

# Estatística e Informática

## Aula 03 - Distribuição de Frequência

Alan Rodrigo Panosso [alan.panosso@unesp.br](mailto:alan.panosso@unesp.br)

Departamento de Engenharia e Ciências Exatas FCAV/UNESP

(14-03-2024)

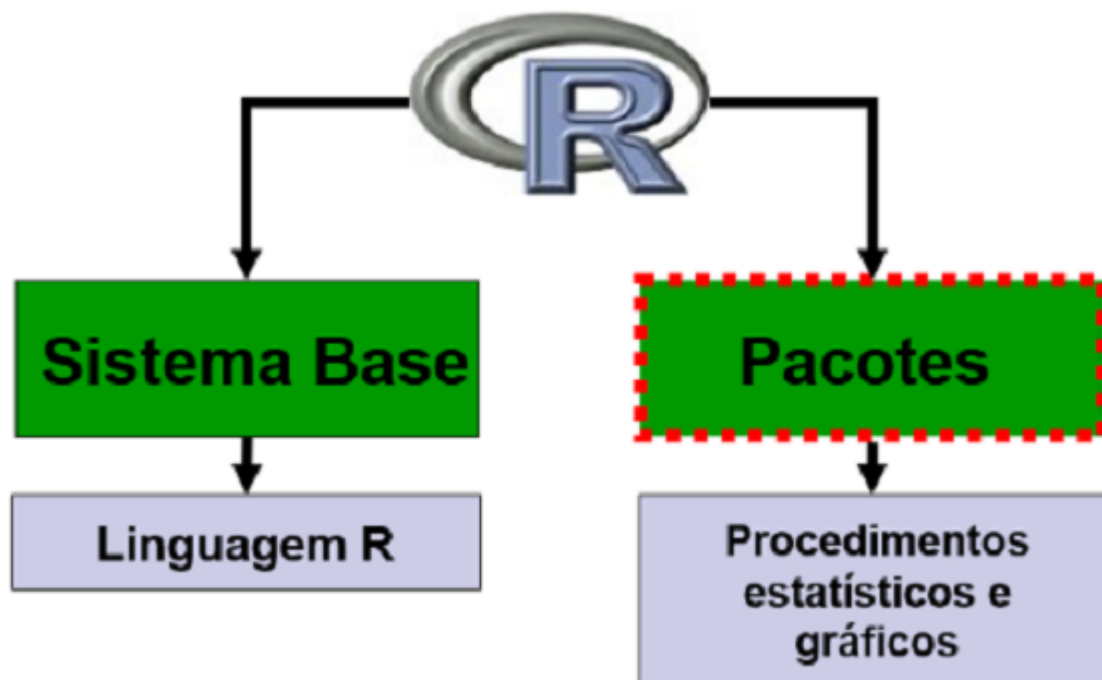
# Pré-requisito: Instalação de Patocotes no R



<https://youtu.be/is32jC4LRjA>

## Pacote em R

Um pacote é uma coleção de funções, exemplos e documentação. A funcionalidade de um pacote é frequentemente focada em uma metodologia estatística especial" (Everitt & Hothorn).



Pacotes no R são coleções de funções, exemplos e documentações, que devem ser previamente instalados e alocados no ambiente pela função `library`.

# Pacotes básicos

Liste os pacotes carregados no ambiente com:

```
(.packages())
```

```
#> [1] "readxl"      "agricolae"  "lubridate"  "forcats"  
#> [5] "stringr"     "dplyr"      "purrr"      "readr"  
#> [9] "tidyr"       "tibble"     "ggplot2"    "tidyverse"  
#> [13] "MASS"        "stats"      "graphics"   "grDevices"  
#> [17] "utils"       "datasets"   "methods"    "base"
```

O retorno da função é uma lista de nomes, caracteres (ou strings), na forma de um *objeto* denominado **veter**. Observe que cada pacote (elemento) é referenciado dentro do vetor por um índice, um número inteiro [ *i* ] apresentado entre colchetes [**i**].

Carregue um pacote chamando a função `library`.

```
library(MASS)
```

Agora, liste novamente os pacotes e observe a diferença no retorno da função.

```
(.packages())
```

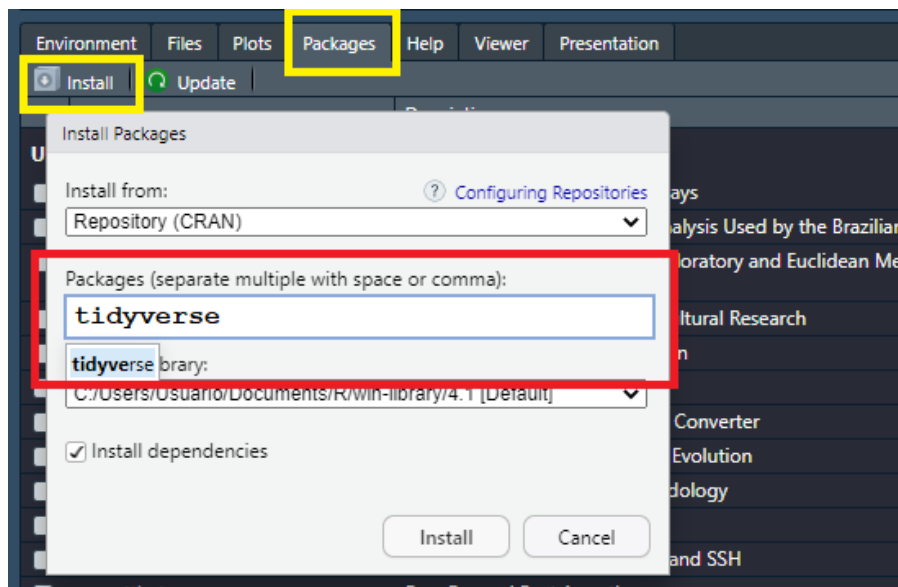
```
#> [1] "readxl"      "agricolae"  "lubridate"  "forcats"  
#> [5] "stringr"     "dplyr"      "purrr"      "readr"  
#> [9] "tidyr"       "tibble"     "ggplot2"    "tidyverse"  
#> [13] "MASS"        "stats"      "graphics"   "grDevices"  
#> [17] "utils"       "datasets"   "methods"    "base"
```

Observe que o pacote MASS agora está no ambiente de trabalho.

# Instalando pacotes

Para a realização de procedimentos estatístico e manipulação de arquivos durante o curso, serão necessários vários pacotes que não fazem parte do base do R, que deverão ser instalados.

**Utilizando a opção** Packages\Install\nome do pacote



Instale os pacotes:

- tidyverse
- agricolae

Os pacotes também podem ser instalados a partir das linhas de comandos:

```
install.packages("tidyverse")  
install.packages("agricolae")
```

Agora podemos carregar esses pacotes em nosso ambiente de trabalho.

```
library(tidyverse)  
library(agricolae)
```

Vamos agora criar uma "Pipe Line" para visualização de um banco de dados

```
mtcars %>%  
  glimpse()
```

```
#> Rows: 32  
#> Columns: 11  
#> $ mpg   <dbl> 21.0, 21.0, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4,...  
#> $ cyl   <dbl> 6, 6, 4, 6, 8, 6, 8, 4, 4, 6, 6, 8, 8, 8, 8, 8,...  
#> $ disp  <dbl> 160.0, 160.0, 108.0, 258.0, 360.0, 225.0, 360.0...  
#> $ hp    <dbl> 110, 110, 93, 110, 175, 105, 245, 62, 95, 123, ...  
#> $ drat  <dbl> 3.90, 3.90, 3.85, 3.08, 3.15, 2.76, 3.21, 3.69,...  
#> $ wt    <dbl> 2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570...  
#> $ qsec  <dbl> 16.46, 17.02, 18.61, 19.44, 17.02, 20.22, 15.84...  
#> $ vs    <dbl> 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0,...  
#> $ am    <dbl> 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
```

# Distribuição de frequência de uma variável

Quando analisamos uma variável aleatória (qualitativa ou quantitativa), deve-se conhecer a distribuição de frequência dessa variável por meio de suas possíveis realizações (observações).

Nesse sentido, o objetivo dessa aula será apresentar as principais formas e visualização gráfica de variáveis quali e quantitativas.

**Exemplo:** Considerando os `dados_turmas.xlsx` amostrados das turmas de Estatísticas temos:

	A	B	C	D	E	F
1	id	sexo	cor_cabelo	GA	altura	idade
2	1	F	CC	mais_social	1.68	19
3	2	F	CE	nao_consomme	1.59	20
4	3	F	CC	pouco	1.7	49
5	4	F	CE	socialmente	1.5	20
6	5	M	CE	mais_social	1.76	23
7	6	M	CC	pouco	1.6	28
8	7	M	L	nao_consomme	1.84	19

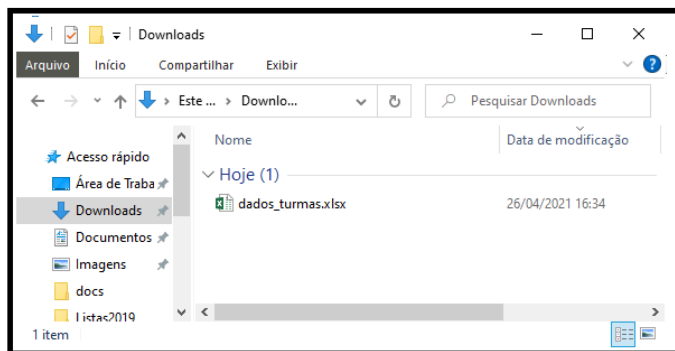
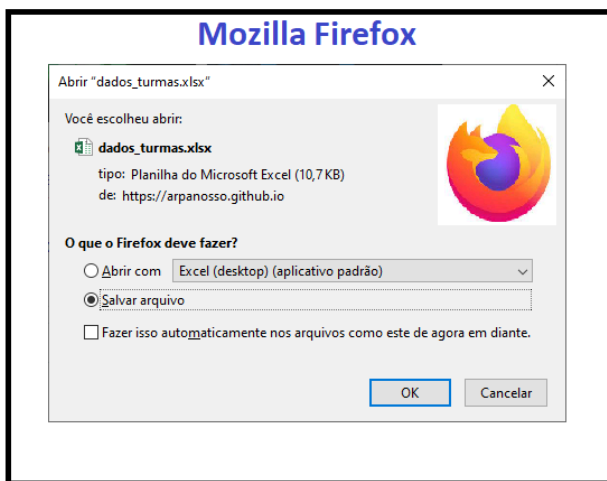
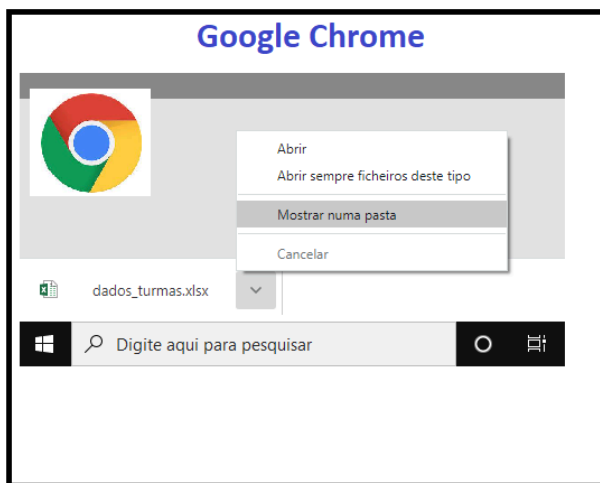


# Carregando os dados no R

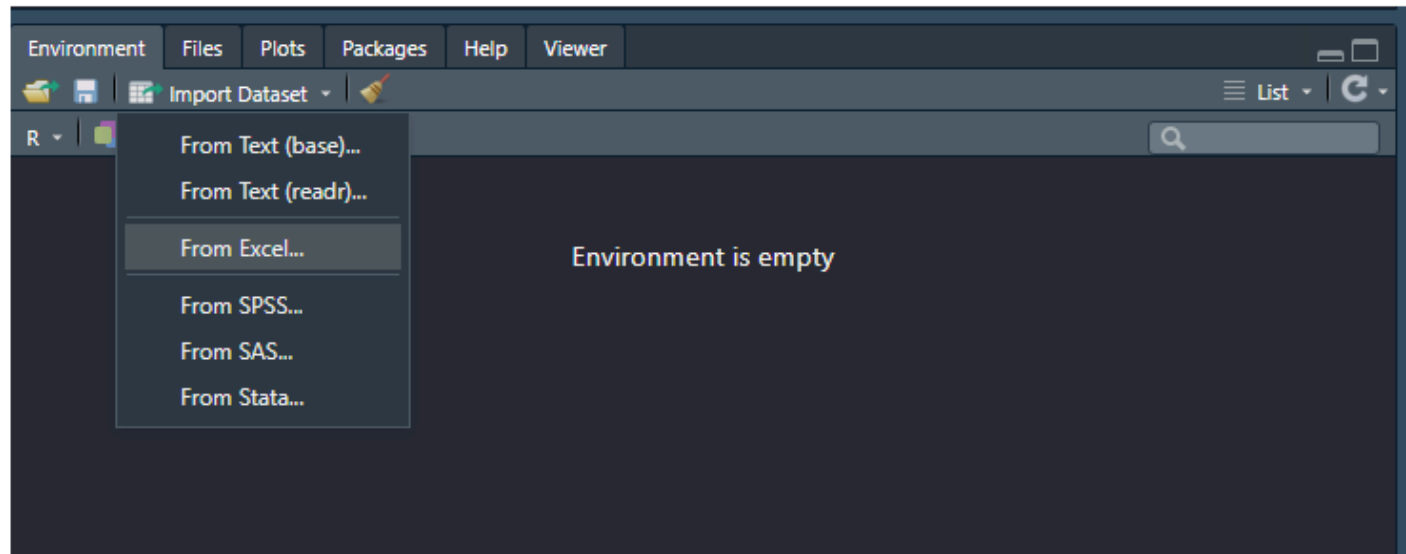
Para carregar o banco de dados da turma no R, siga os passos:

1. Faça o Download dos **dados\_turmas.xlsx**.

2. Salve em uma pasta data dentro de um projeto do R previamente criado.

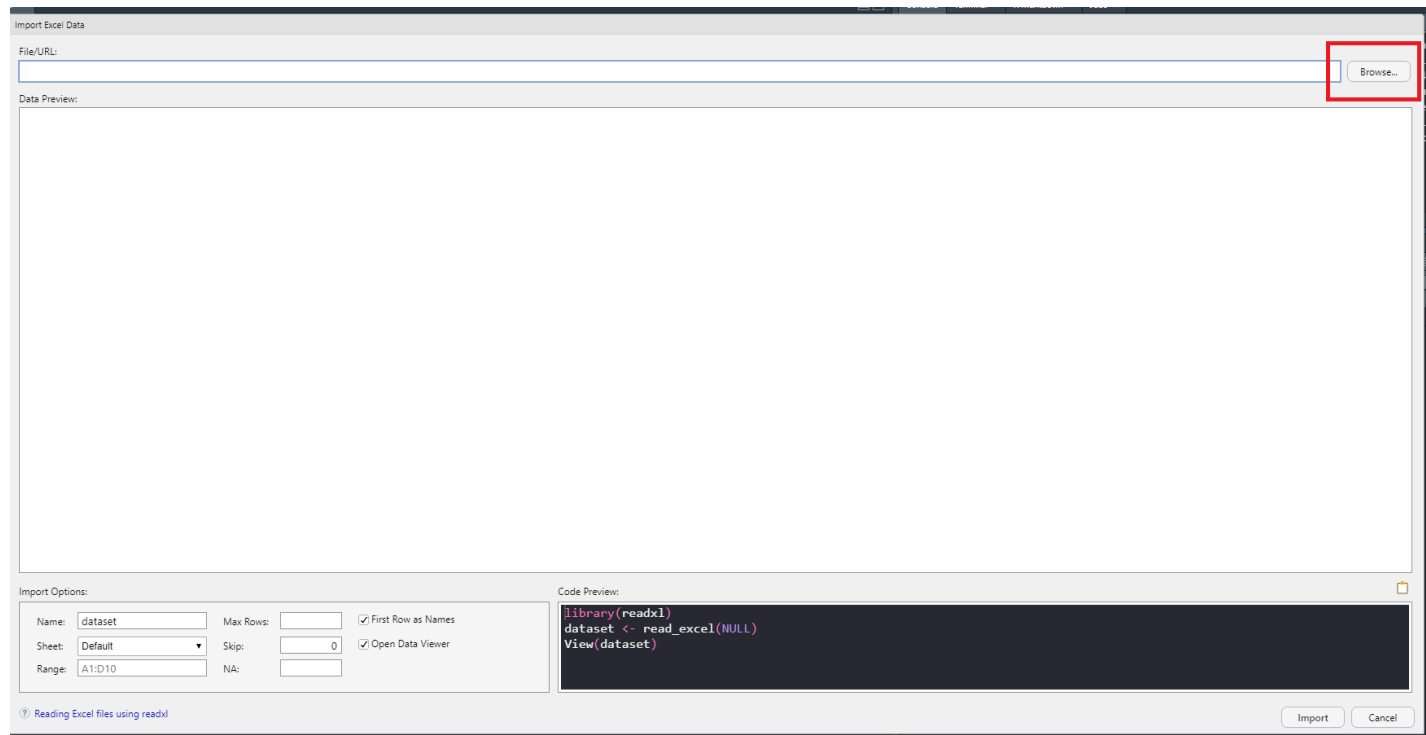


3. Na aba **Environment** do RStudio selecione **Import Dataset/From Excel...** como apresentado abaixo.



]

#### 4. Selecione **Browse** (destacado em vermelho no canto direito superior).



Import Excel Data

File/URL:

Browse...

Data Preview:

Import Options:

Name:  Max Rows:  ☒ First Row as Names

Sheet:  Skip:  ☒ Open Data Viewer

Range:  NA:

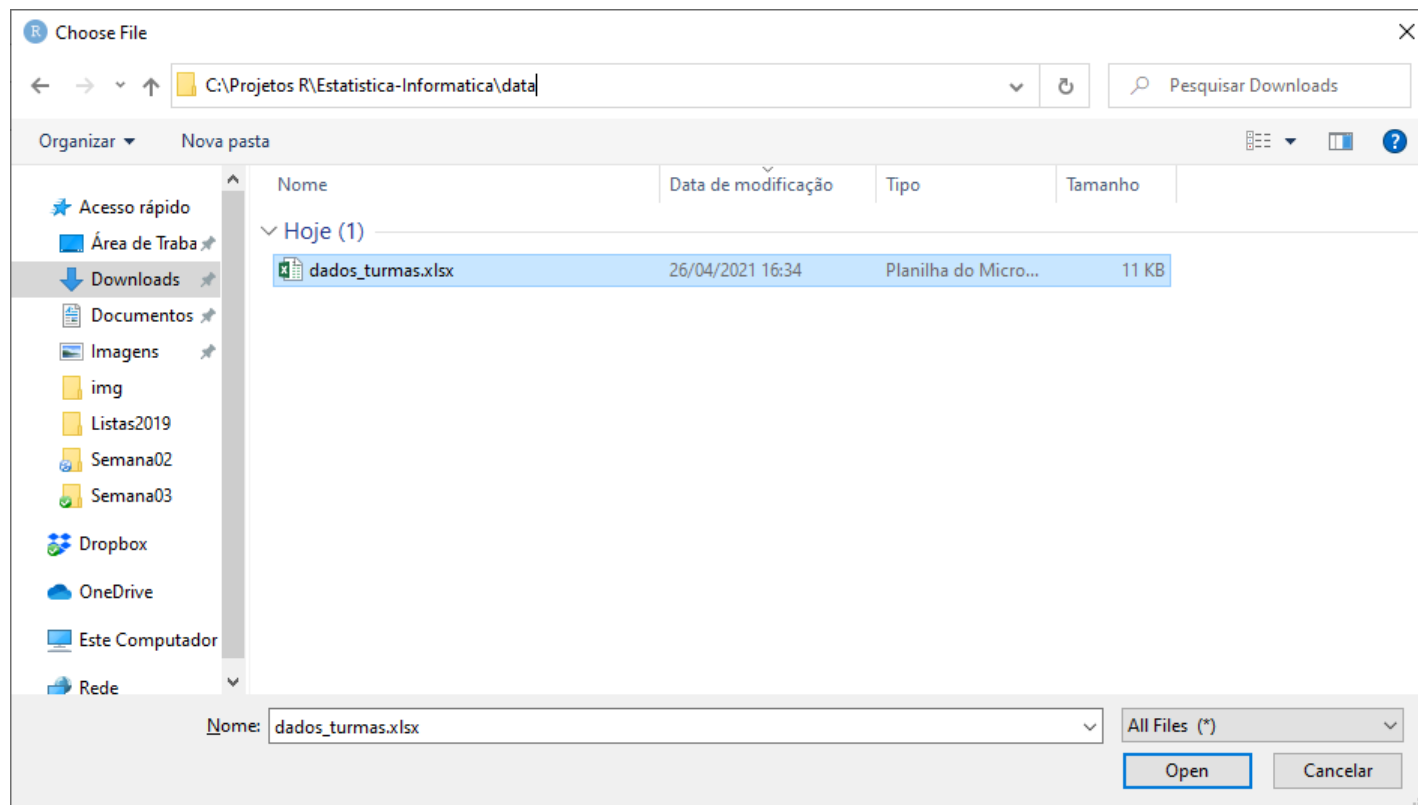
Code Preview:

```
library(readxl)
dataset <- read_excel(NULL)
View(dataset)
```

Reading Excel files using readxl

Import Cancel

5. Na próxima janela busque o arquivo da base de dados **dados\_turmas.xlsx** que salvamos na pasta "*data*", selecione o arquivo e clique em **Open**.



6. Na janela serão apresentados os dados, **NÃO CLIQUE EM IMPORT**, ao invés disso, **selecione e copie o código** para a importação dos dados. Após isso **CLIQUE EM CANCEL**.

Import Excel Data

File/URL:  
C:/Users/Usuario/Downloads/dados\_turmas.xlsx

Data Preview:

id (double)	sexo (character)	cor_cabelo (character)	GA (character)	altura (double)	idade_anos (double)
1	F	CC	mais_social	1.68	19
2	F	CE	nao_consome	1.59	20
3	F	CC	pouco	1.70	49
4	F	CE	sociaimente	1.50	20
5	M	CE	mais_social	1.76	23
6	M	CC	pouco	1.60	28
7	M	L	nao_consome	1.84	19
8	M	CE	pouco	1.88	20
9	M	CC	pouco	1.90	20
10	M	C	mais_social	1.68	19
11	M	CC	sociaimente	1.76	21
12	M	CE	sociaimente	1.91	21
13	M	CC	sociaimente	1.68	21
14	M	CE	pouco	1.70	18
15	M	CE	sociaimente	1.76	19
16	F	CE	pouco	1.65	20
17	F	CC	sociaimente	1.58	19
18	M	CE	sociaimente	1.87	19
19	M	CE	sociaimente	1.75	18
20	F	C	mais_social	1.69	22
21	M	C	sociaimente	1.74	19

Import Options:

Name: dados\_turmas Max Rows:  ☒ First Row as Names

Sheet: Default Skip:  0 ☒ Open Data Viewer

Range: A1:D10 NA:

Code Preview:

```
library(readxl)
dados_turmas <- read_excel("C:/Users/Usuario/Downloads/dados_turmas.xlsx")
View(dados_turmas)
```

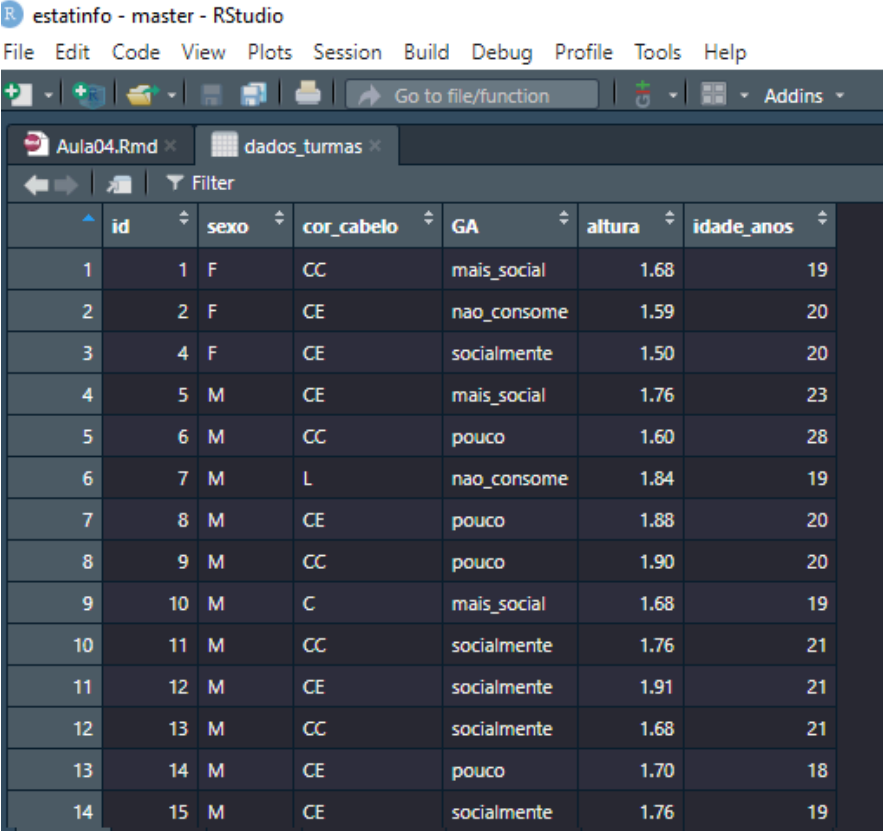
**Copie essas linhas de código, cole no seu script e o execute**

Reading Excel files using readxl

Import Cancel

7. Cole o código no seu script do R e o execute. Os dados serão salvos no objeto `dados_turmas`. Se necessário, instale o pacote `readxl` com as opções da aba **Packages/Install** ou com o comando `install.packages("readxl")`.

```
library(readxl)
dados_turmas <- read_excel("data/dados_turmas.xlsx")
View(dados_turmas)
```



estatinfo - master - RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function

Aula04.Rmd dados\_turmas

Filter

	id	sexo	cor_cabelo	GA	altura	idade_anos
1	1	F	CC	mais_social	1.68	19
2	2	F	CE	nao_consome	1.59	20
3	4	F	CE	socialmente	1.50	20
4	5	M	CE	mais_social	1.76	23
5	6	M	CC	pouco	1.60	28
6	7	M	L	nao_consome	1.84	19
7	8	M	CE	pouco	1.88	20
8	9	M	CC	pouco	1.90	20
9	10	M	C	mais_social	1.68	19
10	11	M	CC	socialmente	1.76	21
11	12	M	CE	socialmente	1.91	21
12	13	M	CC	socialmente	1.68	21
13	14	M	CE	pouco	1.70	18
14	15	M	CE	socialmente	1.76	19

## Tamanho da População ( $N$ )

O tamanho da população  $N$  é o número total dos elementos alvos da pesquisa. Muitas vezes não conhecemos esse valor. Em nosso exemplo, poderíamos entender como  $N$  o número de todos os alunos da Unesp que estão no segundo ano de sua graduação.

## Tamanho da amostra ( $n$ )

É o número total de registros de sua base de dados, ou seja o número total de elementos amostrados da população. O comando `glimpse` permite que veriquemos o tamanho do banco de dados em linhas (Rows -  $n$ ) e colunas (Columns - variáveis). Onde `chr` representa variáveis do tipo **strings**, ou seja textos e `dbl` representa variáveis numéricas.

```
library(tidyverse)
glimpse(dados_turmas)
```

```
#> Rows: 79
#> Columns: 6
#> $ id          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1...
#> $ sexo        <chr> "F", "F", "F", "M", "M", "M", "M", "M", "M", ...
#> $ cor_cabelo  <chr> "CC", "CE", "CE", "CE", "CC", "L", "CE",...
#> $ grau_alcool <chr> "mais_social", "nao_consomme", "socialmen...
#> $ altura      <dbl> 1.68, 1.59, 1.50, 1.76, 1.60, 1.84, 1.88...
#> $ idade_anos  <dbl> 19, 20, 20, 23, 28, 19, 20, 20, 19, 21, ...
```

## Exemplo da base de dados das turmas

Construir uma tabela de frequências para a variável `sexo` contendo as frequências absolutas ( $n_i$ ), as frequências relativas ( $f_i$ ) e a porcentagem ( $perc$ ) para as categoria existentes.

Após isso, realizar a visualização de dados com gráficos de Colunas, Barras e Setores (Pizza ou *Pie*).

Para essa tarefa, vamos utilizar o R. Precisaremos, então, fazer algumas operações nos dados das turmas e vamos usar o operador PIPE (`%>%`) feito com o atalho CTRL + SHIFT + M. Vamos utilizar a função `n()` para contar cada ocorrência das diferentes categorias de `sexo`

```
tab <- dados_turmas %>%  
  group_by(sexo) %>%  
  summarise(ni=n())
```



- `group_by()` agrupa as categorias da variável `sexo`.
- `summarise()` vai criar o resumo dos dados, ou seja, contará o valor de cada categoria usando a função `n()` e salvará na coluna `ni`.

```
tab
```

```
#> # A tibble: 2 × 2  
#>   sexo      ni  
#>   <chr> <int>  
#> 1 F      31  
#> 2 M      48
```

```
dados_turmas %>%  
  arrange(sexo)
```

```
#> # A tibble: 79 × 6  
#>       id sexo cor_cabelo grau_alcool altura idade_anos  
#>   <dbl> <chr> <chr>      <chr>      <dbl>      <dbl>  
#> 1     1  F      CC      mais_social  1.68        19  
#> 2     2  F      CE      nao_consomme 1.59        20  
#> 3     3  F      CE      socialmente  1.5         20  
#> 4    15  F      CE      pouco      1.65        20  
#> 5    16  F      CC      socialmente  1.58        19  
#> # i 74 more rows
```

# Frequência Absoluta ( $n_i$ )

É definida como o número de observações no conjunto de dados pertencentes à uma categoria ou classe da variável em estudo.

Então temos:

$$n_F = 30$$

$$n_M = 48$$

Assim temos a primeira regra da análise de nossa base de dados, a soma da frequência absoluta das classes ( $k$ ) da variável categoria é igual a  $n$ .

$$\sum_{i=1}^k n_i = n$$

Onde  $k$  é o número de categorias da variável em questão, no caso do sexo, temos duas categorias (M e F).

$$\sum_{i=1}^k n_i = 30 + 48 = 78$$

## Frequência Relativa ( $f_i$ )

É definida como a proporção de cada categoria em relação ao **Total de observações** ( $n$ ), ou seja:

$$f_i = \frac{n_i}{n}$$

Vamos, mais uma vez, utilizar o R para esses cálculos.

```
dados_turmas %>%  
  group_by(sexo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni))
```

```
#> # A tibble: 2 × 3  
#>   sexo    ni    fi  
#>   <chr> <int> <dbl>  
#> 1 F      31 0.392  
#> 2 M      48 0.608
```

`mutate()` permitirá criarmos mais uma coluna na nossa tabela de resumo a partir da coluna  $n_i$ , no caso a frequência relativa  $f_i$ .

Portanto temos que,

$$f_F = \frac{30}{78} = 0,385$$

e

$$f_M = \frac{48}{78} = 0,615$$

Assim, temos que a soma das frequências relativas sempre será igual a 1:

$$\sum_{i=1}^k f_i = 1$$

onde  $k$  é o número de categorias da variável sexo, ou seja, 2, nesse caso.

## Porcentagem de frequência (*perc* ou %)

Definida como o resultado da multiplicação da frequência relativa (proporção) por 100.

```
dados_turmas %>%  
  group_by(sexo) %>%  
  summarise(ni=n()) %>% # Frequência Absoluta  
  mutate(fi = ni/sum(ni), # Frequência Relativa  
          perc = fi*100)
```

```
#> # A tibble: 2 × 4  
#>   sexo      ni    fi  perc  
#>   <chr> <int> <dbl> <dbl>  
#> 1 F      31 0.392  39.2  
#> 2 M      48 0.608  60.8
```

A partir da tabela de frequência, poderemos criar representações gráficas que nos auxiliarão na apresentação e interpretação do comportamento dos dados.

Essa etapa é a denominada de **Visualização de Dados**.

# VISUALIZAÇÃO DE DADOS

## (Variáveis Qualitativas)

# Visualização dos dados

Os tipos de gráficos podem variar de acordo com o tipo de variável, geralmente, para as variáveis qualitativas utilizamos os gráficos de Barras, Colunas ou de Setores (Pizza ou Pie).

Para isso, vamos utilizar as funções do pacote `ggplot2` que é carregado junto com o pacote `tidyverse`. Observe que os gráficos serão construídos a partir do objeto `tab` anteriormente definido.

O `ggplot` funciona na forma de camadas representações gráficas e de formatações, que são adicionadas de acordo com a necessidade do usuário por meio do operador de adição `+` digitado ao final da cada linha.

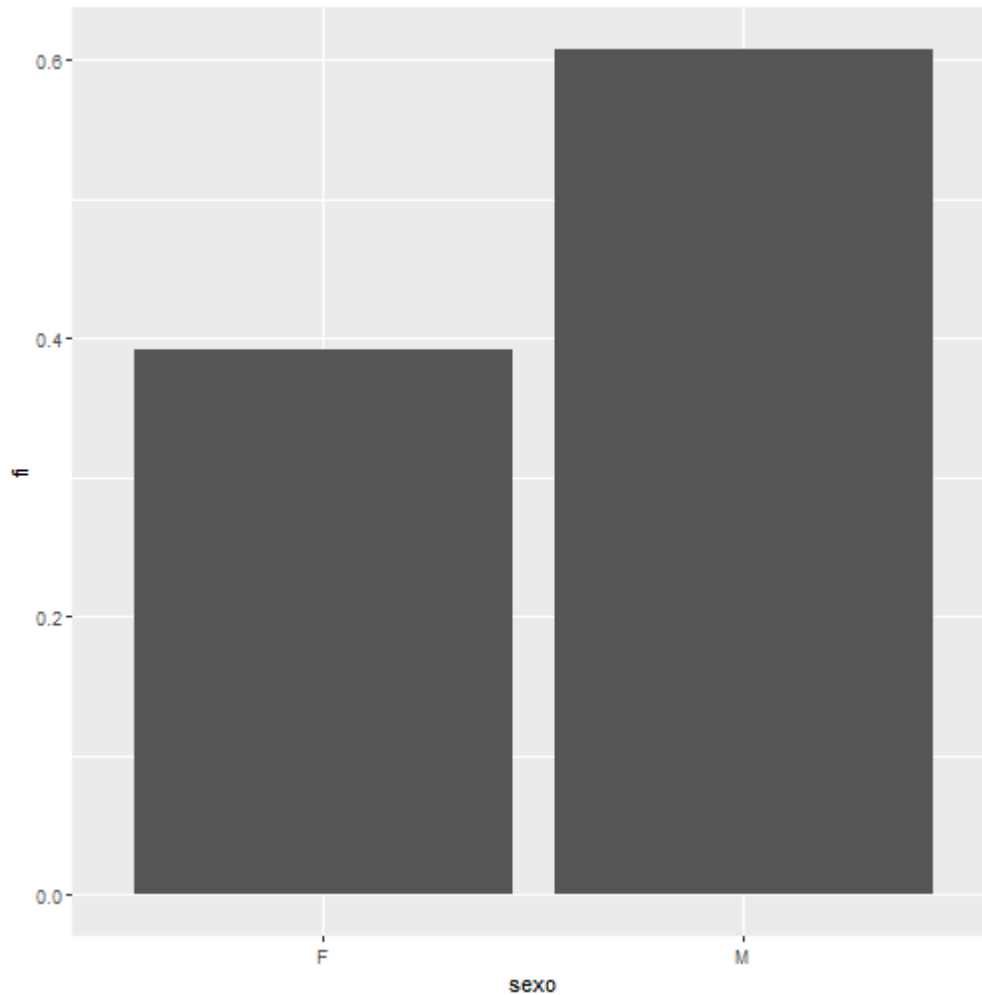


## Gráfico de Colunas para Sexo

Deve ser utilizado para variáveis categóricas (qualitativas ordinais ou nominais).

```
dados_turmas %>%  
  group_by(sexo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
          perc = fi*100) %>%  
  ggplot(aes(x=sexo, y=fi)) +  
  geom_col()
```

## Gráfico de Colunas para Sexo

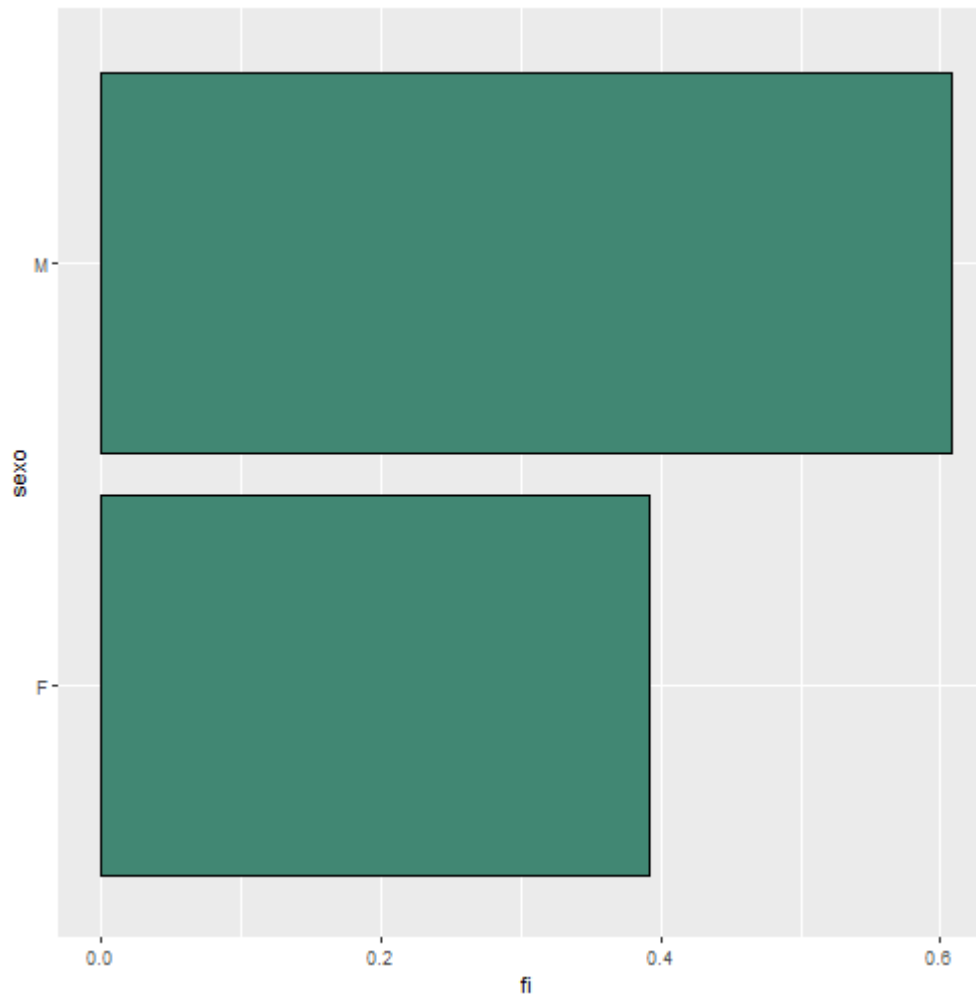


# Gráfico de Barras para Sexo

Semelhante ao gráfico de colunas, contudo, com as barras na horizontal, facilita a leitura do nome das categorias.

```
dados_turmas %>%  
  group_by(sexo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
          perc = fi*100) %>%  
  ggplot(aes(x=fi,y=sexo)) +  
  geom_col(fill="aquamarine4",  
           color="black")
```

- o argumento `fill = "aquamarine4"` permite que possamos alterar a cor do preenchimento (*fill*) da barra e o argumento `color="black"` permite a alteração da cor o contorno das barras.



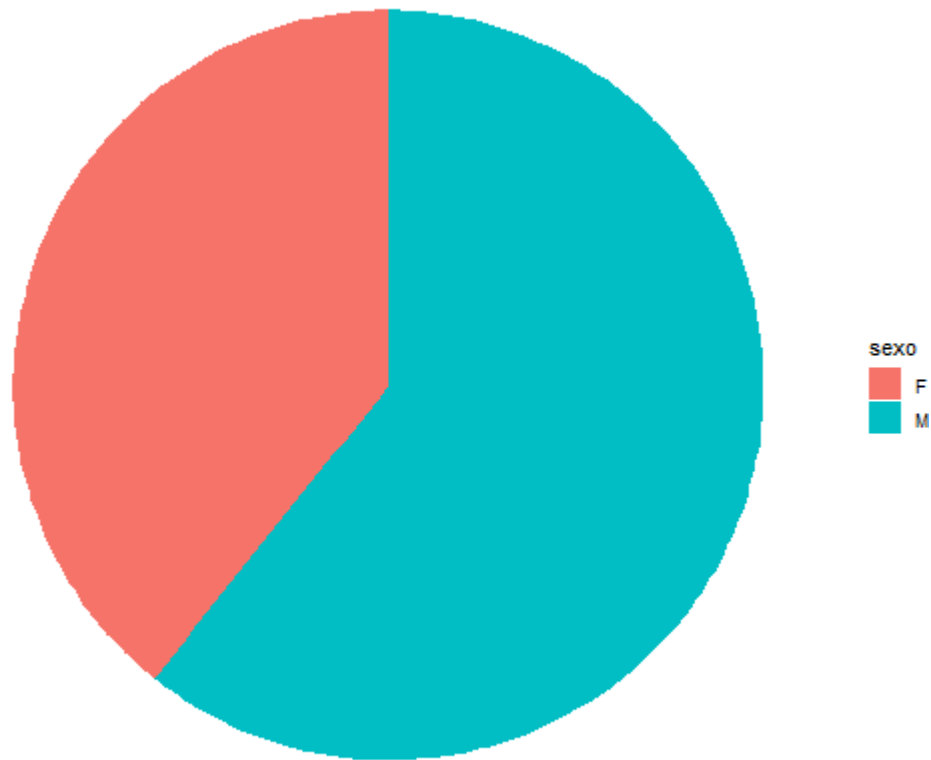
Outras cores são possíveis tente algumas **das cores no R.**

# Gráfico de Setores para Sexo

Também conhecido como gráfico de Pizza (ou torta em inglês - *pie*), ele representa cada valor de frequência relativa das diferentes categorias da variável em uma circunferência.

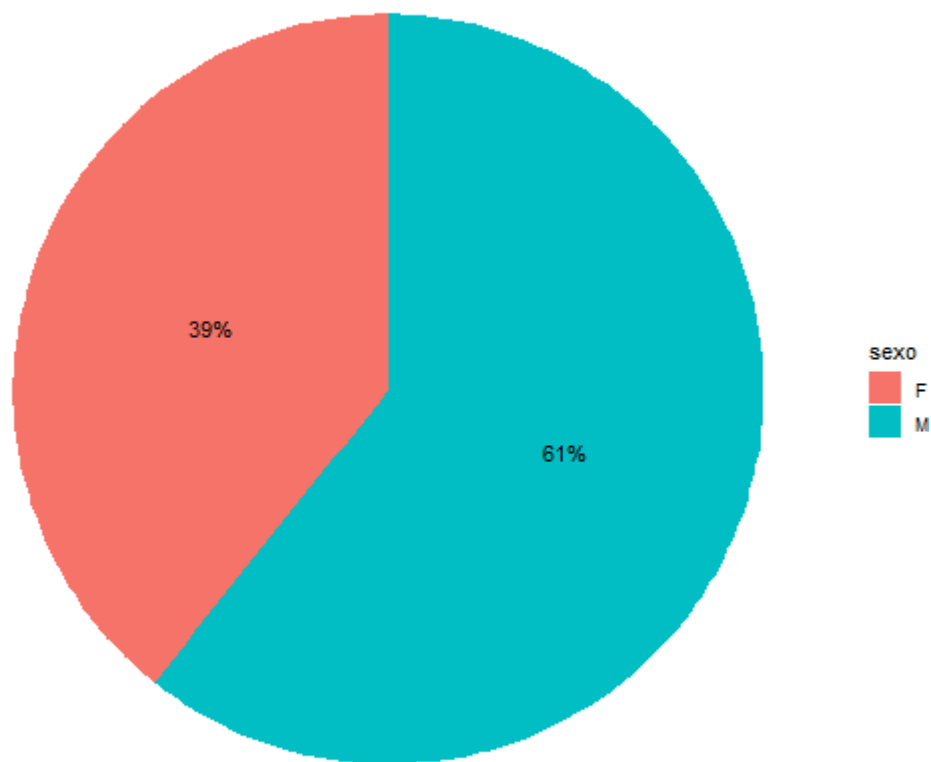
```
dados_turmas %>%  
  group_by(sexo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
          perc = fi*100) %>%  
  ggplot(aes(x="",y=fi, fill=sexo)) +  
  geom_bar(stat="identity") +  
  coord_polar("y", start=0) +  
  theme_void()
```

- A função `geom_bar()` associada à `coord_polar()` permite a transformação do gráfico de barras no gráfico de pizza.
- A função `theme_void()` retira elementos como linhas e nomes e números da representação gráfica.



## Vamos Adicionar os valores de $f_i$ no gráfico

```
dados_turmas %>%  
  group_by(sexo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
         perc = fi*100) %>%  
  ggplot(aes(x="",y=fi, fill=sexo)) +  
  geom_bar(stat="identity") +  
  coord_polar("y", start=0) +  
  theme_void() +  
  geom_text(  
    aes(  
      label=paste0(round(perc), "%"),  
      position=position_stack(vjust = 0.5))
```





# Tabela de Frequência para a variável Cor de cabelo

Vamos, mais uma vez, utilizar o R para conseguirmos as tabelas e a representação gráfica.

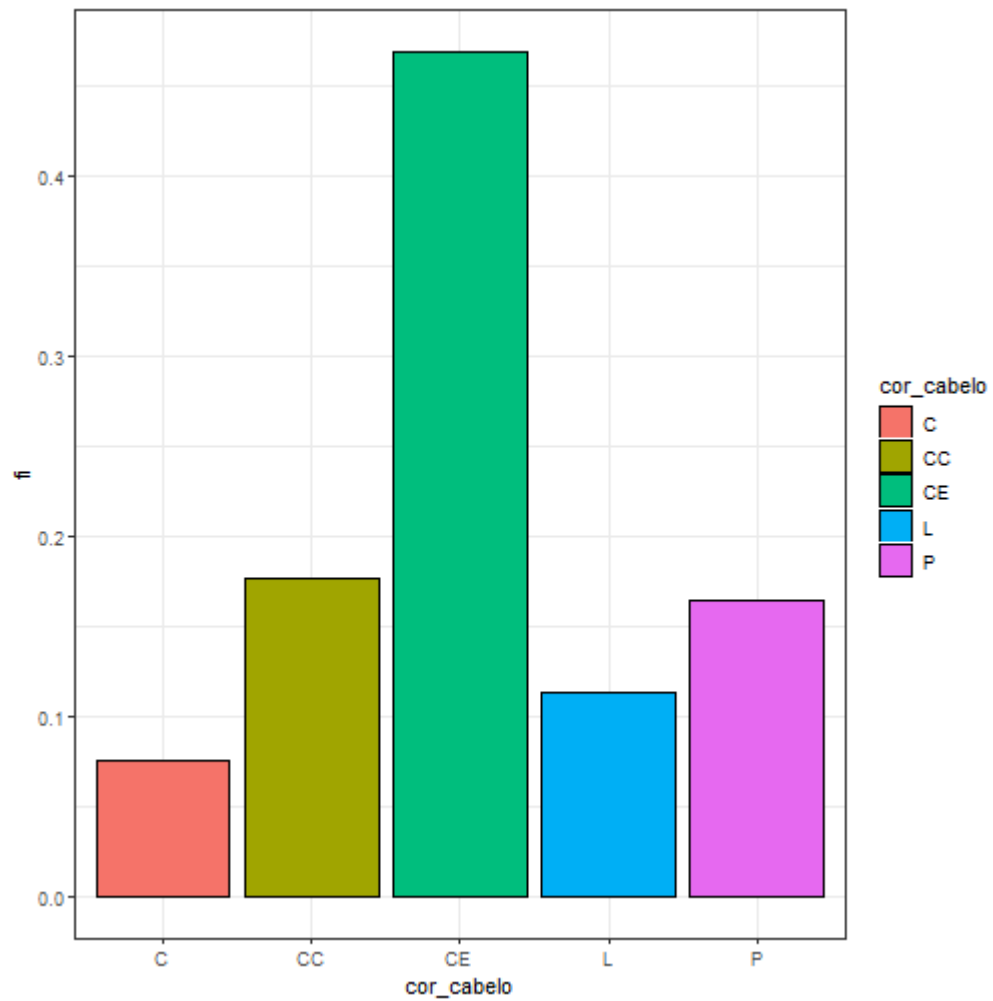
```
dados_turmas %>%  
  group_by(cor_cabelo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
         perc = fi*100)
```

```
#> # A tibble: 5 × 4  
#>   cor_cabelo    ni    fi  perc  
#>   <chr>      <int> <dbl> <dbl>  
#> 1 C         6 0.0759  7.59  
#> 2 CC        14 0.177  17.7  
#> 3 CE        37 0.468  46.8  
#> 4 L         9 0.114  11.4  
#> 5 P        13 0.165  16.5
```

# Gráfico de Colunas para Cor de cabelo

```
dados_turmas %>%  
  group_by(cor_cabelo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
          perc = fi*100) %>%  
  ggplot(aes(x=cor_cabelo,y=fi,  
            fill=cor_cabelo)) +  
  geom_col(color="black")+  
  theme_bw()
```

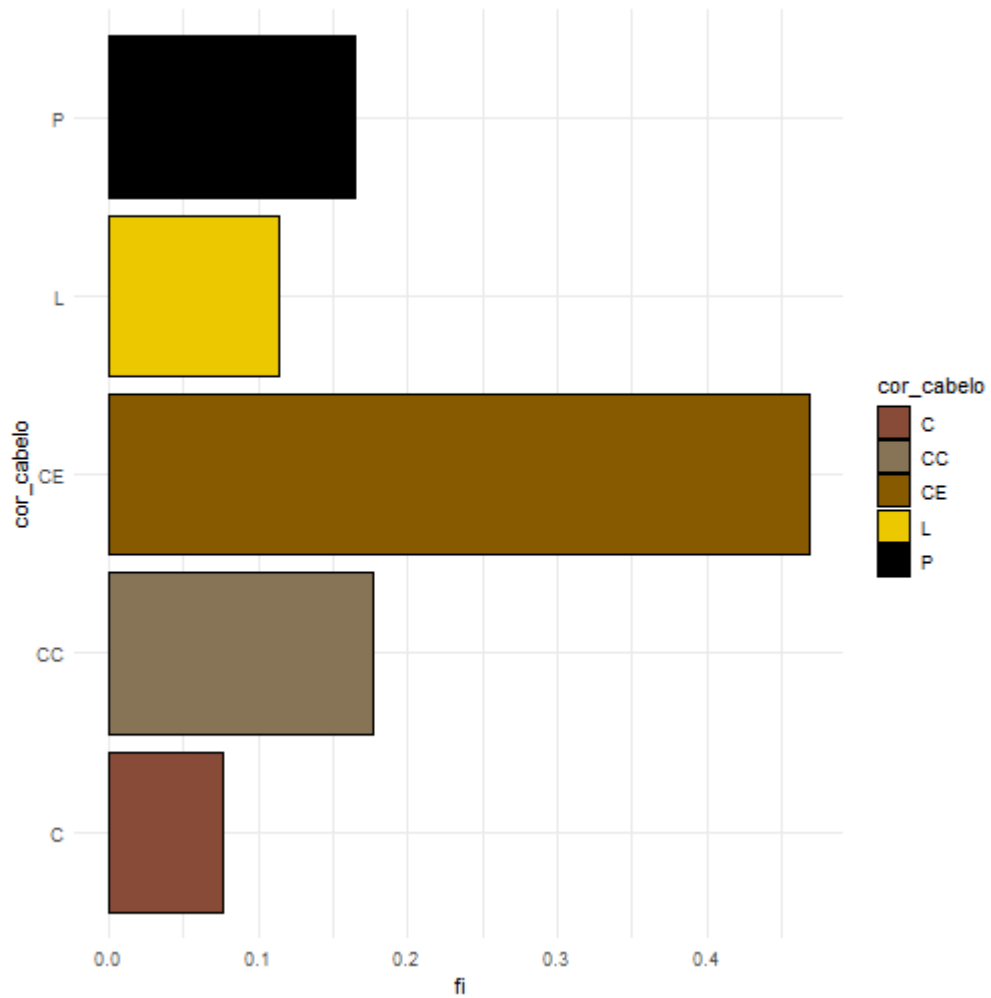
- ao passarmos o argumento `fill=cor_cabelo` dentro da função `aes()` estamos pedindo o mapeamento das cores de cabelo a partir de cores de preenchimento diferentes.
- `aes()` representa a estética do gráfico, ou seja, quem é x, quem é y e quem deve ser mapeado.
- a função `theme_bw()` muda o padrão de cores e de linhas do gráfico. Existem outros padrões como `theme_classic`, `theme_minimal` entre outros.



## Gráfico de Barras para Cor de cabelo

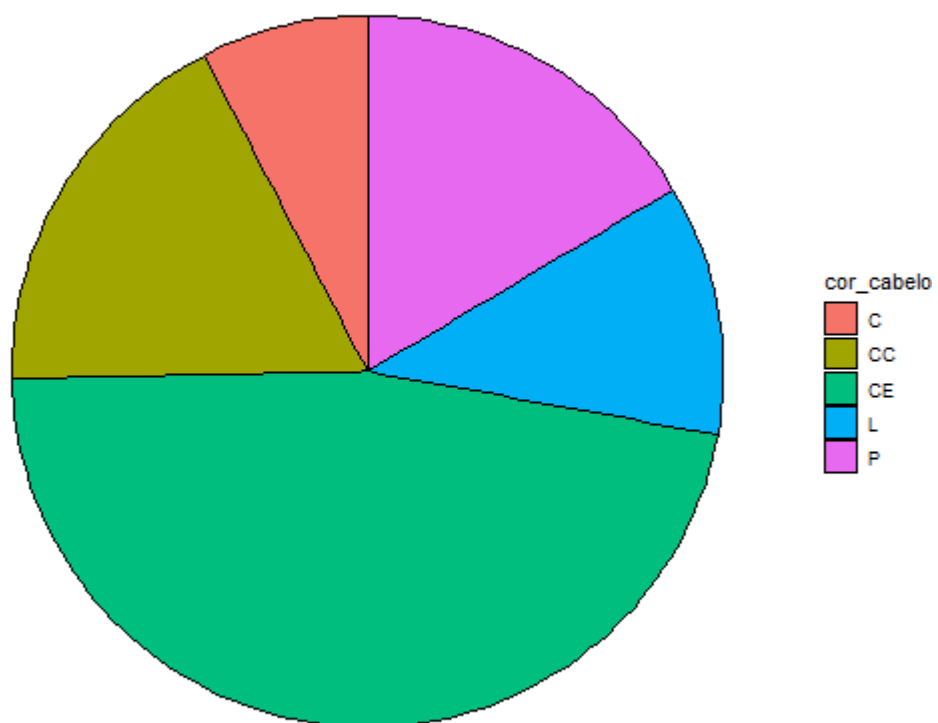
```
dados_turmas %>%  
  group_by(cor_cabelo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
          perc = fi*100) %>%  
  ggplot(aes(x=fi, y=cor_cabelo,  
            fill=cor_cabelo)) +  
  geom_col(color="black")+  
  scale_fill_manual(values = c("salmon4",  
                                "burlywood4",  
                                "orange4",  
                                "gold2",  
                                "black"))+  
  
  theme_minimal()
```

- utilize a função `scale_fill_manual()` para alterar as cores dos preenchimentos, se necessário.



## Gráfico de Setores para cor\_cabelo

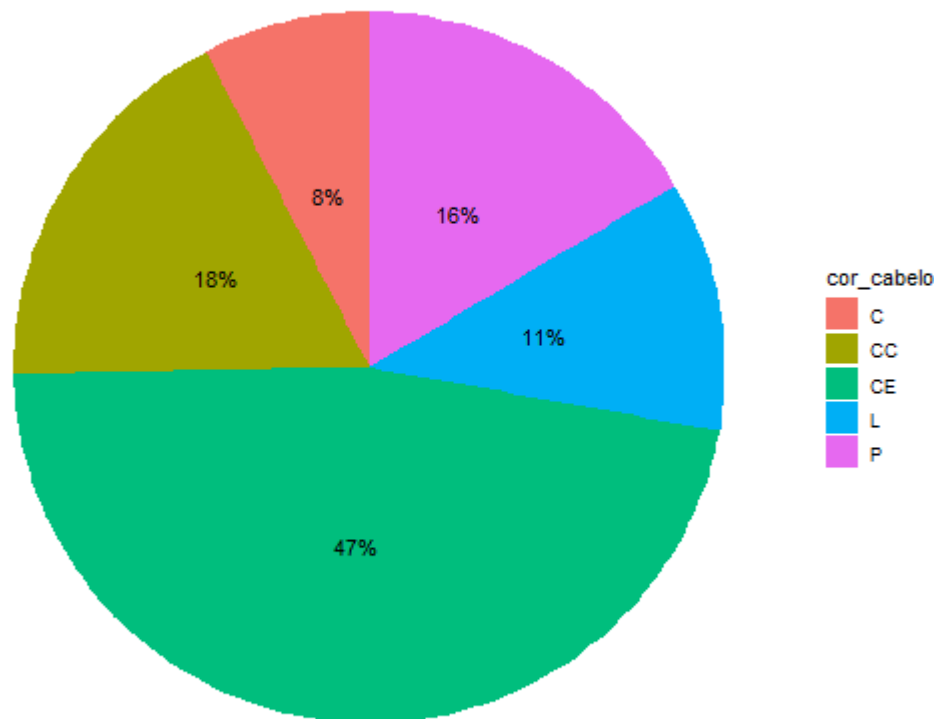
```
dados_turmas %>%  
  group_by(cor_cabelo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
          perc = fi*100) %>%  
  ggplot(aes(x="", y=fi,  
            fill=cor_cabelo)) +  
  geom_bar(stat="identity",color="black") +  
  coord_polar("y", start=0) +  
  theme_void()
```



## Vamos Adicionar os valores de *perc* no gráfico

```
dados_turmas %>%  
  group_by(cor_cabelo) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
         perc = fi*100) %>%  
  ggplot(aes(x="",y=fi, fill=cor_cabelo)) +  
  geom_bar(stat="identity") +  
  coord_polar("y", start=0) +  
  theme_void() +  
  geom_text(  
    aes(  
      label=paste0(round(perc), "%"),  
      position=position_stack(vjust = 0.5))
```





# VISUALIZAÇÃO DE DADOS

(Variáveis Quantitativas)

# Tabela de frequência e visualização para Idade em anos (discreta)

Quando a variável for quantitativa discreta, os mesmos gráficos de variáveis qualitativas podem ser utilizados. Porém, também recomendamos a utilização dos gráficos boxplot, histograma e Função distribuição acumulada.

Pra a variável `idade_anos` vamos criar a tabela de frequência;

```
tab <- dados_turmas %>%  
  group_by(idade_anos) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
          perc = fi*100)  
View(tab)
```

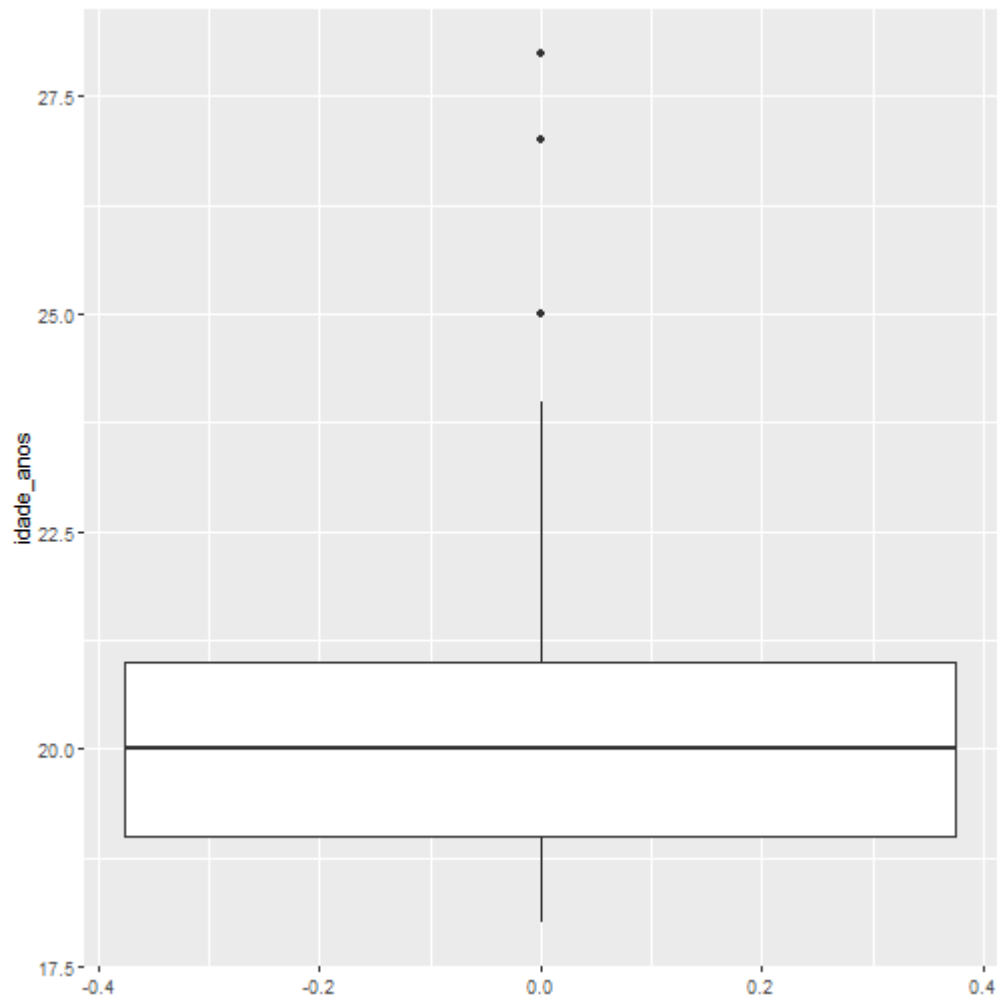
<b>idade_anos</b>	<b>ni</b>	<b>fi</b>	<b>perc</b>
18	11	0.1392405	13.924051
19	28	0.3544304	35.443038
20	18	0.2278481	22.784810
21	5	0.0632911	6.329114
22	3	0.0379747	3.797468
23	8	0.1012658	10.126582
24	2	0.0253165	2.531646
25	1	0.0126582	1.265823
27	1	0.0126582	1.265823
28	2	0.0253165	2.531646

# Gráfico Boxplot

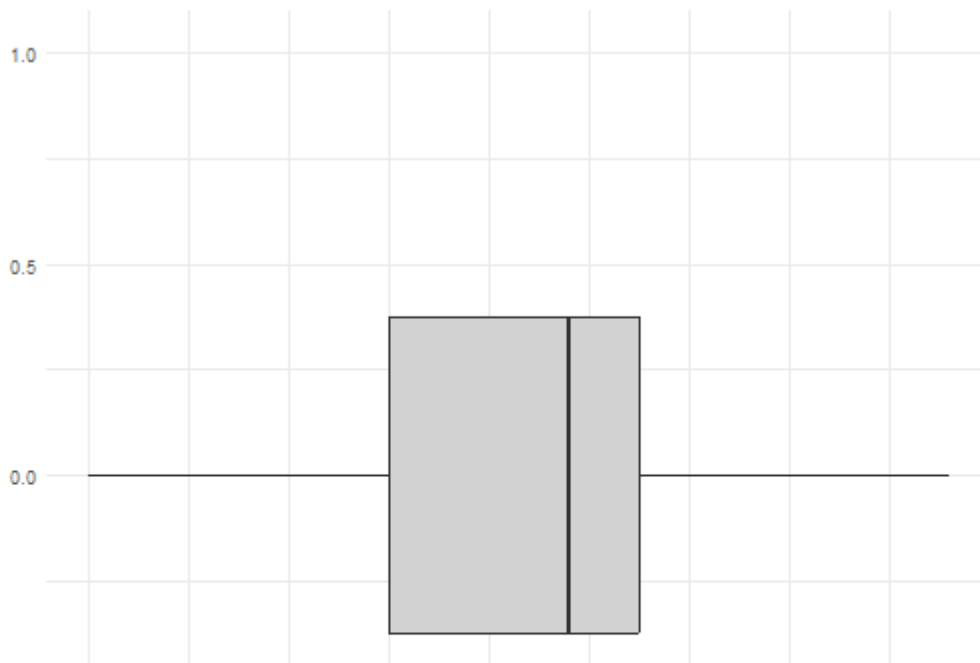
Conhecido como gráfico dos 5 números representa um resumo dos valores *mínimo, primeiro quartil, mediana, terceiro quartil e máximo* de uma variável. É construído a partir de `geom_boxplot()`.

Observe que dentro da função `ggplot()` não é necessário passar o `x`, somente é atribuído a `y` a variável discreta `idade_anos`.

```
dados_turmas %>%  
  ggplot(aes(y=idade_anos)) +  
  geom_boxplot()
```



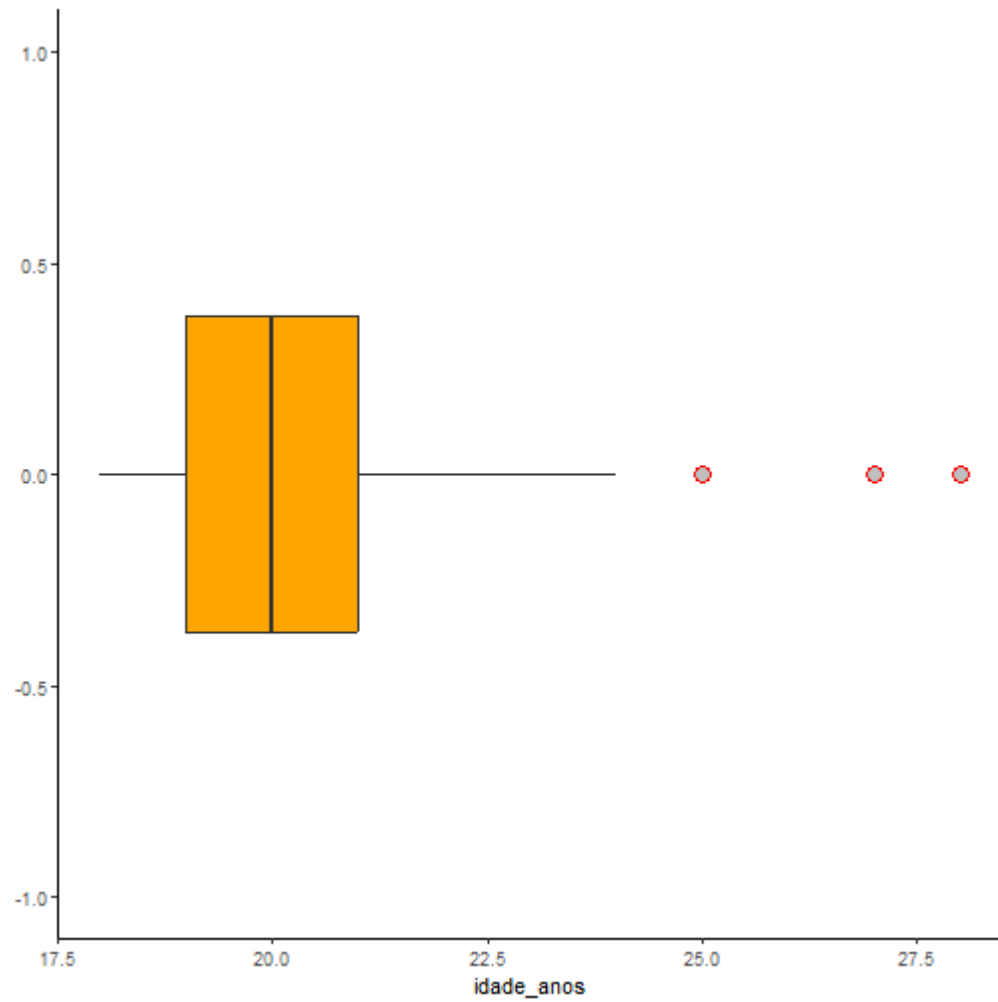
- O boxplot é uma caixa que vai do 25º percentil ao 75º percentil da distribuição, uma distância conhecida como a **amplitude interquartil** (IIQ).
- No meio da caixa há uma linha que exibe a **mediana**, isto é, 50º percentil, da distribuição. Essas três linhas lhe dão um sentido da dispersão da distribuição e se ela é ou não simétrica sobre a mediana ou enviesada para um lado.
- Pontos visuais que exibem observações são aqueles que caíram em mais do que 1,5 vezes o IIQ de cada limite da caixa. Esses pontos fora da curva, denominados **outliers** são incomuns, então são plotados individualmente.
- Uma linha que se estende de cada lado da caixa e vai até o ponto mais distante da distribuição que não seja um valor incomum.



Suas coordenadas podem ser transpostas trocando y por x no código anterior e o tamanho da caixa, nesse caso, pode ser controlado por `coord_cartesian(ylim=c(-1,1))`. Utilize `fill` para controlar a cor de preenchimento da caixa, e as opções `outlier` para controlar tamanhos, formas, cores e preenchimentos dos outliers.

```
dados_turmas %>%  
  ggplot(aes(x=idade_anos)) +  
  geom_boxplot(fill="orange",  
               outlier.size = 4,  
               outlier.shape = 21,  
               outlier.color = "red",  
               outlier.fill = "gray"  
            ) +  
  coord_cartesian(ylim=c(-1,1)) +  
  theme_classic()
```

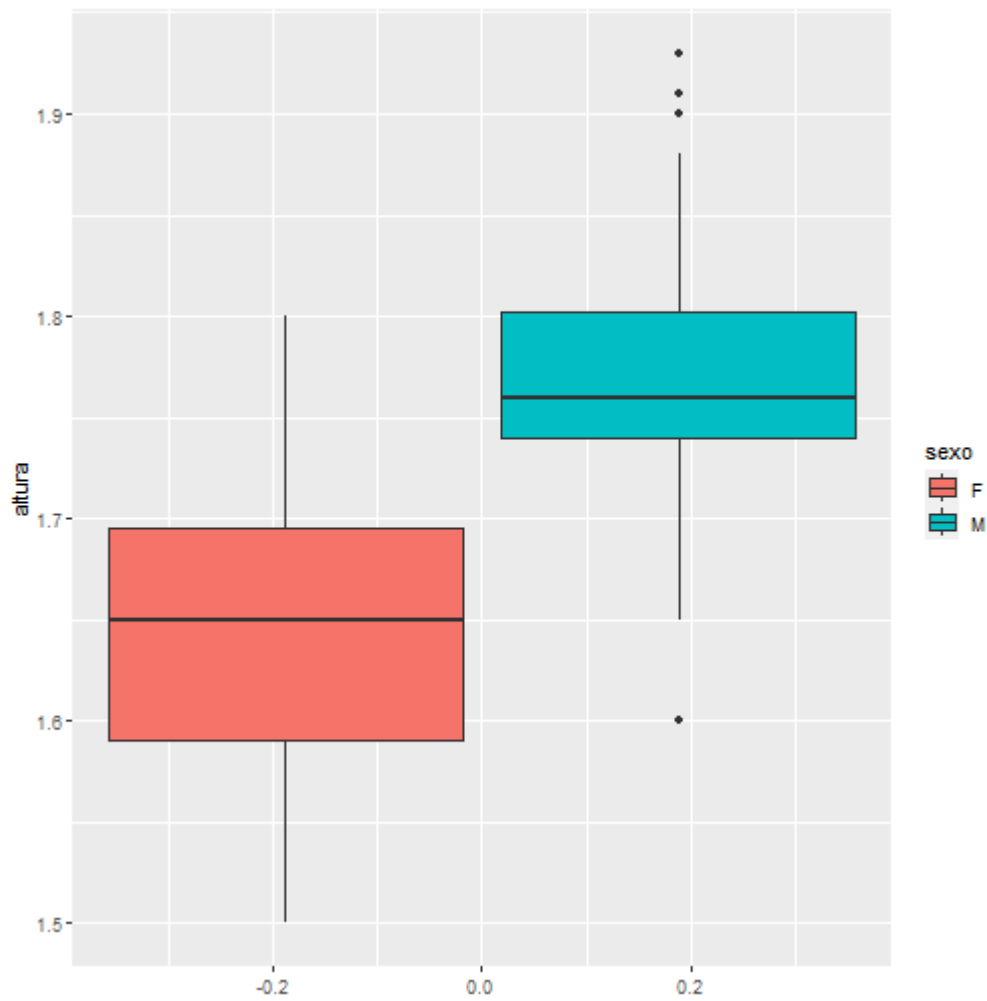




Outra alternativa para exibir a distribuição de uma variável quantitativa, no exemplo `altura`, é desmembrá-la por uma variável categórica aqui no `boxplot`.

```
dados_turmas %>%  
  ggplot(aes(y=altura,  
             fill=sexo)) +  
  geom_boxplot()
```

Para fazer isso basta passarmos como preenchimento o nome da variável categórica, no exemplo, `sexo`.



# Tabela de frequência para variável Altura (Quantitativa Contínua)

Quando a variável quantitativa for contínua, recomendamos a utilização dos gráficos boxplot, histograma e Função de distribuição acumulada.

Devemos, inicialmente construir a tabela de frequência da variável altura. Porém, os valores de uma variável contínua não se repetem, mesmo que isso aparentemente ocorra na base de dados.

Em teoria suas realizações podem assumir qualquer valor dentro da reta dos números reais, portanto, ao mensurarmos uma variável contínua, temos apenas uma aproximação de seu verdadeiro valor dada pelo instrumento de medida.

Para exemplificar, vamos criar 5 classes de alturas a partir da função `cut()`.

```
tab <- dados_turmas %>%  
  mutate(  
    classes_altura = cut(altura, 5)  
  ) %>%  
  group_by(classes_altura) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
         perc = fi*100)  
View(tab)
```

classes_altura	ni	fi	perc
(1.5,1.59]	7	0.0886076	8.86076
(1.59,1.67]	15	0.1898734	18.98734
(1.67,1.76]	28	0.3544304	35.44304
(1.76,1.84]	19	0.2405063	24.05063
(1.84,1.93]	10	0.1265823	12.65823

# Amplitude Total

Para melhor entendermos como a função `cut()` funciona, será necessário conhecermos mais algumas medidas para a construção do histograma. Vamos iniciar com a Amplitude total ( $\Delta$ ), definida como a diferença entre o valor máximo menos o valor mínimo da variável.

$$\Delta = \text{Máximo} - \text{Mínimo}$$

Para os dados de altura temos,  $\Delta = 1,93\text{ m} - 1,50\text{ m} = 0,43\text{ m}$

No R podemos calcular a amplitude total com as funções do pacote base, para isso devemos, primeiramente, extrair de `dados_turmas` a variável (coluna) `altura` por meio do operador de acesso de listas `$`.

```
altura <- dados_turmas$altura
```

Agora vamos encontrar o máximo e o mínimo e calcular a diferença.

```
D <- max(altura) - min(altura)
D
```

```
#> [1] 0.43
```

## Número de intervalos de classes ( $k$ )

Definiremos  $k$  como sendo o número de **sub-intervalos** da Amplitude Total. Uma boa representação apresenta um  $k$  **NUNCA** inferior a 5 ou superior a 15, pois com um pequeno número de classes, perde-se informação, e com um grande número de classes, o objetivo de resumir os dados fica prejudicado.

```
k <- 5
```

## Amplitude de classe ( $\Delta_i$ )

É o tamanho de cada um dos  $k = 5$  sub-intervalos, dado pela amplitude total dividida pelo número de intervalos.

$$\Delta_i = \frac{\Delta}{k}$$

Para os dados de altura:

$$\Delta_i = \frac{\Delta}{k} = \frac{0.43 \text{ m}}{5} = 0,086 \text{ m}$$

Assim, temos

```
Di = D/k  
Di
```

```
#> [1] 0.086
```

Cada um dos 5 intervalos terá uma amplitude de 0,086 m. Ou seja, o cálculo dos limites das classes é feito a partir da adição ao valor Mínimo o valor de  $\Delta_i$   $k$  vezes:

```
limites <- min(altura) + 0:k * Di  
limites
```

```
#> [1] 1.500 1.586 1.672 1.758 1.844 1.930
```



Obervem que foi essa a metodologia qua a função `cut()` utilizou para calcular os limites de classes, e essa é a metodologia clássica para lidar com dados contínuos e os agrupar em classes.

<b>classes_altura</b>	<b>ni</b>	<b>fi</b>	<b>perc</b>
(1.5,1.59]	7	0.0886076	8.86076
(1.59,1.67]	15	0.1898734	18.98734
(1.67,1.76]	28	0.3544304	35.44304
(1.76,1.84]	19	0.2405063	24.05063
(1.84,1.93]	10	0.1265823	12.65823

```
#> [1] 1.500 1.586 1.672 1.758 1.844 1.930
```

# Gráfico histograma (frequências absolutas)

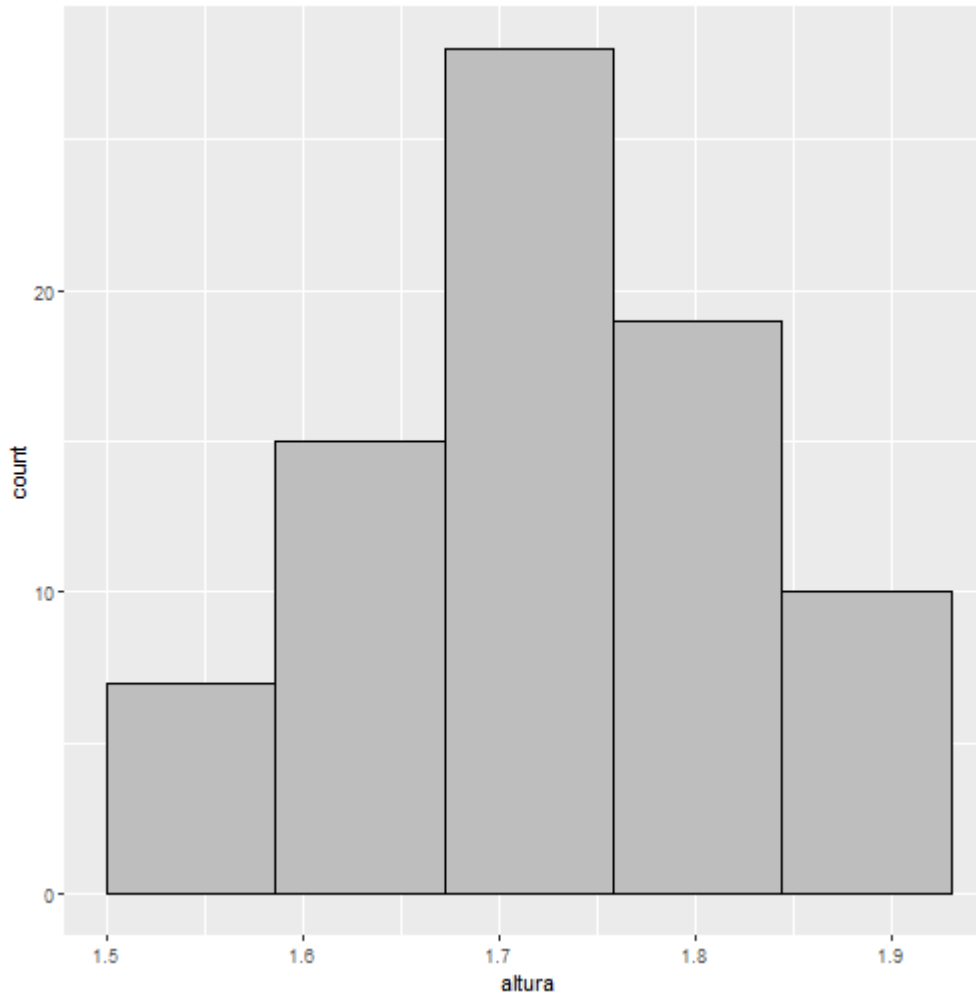
A partir da tabela anterior, pode-se construir o gráfico de frequência de cada classe de valor de altura, denominado **Histograma**.

```
dados_turmas %>%  
  ggplot(aes(x=altura,y=..count..))+  
  geom_histogram(breaks = limites,  
                 color="black",  
                 fill="gray")
```

O código acima gera um histograma com 5 barras onde o eixo y será a frequência absoluta, ou seja, a contagem (`..count..`) de quantos valores de altura estão dentro de uma determinada classe.

A opção `breaks = limites` deixa o histograma igual ao observado na tabela anterior.

# Histograma da altura (m)



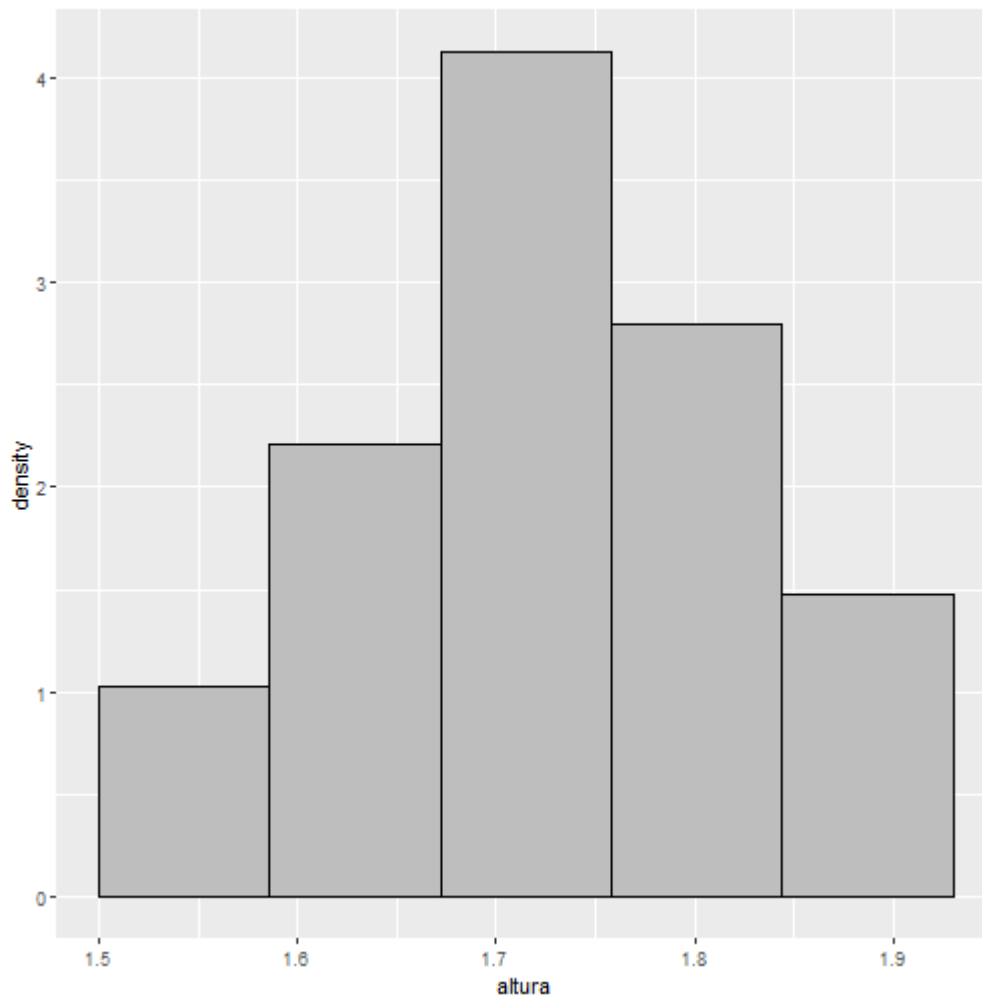
# Gráfico histograma (frequências relativas)

Ao longo de nosso curso, vamos estudar que a frequência relativa  $f_i$  é uma estimativa empírica da probabilidade  $P(X = x_i)$ , assim é interessante que a área total da figura do histograma seja igual a 1, correspondendo à soma total das frequências relativas ( $f_i$ ).

Então, para construção do histograma, sugere-se usar no eixo das ordenadas os valores de  $f_i/\Delta_i$  (denominado densidade de frequência), ou seja, da medida que indica qual a concentração por unidade da variável.

```
dados_turmas %>%  
  ggplot(aes(x=altura,y=..density..))+  
  geom_histogram(breaks = limites,  
                 color="black",  
                 fill="gray")
```

Para isso utilizamos `y=..density...`



# Densidade de frequência ( $d_i$ )

Agora vamos atualizar a tabela com o valor de densidade de frequência, dado por:

$$d_i = \frac{f_i}{\Delta_i}$$

```
tab <- dados_turmas %>%  
  mutate(  
    classes_altura = cut(altura, 5)  
  ) %>%  
  group_by(classes_altura) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
         perc = fi*100,  
         di=fi/Di)  
View(tab)
```

<b>classes_altura</b>	<b>ni</b>	<b>fi</b>	<b>perc</b>	<b>di</b>
(1.5,1.59]	7	0.0886076	8.86076	1.030321
(1.59,1.67]	15	0.1898734	18.98734	2.207830
(1.67,1.76]	28	0.3544304	35.44304	4.121283
(1.76,1.84]	19	0.2405063	24.05063	2.796585
(1.84,1.93]	10	0.1265823	12.65823	1.471887

# Medidas de Frequências Acumuladas

As medidas acumuladas são interessantes para compor algumas visualizações:

$N_i$ : Frequência Absoluta Acumulada.

$F_i$ : Frequência Relativa Acumulada.

*Perc*: Porcentagem de Frequência Acumulada.

```
tab <- dados_turmas %>%  
  mutate(  
    classes_altura = cut(altura, 5)  
  ) %>%  
  group_by(classes_altura) %>%  
  summarise(ni=n()) %>%  
  mutate(fi = ni/sum(ni),  
         perc = fi*100,  
         di=fi/Di,  
         Ni = cumsum(ni),  
         Fi = cumsum(fi),  
         Perc = cumsum(perc))
```

```
View(tab)
```



# Tabela de Frequência para Altura

classes_altura	ni	fi	perc	di	Ni	Fi	Perc
(1.5,1.59]	7	0.0886076	8.86076	1.030321	7	0.0886076	8.86076
(1.59,1.67]	15	0.1898734	18.98734	2.207830	22	0.2784810	27.84810
(1.67,1.76]	28	0.3544304	35.44304	4.121283	50	0.6329114	63.29114
(1.76,1.84]	19	0.2405063	24.05063	2.796585	69	0.8734177	87.34177
(1.84,1.93]	10	0.1265823	12.65823	1.471887	79	1.0000000	100.00000

# Histograma e Polígono de Frequência

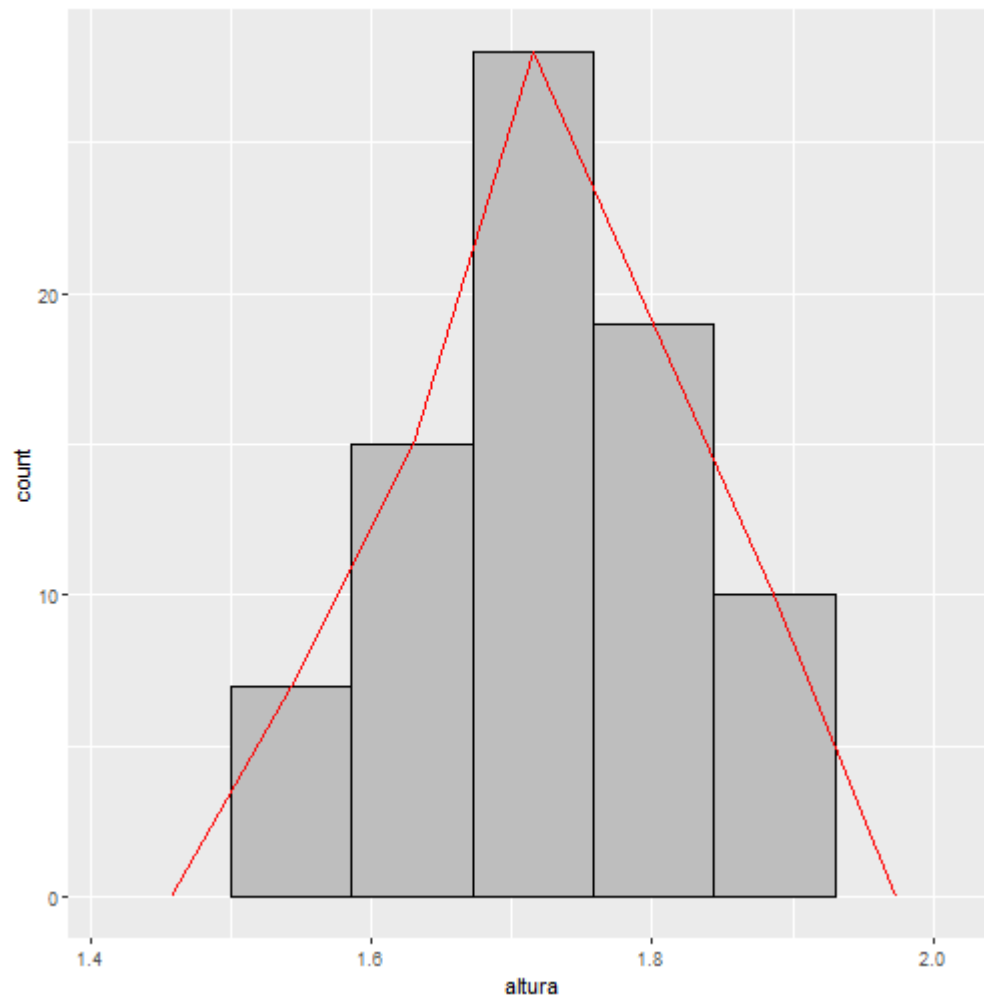
Obtemos o polígono de frequências unindo por uma poligonal (segmentos de retas) os pontos correspondentes às frequências, das classes, centradas nos pontos médios de cada classe.

Para se obter as interseções do polígono com o eixo horizontal, cria-se em cada extremo do histograma uma classe com frequência nula.

No R:

```
dados_turmas %>%  
  ggplot(aes(x=altura, y=..count..))+  
  geom_histogram(breaks = limites,  
                 color="black",  
                 fill="gray") +  
  geom_freqpoly(breaks=limites,color="red")
```

Observe que o histograma foi construído com as frequências absolutas ( $n_i$ ), ou seja, `y=..count...`

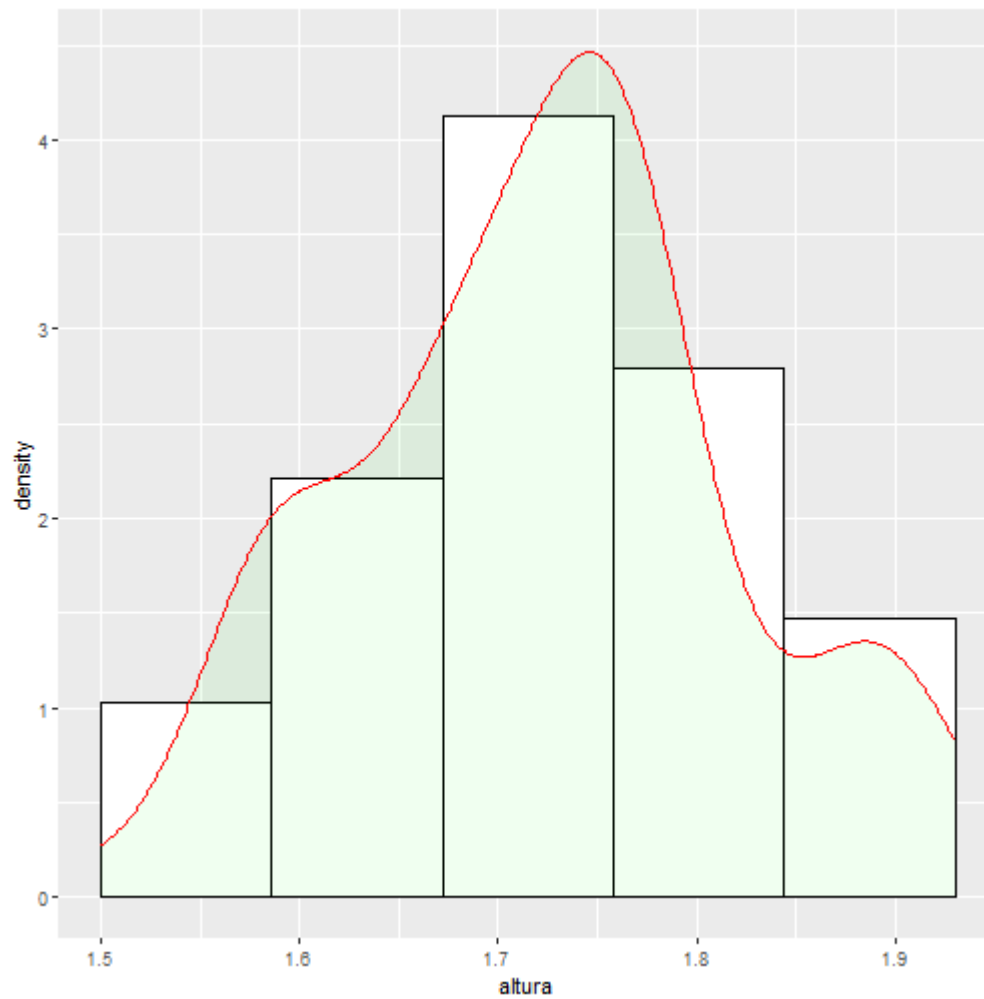


# Histograma e Estimativas de densidade suavizadas

Calcula e desenha a estimativa da densidade, que é uma versão suavizada do histograma. Esta é uma alternativa útil para dados contínuos que vêm de uma distribuição suave subjacente.

```
dados_turmas %>%  
  ggplot(aes(x=altura,y=..density..))+  
  geom_histogram(breaks = limites,  
                 color="black",  
                 fill="white") +  
  geom_density(color="red",  
               fill="green",  
               alpha=0.05)
```

- Observe que o histograma foi construído com as frequências absolutas ( $n_i$ ), ou seja,  $y=..density..$ . Utilizamos a função `geom_density()`.
- O argumento `alpha=0.05` controla a transparência do preenchimento.



# Função de Distribuição Acumulada

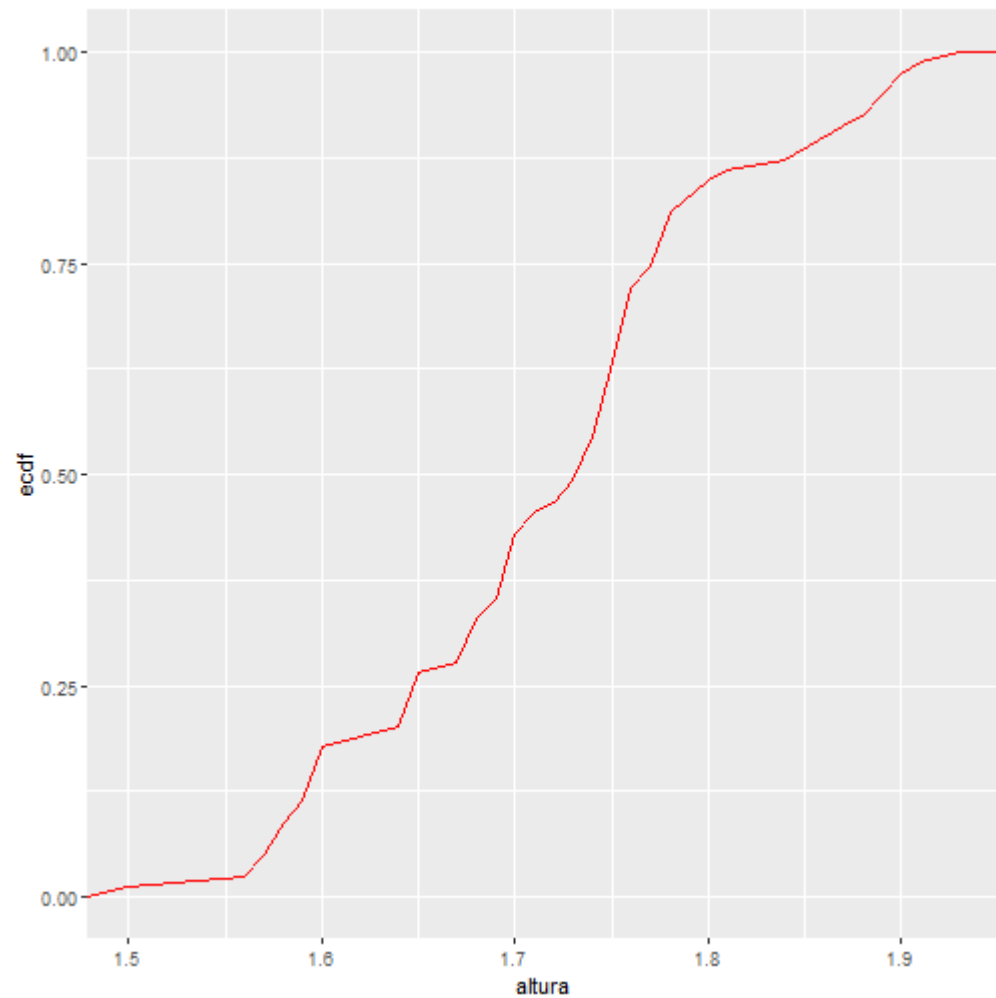
A função de distribuição acumulada descreve como probabilidades são associadas aos valores ou aos intervalos de valores de uma variável aleatória. Em outras palavras, ela representa a probabilidade de uma variável aleatória ser menor ou igual a um valor real qualquer  $x$ .

$$F(x) = P(X \leq x) \in [0, 1].$$

Para uma variável aleatória contínua (altura):

$$\int_{-\infty}^x f(x_i) dx$$

```
dados_turmas %>%  
  ggplot(aes(x=altura))+  
  stat_ecdf(geom = "line",color="red")
```



# Função de Distribuição Acumulada

No caso da variável aleatória discreta (idade\_anos):

$$F(x) = \sum_{x_i < x} f(x_i)$$

```
dados_turmas %>%  
  ggplot(aes(x=idade_anos))+  
  stat_ecdf(geom = "line",color="red")
```



