaturas que estão acima da média do vetor `temperatura`

**Instrução 3/4 \*** Mostre quanto dos dados do vetor de temperaturas apresentam valores `NA`

**Instrução 4/4 \*** Mostre os índices onde os dados do vetor de temperaturas apresentam valores NA

## 10 Testes Lógicos com Vetores

- `any()` Testa se algum elemento do vetor atende a uma condição específica

**Exemplo:** Dado o vetor de nome `dap`, o qual armazena dados de mensuração de diâmetro de milhares de árvores na Floresta Nacional de Altamira, teste se algum elemento é menor ou igual a 40.

```
any(dap >= 40)
```

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

```
TRUE
```

- `all()` Testa se todos os elementos de um vetor atendem a uma condição.

**Exemplo:** Dado o vetor de nome `dap`, testar se algum elemento é menor do que 0:

```
all(dap < 40)
```

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

```
FALSE
```

- `is.na( )` Testa se o vetor contém valores ausentes (*Not Availables*)

### 10.1 Exercício Prático - Índice de Vetor

Considerando o vetor de nome `dap`, o qual armazena dados de mensuração de diâmetro de milhares de árvores na Floresta Nacional de Altamira, mostre:

**Instrução 1/5 \*** Quantas árvores foram inventariadas.

**Instrução 2/5 \*** Apenas o penúltimo elemento desse vetor.

**Instrução 3/5 \*** O diâmetro mínimo de medição

**Instrução 4/5 \*** O diâmetro máximo mensurado

**Instrução 5/5 \*** O diâmetro médio mensurado

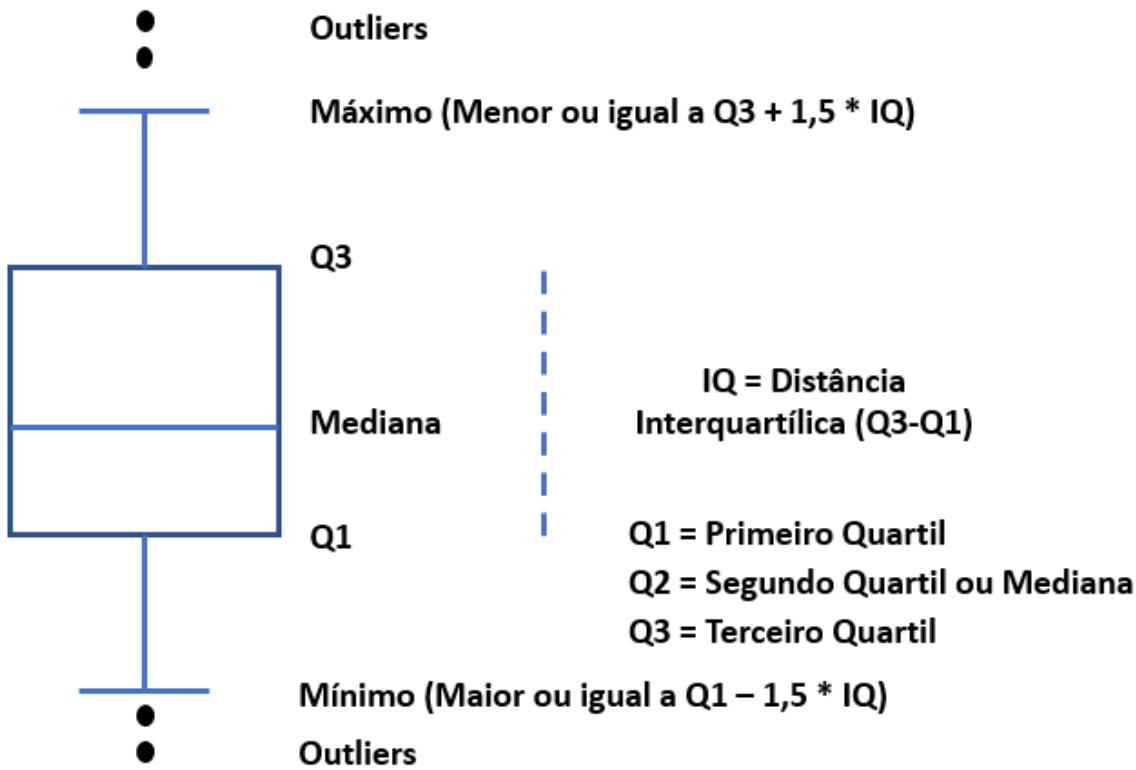
### 10.2 Visualização Gráfica de Vetores

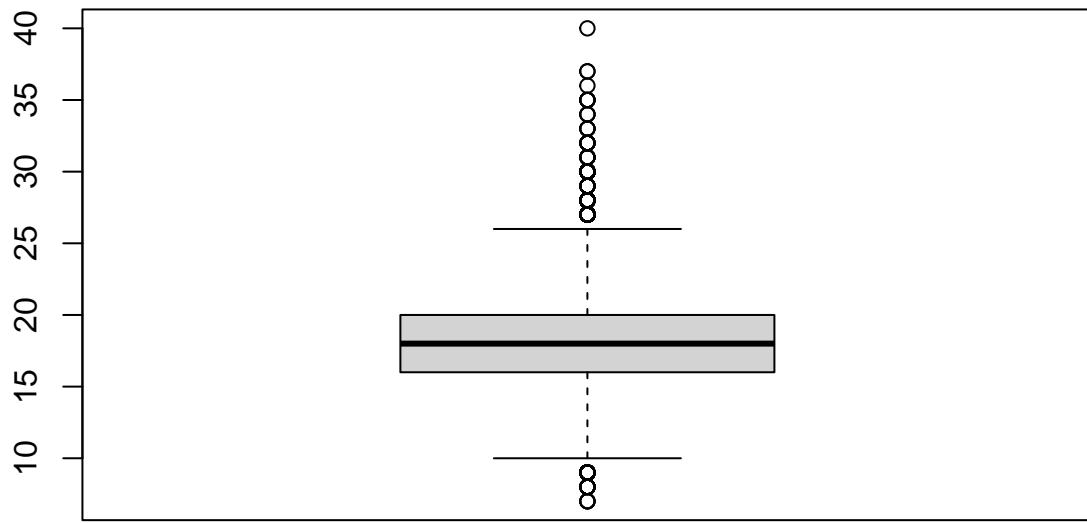
Hora de tentar algo um pouco diferente. Até agora, você programou script e observou seus dados imprimindo-os. Para uma visualização mais informativa de dados, experimente uma saída gráfica.

Para este exercício, você irá trabalhar com dados de inventário florestal realizado em uma unidade de produção anual da Floresta Nacional de Altamira. Para tal utilizaremos apenas duas variáveis, a saber: diâmetro a altura do peito (DAP) e altura comercial.

### 10.2.1 Boxplot

O *boxplot* ou diagrama de caixa é uma ferramenta gráfica da estatística que nos permite visualizar a distribuição e valores discrepantes (outliers) de dados.





### 10.2.2 Histograma

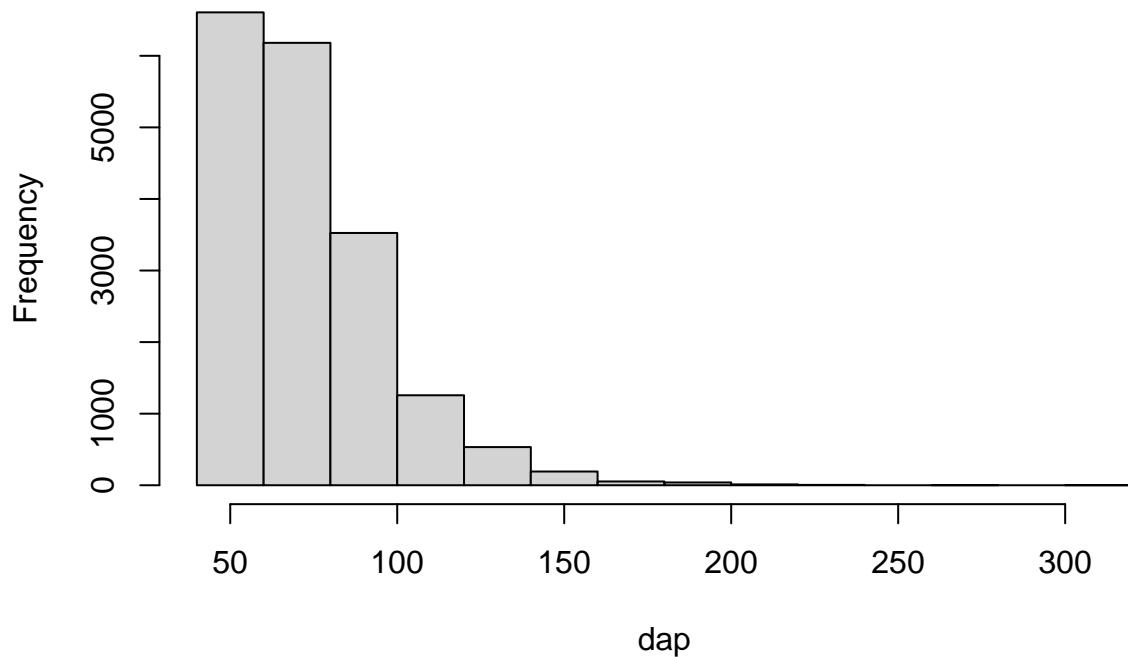
Utilziamos histogramas para visualizar a distribuição de uma variável contínua. Em R o pacote “base” nos fornece a função `hist( )`.

#### Exercício Prático

- Mostrar o histograma para os dados da variável DAP disponível no vetor `dap`

```
hist(dap)
```

### Histogram of dap



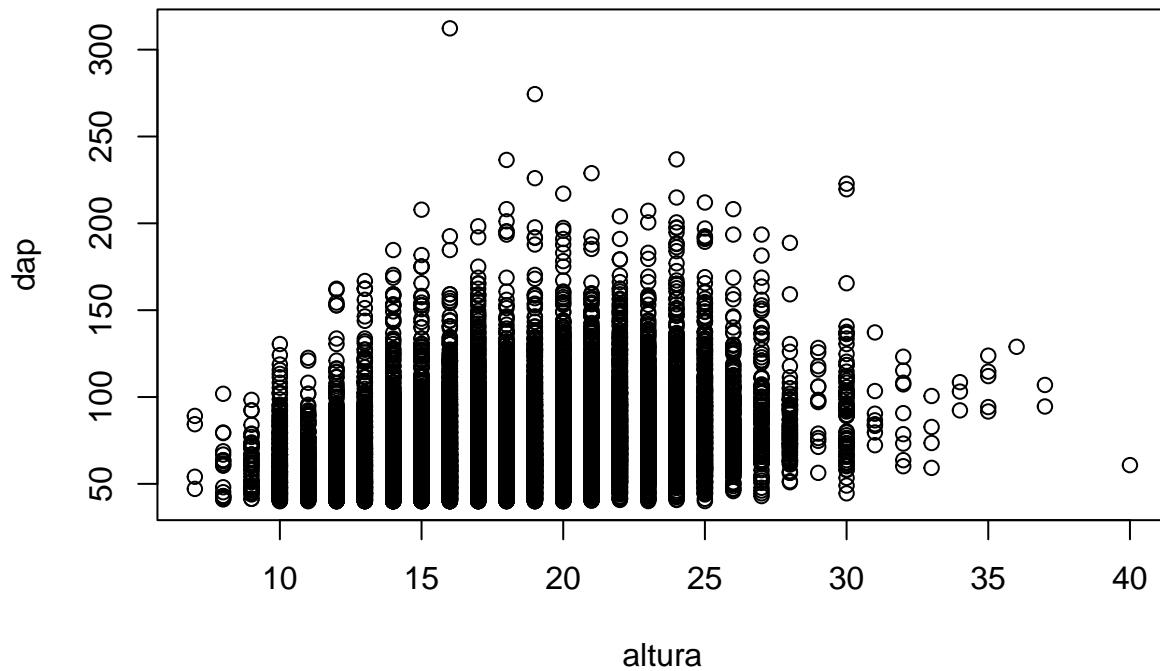
#### 10.2.3 Gráfico Dispersão

O gráfico de dispersão é utilizado para visualizar a relação entre duas variáveis contínuas. Para gerar um gráfico de dispersão em R devemos utilizar a função `plot` do pacote “base”.

##### Exercício Prático

Visualizar a relação entre a varável altura e diâmetro.

```
plot(altura, dap)
```



### 10.3 Modificar a Aparência dos Gráficos

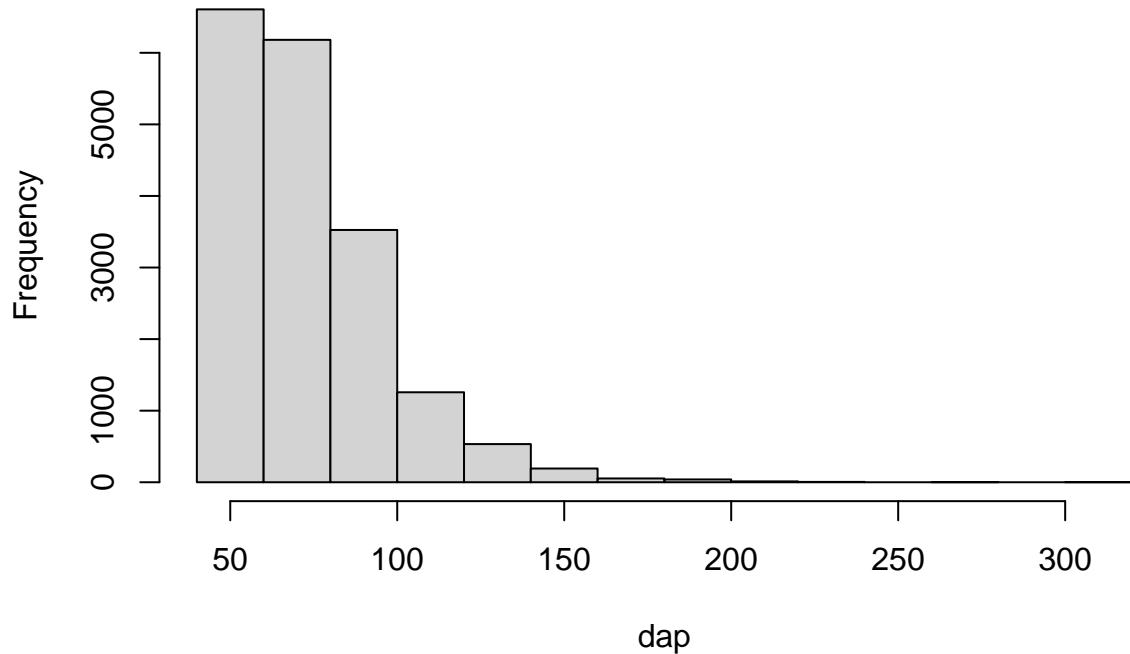
A configuração dos parâmetros de estilo, tamanho e agrupamento dos gráficos pode ser obtida digitando o comando `? par`. Para este curso introdutório utilizaremos apenas o básico da configuração da aparência de gráficos no pacote “base” do R.

#### 10.3.1 Personalizando Histogramas

- `main` Utilizado para atribuir ou modificar um título do gráfico

```
# Modificar o título do gráfico
hist(dap, main = 'Distribuição da variável DAP (cm)')
```

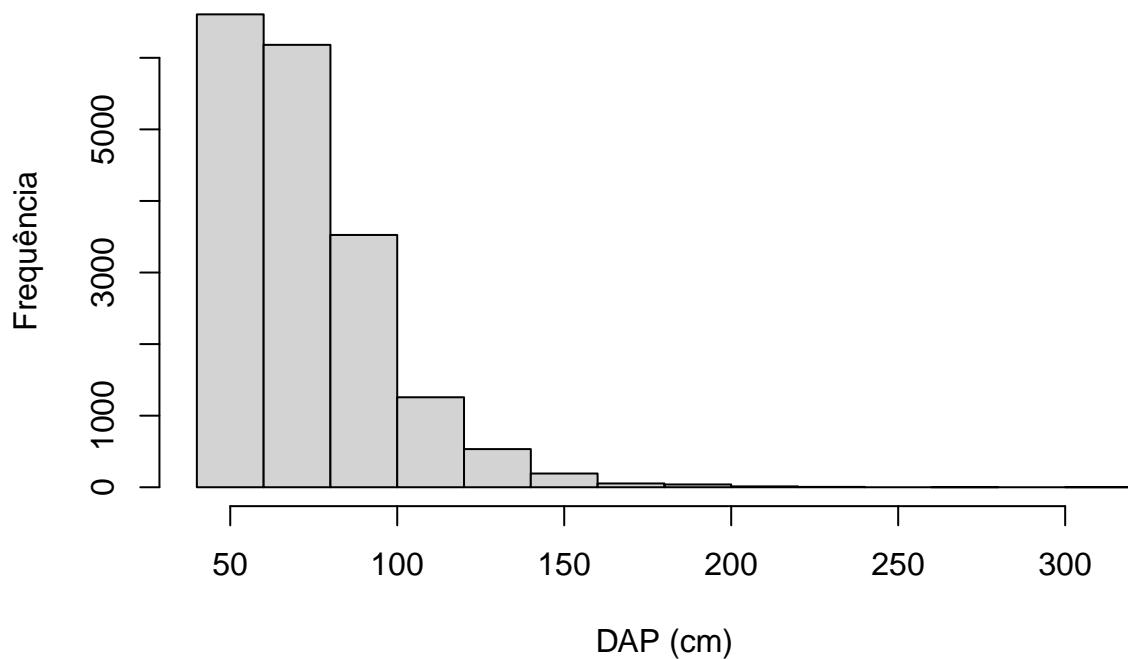
## Distribuição da variável DAP (cm)



- `xlab` e `ylab` - Modificar os nomes dos eixos x e y.

```
# Modificar os rótulos dos eixos x e y
hist(dap,
      main = 'Distribuição da variável DAP', # título do gráfico
      xlab = 'DAP (cm)',    # Rótulo do eixo x
      ylab = 'Frequência') # Rótulo do eixo y
```

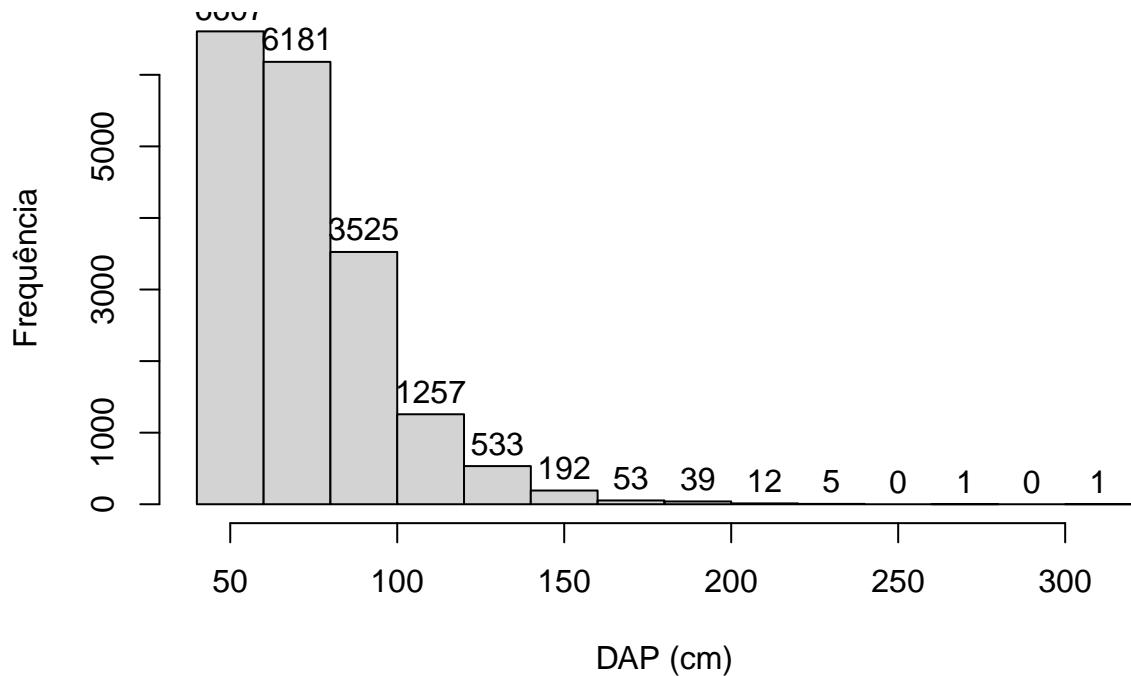
## Distribuição da variável DAP



- `labels` - Mostra os valores de cada barra do histograma.

```
hist(dap,
  main = 'Distribuição da variável DAP',
  xlab = 'DAP (cm)',
  ylab = 'Frequência',
  labels = TRUE)
```

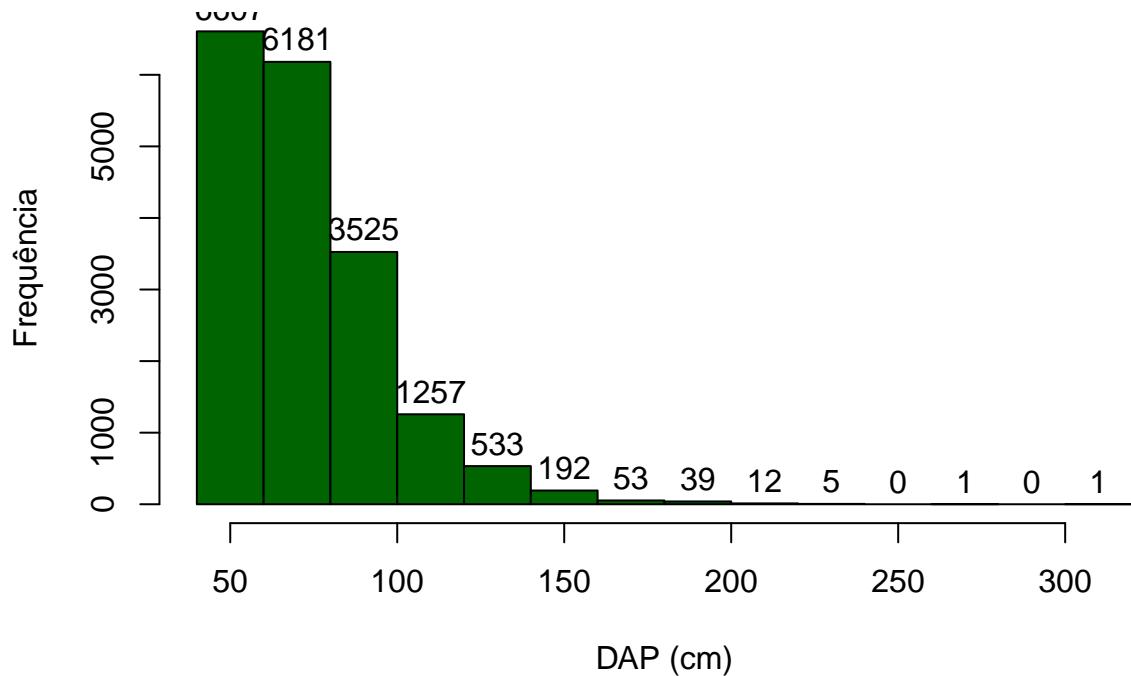
## Distribuição da variável DAP



- col - Muda a cor das barras do histograma.

```
hist(dap,
  main = 'Distribuição da variável DAP',
  xlab = 'DAP (cm)',
  ylab = 'Frequência',
  labels = TRUE,
  col = 'darkgreen')
```

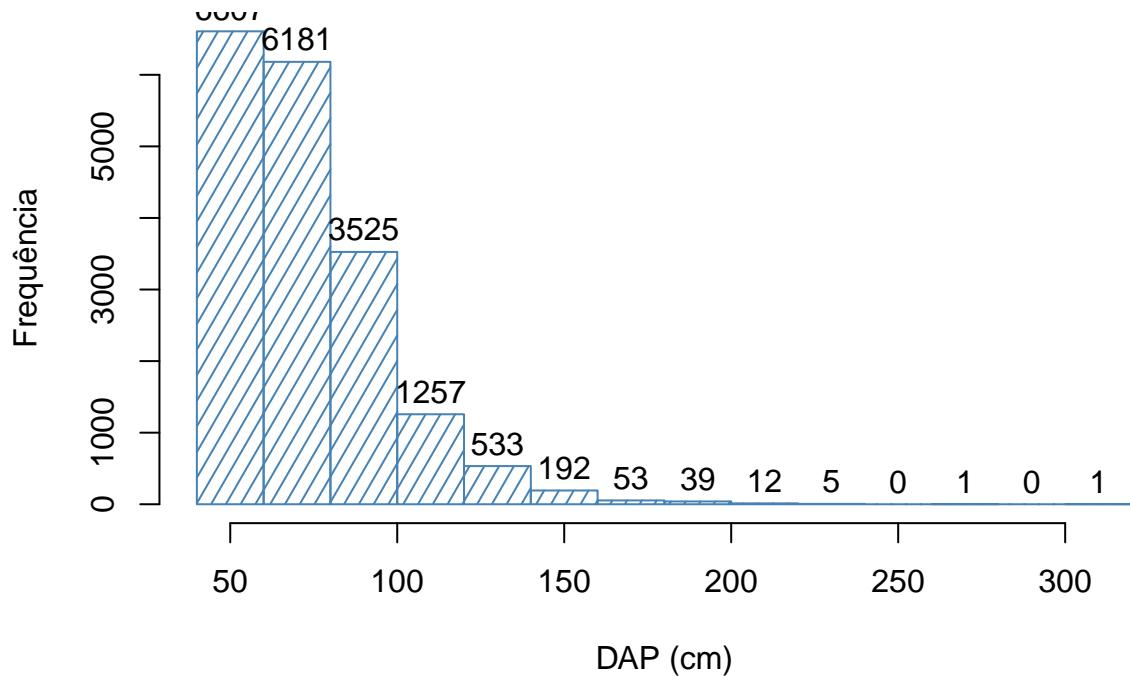
## Distribuição da variável DAP



- `density` e `angle` - Mostram as barras do histograma hachuradas

```
hist(dap,
  main = 'Distribuição da variável DAP',
  xlab = 'DAP (cm)',
  ylab = 'Frequência',
  labels = TRUE,
  col = 'steelblue',
  density = 15,
  angle = 60)
```

## Distribuição da variável DAP



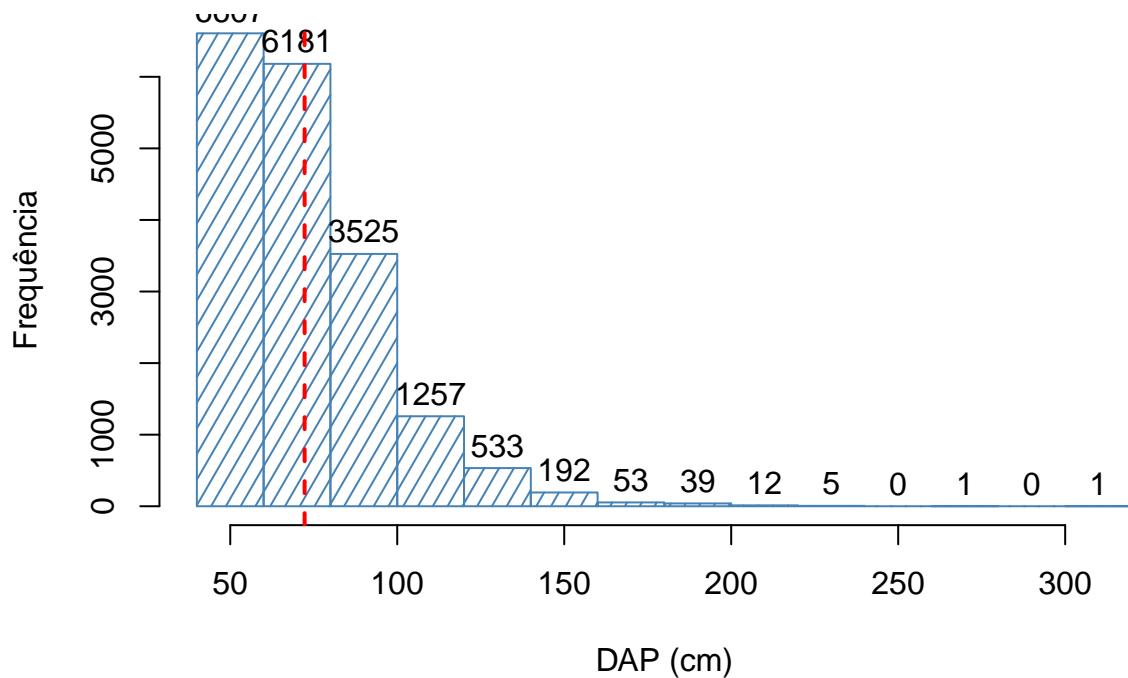
- `abline( )` - Função para adicionar uma linha reta ao histograma. Para adicionar uma linha vertical deve-se utilizar o argumento `v` e para linha horizontal `h`. O tipo de linha é modificado através do argumento `lty` (*line type*) e a espessura da linha através do argumento `lwd` (*line width*).

O exemplo a seguir mostra como adicionar uma linha ao histograma para representar a média dos diâmetros.

```
hist(dap,
  main = 'Distribuição da variável DAP',
  xlab = 'DAP (cm)',
  ylab = 'Frequência',
  labels = TRUE,
  col = 'steelblue',
  density = 15,
  angle = 60)

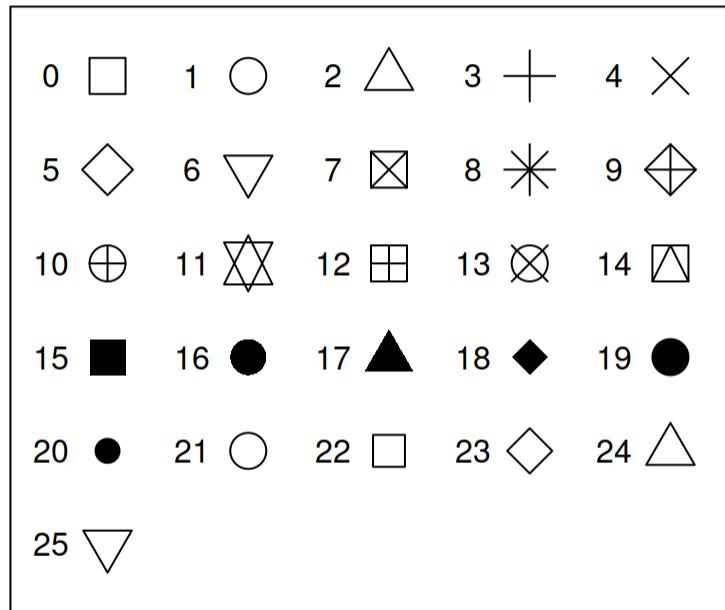
abline(v = mean(dap),
       col = 'red',
       lty = 2,
       lwd = 2)
```

## Distribuição da variável DAP



### 10.3.2 Personalizando Gráficos de Dispersão

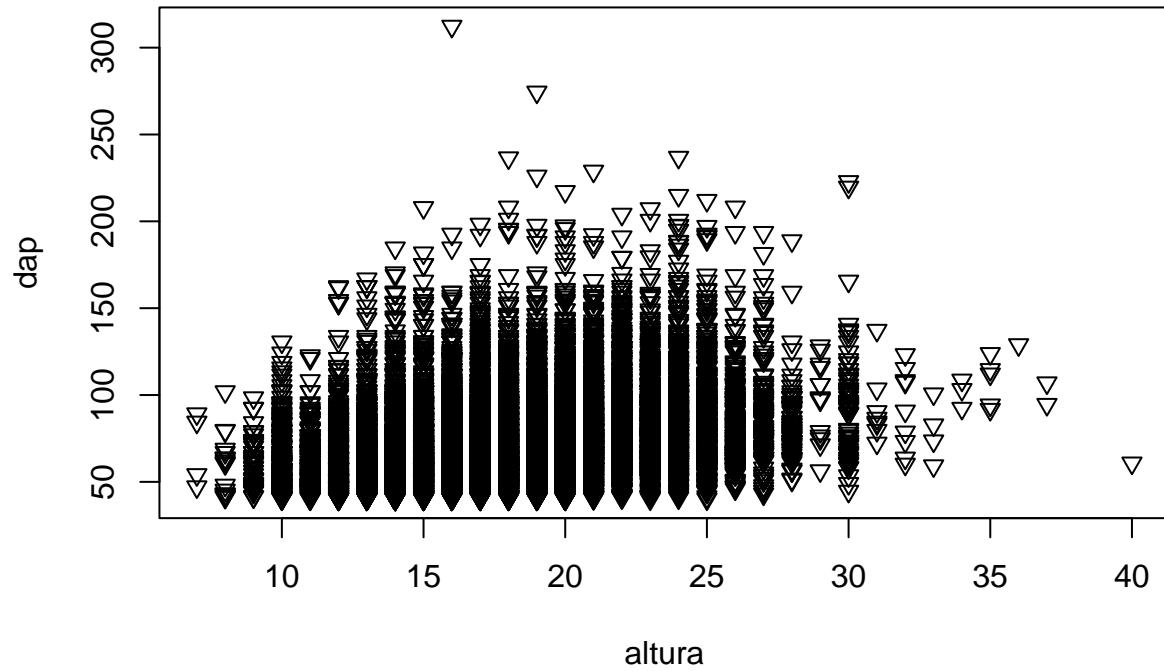
- `pch` - Altera o tipo de caractere dos pontos



- Forma, Tamanho e Cor dos Pontos

Argumento	Saída
<code>col</code>	Cor da borda do ponto.
<code>bg</code>	Cor do Fundo do ponto.
<code>cex</code>	Tamanho do ponto.
<code>lwd</code>	Espessura da Borda do Ponto.

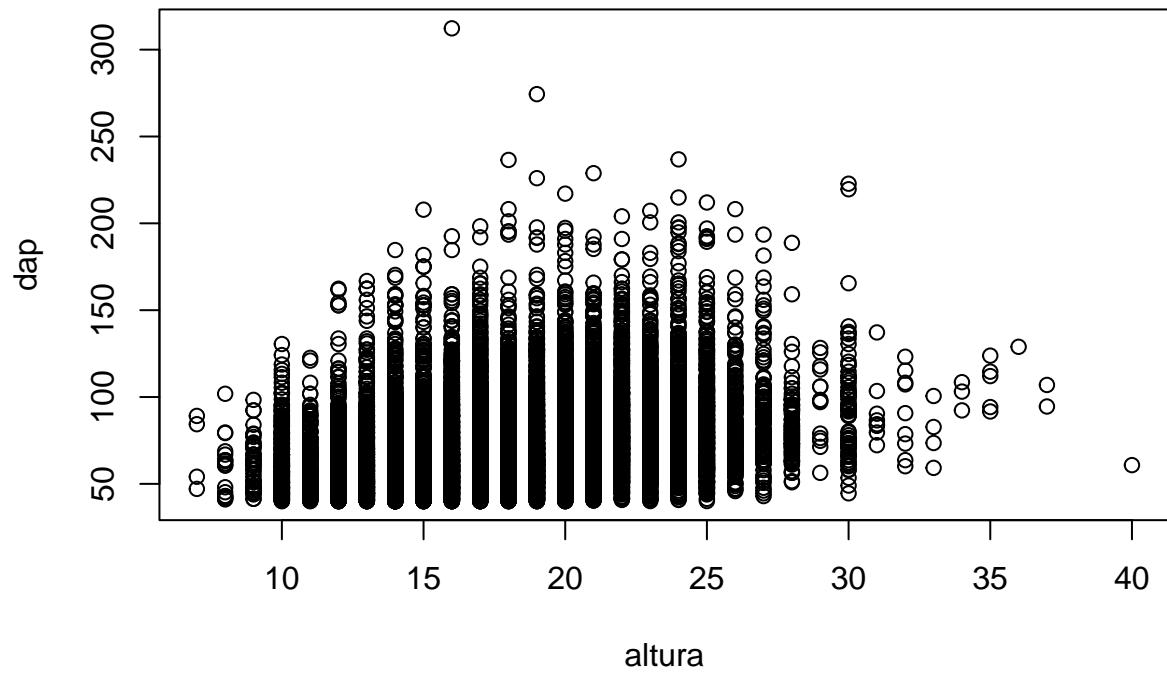
```
# Alterar o tipo de caractere dos pontos
plot(altura, dap, pch = 25)
```



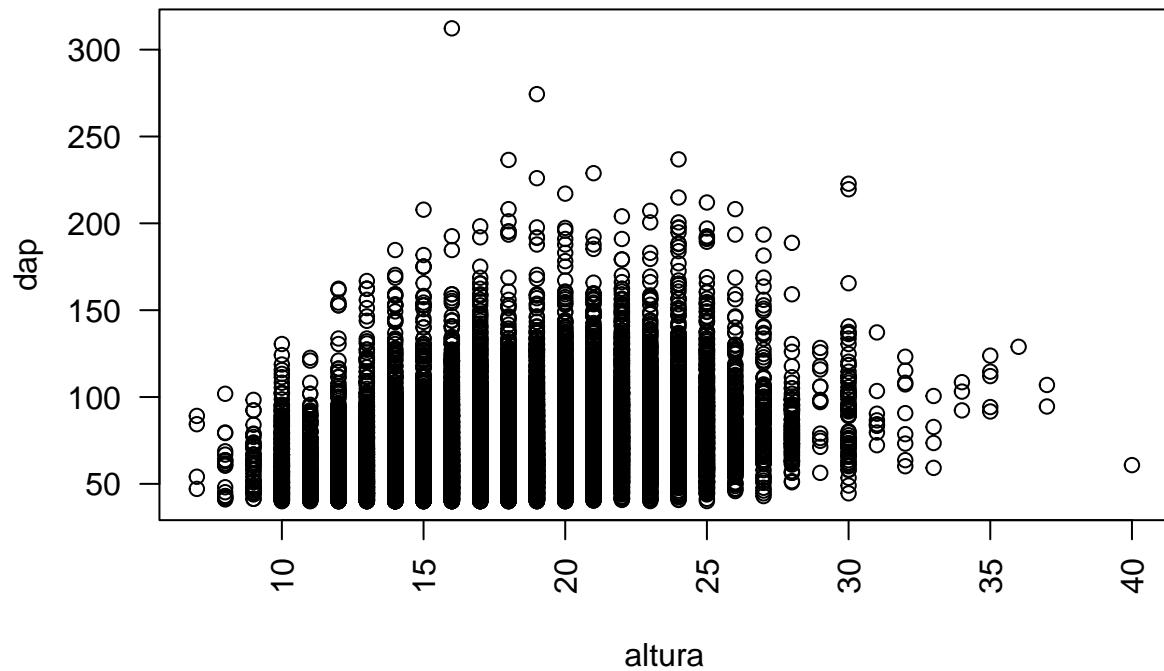
- **las (label style)** - Rotação dos rótulos dos eixos x e y.

las	Rótulo
0	Paralelo aos eixos
1	Sempre na Horizontal
2	Sempre na Perpendicular
3	Sempre na Vertical

```
plot(altura, dap, pch = 21, las = 0)
```



```
plot(altura, dap, pch = 21, las = 2)
```

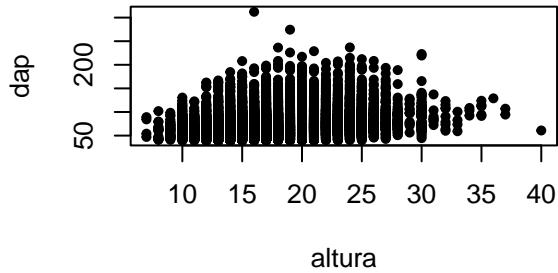
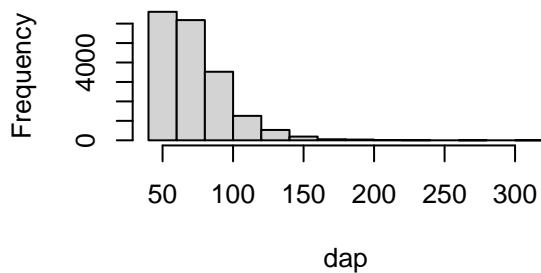


### 10.3.3 Agrupar Gráficos em uma Única Figura

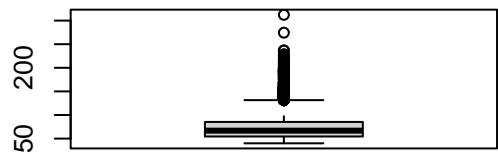
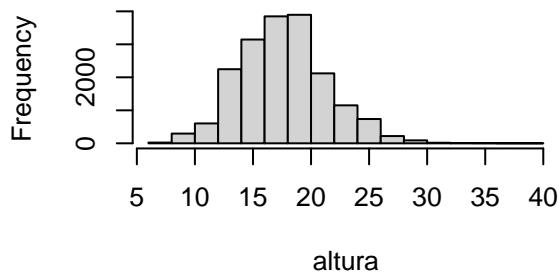
```
# Parâmetros gráficos
par(mfcol = c(2, 2))

hist(dap, pin = c(12, 8))
hist(altura)
plot(altura, dap, pch = 20)
boxplot(dap)
```

### Histogram of dap



### Histogram of altura



## 11 Matriz

Matriz é uma estrutura de dados semelhante a vetor, exceto que na matriz temos 2 dimensões, uma para as linhas e outra para as colunas. O código a seguir mostra a criação de uma matriz 3x3.

```
matriz <- matrix(1:9, nrow = 3, ncol = 3)
matriz
```

```
##      [,1] [,2] [,3]
## [1,]     1     4     7
## [2,]     2     5     8
## [3,]     3     6     9
```

### 11.1 Somar linhas e Colunas de uma Matriz

A função `apply`, parte do pacote `base` do R, pode ser usada para aplicar uma determinada função a uma matriz, e recebe 3 argumentos como parâmetro: a matriz contendo os dados, a indicação do sentido de aplicação da função, representado pelos números 1 (linha) ou 2 (coluna) e a função a ser aplicada.

Somar as linhas de uma matriz:

```
print(apply(matriz, 1, sum))
```

```
## [1] 12 15 18
```

Somar os valores das colunas de uma matriz:

```
print(apply(matriz, 2, sum))
```

```
## [1] 6 15 24
```

## 11.2 Somar os Elementos da Diagonal de uma Matriz

```
m <- matrix(1:9, nrow = 3, ncol = 3)
print(m)
```

```
##      [,1] [,2] [,3]
## [1,]     1     4     7
## [2,]     2     5     8
## [3,]     3     6     9
```

```
sum(diag(m))
```

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

## 11.3 Sentido de Preenchimento dos Dados em uma Matriz

A função `matrix()` tem por padrão o preenchimento no sentido das colunas, porém, em alguns casos podemos necessitar preencher uma matriz no sentido das linhas, para isso devemos definir o valor do argumento `byrow = TRUE`

```
matriz <- matrix(1:9, nrow = 3, ncol = 3, byrow = TRUE)
print(matriz)
```

```
##      [,1] [,2] [,3]
## [1,]     1     2     3
## [2,]     4     5     6
## [3,]     7     8     9
```

## 11.4 Atribuir Nomes as Linhas e Colunas de uma Matriz

```
matriz <- matrix(1:9, nrow = 3, ncol = 3, byrow = TRUE)
print(matriz)
```

```
##      [,1] [,2] [,3]
## [1,]     1     2     3
## [2,]     4     5     6
## [3,]     7     8     9
```

# *Atribuir Nomes as Linhas da matriz*

```
rownames(matriz) <- c('Linha 1', 'Linha 2', 'Linha 3')
```

```
print(matriz)
```

```
##      [,1] [,2] [,3]
## Linha 1     1     2     3
## Linha 2     4     5     6
## Linha 3     7     8     9
```

# *Atribuir Nomes as colunas da matriz*

```
colnames(matriz) <- c('Coluna 1', 'Coluna 2', 'Coluna 3')
```

```
matriz
```

```

##          Coluna 1 Coluna 2 Coluna 3
## Linha 1      1      2      3
## Linha 2      4      5      6
## Linha 3      7      8      9

```

## 11.5 Obter os nomes das Linhas e Colunas de uma Matriz

Somente os Nomes das Linhas

```
rownames(matriz)
```

```
## [1] "Linha 1" "Linha 2" "Linha 3"
```

Somente os Nomes das Colunas

```
colnames(matriz)
```

```
## [1] "Coluna 1" "Coluna 2" "Coluna 3"
```

Nomes das Linhas e Colunas

```
dimnames(matriz)
```

```

## [[1]]
## [1] "Linha 1" "Linha 2" "Linha 3"
##
## [[2]]
## [1] "Coluna 1" "Coluna 2" "Coluna 3"

```

## 11.6 Acessar Linhas e Colunas da Matriz

```
# mostrar a primeira linha da matriz
print(matriz[1, ])
```

```
## Coluna 1 Coluna 2 Coluna 3
##      1      2      3
```

```
# mostrar a segunda Coluna da matriz
print(matriz[, 2])
```

```
## Linha 1 Linha 2 Linha 3
##      2      5      8
```

## 11.7 Acessar Elementos da Matriz

```
# Mostrar o elemento pertencente a segunda linha e segunda coluna
print(matriz[2, 2])
```

```
## [1] 5
```

## 11.8 Alterar os Elementos de uma Matriz

```
# alterar o elemento da linha 2 coluna 2, número 5, para 0
matriz[2, 2] <- 0
print(matriz)
```

```

##          Coluna 1 Coluna 2 Coluna 3
## Linha 1      1      2      3

```

```
## Linha 2      4      0      6
## Linha 3      7      8      9
```

## 11.9 Operações com Matrizes

### 11.9.1 Maior e menor valor entre os elementos da matriz

```
# maior valor entre os elementos da matriz
max(matriz)

## [1] 9

# menor valor entre os elementos da matriz
min(matriz)

## [1] 0
```

### 11.9.2 Maior e menor valor de uma linha ou coluna da matriz

```
# maior valor entre os elementos da primeira linha
max(matriz[1,])

## [1] 3

# menor valor entre os elementos da terceira coluna
min(matriz[,3])

## [1] 3
```

### 11.9.3 Média dos elementos da matriz

```
mean(matriz)

## [1] 4.444444
```

### 11.9.4 Somar os valores das linhas e colunas

#### 11.9.5 Soma de elementos da matriz

```
# somar os valores da primeira linha
sum(matriz[1, ])

## [1] 6

# somar os valores da terceira coluna
sum(matriz[, 3])

## [1] 18

# somar os elementos da segunda linha da matriz
sum(matriz[2, ])

## [1] 10
```

### 11.9.6 Diagonal da matriz

```
# Obter a diagonal da matriz
print(diag(matriz))
```

```

## [1] 1 0 9
# Obter a soma entre os elementos da diagonal da matriz
sum(diag(matriz))

## [1] 10

```

### 11.9.7 Transposição de Matriz

```

# Transpor a matriz
t(matriz)

##           Linha 1 Linha 2 Linha 3
## Coluna 1      1      4      7
## Coluna 2      2      0      8
## Coluna 3      3      6      9

```

### 11.9.8 Soma entre matrizes

```

# Definição das matrizes "a" e "b"
a <- matrix(1:6, nrow = 3, byrow = TRUE)
b <- matrix(1:6, nrow = 3, byrow = TRUE)

print(a)

##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
## [3,]    5    6

print(b)

##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
## [3,]    5    6

# soma das matrizes a e b
a + b

##      [,1] [,2]
## [1,]    2    4
## [2,]    6    8
## [3,]   10   12

```

## 11.10 Combinar Vetores em Matriz

Em R podemos combinar vetores para formar uma matriz em que cada vetor fará parte de uma coluna ou linha da matriz. Para combinar vetores em linhas matriciais usamos a função `rbind()`, e para combinar vetores em colunas da matriz usamos a função `cbind()`. O exemplo a seguir mostra como combina três vetores com orientação nas linhas de uma matriz.

```

# Vetor referente a uma amostra de valores de ações da Apple
apple <- c(109.49, 109.90, 109.11, 109.95, 111.03)

# Vetor referente a uma amostra de valores de ações da IBM
ibm <- c(159.82, 160.02, 159.84, 160.35, 164.79)

```

```
# Vetor referente a uma amostra de valores de ações da Microsoft
microsoft <- c(59.20, 59.25, 60.22, 59.95, 61.37)
```

```
# combinar os vetores em uma matriz onde cada linha receberá os valores dos vetores
print(rbind(apple, ibm, microsoft))
```

```
##      [,1]  [,2]  [,3]  [,4]  [,5]
## apple    109.49 109.90 109.11 109.95 111.03
## ibm      159.82 160.02 159.84 160.35 164.79
## microsoft 59.20 59.25 60.22 59.95 61.37
```

A seguir é demonstrado como combinar os elementos de vetores em colunas de uma matriz.

```
# combinar os vetores em uma matriz onde cada coluna receberá os valores dos vetores
cbind(apple, ibm, microsoft)
```

```
##      apple    ibm microsoft
## [1,] 109.49 159.82     59.20
## [2,] 109.90 160.02     59.25
## [3,] 109.11 159.84     60.22
## [4,] 109.95 160.35     59.95
## [5,] 111.03 164.79     61.37
```

## 11.11 Matriz de Correlação

Como exemplo prático para demonstrar o uso de matriz para calcular a correlação entre variáveis, usaremos os dados referente a publicação:

Ramsey, F.L. and Schafer, D.W. (2013). *The Statistical Sleuth: A Course in Methods of Data Analysis* (3rd ed), Cengage Learning.

Os dados são os valores médios de peso cerebral (g), peso corporal (g), duração da gestação (dias) e tamanho da prole de 96 espécies de mamíferos.

```
# Carregar os dados vetoriais
load('./data/dados_modulo_1_aula_3.rda')

# listar os objetos no ambiente R
ls()

## [1] "a"                  "altura"            "apple"
## [4] "b"                  "categoria"         "cerebro"
## [7] "corpo"              "dap"                "dias_semana"
## [10] "especies"           "gestacao"          "ibm"
## [13] "int"                "inteiros"           "letra"
## [16] "m"                  "matriz"             "microsoft"
## [19] "nomes"              "nomes_cientificos" "num"
## [22] "prole"              "sequencia"         "string"
## [25] "temperatura"        "vetor"

# Combinar os vetores em uma matriz
m <- cbind(cerebro, corpo, gestacao, prole)

# Mostrar as primeiras 6 linhas da matriz
head(m)

##      cerebro  corpo gestacao prole
## [1,]      9.6    2.20       31    5.0
```

```

## [2,]    9.9    0.78     98   1.2
## [3,] 4480.0 2800.00    655   1.0
## [4,]   20.3    2.80    104   1.3
## [5,]  219.0   89.00    218   1.0
## [6,]   53.0    6.00     60   2.2
# Mostrar as últimas 6 linhas da matriz
tail(m)

##      cerebro corpo gestacao prole
## [91,]    198  45.0     300   1.1
## [92,]    550 400.0     310   1.0
## [93,]    179  32.0     180   1.0
## [94,]    102   5.5     210   1.0
## [95,]    185 150.0     120   4.0
## [96,]    334 250.0     255   1.0

```

## 11.12 Atribuir um Atributo a uma Matriz

Para inserir um atributo a matriz utilizamos a função `attr()`, passando como argumentos a matriz e um rótulo para nomear o atributo. Como demonstração iremos inserir um atributo a nossa matriz definida anteriormente, este atributo será a referência bibliográfica dos dados.

```

# Obter os atributos da matriz
attributes(m)

## $dim
## [1] 96  4
##
## $dimnames
## $dimnames[[1]]
## NULL
##
## $dimnames[[2]]
## [1] "cerebro"  "corpo"    "gestacao" "prole"
# Inserir o atributo
attr(m, 'Fonte') <- 'Ramsey, F.L. and Schafer, D.W. (2013). The Statistical Sleuth: A Course in Methods of Data Analysis'

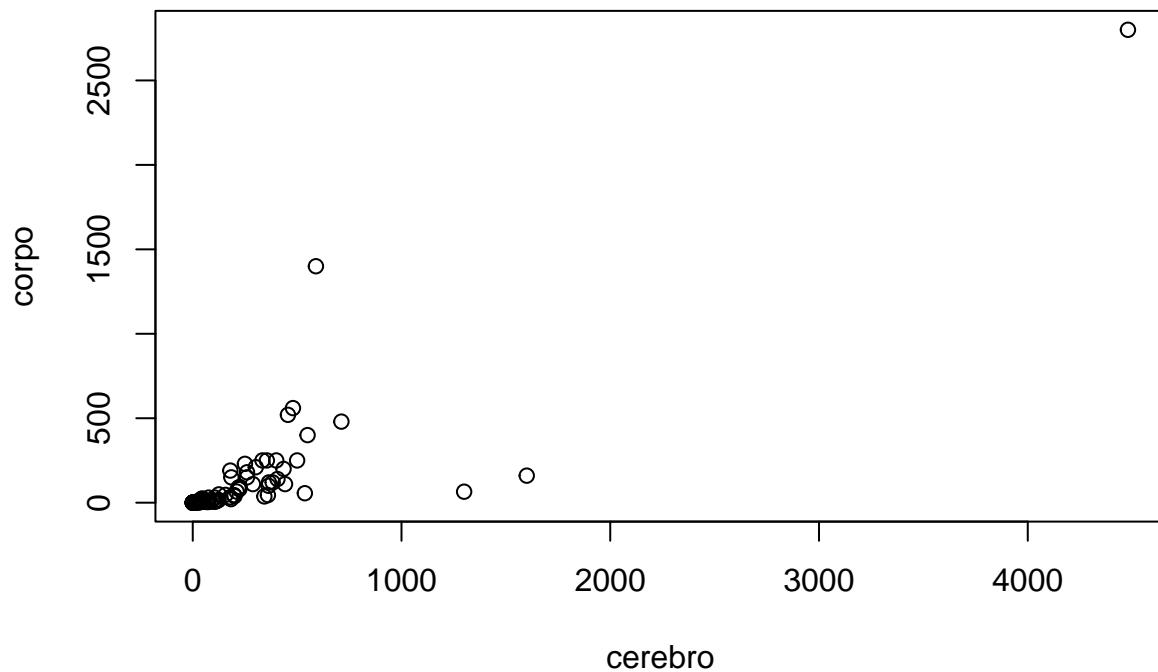
# conferir os atributos da matriz
attributes(m)

## $dim
## [1] 96  4
##
## $dimnames
## $dimnames[[1]]
## NULL
##
## $dimnames[[2]]
## [1] "cerebro"  "corpo"    "gestacao" "prole"
##
## $Fonte
## [1] "Ramsey, F.L. and Schafer, D.W. (2013). The Statistical Sleuth: A Course in Methods of Data Analysis"

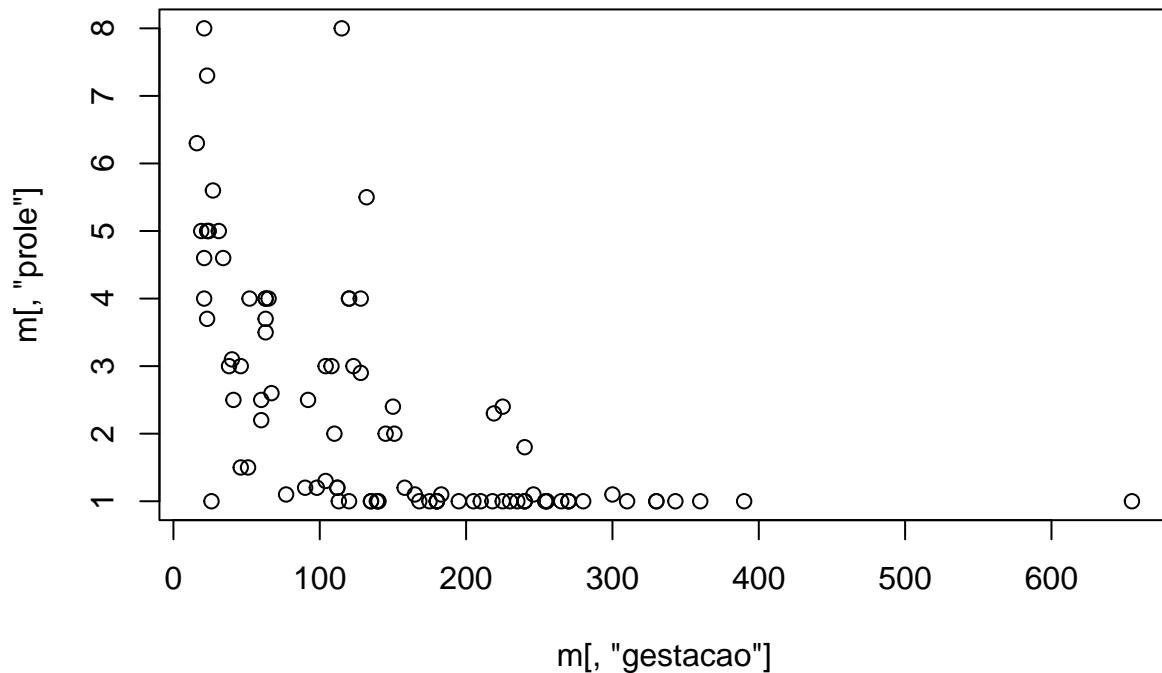
```

### 11.13 Gerar Gráficos a partir dos Dados de uma Matriz

```
# gráfico da relação entre as duas primeiras colunas (cerebro e corpo)
plot(m)
```



```
# gráfico da relação entre as duas primeiras colunas (gestacao e prole)
# plot(m[, 3], m[, 4])
plot(m[, 'gestacao'], m[, 'prole'])
```



## 12 Array

Em R array é uma estrutura de dados tridimensional. Criamos um array através da função `array(x, dim)`, onde o parâmetros `x` é um vetor e `dim` são as dimensões do array.

```
a <- array(c(1:24), dim = c(3, 3, 2))

print(a)

## , , 1
##
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
##
## , , 2
##
##      [,1] [,2] [,3]
## [1,]   10   13   16
## [2,]   11   14   17
## [3,]   12   15   18
```

## 12.1 Acessar Elementos do Array

```
# Acessar a primeira tabela
a[, , 1]

## [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9

# Acessar a primeira linha da tabela 1
print(a[1, , 1])

## [1] 1 4 7

# Acessar a primeira coluna da segunda tabela
print(a[, 1, 2])

## [1] 10 11 12
```

## 12.2 Operações com Arrays

```
# Obter o maior valor da primeira tabela
max(a[, , 1])

## [1] 9

# Obter a soma da primeira coluna da tabela 1
sum(a[, 1, 1])

## [1] 6

# obter a média dos valores da segunda linha da segunda tabela
mean(a[, 2, 2])

## [1] 14

# Obter a soma entre os valores da primeira coluna da table 1 com os da
# primeira coluna da tabela 2
sum(a[, 1, 1], a[, 1, 2])

## [1] 39

# obter a soma dos valores da diagonal da primeira tabele
sum(diag(a[, , 1]))

## [1] 15
```

## 12.3 Atribuir Nomes as Dimensões do Array

Assim como podemos atribuir nomes aos elementos de um vetor e as duas dimensões de uma matriz, também é possível o fazer para arrays. Para tal utilizamos a função `dimnames()`, passando como parâmetros três vetores com os nomes das linhas da matriz, nomes das colunas e nomes das matrizes.

```
a <- array(c(1:24), # Vetor
            dim = c(3, 3, 2), # Dimensões do array
            dimnames = list(c('L1', 'L2', 'L3'), # Nome das linhas das matrizes
                           c('C1', 'C2', 'C3'), # Nome das colunas das matrizes
                           c('Matriz 1', 'Matriz 2')))) # Nomes das Matrizes
```

```

print(a)

## , , Matriz 1
##
##      C1  C2  C3
## L1   1   4   7
## L2   2   5   8
## L3   3   6   9
##
## , , Matriz 2
##
##      C1  C2  C3
## L1  10  13  16
## L2  11  14  17
## L3  12  15  18

```

## 12.4 Inserir Atributo em um Array

```

# inserir um atributo ao array "a"
attr(a, 'Observação') <- 'Meu primeiro array em R!!'

# checar os atributos do array
print(attributes(a))

```

```

## $dim
## [1] 3 3 2
##
## $dimnames
## $dimnames[[1]]
## [1] "L1" "L2" "L3"
##
## $dimnames[[2]]
## [1] "C1" "C2" "C3"
##
## $dimnames[[3]]
## [1] "Matriz 1" "Matriz 2"
##
## 
## $Observação
## [1] "Meu primeiro array em R!!"

```

## 13 Data Frames

dataframe indiscutivelmente é a estrutura de dados mais importante em R, é nesta estrutura que a maioria dos seus dados será armazenada para análise. Combina a estrutura de uma matriz com a flexibilidade de ter diferentes tipos de dados em cada coluna. Pense em cada coluna como um vetor armazendo um tipo de dado específico. Para criar um dataframe utilizamos a função `dataframe( )`.

### 13.1 Criar um data frame

```

# criar um data frame
df <- data.frame(
  # Coluna id

```

```

id = c(1, 2, 3, 4, 5),
# Coluna nome
nome = c('Mezilaurus itauba', 'Apuleia leiocarpa', 'Cedrela odorata',
        'Amburana acreana', 'Hymenolobium excelsum'),
# Coluna volume
volume = c(3.25, 6.51, 7.45, 8.81, 4.35)
)

# mostrar os dados na tela
df

##   id           nome volume
## 1  1   Mezilaurus itauba  3.25
## 2  2     Apuleia leiocarpa  6.51
## 3  3      Cedrela odorata  7.45
## 4  4    Amburana acreana  8.81
## 5  5 Hymenolobium excelsum  4.35

```

## 13.2 Carregar um data frame a partir de um arquivo

Em R é possível a leitura de vários formatos de arquivos utilizados para armazenamento de dados, a exemplo de:

- .csv
- .dbf
- .dta (Stata)
- .fst
- .h5
- .mtp (Minitab)
- .parquet
- .rda
- .rds
- .RData
- .spss (SPSS)
- .txt
- .xls e .xlsx
- .xml
- .xport (SAS)

Abordaremos neste módulo apenas os formatos mais frequentemente utilizados, a saber: .csv, .txt, .rds e .xls

Para leitura de arquivos nos formatos .csv e .txt, podemos utilizar a função `read.csv()` ou `read.csv2()`, a primeira por padrão lê dados de planilhas onde o separador decimal é o `.` e o separador de colunas é o `,`; ao passo que a segunda função é utilizada para leitura de planilhas onde o separador decimal é o `,` e o separador de colunas é o `;`

### 13.3 Leitura de Arquivos .csv e .txt

O trecho de código a seguir faz a leitura de uma planilha em formato .csv, a qual é atribuída ao objeto R denominado inventario, o qual após leitura dos dados passa a ser um dataframe, pois as colunas da planilha são importadas como vetores. Assim, podemos definir dataframe como um conjunto de vetores, mas pode também armazenar listas.

```
# Lê uma planilha csv com dados de inventário florestal
inventario <- read.csv2('./data/UMF_4_UPA_4F_SINAFLOR_v03.csv',
                        encoding = 'latin1')
```

A seguir usamos a função head( ) para mostrar as linhas iniciais do dataframe inventario. Por padrão esta função mostra apenas as seis primeiras linhas do dataframe, caso queiramos ler as dez primeiras linhas, devemos especificar este número após o nome do dataframe: head(inventario, 10)

```
# Mostrar as seis primeiras linhas dos dados no console
head(inventario)
```

```
##   N_arv UPA UT      Nome_Cientifico Nome_Popular DAP_cm Alt    Categoria QF
## 1 10001   6 1 Parkia gigantocarpa   Fava-atanã 114.59 19 Remanescente 1
## 2 10002   6 1 Bagassa guianensis   Tatajuba  67.48 17 Remanescente 1
## 3 10003   6 1       Castilla ulei     Caucho  52.52 11 Explorar  1
## 4 10004   6 1       Castilla ulei     Caucho  40.74 13 Remanescente 1
## 5 10005   6 1      Vochysia maxima   Quaruba 47.75 15 Remanescente 2
## 6 10006   6 1   Copaifera duckei   Copaíba 43.93 18 Remanescente 1
##          Vol      g     lat      lon
## 1 12.4891 1.0313 -5.907904 -54.96910
## 2  4.3893 0.3577 -5.907938 -54.96912
## 3  1.8263 0.2166 -5.907999 -54.96991
## 4  1.1068 0.1304 -5.906353 -54.96999
## 5  1.8325 0.1790 -5.906086 -54.97024
## 6  1.7038 0.1515 -5.905871 -54.97024
```

A seguir usamos a função tail( ) para mostrar as últimas linhas do dataframe inventario. Por padrão esta função mostra apenas as seis últimas linhas do dataframe, caso queiramos ler as dez últimas linhas, devemos especificar este número após o nome do dataframe: tail(inventario, 10)

```
# Mostrar as seis últimas linhas dos dados no console
tail(inventario)
```

```
##   N_arv UPA UT      Nome_Cientifico Nome_Popular DAP_cm Alt    Categoria QF
## 18583 290695   6 29      Parkia multijuga   Fava-bengô 52.52 15 Remanescente
## 18584 290696   6 29 Couratari guianensis   Tauari  92.95 24 Explorar
## 18585 290697   6 29      Vochysia maxima   Quaruba 112.68 20 Remanescente
## 18586 290698   6 29 Hymenaea parvifolia Jutaí-mirim 66.53 18 Remanescente
## 18587 290699   6 29      Vochysia maxima   Quaruba 127.32 19 Explorar
## 18588 290700   6 29      Vochysia maxima   Quaruba  54.75 17 Remanescente
##          QF      Vol      g     lat      lon
## 18583  2  2.3001 0.2166 -5.853998 -54.97232
## 18584  1 10.4347 0.6785 -5.853668 -54.97245
## 18585  2 12.6251 0.9972 -5.853556 -54.97230
## 18586  2  4.4416 0.3476 -5.853985 -54.96238
## 18587  1 14.6343 1.2732 -5.854445 -54.96315
## 18588  2  2.7760 0.2354 -5.854735 -54.96475
```

### 13.4 Obter os nomes das colunas de um Dataframe

Alguns Dataframes contém um número muito grande de colunas e precisamos atribuir os mesmos nomes a outros dataframes que formos criando, para evitar ter que digitar estes nomes outras vezes, podemos usar as função names( ) e attributes( ).

A primeira retornará um saída vetorial ao passo que a segunda uma lista com três elementos, o primeiro é names: nomes das colunas; class: mostra a estrutura de dados, no caso data.frame; row.names: referente ao nomes dos rótulos das linhas, se houver. Para acessarmos somente os nomes das colunas usando a função attributes() devemos chamá-la da seguinte forma: attributes(dataframe)[1]

```
# Mostrar os nomes das colunas do data frame  
names(inventario)
```

```
## [1] "N_arv"           "UPA"            "UT"             "Nome_Cientifico"  
## [5] "Nome_Popular"    "DAP_cm"          "Alt"            "Categoria"  
## [9] "QF"               "Vol"            "g"              "lat"  
## [13] "lon"
```

```
# Mostrar todos os atributos do data frame, menos os rótulos das linhas  
attributes(inventario)[-3]
```

```
## $names  
## [1] "N_arv"           "UPA"            "UT"             "Nome_Cientifico"  
## [5] "Nome_Popular"    "DAP_cm"          "Alt"            "Categoria"  
## [9] "QF"               "Vol"            "g"              "lat"  
## [13] "lon"  
##  
## $class  
## [1] "data.frame"  
  
# Mostrar apenas o primeiro atributo (nomes das colunas)  
attributes(inventario)[1]
```

```
## $names  
## [1] "N_arv"           "UPA"            "UT"             "Nome_Cientifico"  
## [5] "Nome_Popular"    "DAP_cm"          "Alt"            "Categoria"  
## [9] "QF"               "Vol"            "g"              "lat"  
## [13] "lon"
```

### 13.5 Obter número de linhas e colunas de uma Dataframe

A função dim( ) recebe como parâmetro um dataframe e retorna o número de linhas e colunas.

```
# Mostrar as dimensões do data frame  
dim(inventario)
```

```
## [1] 18588   13
```

- Podemos acessar apenas o número de linhas ou de colunas: dim(inventario)[1] e dim(inventario)[2], respectivamente.

```
# Mostrar apenas o número de linhas do dataframe  
dim(inventario)[1]
```

```
## [1] 18588
```

```
# Mostrar apenas o número de colunas do dataframe  
dim(inventario)[2]
```

```
## [1] 13
```

Os comandos acima podem ser simplificados com o uso apenas das funções `nrow()` e `ncol()`. veja os exemplos a seguir:

```
# Obter o número de linhas de um dataframe  
nrow(inventario)
```

```
## [1] 18588  
  
# Obter o número de colunas de um dataframe  
ncol(inventario)
```

```
## [1] 13
```

Dos exemplos acima podemos observar que foram inventariadas 18.588 árvores (número de linhas) e que o dataframe contém 13 variáveis (colunas)

## 13.6 Obter a Estrutura dos Dados de um DataFrame

```
# Verificar a estrutura dos dados  
str(inventario)
```

```
## 'data.frame': 18588 obs. of 13 variables:  
## $ N_arv : int 10001 10002 10003 10004 10005 10006 10007 10008 10009 10010 ...  
## $ UPA : int 6 6 6 6 6 6 6 6 6 6 ...  
## $ UT : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ Nome_Cientifico: chr "Parkia gigantocarpa" "Bagassa guianensis" "Castilla ulei" "Castilla ulei"  
## $ Nome_Popular : chr "Fava-atanã" "Tatajuba" "Caucho" "Caucho" ...  
## $ DAP_cm : num 114.6 67.5 52.5 40.7 47.8 ...  
## $ Alt : int 19 17 11 13 15 18 16 20 20 15 ...  
## $ Categoria : chr "Remanescente" "Remanescente" "Explorar" "Remanescente" ...  
## $ QF : int 1 1 1 1 2 1 2 1 1 1 ...  
## $ Vol : num 12.49 4.39 1.83 1.11 1.83 ...  
## $ g : num 1.031 0.358 0.217 0.13 0.179 ...  
## $ lat : num -5.91 -5.91 -5.91 -5.91 -5.91 ...  
## $ lon : num -55 -55 -55 -55 -55 ...
```

## 13.7 Resumo dos Dados de um Dataframe

```
# Mostrar o resumo dos dados  
summary(inventario)
```

```
##      N_arv          UPA          UT      Nome_Cientifico  
##  Min.   : 10001   Min.   :6   Min.   : 1.00  Length:18588  
##  1st Qu.: 90010   1st Qu.:6   1st Qu.: 9.00  Class  :character  
##  Median :160623   Median :6   Median :16.00  Mode   :character  
##  Mean   :161674   Mean   :6   Mean   :16.13  
##  3rd Qu.:240164   3rd Qu.:6   3rd Qu.:24.00  
##  Max.   :290700   Max.   :6   Max.   :29.00  
##  
##      Nome_Popular        DAP_cm          Alt      Categoria  
##  Length:18588   Min.   : 39.79   Min.   : 7.00  Length:18588  
##  Class  :character  1st Qu.: 54.11   1st Qu.:16.00  Class  :character  
##  Mode   :character  Median  : 66.85   Median  :18.00  Mode   :character  
##  
##      Mean   : 71.95   Mean   :18.12  
##  
##      3rd Qu.: 84.99   3rd Qu.:20.00  
##  
##      Max.   :312.26   Max.   :40.00  
##  
##      QF            Vol           g          lat
```

```

##  Min.   :1.000   Min.   : 0.7213   Min.   :0.1243   Min.   :-5.918
##  1st Qu.:1.000   1st Qu.: 2.7023   1st Qu.:0.2300   1st Qu.:-5.897
##  Median :1.000   Median : 4.4863   Median :0.3509   Median :-5.881
##  Mean   :1.282   Mean   : 5.5925   Mean   :0.4523   Mean   :-5.883
##  3rd Qu.:2.000   3rd Qu.: 7.3581   3rd Qu.:0.5673   3rd Qu.:-5.871
##  Max.   :3.000   Max.   :39.0858   Max.   :7.6582   Max.   :-5.848
##
##          lon
##  Min.   :-55.00
##  1st Qu.:-54.97
##  Median :-54.96
##  Mean   :-54.96
##  3rd Qu.:-54.95
##  Max.   :-54.94

```

### 13.8 Acessar colunas do data frame

Para acessar uma coluna específica de um data frame usamos o sinal \$, se queremos acessar somente a coluna referente a variável volume, denominada em nossa data frame de Vol: inventario\$Vol. Isso nos permite aplicar uma função apenas a esta coluna:

```
# Selecionar a coluna "Vol" e Mostrar o resumo dos dados da coluna "Vol"
summary(inventario$Vol)
```

```

##      Min. 1st Qu. Median Mean 3rd Qu. Max.
##  0.7213 2.7023 4.4863 5.5925 7.3581 39.0858

```

### 13.9 Filtrar Linhas e/ou Colunas de uma Data Frame

Em função do data frame possuir duas dimensões, é possível localizar uma linha e/ou uma coluna através das coordenadas de suas dimensões. Vamos considerar como exemplo os dados referente a lista de espécies da fauna brasileira ameaçadas de extinção.

```
fauna <- read.csv2('./data/DF_port_MMA_300-2022_fauna.csv')
```

```
head(fauna)
```

```

##   n port443 classe       ordem      familia      especie_subespecie
## 1 1      *  Aves Accipitriformes Accipitridae Amadonastur lacernulatus
## 2 2      *  Aves Accipitriformes Accipitridae           Circus cinereus
## 3 3      *  Aves Accipitriformes Accipitridae           Harpia harpyja
## 4 4      *  Aves Accipitriformes Accipitridae           Leptodon forbesi
## 5 5      *  Aves Accipitriformes Accipitridae           Morphnus guianensis
## 6 6      *  Aves Accipitriformes Accipitridae           Urubitinga coronata
##
##      categoria
## 1      VU
## 2      VU
## 3      VU
## 4      EN
## 5      VU
## 6      EN

```

Para filtrar a segunda linha passamos o endereço desta linha no data frame

```
fauna[2, ]
```

```

##   n port443 classe       ordem      familia      especie_subespecie categoria
## 2 2      *  Aves Accipitriformes Accipitridae           Circus cinereus      VU

```

Para filtrar apenas a terceira coluna

```
fauna[, 3]
```

```
## [1] "Aves"      "Aves"      "Aves"      "Aves"      "Aves"
## [5] "Aves"      "Aves"      "Anthozoa"   "Arachnida"  "Arachnida"
## [9] "Arachnida"  "Arachnida"  "Arachnida"  "Arachnida"  "Arachnida"
## [13] "Arachnida"  "Arachnida"  "Arachnida"  "Arachnida"  "Arachnida"
## [17] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [21] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [25] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [29] "Malacostraca" "Malacostraca" "Malacostraca" "Aves"       "Hydrozoa"
## [33] "Amphibia"    "Amphibia"    "Diplopoda"   "Diplopoda"   "Diplopoda"
## [37] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [41] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [45] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [49] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [53] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [57] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [61] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [65] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [69] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [73] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [77] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [81] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [85] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [89] "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"    "Amphibia"
## [93] "Amphibia"    "Holothuroidea" "Aves"       "Aves"       "Aves"
## [97] "Aves"       "Aves"       "Aves"       "Aves"       "Aves"
## [101] "Aves"      "Aves"       "Aves"       "Aves"       "Aves"
## [105] "Arachnida"  "Arachnida"   "Arachnida"   "Arachnida"   "Arachnida"
## [109] "Arachnida"  "Arachnida"   "Arachnida"   "Arachnida"   "Arachnida"
## [113] "Arachnida"  "Arachnida"   "Arachnida"   "Arachnida"   "Arachnida"
## [117] "Arachnida"  "Arachnida"   "Arachnida"   "Arachnida"   "Arachnida"
## [121] "Arachnida"  "Arachnida"   "Arachnida"   "Arachnida"   "Arachnida"
## [125] "Arachnida"  "Arachnida"   "Arachnida"   "Arachnida"   "Arachnida"
## [129] "Actinopterygii" "Insecta"    "Gastropoda"  "Gastropoda"  "Gastropoda"
## [133] "Gastropoda"  "Echinoidea"  "Aves"       "Aves"       "Aves"
## [137] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [141] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [145] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [149] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [153] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [157] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Mammalia"    "Mammalia"
## [161] "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"
## [165] "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"
## [169] "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"
## [173] "Echinoidea"  "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"
## [177] "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"
## [181] "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"    "Mammalia"
## [185] "Mammalia"    "Mammalia"    "Mammalia"    "Actinopterygii" "Actinopterygii"
## [189] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [193] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [197] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [201] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
```



```

## [421] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [425] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [429] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [433] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [437] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [441] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Malacostraca"
## [445] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [449] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [453] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [457] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [461] "Malacostraca" "Malacostraca" "Malacostraca" "Diplura"
## [465] "Actinopterygii" "Enteropneusta" "Collembola" "Collembola"
## [469] "Collembola" "Collembola" "Collembola" "Collembola"
## [473] "Collembola" "Collembola" "Collembola" "Collembola"
## [477] "Collembola" "Collembola" "Insecta" "Insecta"
## [481] "Insecta" "Insecta" "Insecta" "Insecta"
## [485] "Insecta" "Polychaeta" "Polychaeta" "Udeonychophora"
## [489] "Udeonychophora" "Udeonychophora" "Udeonychophora" "Asteroidea"
## [493] "Aves" "Aves" "Aves" "Aves"
## [497] "Aves" "Aves" "Aves" "Aves"
## [501] "Aves" "Aves" "Aves" "Aves"
## [505] "Aves" "Diplopoda" "Aves" "Aves"
## [509] "Aves" "Aves" "Aves" "Aves"
## [513] "Gastropoda" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [517] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [521] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [525] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [529] "Oligochaeta" "Insecta" "Insecta" "Insecta"
## [533] "Chondrichthyes" "Insecta" "Insecta" "Insecta"
## [537] "Insecta" "Insecta" "Insecta" "Insecta"
## [541] "Insecta" "Insecta" "Insecta" "Insecta"
## [545] "Insecta" "Insecta" "Insecta" "Insecta"
## [549] "Insecta" "Malacostraca" "Malacostraca" "Malacostraca"
## [553] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [557] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca"
## [561] "Malacostraca" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [565] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Insecta"
## [569] "Insecta" "Insecta" "Insecta" "Insecta"
## [573] "Insecta" "Insecta" "Insecta" "Insecta"
## [577] "Insecta" "Insecta" "Insecta" "Insecta"
## [581] "Insecta" "Insecta" "Insecta" "Insecta"
## [585] "Insecta" "Insecta" "Insecta" "Insecta"
## [589] "Insecta" "Insecta" "Insecta" "Insecta"
## [593] "Insecta" "Insecta" "Insecta" "Insecta"
## [597] "Insecta" "Insecta" "Insecta" "Insecta"
## [601] "Insecta" "Insecta" "Insecta" "Insecta"
## [605] "Insecta" "Insecta" "Insecta" "Insecta"
## [609] "Insecta" "Insecta" "Insecta" "Insecta"
## [613] "Insecta" "Insecta" "Insecta" "Insecta"
## [617] "Insecta" "Insecta" "Insecta" "Insecta"
## [621] "Insecta" "Insecta" "Insecta" "Insecta"
## [625] "Insecta" "Insecta" "Insecta" "Insecta"
## [629] "Insecta" "Insecta" "Insecta" "Insecta"
## [633] "Insecta" "Gastropoda" "Gastropoda" "Gastropoda"

```

```

## [637] "Gastropoda"      "Aves"          "Aves"          "Insecta"
## [641] "Insecta"          "Insecta"        "Insecta"        "Insecta"
## [645] "Insecta"          "Insecta"        "Insecta"        "Insecta"
## [649] "Insecta"          "Insecta"        "Insecta"        "Insecta"
## [653] "Insecta"          "Insecta"        "Actinopterygii" "Actinopterygii"
## [657] "Arachnida"        "Arachnida"      "Arachnida"      "Arachnida"
## [661] "Arachnida"        "Arachnida"      "Arachnida"      "Arachnida"
## [665] "Arachnida"        "Arachnida"      "Arachnida"      "Chondrichthyes"
## [669] "Chondrichthyes"    "Insecta"        "Insecta"        "Bivalvia"
## [673] "Arachnida"        "Arachnida"      "Arachnida"      "Arachnida"
## [677] "Arachnida"        "Arachnida"      "Arachnida"      "Arachnida"
## [681] "Arachnida"        "Arachnida"      "Arachnida"      "Arachnida"
## [685] "Aves"              "Aves"          "Aves"          "Aves"
## [689] "Aves"              "Aves"          "Aves"          "Aves"
## [693] "Aves"              "Aves"          "Aves"          "Aves"
## [697] "Aves"              "Aves"          "Aves"          "Aves"
## [701] "Aves"              "Aves"          "Aves"          "Aves"
## [705] "Aves"              "Aves"          "Aves"          "Aves"
## [709] "Aves"              "Aves"          "Aves"          "Aves"
## [713] "Aves"              "Aves"          "Aves"          "Aves"
## [717] "Aves"              "Aves"          "Aves"          "Aves"
## [721] "Aves"              "Aves"          "Aves"          "Aves"
## [725] "Aves"              "Aves"          "Aves"          "Aves"
## [729] "Aves"              "Aves"          "Aves"          "Aves"
## [733] "Aves"              "Aves"          "Aves"          "Aves"
## [737] "Aves"              "Aves"          "Aves"          "Aves"
## [741] "Aves"              "Aves"          "Aves"          "Aves"
## [745] "Aves"              "Aves"          "Aves"          "Aves"
## [749] "Aves"              "Aves"          "Aves"          "Aves"
## [753] "Aves"              "Aves"          "Aves"          "Aves"
## [757] "Aves"              "Aves"          "Aves"          "Aves"
## [761] "Aves"              "Aves"          "Aves"          "Aves"
## [765] "Aves"              "Aves"          "Aves"          "Aves"
## [769] "Aves"              "Aves"          "Aves"          "Aves"
## [773] "Aves"              "Aves"          "Aves"          "Aves"
## [777] "Aves"              "Aves"          "Aves"          "Aves"
## [781] "Aves"              "Aves"          "Aves"          "Aves"
## [785] "Aves"              "Aves"          "Aves"          "Aves"
## [789] "Aves"              "Aves"          "Aves"          "Aves"
## [793] "Aves"              "Aves"          "Aves"          "Aves"
## [797] "Aves"              "Aves"          "Aves"          "Aves"
## [801] "Aves"              "Aves"          "Aves"          "Aves"
## [805] "Aves"              "Aves"          "Aves"          "Aves"
## [809] "Aves"              "Aves"          "Aves"          "Aves"
## [813] "Aves"              "Aves"          "Aves"          "Aves"
## [817] "Aves"              "Aves"          "Aves"          "Aves"
## [821] "Aves"              "Aves"          "Aves"          "Aves"
## [825] "Aves"              "Aves"          "Aves"          "Aves"
## [829] "Aves"              "Asteroidea"    "Asteroidea"    "Asteroidea"
## [833] "Asteroidea"        "Aves"          "Actinopterygii" "Actinopterygii"
## [837] "Actinopterygii"    "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [841] "Actinopterygii"    "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [845] "Actinopterygii"    "Actinopterygii" "Atheriniformes" "Atheriniformes"
## [849] "Atheriniformes"   "Actinopterygii" "Actinopterygii" "Actinopterygii"

```

```

## [853] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [857] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [861] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [865] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [869] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [873] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [877] "Mammalia" "Aves" "Aves" "Aves"
## [881] "Aves" "Aves" "Aves" "Aves"
## [885] "Aves" "Aves" "Aves" "Aves"
## [889] "Aves" "Mammalia" "Mammalia" "Actinopterygii"
## [893] "Collembola" "Collembola" "Collembola" "Collembola"
## [897] "Collembola" "Collembola" "Collembola" "Collembola"
## [901] "Demospongea" "Diplopoda" "Diplopoda" "Diplopoda"
## [905] "Diplopoda" "Diplopoda" "Diplopoda" "Mammalia"
## [909] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [913] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [917] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [921] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [925] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [929] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [933] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [937] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [941] "Mammalia" "Mammalia" "Chondrichthyes" "Chondrichthyes"
## [945] "Aves" "Aves" "Aves" "Aves"
## [949] "Aves" "Aves" "Aves" "Aves"
## [953] "Aves" "Aves" "Aves" "Aves"
## [957] "Arachnida" "Arachnida" "Arachnida" "Arachnida"
## [961] "Arachnida" "Arachnida" "Arachnida" "Arachnida"
## [965] "Aves" "Aves" "Aves" "Aves"
## [969] "Aves" "Aves" "Aves" "Aves"
## [973] "Aves" "Aves" "Aves" "Aves"
## [977] "Aves" "Aves" "Aves" "Aves"
## [981] "Aves" "Aves" "Aves" "Aves"
## [985] "Aves" "Aves" "Gastropoda" "Gastropoda"
## [989] "Gastropoda" "Gastropoda" "Gastropoda" "Gastropoda"
## [993] "Gastropoda" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [997] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [1001] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [1005] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [1009] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [1013] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [1017] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [1021] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [1025] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [1029] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [1033] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [1037] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [1041] "Mammalia" "Mammalia" "Mammalia" "Mammalia"
## [1045] "Mammalia" "Mammalia" "Mammalia" "Arachnida"
## [1049] "Anthozoa" "Anthozoa" "Chilopoda" "Chilopoda"
## [1053] "Chilopoda" "Chilopoda" "Chilopoda" "Chilopoda"
## [1057] "Actinopterygii" "Arachnida" "Arachnida" "Arachnida"
## [1061] "Arachnida" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1065] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"

```

```

## [1069] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1073] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1077] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1081] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1085] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1089] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1093] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1097] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1101] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1105] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1109] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1113] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1117] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1121] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1125] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1129] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1133] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1137] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1141] "Actinopterygii" "Actinopterygii" "Mammalia" "Mammalia"
## [1145] "Diplopoda" "Diplopoda" "Diplopoda" "Diplopoda"
## [1149] "Diplopoda" "Demospongiae" "Demospongiae" "Reptilia"
## [1153] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1157] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1161] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1165] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1169] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1173] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1177] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1181] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1185] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1189] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1193] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1197] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1201] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1205] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1209] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1213] "Reptilia" "Reptilia" "Reptilia" "Reptilia"
## [1217] "Reptilia" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes"
## [1221] "Aves" "Aves" "Aves" "Gastropoda"
## [1225] "Gastropoda" "Gastropoda" "Demospongiae" "Demospongiae"
## [1229] "Aves" "Aves" "Aves" "Collembola"
## [1233] "Collembola" "Collembola" "Collembola" "Collembola"
## [1237] "Collembola" "Actinopterygii" "Actinopterygii" "Actinopterygii"
## [1241] "Rhynchonellata" "Reptilia" "Reptilia" "Reptilia"
## [1245] "Reptilia" "Reptilia" "Aves" "Aves"
## [1249] "Aves" "Aves" "Chondrichthyes" "Turbellaria"
## [1253] "Turbellaria" "Turbellaria" "Turbellaria" "Turbellaria"
## [1257] "Turbellaria" "Aves" "Bivalvia" "Bivalvia"
## [1261] "Asteroidea" "Asteroidea" "Insecta" "Insecta"
## [1265] "Insecta" "Insecta"

```

Também podemos filtrar um data frame com base em uma coluna

```

fauna[fauna$especie_subespecie == 'Euvola ziczac', ]

##      n port443 classe     ordem familia especie_subespecie categoria
## 672 679      * Bivalvia Ostreoida Pectinidae      Euvola ziczac      EN

```

### 13.10 Filtrar dados com a função subset

```

# Gerar um subconjunto das árvores selecionadas para corte
especies_explorar <- subset(inventario, Categoria == 'Explorar')

```

```

# Mostrar o subconjunto
head(especies_explorar)

```

	N_arv	UPA	UT	Nome_Cientifico	Nome_Popular	DAP_cm	Alt
## 3	10003	6	1	Castilla ulei	Cauchó	52.52	11
## 7	10007	6	1	Pseudopiptadenia psilostachya	Timborana	59.52	16
## 9	10009	6	1	Hymenolobium petraeum	Angelim-pedra	72.57	20
## 13	10013	6	1	Gouphia glabra	Cupiúba	74.80	16
## 23	10023	6	1	Manilkara elata	Maçaranduba	74.17	19
## 24	10024	6	1	Gouphia glabra	Cupiúba	63.66	14
##	Categoria	QF	Vol	g	lat	lon	
## 3	Explorar	1	1.8263	0.2166	-5.907999	-54.96991	
## 7	Explorar	2	3.2062	0.2783	-5.905860	-54.97020	
## 9	Explorar	1	5.7342	0.4137	-5.905200	-54.97010	
## 13	Explorar	2	5.1779	0.4395	-5.903429	-54.97043	
## 23	Explorar	1	5.7705	0.4320	-5.906069	-54.97048	
## 24	Explorar	1	3.3641	0.3183	-5.906245	-54.97040	

```

# Número de Árvores a Explorar
nrow(especies_explorar)

```

```

## [1] 8221

```

### 13.11 Criar uma nova coluna

Iremos criar uma coluna em função de um filtro baseado em dados de outro data frame. Vamos filtrar os dados onde as espécies do `inventario` são espécies ameaçadas de extinção, as espécies que forem ameaçadas receberão a denominação conforme sua categoria definida na Portaria MMA 300/2022, para isso teremos que utilizar carregar um *data frame* com os dados da portaria e criar uma coluna no `inventario` denominada `status_ecologico`, a qual receberá a categoria de ameaça.

```

# ler os dados da portaria MMA 300/2022
port_300_flora <- read.csv2('~/data/DF_port_MMA_300-2022_flora.csv')

```

```

# Mostrar na tela apenas 6 linhas do data frame
head(port_300_flora)

```

	n	port443	familia	especie_subespecie_var	categoria
## 1	1	* Acanthaceae	Aphelandra espirito-santensis	EN	
## 2	2	* Acanthaceae	Aphelandra margaritae	VU	
## 3	3	* Acanthaceae	Aphelandra maximiliana	EN	
## 4	4	Acanthaceae	Aphelandra rigida	EN	
## 5	5	Acanthaceae	Aphelandra stephanophysa	VU	
## 6	6	* Acanthaceae	Dyschoriste lavandulacea	EN	

```
# Filtar as espécies do inventario que coincidem as espécies da Portaria
# MMA 300/2022 e atribuir a categoria de ameaça a uma nova coluna em inventario,
# a ser denominada status_ecologico.
inventario_ameacadas <- inventario[inventario$Nome_Cientifico %in% port_300_flora$especie_subespecie_van]
```

Agora podemos saber quais espécies do inventário florestal são ameaçadas de extinção:

```
unique(inventario_ameacadas$Nome_Cientifico)
```

```
## [1] "Bertholletia excelsa" "Apuleia leiocarpa"     "Cedrela odorata"
## [4] "Hymenaea parvifolia"   "Virola surinamensis" "Mezilaurus itauba"
```

### 13.12 Excluir Linhas e Colunas de um Data Frame

```
# Excluir a linha 2
fauna <- fauna[-2, ]
```

```
# Conferir a remoção da linha 2
head(fauna)
```

	n	port443	classe	ordem	familia	especie_subespecie
## 1	1	*	Aves	Accipitriformes	Accipitridae	Amadonastur lacernulatus
## 3	3	*	Aves	Accipitriformes	Accipitridae	Harpia harpyja
## 4	4	*	Aves	Accipitriformes	Accipitridae	Leptodon forbesi
## 5	5	*	Aves	Accipitriformes	Accipitridae	Morphnus guianensis
## 6	6	*	Aves	Accipitriformes	Accipitridae	Urubitinga coronata
## 7	7	*	Anthozoa	Actiniaria	Actiniidae	Condylactis gigantea
			categoria			
## 1			VU			
## 3			VU			
## 4			EN			
## 5			VU			
## 6			EN			
## 7			EN			

```
# Excluir a coluna n
fauna$n <- NULL
```

```
# Conferir a remoção da coluna 1
head(fauna)
```

	port443	classe	ordem	familia	especie_subespecie
## 1	*	Aves	Accipitriformes	Accipitridae	Amadonastur lacernulatus
## 3	*	Aves	Accipitriformes	Accipitridae	Harpia harpyja
## 4	*	Aves	Accipitriformes	Accipitridae	Leptodon forbesi
## 5	*	Aves	Accipitriformes	Accipitridae	Morphnus guianensis
## 6	*	Aves	Accipitriformes	Accipitridae	Urubitinga coronata
## 7	*	Anthozoa	Actiniaria	Actiniidae	Condylactis gigantea
			categoria		
## 1			VU		
## 3			VU		
## 4			EN		
## 5			VU		
## 6			EN		
## 7			EN		

### 13.13 Ler Dados de Arquivo Demilitado por Tabulação

```
dados <- read.delim2('~/data/Bivalvia_tab.txt')
head(dados)

##      n port443 classe     ordem familia especie_subespecie categoria
## 1  679       * Bivalvia Ostreoida Pectinidae   Euvola ziczac      EN
## 2 1275       * Bivalvia Unionoida Hyriidae Diplodon koseritzii    EN
## 3 1276       * Bivalvia Unionoida Mycetopodidae Mycetopoda legumen      EN
```

### 13.14 Ler Dados de Arquivo Delimitado por Espaço

```
dados <- read.delim2('~/data/Separado_por_Espaco.txt', sep = ' ')
head(dados)

##   id           nome volume
## 1 1   Mezilaurus itauba  3.25
## 2 2   Apuleia leiocarpa 6.51
## 3 3   Cedrela odorata  7.45
## 4 4   Amburana acreana 8.81
## 5 5 Hymenolobium excelsum 4.35
```

### 13.15 Ler Dados de Arquivo Hospedado na Internet

A seguir é mostrado como ler dados diretamente da internet, como exemplo iremos ler um conjunto de dados do SISTAXON.

```
# Link da planilha com dados do Sistaxon
link <- 'http://www.ibama.gov.br/phocadownload/sinaflor/2022/2022-07-22_Lista_especies_DOF.csv'

# Ler o arquivo csv diretamente do link
#sp_sistaxon <- read.csv(link)

# Mostrar os dados no console
#head(sp_sistaxon)
```

### 13.16 Leitura de Dados em Formatos de Arquivos de Softwares Proprietários

Arquivo	Pacote	Função
.dbf	foreign	read.dbf( )
.dta	foreign	read.tda( )
.fst	fst	read_fst( )
.mtp	foreign	read.mtp( )
.spss	foreign	read.spss( )
.xls		read_xls( )
.xls		read_xlsx( )
.xport	foreign	read.xport( )

### 13.17 Ler Dados de Arquivos .xls e .xlsx

```
# Carregar o pacote readxl
library(readxl)
```

```

# Listar as planilhas presentes no arquivo .xls
xls <- excel_sheets('./data/aves_reptilia.xls')
xls

## [1] "aves"      "reptilia"

Por padrão a função read_xls() lê a primeira planilha do arquivo.

# Ler dados da primeira planilha "aves" e atribuir a um dataframe
dados_fauna <- read_xls('./data/aves_reptilia.xls')

# Mostrar as primeiras seis linhas do dataframe
head(dados_fauna)

## # A tibble: 6 x 7
##       n port443 classe ordem      familia      especie_subespecie categoria
##   <dbl> <chr>   <chr> <chr>      <chr>      <chr>           <chr>
## 1     1 *      Aves  Accipitriformes Accipitridae Amadonastur lacer~ VU
## 2     2 *      Aves  Accipitriformes Accipitridae Circus cinereus VU
## 3     3 *      Aves  Accipitriformes Accipitridae Harpia harpyja VU
## 4     4 *      Aves  Accipitriformes Accipitridae Leptodon forbesi EN
## 5     5 *      Aves  Accipitriformes Accipitridae Morphnus guianens~ VU
## 6     6 *      Aves  Accipitriformes Accipitridae Urubitinga corona~ EN

```

Para ler uma outra planilha devemos passar essa informação para o parâmetro `sheets`. Se quisermos ler a planilha `reptilia` o código seria: `read_xls('./data/aves_reptilia.xls', sheet = 'reptilia')` ou `read_xls('./data/aves_reptilia.xls', sheet = 2)`

```

# Ler dados da primeira planilha "reptilia" e atribuir a um dataframe
dados_fauna <- read_xls('./data/aves_reptilia.xls', sheet = 'reptilia')

# Mostrar as primeiras seis linhas do dataframe
head(dados_fauna)

```

```

## # A tibble: 6 x 7
##       n port443 classe ordem      familia      especie_subespecie categoria
##   <dbl> <chr>   <chr> <chr>      <chr>      <chr>           <chr>
## 1 1168 *      Reptilia Squamata Amphisbaenidae Amphisbaena arda EN
## 2 1169 *      Reptilia Squamata Amphisbaenidae Amphisbaena frontalis EN
## 3 1170 *      Reptilia Squamata Amphisbaenidae Amphisbaena nigricau~ EN
## 4 1171 *      Reptilia Squamata Amphisbaenidae Amphisbaena supernum~ EN
## 5 1172 *      Reptilia Squamata Amphisbaenidae Amphisbaena uroxena EN
## 6 1173 *      Reptilia Squamata Amphisbaenidae Leposternon kisteuma~ VU

```

## 14 Salvar Data Frame

### 14.1 .csv

```

# criar um data frame
df <- data.frame(
  # Definir a coluna id
  id = c(1, 2, 3, 4, 5),
  # Definir a coluna nome
  nome = c('Mezilaurus itauba', 'Apuleia leiocarpa', 'Cedrela odorata',
          'Amburana acreana', 'Hymenolobium excelsum'),

```

```

# Definir a coluna volume
volume = c(3.25, 6.51, 7.45, 8.81, 4.35)
)

# Salvar no disco como arquivo .csv
write.csv2(df, './output/data_frame.csv')

```

## 14.2 .txt

```

# Salvar no disco como arquivo .txt
write.table(df, './data_frame.txt', row.names = FALSE)

```

## 15 2. Fatores

Em R as variáveis categóricas podem ser classificadas e ordenadas através da função factor(). Ao converter ordenar categoricamente um vetor através desta função, nosso vetor passa a ser denominado de fator.

```

# Definir o vetor de dados categóricos "sexo"
sexo <- c("macho", "femea", "femea", "macho",
        "macho", "macho", "femea", "macho")

```

O trecho de código a seguir mostra a estrutura do vetor sexo, obtida através da função str(), o retorno desta função nos diz que o tipo de vetor em questão é caracteres e possui oito elementos.

```

# Checar a estrutura do vetor
str(sexo)

##  chr [1:8] "macho" "femea" "femea" "macho" "macho" "macho" "femea" "macho"
# Resumir os dados do vetor sexo
summary(sexo)

```

```

##      Length     Class      Mode
##          8 character character

```

A função levels() é utilizada para inspecionar os níveis de um fator, como nosso vetor não foi definido como fator retorna NULL.

```

# checar os níveis do vetor
levels(sexo)

```

```

## NULL

```

Outra forma de verificar se um vetor de variáveis categóricas possui fatores é através da função is.factor(). Esta função retorna TRUE em caso do vetor possuir fatores definidos, caso contrário retorna FALSE.

```

# Checar se uma vetor é um fator
is.factor(sexo)

```

```

## [1] FALSE

```

A seguir definimos o vetor sexo como fator

```

# Converter o vetor sexo para fator
sexo_fator <- factor(sexo, levels = c("macho", "femea"))

# checar a estrutura do fator
str(sexo_fator)

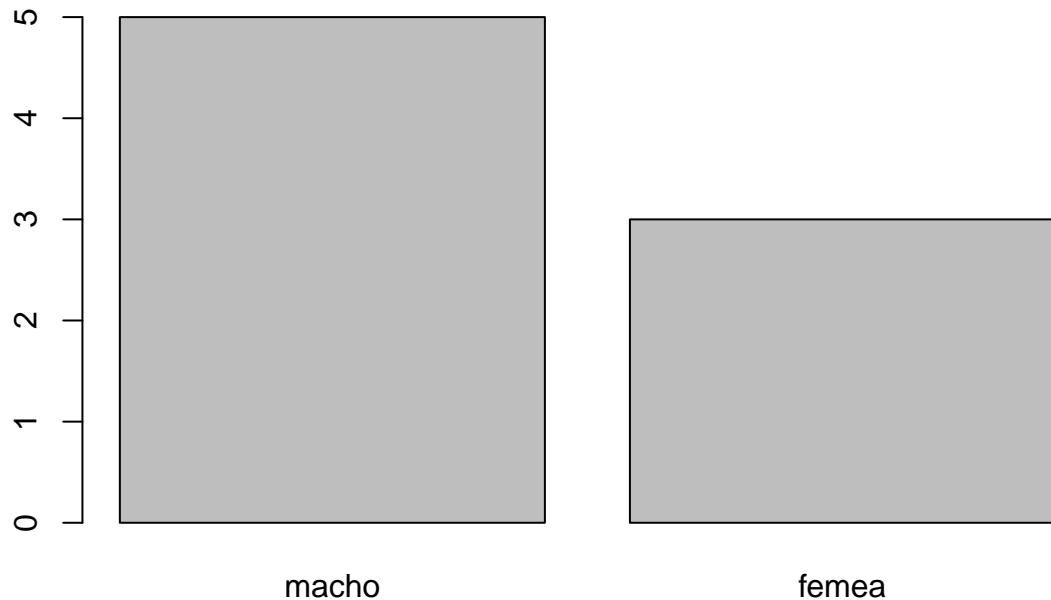
```

```

## Factor w/ 2 levels "macho","femea": 1 2 2 1 1 1 2 1
# checar os níveis do fator
levels(sexo_fator)

## [1] "macho" "femea"
# Resumir os dados do fator sexo através de um gráfico
plot(sexo_fator)

```



O código abaixo mostra como atribuir rótulos ao níveis do fatores:

```

sexo_fator <- factor(sexo, levels = c("macho", "femea"),
                      labels = c("M", "F"))

str(sexo_fator)

## Factor w/ 2 levels "M","F": 1 2 2 1 1 1 2 1
levels(sexo_fator)

## [1] "M" "F"
sexo_fator

## [1] M F F M M M F M
## Levels: M F

```

## 15.1 Converter colunas de uma data frame para factor

### 15.1.1 Fatores em Inventário Florestal

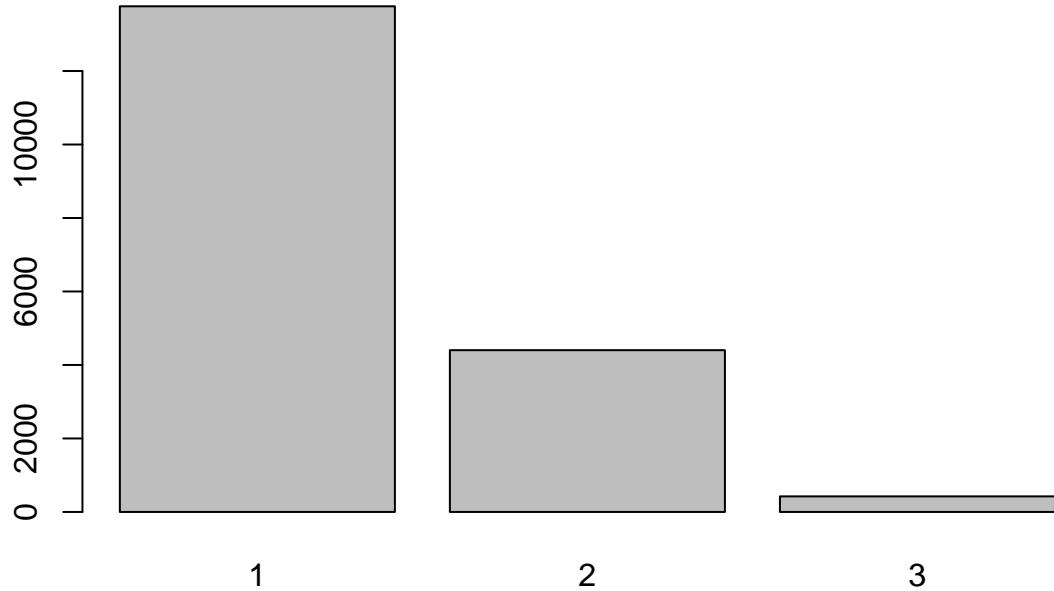
```
# Lê uma planilha csv com dados de inventário florestal
inventario <- read.csv2('./data/UMF_4_UPA_4F_SINAFLOR_v03.csv',
                        encoding = 'latin1')

# Converter a coluna "QF" (qualidade do fuste) para fatores
inventario$QF <- factor(inventario$QF,
                           levels = c(1, 2, 3),
                           ordered = TRUE)

# Resumo da coluna QF
summary(inventario$QF)

##      1      2      3
## 13761  4403   424

# Visualização gráfica dos fatores
plot(inventario$QF)
```



## 15.2 Atribuir Rótulos ao Fator

Agora iremos atribuir rótulos ao nosso fator, os rótulos serão: Fuste 1, Fuste 2 e Fuste 3.

```
# Converter a coluna "QF" (qualidade do fuste) para fatores e atribuir rótulos
inventario$QF <- factor(
  inventario$QF,
```

```

  levels = c(1, 2, 3),
  label = c('Fuste 1', 'Fuste 2', 'Fuste 3'),
  ordered = TRUE
)

# Resumo da coluna QF
summary(inventario$QF)

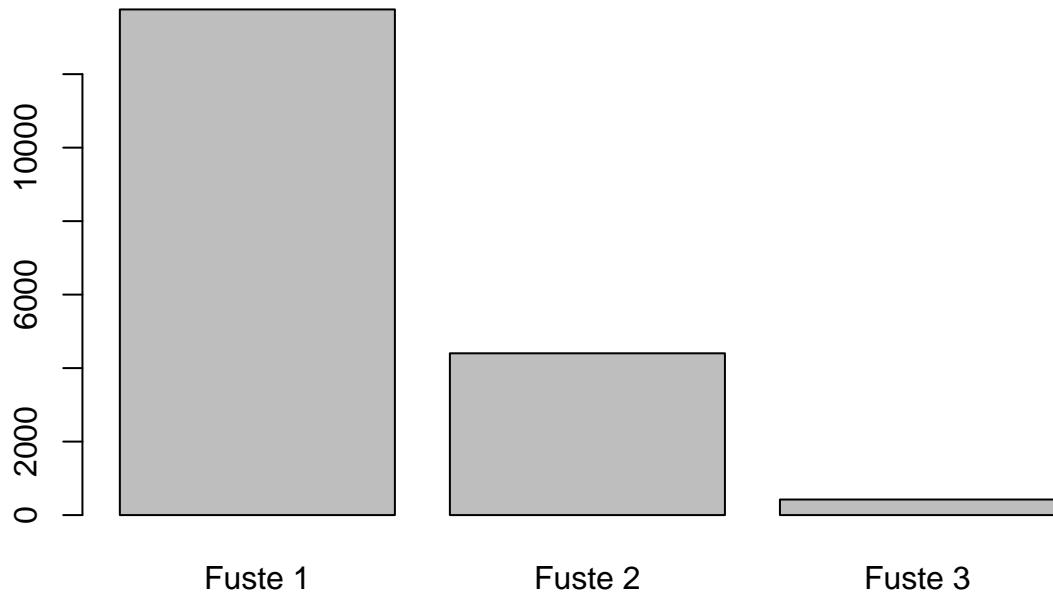
## Fuste 1 Fuste 2 Fuste 3
##   13761    4403     424

# Mostrar os dados com a inserção de rótulos aos fatores
head(inventario)

##   N_arv UPA UT      Nome_Cientifico Nome_Popular DAP_cm Alt   Categoria      QF
## 1 10001   6 1 Parkia gigantocarpa   Fava-atanã 114.59 19 Remanescente Fuste 1
## 2 10002   6 1 Bagassa guianensis   Tatajuba  67.48 17 Remanescente Fuste 1
## 3 10003   6 1       Castilla ullei   Cauchó  52.52 11 Explorar Fuste 1
## 4 10004   6 1       Castilla ullei   Cauchó  40.74 13 Remanescente Fuste 1
## 5 10005   6 1      Vochysia maxima   Quaruba  47.75 15 Remanescente Fuste 2
## 6 10006   6 1      Copaifera duckei   Copáiba  43.93 18 Remanescente Fuste 1
##           Vol      g      lat      lon
## 1 12.4891 1.0313 -5.907904 -54.96910
## 2  4.3893 0.3577 -5.907938 -54.96912
## 3  1.8263 0.2166 -5.907999 -54.96991
## 4  1.1068 0.1304 -5.906353 -54.96999
## 5  1.8325 0.1790 -5.906086 -54.97024
## 6  1.7038 0.1515 -5.905871 -54.97024

# Visualização gráfica dos fatores
plot(inventario$QF)

```



### 15.3 Fatores - Lista Espécies Fauna Ameaçada de Extinção

```
# Leitura dos dados
fauna <- read.csv("./data/DF_port_MMA_300-2022_fauna.csv")

# Mostrar as seis primeiras linhas dos dados
head(fauna)

##   n port443 classe      ordem     familia      especie_subespecie
## 1 1       *  Aves Accipitriformes Accipitridae Amadonastur lacernulatus
## 2 2       *  Aves Accipitriformes Accipitridae           Circus cinereus
## 3 3       *  Aves Accipitriformes Accipitridae          Harpia harpyja
## 4 4       *  Aves Accipitriformes Accipitridae        Leptodon forbesi
## 5 5       *  Aves Accipitriformes Accipitridae Morphnus guianensis
## 6 6       *  Aves Accipitriformes Accipitridae    Urubitinga coronata
##   categoria
## 1       VU
## 2       VU
## 3       VU
## 4       EN
## 5       VU
## 6       EN

# Valores únicos para categoria de ameaça
unique(fauna$categoria)

## [1] "VU"      "EN"      "CR"      "CR (PEX)" "EX"      "RE"      "EW"
```

```

# Resumo da coluna categoria
summary(fauna$categoria)

##      Length     Class    Mode
##      1266 character character

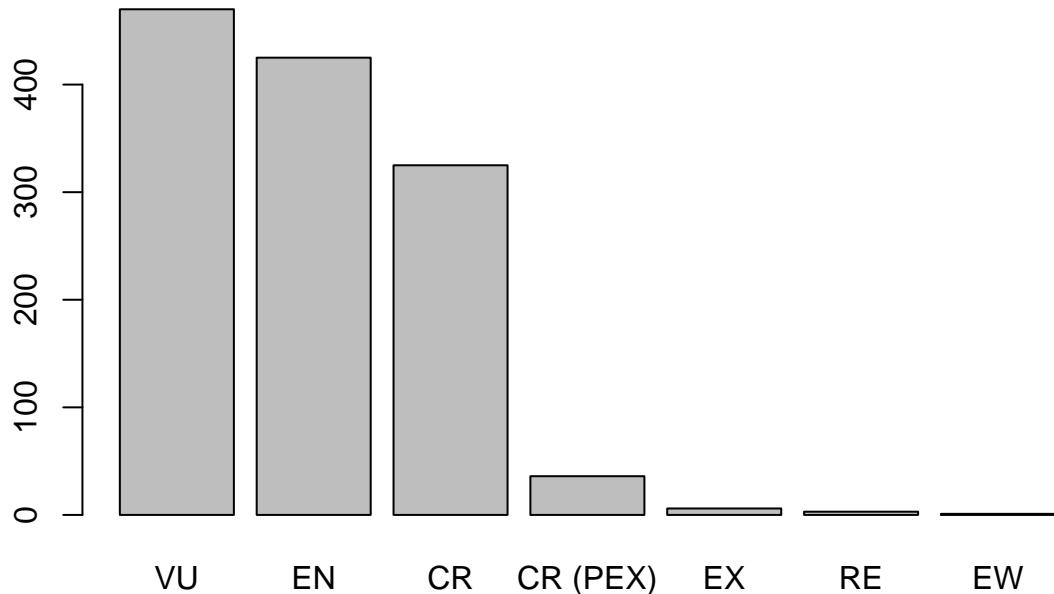
# Converter os dados da coluna categoria para fatores
fauna$categoria <- factor(
  fauna$categoria,
  levels = c('VU', 'EN', 'CR', 'CR (PEX)', 'EX', 'RE', 'EW'),
  ordered = TRUE
)

# Resumo da coluna categoria
summary(fauna$categoria)

##      VU        EN       CR CR (PEX)      EX        RE        EW
##      470       425      325       36       6       3       1

# Visualização Gráfica dos Fatores
plot(fauna$categoria)

```



## 16 Listas

Imagine que você precisa ir ao supermercado, para se manter organizado e lembrar o que comprar, você provavelmente fará uma lista de compras. Pense no que pode haver nessa lista: Cereais, papel toalha, frutas, detergente... Vários tipos de coisas com formas e estruturas diferentes. Não importa o que seja, se você

puder lembar, poderá colocá-los em sua lista. Assim como sua lista de compras, em R, existe um tipo de superestrutura de dados, também chamada de Lista, que permite agrupar outras estruturas de dados em, digamos, uma lista.

Para criar uma lista é simples, basta usar a função `list()` e passar todas as estruturas de dados que você deseja dentro de sua lista.

Agora iremos criar a seguinte lista em R:

- farinha de trigo
- farinha de mandioca
- farinha de milho
- creme dental
- papel higiênico
- sabonete
- água sanitária
- detergente
- carne
- frango
- peixe
- cerveja
- sonrisal
- eparema

```
lista <- list(c('farinha de trigo', 'farinha de mandioca', 'farinha de milho'),
              c('creme dental', 'papel higiênico', 'sabonete'),
              c('água sanitária', 'detergente'),
              c('carne', 'frango', 'peixe'),
              c('cerveja', 'sonrisal', 'eparema'))

print(lista)

## [[1]]
## [1] "farinha de trigo"      "farinha de mandioca" "farinha de milho"
##
## [[2]]
## [1] "creme dental"       "papel higiênico" "sabonete"
##
## [[3]]
## [1] "água sanitária" "detergente"
##
## [[4]]
## [1] "carne"    "frango"   "peixe"
##
## [[5]]
## [1] "cerveja"  "sonrisal" "eparema"
```

## 16.1 Acessando elementos em uma lista

```
# Selecionar todos os valores do primeiro ítem da lista
print(lista[[1]])

## [1] "farinha de trigo"      "farinha de mandioca" "farinha de milho"

# Todos os elementos do terceiro vetor da lista
print(lista[[3]])
```

```

## [1] "água sanitária" "detergente"
# Primeiro valor do primeiro da lista
print(lista[[1]][[1]])

## [1] "farinha de trigo"
# Quarto e Quinto ítems da lista
nao_esquecer <- lista[c(4, 5)]
print(nao_esquecer)

## [[1]]
## [1] "carne"   "frango"  "peixe"
##
## [[2]]
## [1] "cerveja" "sonrisal" "eparema"

```

## 16.2 Nomear os elementos de uma lista

```

# Definir nomes para os elementos da lista
names(lista) <- c('cereais', 'higiene', 'limpeza',
                   'proteína', 'birita')

# No primeiro ítem da lista, filtrar apenas o primeiro sub-ítem
lista$cereais[[1]]

## [1] "farinha de trigo"

# No segundo ítem da lista, filtrar o primeiro sub-ítem
# No elemento flora, filtrar a primeira espécie
lista$higiene[[1]]

## [1] "creme dental"

```

## 16.3 Funções Aplicadas à lista

Praticamente todas as funções vistas nas aulas anteriores se aplicam à lista.

### 16.3.1 str e summary

```

# Verificar a estrutura da lista
str(lista)

## List of 5
## $ cereais : chr [1:3] "farinha de trigo" "farinha de mandioca" "farinha de milho"
## $ higiene : chr [1:3] "creme dental" "papel higiênico" "sabonete"
## $ limpeza : chr [1:2] "água sanitária" "detergente"
## $ proteína: chr [1:3] "carne" "frango" "peixe"
## $ birita  : chr [1:3] "cerveja" "sonrisal" "eparema"

# Obter um resumo dos dados armazenados na lista
summary(lista)

##          Length Class  Mode
## cereais     3    -none- character
## higiene     3    -none- character
## limpeza     2    -none- character
## proteína    3    -none- character

```

```
## biritá 3      -none- character
```

### 16.3.2 lapply e sapply

As funções `lapply` e `sapply` são uma variação da função `apply` para iterar sobre aos elementos de uma lista. Esta funções diferem somente na forma como os resultados são mostrados na tela. Para a função `lapply` a saída é na forma de lista, enquanto `sapply` apresenta uma saída vetorial.

```
lista = list(x = c(1.55, 1.15, 1.71, 1.89, 1.65, 1.44),  
             y = c(12.51, 9.89, 21.22, 22.11, 9.98))
```

```
print(lapply(lista, mean))
```

```
## $x  
## [1] 1.565  
##  
## $y  
## [1] 15.142
```

*# Média para cada elemento da lista*  
print(sapply(lista, mean))

```
##      x      y  
## 1.565 15.142
```

*# Média apenas do elemento x da lista*  
print(mean(lista[[1]]))

```
## [1] 1.565
```

### 16.3.3 toString()

```
print(toString(lista))
```

```
## [1] "c(1.55, 1.15, 1.71, 1.89, 1.65, 1.44), c(12.51, 9.89, 21.22, 22.11, 9.98)"
```

## 16.4 Converter Vetor para Lista

Há duas maneiras de converter um vetor para lista, uma delas é usar a função `list()` e passar como parâmetro o nome do vetor, a outra forma é usar a função `as.list()`. A diferença entre essas essas função é que `list()` produzirá uma lista formada pelos vários elementos do vetor, ao passo que a função `as.list()` irá gerar uma lista para cada elemento do vetor, conforme demonstrado no código a seguir.

```
# Definição do vetor "numeros"  
numeros <- c(1:26)
```

```
# Definição do vetor "letras"  
letras <- letters
```

```
lista <- list(numeros, toupper(letras))
```

```
print(lista)
```

```
## [[1]]  
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25  
## [26] 26  
##  
## [[2]]
```

```
## [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S"  
## [20] "T" "U" "V" "W" "X" "Y" "Z"
```

#### 16.4.1 as.list

```
x = c(1.55, 1.15, 1.71, 1.89, 1.65, 1.44)  
  
lista2 <- as.list(x)  
  
print(head(lista2))  
  
## [[1]]  
## [1] 1.55  
##  
## [[2]]  
## [1] 1.15  
##  
## [[3]]  
## [1] 1.71  
##  
## [[4]]  
## [1] 1.89  
##  
## [[5]]  
## [1] 1.65  
##  
## [[6]]  
## [1] 1.44
```

#### 16.4.2 list

```
# Converter o vetor "numeros" para lista usando a função list()  
print(list(x))
```

```
## [[1]]  
## [1] 1.55 1.15 1.71 1.89 1.65 1.44
```

Podemos aplicar função aos vetores no momento de criação da lista

```
# Definição do vetor "numeros"  
numeros <- c(1:26)  
  
# Definição do vetor "letras"  
letras <- letters  
  
lista <- list(numeros, toupper(letras))  
  
print(lista)  
  
## [[1]]  
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25  
## [26] 26  
##  
## [[2]]  
## [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S"
```

```

## [20] "T" "U" "V" "W" "X" "Y" "Z"
## Converter Dataframe para Lista

# definir o dataframe
df <- data.frame(a = c(1:5), b = c(6:10))

# Mostrar os dados do dataframe
df

##   a   b
## 1 1   6
## 2 2   7
## 3 3   8
## 4 4   9
## 5 5  10

# converter o dataframe para lista utilizando a função list()
lista_df <- list(df)
print(lista_df)

## [[1]]
##   a   b
## 1 1   6
## 2 2   7
## 3 3   8
## 4 4   9
## 5 5  10

# converter o dataframe para lista utilizando a função as.list()
lista_df <- as.list(df)
print(lista_df)

## $a
## [1] 1 2 3 4 5
##
## $b
## [1] 6 7 8 9 10

```

`as.list()` converte cada elemento do vetor em um elemento de lista, no caso de data frame cada coluna será uma elemento de lista.

## 16.5 Conhecendo a Versatilidade de uma lista

A lista a seguir contém todas as estruturas de dados que conhecemos neste curso, e formada pelo vetor de diâmetros de árvores (`dap`), uma matriz, um *array*, um *data frame* e uma lista. Observe que podemos inserir lista dentro de lista.

```

li <- readRDS('./data/lista_exemplo.rds')

str(li)

## List of 5
## $ Vetor      : num [1:18406] 114.6 67.5 52.5 40.7 47.8 ...
##   ..- attr(*, "Diâmetro de Árvores")= chr "Vetor de diâmetros"
## $ Matriz     : int [1:3, 1:3] 1 2 3 4 5 6 7 8 9
##   ..- attr(*, "Observação")= chr "Exemplo de Matriz criada no curso"
## $ Array      : int [1:3, 1:3, 1:2] 1 2 3 4 5 6 7 8 9 10 ...

```

```

## ..- attr(*, "Observação")= chr "Exemplo de Array criado no curso"
## $ Data frame:'data.frame': 5 obs. of 3 variables:
##   ..$ id    : num [1:5] 1 2 3 4 5
##   ..$ nome  : chr [1:5] "Mezilaurus itauba" "Apuleia leiocarpa" "Cedrela odorata" "Amburana acreana"
##   ..$ volume: num [1:5] 3.25 6.51 7.45 8.81 4.35
##   ..- attr(*, "Observação")= chr "Primeiro data frame criado no curso"
## $ Lista   :List of 5
##   ..$ cereais : chr [1:3] "farinha de trigo" "farinha de mandioca" "farinha de milho"
##   ..$ higiene : chr [1:3] "creme dental" "papel higiênico" "sabonete"
##   ..$ limpeza : chr [1:2] "água sanitária" "detergente"
##   ..$ proteína: chr [1:3] "carne" "frango" "peixe"
##   ..$ biritá  : chr [1:3] "cerveja" "sonrisal" "eparema"
##   ..- attr(*, "Observação")= chr "Primeira lista criado no curso"

```

### 16.5.1 Mostrar o Vetor Presente na lista

```

head(li$Vetor)

## [1] 114.59 67.48 52.52 40.74 47.75 43.93

```

### 16.5.2 Extrairndo a Matriz Presente na Lista

```

li$Matriz

##      [,1] [,2] [,3]
## [1,]     1     4     7
## [2,]     2     5     8
## [3,]     3     6     9
## attr(,"Observação")
## [1] "Exemplo de Matriz criada no curso"

```

### 16.5.3 Extrairndo a Array Presente na Lista

```

li$array

## , , 1
##
##      [,1] [,2] [,3]
## [1,]     1     4     7
## [2,]     2     5     8
## [3,]     3     6     9
##
## , , 2
##
##      [,1] [,2] [,3]
## [1,]    10    13    16
## [2,]    11    14    17
## [3,]    12    15    18
##
## attr(,"Observação")
## [1] "Exemplo de Array criado no curso"

```

#### 16.5.4 Extraindo o Data Frame Presente na Lista

```
li$`Data frame`  
  
##   id           nome volume  
## 1 1    Mezilaurus itauba 3.25  
## 2 2     Apuleia leiocarpa 6.51  
## 3 3      Cedrela odorata 7.45  
## 4 4    Amburana acreana 8.81  
## 5 5 Hymenolobium excelsum 4.35
```

#### 16.5.5 Extraindo a Lista Presente na Lista

```
li$Lista  
  
## $cereais  
## [1] "farinha de trigo"    "farinha de mandioca" "farinha de milho"  
##  
## $higiene  
## [1] "creme dental"       "papel higiênico" "sabonete"  
##  
## $limpeza  
## [1] "água sanitária" "detergente"  
##  
## $proteína  
## [1] "carne"    "frango"    "peixe"  
##  
## $birita  
## [1] "cerveja"   "sonrisal"   "eparema"  
##  
## attr(,"Observação")  
## [1] "Primeira lista criado no curso"
```

## 17 Aplicação do uso de lista em R

A seguir conhiceremos uma lista gerada através de webscraping na página da Impresa Nacional, utilizando o pacote R Rvest. O objetivo foi extrair os dados das espécies da flora e fauna ameaçadas de extinção.

```
# carregar a lista das espécies da fauna  
lista_fauna <- readRDS('./data/lista_classes_fauna_ameacada.rds')  
  
# Resumir os dados da lista  
str(lista_fauna)  
  
## List of 28  
## $ Actinopterygii:'data.frame': 329 obs. of 7 variables:  
##   ..$ n          : int [1:329] 129 191 192 193 194 195 196 197 198 199 ...  
##   ..$ port443    : chr [1:329] "*" "*" "*" "*" ...  
##   ..$ classe     : chr [1:329] "Actinopterygii" "Actinopterygii" "Actinopterygii" "Actinopterygii" ...  
##   ..$ ordem      : chr [1:329] "Atheriniformes" "Characiformes" "Characiformes" "Characiformes" ...  
##   ..$ familia    : chr [1:329] "Atherinopsidae" "Anostomidae" "Anostomidae" "Anostomidae" ...  
##   ..$ especie_subespecie: chr [1:329] "Odontesthes bicudo" "Hypomasticus thayeri" "Leporinus guttatus" "Leporinus guttatus" ...  
##   ..$ categoria   : chr [1:329] "EN" "VU" "VU" "CR" ...  
## $ Amphibia     :'data.frame': 59 obs. of 7 variables:  
##   ..$ n          : int [1:59] 33 34 37 38 39 40 41 42 43 44 ...
```

```

##   ..$ port443      : chr [1:59] "" "*" "" "*" ...
##   ..$ classe       : chr [1:59] "Amphibia" "Amphibia" "Amphibia" "Amphibia" ...
##   ..$ ordem        : chr [1:59] "Anura" "Anura" "Anura" "Anura" ...
##   ..$ familia      : chr [1:59] "Hylidae" "Phyllomedusidae" "Brachycephalidae" "Brachycephalidae"
##   ..$ especie_subespecie: chr [1:59] "Boana cymbalum" "Phrynomedusa fimbriata" "Brachycephalus mirissimus"
##   ..$ categoria    : chr [1:59] "EX" "EX" "VU" "CR" ...
## $ Anthozoa       :'data.frame': 3 obs. of 7 variables:
##   ..$ n            : int [1:3] 7 1065 1066
##   ..$ port443     : chr [1:3] "*" "*" "*"
##   ..$ classe      : chr [1:3] "Anthozoa" "Anthozoa" "Anthozoa"
##   ..$ ordem       : chr [1:3] "Actiniaria" "Scleractinia" "Scleractinia"
##   ..$ familia     : chr [1:3] "Actiniidae" "Mussidae" "Mussidae"
##   ..$ especie_subespecie: chr [1:3] "Condylactis gigantea" "Mussismilia brasiliensis" "Mussismilia haematocheila"
##   ..$ categoria    : chr [1:3] "EN" "VU" "EN"
## $ Arachnida      :'data.frame': 69 obs. of 7 variables:
##   ..$ n            : int [1:69] 8 9 10 11 12 13 14 15 16 105 ...
##   ..$ port443     : chr [1:69] "*" "*" "" "*" ...
##   ..$ classe      : chr [1:69] "Arachnida" "Arachnida" "Arachnida" "Arachnida" ...
##   ..$ ordem       : chr [1:69] "Amblypygi" "Amblypygi" "Amblypygi" "Amblypygi" ...
##   ..$ familia     : chr [1:69] "Charinidae" "Charinidae" "Charinidae" "Charinidae" ...
##   ..$ especie_subespecie: chr [1:69] "Charinus acaraje" "Charinus asturius" "Charinus caatingae" "Charinus caatingae"
##   ..$ categoria    : chr [1:69] "VU" "EN" "CR" "EN" ...
## $ Asteroidea      :'data.frame': 7 obs. of 7 variables:
##   ..$ n            : int [1:7] 499 845 846 847 848 1277 1278
##   ..$ port443     : chr [1:7] "*" "*" "*" "*" ...
##   ..$ classe      : chr [1:7] "Asteroidea" "Asteroidea" "Asteroidea" "Asteroidea" ...
##   ..$ ordem       : chr [1:7] "Forcipulatida" "Paxillosida" "Paxillosida" "Paxillosida" ...
##   ..$ familia     : chr [1:7] "Asteriidae" "Astropectinidae" "Astropectinidae" "Astropectinidae"
##   ..$ especie_subespecie: chr [1:7] "Coscinasterias tenuispina" "Astropecten articulatus" "Astropecten articulatus"
##   ..$ categoria    : chr [1:7] "VU" "VU" "VU" "VU" ...
## $ Atheriniformes:'data.frame': 3 obs. of 7 variables:
##   ..$ n            : int [1:3] 862 863 864
##   ..$ port443     : chr [1:3] "*" "*" "*"
##   ..$ classe      : chr [1:3] "Atheriniformes" "Atheriniformes" "Atheriniformes"
##   ..$ ordem       : chr [1:3] "Perciformes" "Perciformes" "Perciformes"
##   ..$ familia     : chr [1:3] "Epinephelidae" "Epinephelidae" "Epinephelidae"
##   ..$ especie_subespecie: chr [1:3] "Epinephelus itajara" "Epinephelus marginatus" "Epinephelus morio"
##   ..$ categoria    : chr [1:3] "CR" "VU" "VU"
## $ Aves            :'data.frame': 263 obs. of 7 variables:
##   ..$ n            : int [1:263] 1 2 3 4 5 6 31 95 96 97 ...
##   ..$ port443     : chr [1:263] "*" "*" "*" "*" ...
##   ..$ classe      : chr [1:263] "Aves" "Aves" "Aves" "Aves" ...
##   ..$ ordem       : chr [1:263] "Accipitriformes" "Accipitriformes" "Accipitriformes" "Accipitriformes"
##   ..$ familia     : chr [1:263] "Accipitridae" "Accipitridae" "Accipitridae" "Accipitridae" ...
##   ..$ especie_subespecie: chr [1:263] "Amadonastur lacernulatus" "Circus cinereus" "Harpia harpyja"
##   ..$ categoria    : chr [1:263] "VU" "VU" "VU" "EN" ...
## $ Bivalvia        :'data.frame': 3 obs. of 7 variables:
##   ..$ n            : int [1:3] 679 1275 1276
##   ..$ port443     : chr [1:3] "*" "*" "*"
##   ..$ classe      : chr [1:3] "Bivalvia" "Bivalvia" "Bivalvia"
##   ..$ ordem       : chr [1:3] "Ostreoida" "Unionoida" "Unionoida"
##   ..$ familia     : chr [1:3] "Pectinidae" "Hyriidae" "Mycetopodidae"
##   ..$ especie_subespecie: chr [1:3] "Euvola ziczac" "Diplodon koseritzii" "Mycetopoda legumen"
##   ..$ categoria    : chr [1:3] "EN" "EN" "EN"

```

```

## $ Chilopoda      :'data.frame': 6 obs. of 7 variables:
##   ..$ n           : int [1:6] 1067 1068 1069 1070 1071 1072
##   ..$ port443    : chr [1:6] "*" "*" "*" " "
##   ..$ classe     : chr [1:6] "Chilopoda" "Chilopoda" "Chilopoda" "Chilopoda" ...
##   ..$ ordem      : chr [1:6] "Scolopendromorpha" "Scolopendromorpha" "Scolopendromorpha" "Scolopendromorpha" ...
##   ..$ familia    : chr [1:6] "Cryptopidae" "Cryptopidae" "Scolopendridae" "Scolopocryptopidae"
##   ..$ especie_subespecie: chr [1:6] "Cryptopsiporangensis" "Cryptopsspelaeoraptor" "Scolopendropsisdi"
##   ..$ categoria   : chr [1:6] "EN" "CR" "CR" "VU" ...
## $ Chondrichthyes:'data.frame': 61 obs. of 7 variables:
##   ..$ n           : int [1:61] 137 138 139 140 141 142 143 144 145 146 ...
##   ..$ port443    : chr [1:61] " " " " " "
##   ..$ classe     : chr [1:61] "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" "Chondrichthyes" ...
##   ..$ ordem      : chr [1:61] "Carcharhiniformes" "Carcharhiniformes" "Carcharhiniformes" "Carcharhiniformes"
##   ..$ familia    : chr [1:61] "Carcharhinidae" "Carcharhinidae" "Carcharhinidae" "Carcharhinidae"
##   ..$ especie_subespecie: chr [1:61] "Carcharhinus acronotus" "Carcharhinus brevipinna" "Carcharhinus ...
##   ..$ categoria   : chr [1:61] "VU" "VU" "CR" "VU" ...
## $ Collembola     :'data.frame': 39 obs. of 7 variables:
##   ..$ n           : int [1:39] 293 294 295 296 297 298 299 300 301 302 ...
##   ..$ port443    : chr [1:39] "*" " " " " "
##   ..$ classe     : chr [1:39] "Collembola" "Collembola" "Collembola" ...
##   ..$ ordem      : chr [1:39] "Collembola" "Collembola" "Collembola" ...
##   ..$ familia    : chr [1:39] "Arrhopalitidae" "Arrhopalitidae" "Entomobryidae" "Entomobryidae"
##   ..$ especie_subespecie: chr [1:39] "Arrhopalites amorimi" "Arrhopalites glabrofasciatus" "Pseudosinell ...
##   ..$ categoria   : chr [1:39] "VU" "EN" "VU" "CR" ...
## $ Demospongea    :'data.frame': 5 obs. of 7 variables:
##   ..$ n           : int [1:5] 916 1166 1167 1243 1244
##   ..$ port443    : chr [1:5] "*" "*" "*" "*"
##   ..$ classe     : chr [1:5] "Demospongea" "Demospongea" "Demospongea" ...
##   ..$ ordem      : chr [1:5] "Poecilosclerida" "Spongillida" "Suberitida" ...
##   ..$ familia    : chr [1:5] "Latrunculiidae" "Spongillidae" "Metaniidae" "Halichondriidae" ...
##   ..$ especie_subespecie: chr [1:5] "Latrunculia janeirensis" "Racekiela cavernicola" "Corvomeyen ...
##   ..$ categoria   : chr [1:5] "VU" "VU" "VU" "VU" ...
## $ Diplopoda      :'data.frame': 14 obs. of 7 variables:
##   ..$ n           : int [1:14] 35 36 513 917 918 919 920 921 922 1161 ...
##   ..$ port443    : chr [1:14] " " " *" "*"
##   ..$ classe     : chr [1:14] "Diplopoda" "Diplopoda" "Diplopoda" ...
##   ..$ ordem      : chr [1:14] "Anura" "Anura" "Glomeridesmida" "Polydesmida" ...
##   ..$ familia    : chr [1:14] "Aromobatidae" "Aromobatidae" "Glomerodesmidae" "Chelodesmidae"
##   ..$ especie_subespecie: chr [1:14] "Anomaloglossus apiau" "Anomaloglossus tequem" "Glomeridesmus ...
##   ..$ categoria   : chr [1:14] "EN" "CR (PEX)" "EN" "VU" ...
## $ Diplura        :'data.frame': 1 obs. of 7 variables:
##   ..$ n           : int 471
##   ..$ port443    : chr ""
##   ..$ classe     : chr "Diplura"
##   ..$ ordem      : chr "Diplura"
##   ..$ familia    : chr "Campodeidae"
##   ..$ especie_subespecie: chr "Oncinocampa trajanoae"
##   ..$ categoria   : chr "CR"
## $ Echinoidea     :'data.frame': 2 obs. of 7 variables:
##   ..$ n           : int [1:2] 134 176
##   ..$ port443    : chr [1:2] "*" "*"
##   ..$ classe     : chr [1:2] "Echinoidea" "Echinoidea"
##   ..$ ordem      : chr [1:2] "Camarodonta" "Cassiduloida"
##   ..$ familia    : chr [1:2] "Toxopneustidae" "Cassidulidae"

```

```

##   ..$ especie_subespecie: chr [1:2] "Lytechinus variegatus" "Cassidulus mitis"
##   ..$ categoria      : chr [1:2] "VU" "EN"
## $ Enteropneusta :'data.frame': 1 obs. of 7 variables:
##   ..$ n            : int 473
##   ..$ port443     : chr "*"
##   ..$ classe       : chr "Enteropneusta"
##   ..$ ordem        : chr "Enteropneusta"
##   ..$ familia      : chr "Spengelidae"
##   ..$ especie_subespecie: chr "Willeya loya"
##   ..$ categoria      : chr "CR"
## $ Gastropoda    :'data.frame': 18 obs. of 7 variables:
##   ..$ n            : int [1:18] 131 132 133 520 641 642 643 644 1003 1004 ...
##   ..$ port443     : chr [1:18] "*" "*" "*" ...
##   ..$ classe       : chr [1:18] "Gastropoda" "Gastropoda" "Gastropoda" ...
##   ..$ ordem        : chr [1:18] "Caenogastropoda" "Caenogastropoda" "Caenogastropoda" "Gymnomor...
##   ..$ familia      : chr [1:18] "Hydrobiidae" "Ampullariidae" "Hydrobiidae" "Veronicellidae" ...
##   ..$ especie_subespecie: chr [1:18] "Potamolithus troglobius" "Pomacea sordida" "Potamolithus karst...
##   ..$ categoria      : chr [1:18] "CR" "EN" "CR" "CR" ...
## $ Holothuroidea :'data.frame': 1 obs. of 7 variables:
##   ..$ n            : int 94
##   ..$ port443     : chr "*"
##   ..$ classe       : chr "Holothuroidea"
##   ..$ ordem        : chr "Apodida"
##   ..$ familia      : chr "Synaptidae"
##   ..$ especie_subespecie: chr "Synaptula secreta"
##   ..$ categoria      : chr "CR"
## $ Hydrozoa       :'data.frame': 1 obs. of 7 variables:
##   ..$ n            : int 32
##   ..$ port443     : chr "*"
##   ..$ classe       : chr "Hydrozoa"
##   ..$ ordem        : chr "Anthoathecata"
##   ..$ familia      : chr "Milleporidae"
##   ..$ especie_subespecie: chr "Millepora laboreli"
##   ..$ categoria      : chr "VU"
## $ Insecta         :'data.frame': 146 obs. of 7 variables:
##   ..$ n            : int [1:146] 130 261 262 263 264 265 266 267 268 269 ...
##   ..$ port443     : chr [1:146] "" "*" "*" ...
##   ..$ classe       : chr [1:146] "Insecta" "Insecta" "Insecta" ...
##   ..$ ordem        : chr [1:146] "Blattodea" "Coleoptera" "Coleoptera" "Coleoptera" ...
##   ..$ familia      : chr [1:146] "Blattellidae" "Dytiscidae" "Carabidae" "Carabidae" ...
##   ..$ especie_subespecie: chr [1:146] "Litoblatta camargoi" "Copelatus cessaima" "Coarazuphium amazon...
##   ..$ categoria      : chr [1:146] "EN" "EN" "CR" "EN" ...
## $ Malacostraca   :'data.frame': 46 obs. of 7 variables:
##   ..$ n            : int [1:46] 17 18 19 20 21 22 23 24 25 26 ...
##   ..$ port443     : chr [1:46] "" "" "" ...
##   ..$ classe       : chr [1:46] "Malacostraca" "Malacostraca" "Malacostraca" "Malacostraca" ...
##   ..$ ordem        : chr [1:46] "Amphipoda" "Amphipoda" "Amphipoda" "Amphipoda" ...
##   ..$ familia      : chr [1:46] "Artesiidae" "Artesiidae" "Artesiidae" "Artesiidae" ...
##   ..$ especie_subespecie: chr [1:46] "Spelaeogammarus bahiensis" "Spelaeogammarus sanctus" "Spelaeog...
##   ..$ categoria      : chr [1:46] "VU" "CR" "CR" "EN" ...
## $ Mammalia        :'data.frame': 104 obs. of 7 variables:
##   ..$ n            : int [1:104] 160 161 162 163 164 168 169 170 171 172 ...
##   ..$ port443     : chr [1:104] "*" "*" "*" ...
##   ..$ classe       : chr [1:104] "Mammalia" "Mammalia" "Mammalia" "Mammalia" ...

```

```

##   ..$ ordem      : chr [1:104] "Carnivora" "Carnivora" "Carnivora" "Carnivora" ...
##   ..$ familia     : chr [1:104] "Canidae" "Canidae" "Canidae" "Canidae" ...
##   ..$ especie_subespecie: chr [1:104] "Atelocynus microtis" "Chrysocyon brachyurus" "Lycalopex vetulus"
##   ..$ categoria    : chr [1:104] "VU" "VU" "VU" "VU" ...
## $ Oligochaeta   :'data.frame': 1 obs. of  7 variables:
##   ..$ n          : int 536
##   ..$ port443    : chr "*"
##   ..$ classe     : chr "Oligochaeta"
##   ..$ ordem      : chr "Haplotauxida"
##   ..$ familia     : chr "Glossoscolecidae"
##   ..$ especie_subespecie: chr "Fimoscolex sporadochaetus"
##   ..$ categoria    : chr "EN"
## $ Polychaeta    :'data.frame': 2 obs. of  7 variables:
##   ..$ n          : int [1:2] 493 494
##   ..$ port443    : chr [1:2] "*" "*"
##   ..$ classe     : chr [1:2] "Polychaeta" "Polychaeta"
##   ..$ ordem      : chr [1:2] "Eunicida" "Eunicida"
##   ..$ familia     : chr [1:2] "Eunicidae" "Onuphidae"
##   ..$ especie_subespecie: chr [1:2] "Eunice sebastiani" "Diopatra cuprea"
##   ..$ categoria    : chr [1:2] "EN" "VU"
## $ Reptilia      :'data.frame': 71 obs. of  7 variables:
##   ..$ n          : int [1:71] 1168 1169 1170 1171 1172 1173 1174 1175 1176 1177 ...
##   ..$ port443    : chr [1:71] "*" "*" "*" "*"
##   ..$ classe     : chr [1:71] "Reptilia" "Reptilia" "Reptilia" "Reptilia" ...
##   ..$ ordem      : chr [1:71] "Squamata" "Squamata" "Squamata" "Squamata" ...
##   ..$ familia     : chr [1:71] "Amphisbaenidae" "Amphisbaenidae" "Amphisbaenidae" "Amphisbaenidae"
##   ..$ especie_subespecie: chr [1:71] "Amphisbaena arda" "Amphisbaena frontalis" "Amphisbaena nigricans"
##   ..$ categoria    : chr [1:71] "EN" "EN" "EN" "EN" ...
## $ Rhynchonellata:'data.frame': 1 obs. of  7 variables:
##   ..$ n          : int 1257
##   ..$ port443    : chr "*"
##   ..$ classe     : chr "Rhynchonellata"
##   ..$ ordem      : chr "Terebratulida"
##   ..$ familia     : chr "Bouchardiidae"
##   ..$ especie_subespecie: chr "Bouchardia rosea"
##   ..$ categoria    : chr "EN"
## $ Turbellaria    :'data.frame': 6 obs. of  7 variables:
##   ..$ n          : int [1:6] 1268 1269 1270 1271 1272 1273
##   ..$ port443    : chr [1:6] "" "" "" ""
##   ..$ classe     : chr [1:6] "Turbellaria" "Turbellaria" "Turbellaria" "Turbellaria" "Turbellaria" ...
##   ..$ ordem      : chr [1:6] "Tricladida" "Tricladida" "Tricladida" "Tricladida" ...
##   ..$ familia     : chr [1:6] "Dimarcusidae" "Dugesiidae" "Dugesiidae" "Dugesiidae" ...
##   ..$ especie_subespecie: chr [1:6] "Hausera hauseri" "Girardia arenicola" "Girardia desiderensis" "Girardia desiderensis"
##   ..$ categoria    : chr [1:6] "VU" "CR" "CR" "CR" ...
## $ Udeonychophora:'data.frame': 4 obs. of  7 variables:
##   ..$ n          : int [1:4] 495 496 497 498
##   ..$ port443    : chr [1:4] "*" "*" "*" "*"
##   ..$ classe     : chr [1:4] "Udeonychophora" "Udeonychophora" "Udeonychophora" "Udeonychophora"
##   ..$ ordem      : chr [1:4] "Euonychophora" "Euonychophora" "Euonychophora" "Euonychophora"
##   ..$ familia     : chr [1:4] "Peripatidae" "Peripatidae" "Peripatidae" "Peripatidae"
##   ..$ especie_subespecie: chr [1:4] "Epiperipatus adenocryptus" "Epiperipatus diadenoproctus" "Epiperipatus diadenoproctus"
##   ..$ categoria    : chr [1:4] "CR" "VU" "EN" "CR"

```

```

# Selecionar e mostrar a classe das aves
head(lista_fauna$Aves)

##   n port443 classe      ordem     familia      especie_subespecie
## 1 1      * Aves Accipitriformes Accipitridae Amadonastur lacernulatus
## 2 2      * Aves Accipitriformes Accipitridae          Circus cinereus
## 3 3      * Aves Accipitriformes Accipitridae          Harpia harpyja
## 4 4      * Aves Accipitriformes Accipitridae          Leptodon forbesi
## 5 5      * Aves Accipitriformes Accipitridae        Morphnus guianensis
## 6 6      * Aves Accipitriformes Accipitridae       Urubitinga coronata
##   categoria
## 1      VU
## 2      VU
## 3      VU
## 4      EN
## 5      VU
## 6      EN

tail(lista_fauna$Aves)

##      n port443 classe      ordem     familia      especie_subespecie
## 1231 1247      * Aves Suliformes    Sulidae           Sula sula
## 1247 1263      * Aves Tinamiformes Tinamidae      Crypturellus zabele
## 1248 1264      * Aves Tinamiformes Tinamidae          Nothura minor
## 1249 1265      * Aves Tinamiformes Tinamidae          Taoniscus nanus
## 1250 1266      * Aves Tinamiformes Tinamidae          Tinamus tao
## 1258 1274      * Aves Trogoniformes Trogonidae Tropidon collaris eytoni
##   categoria
## 1231      EN
## 1247      VU
## 1248      EN
## 1249      EN
## 1250      VU
## 1258      EN

```