

Estatística e Informática

Aula 04 - Distribuição de Frequência

Alan Rodrigo Panosso alan.panosso@unesp.br

Departamento de Engenharia e Ciências Exatas FCAV/UNESP

(06-05-2021)

Distribuição de Frequência

Distribuição de frequência de uma variável

Quando se estuda uma variável (qualitativa ou quantitativa), deve-se conhecer a distribuição de frequência dessa variável por meio de suas possíveis realizações (dados). Portanto, o objetivo dessa aula será apresentar as principais formas e visualização de variáveis quali e quantitativas.

Exemplo: Considerando os **Dados** (**exemplo_dados.xlsx**) amostrados da turma de Estatística e Informática, temos:

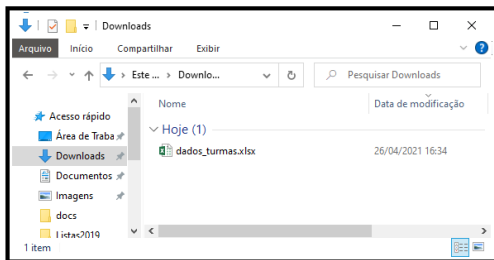
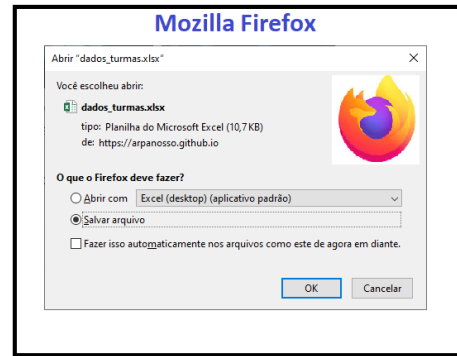
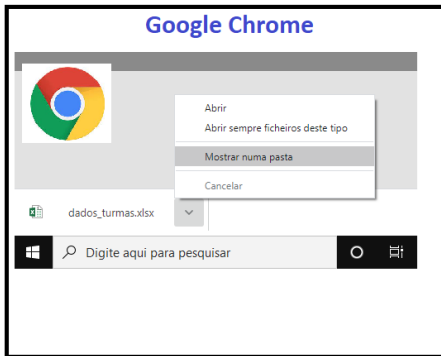
	A	B	C	D	E	F
1	id	sexo	cor_cabelo	GA	altura	idade
2	1	F	CC	mais_social	1.68	19
3	2	F	CE	nao_consomme	1.59	20
4	3	F	CC	pouco	1.7	49
5	4	F	CE	socialmente	1.5	20
6	5	M	CE	mais_social	1.76	23
7	6	M	CC	pouco	1.6	28
8	7	M	L	nao_consomme	1.84	19

Carregando os dados no R

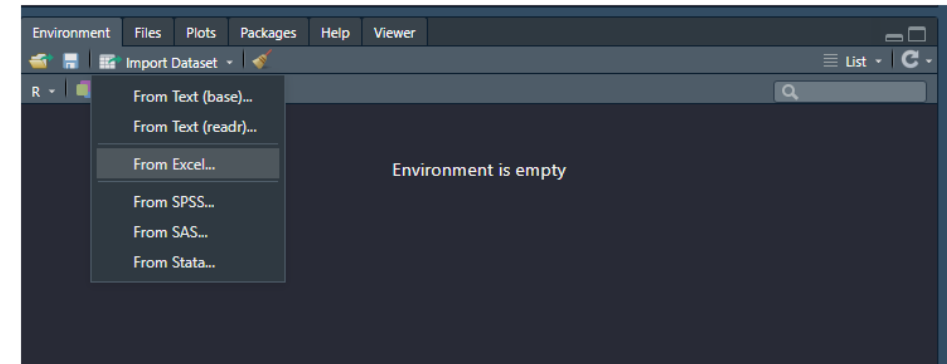
Para carregar o banco de dados da turma no R, siga os passos:

1. Faça o Download dos **Dados**.

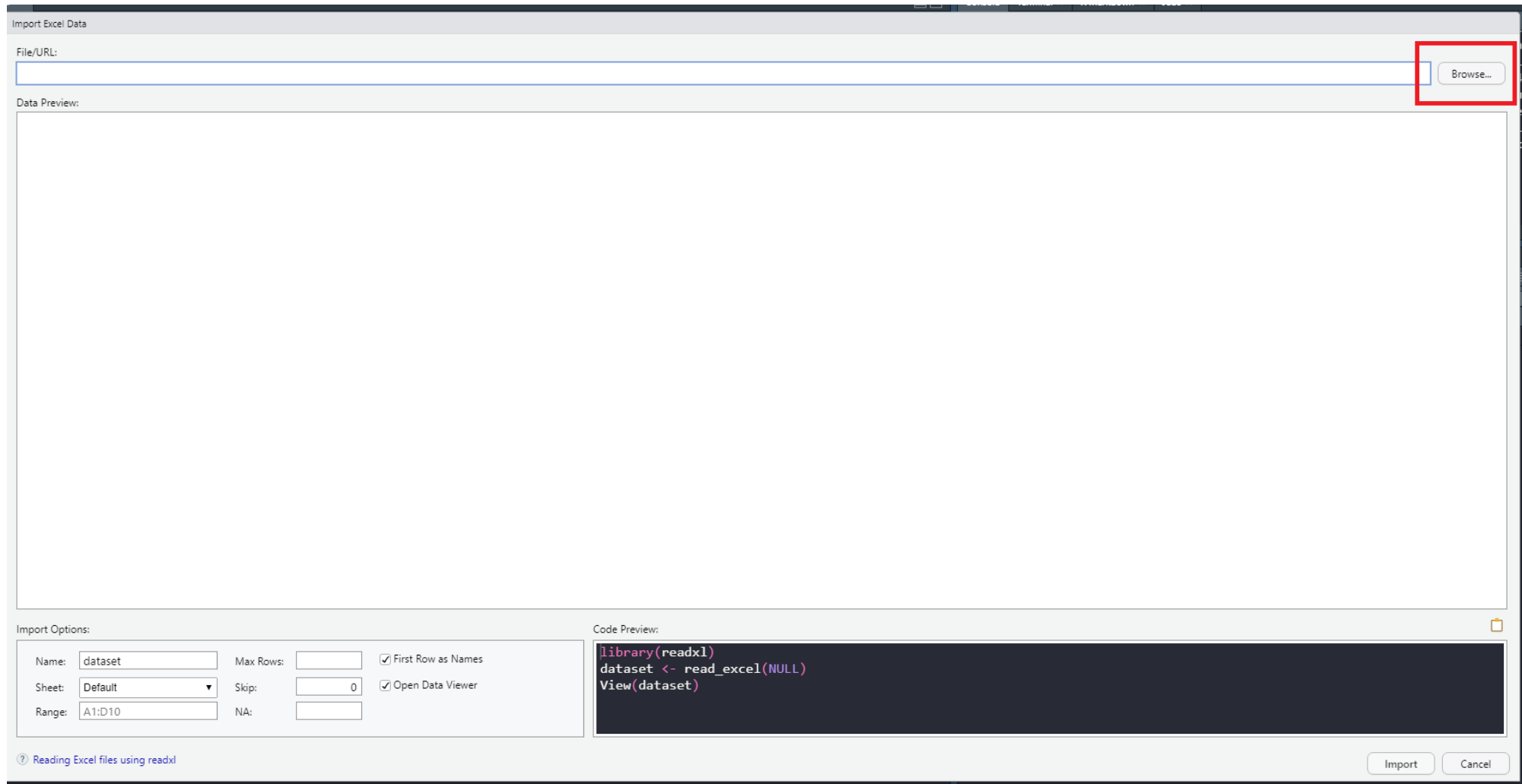
2. Salve em uma pasta do seu computador (no exemplo a pasta é "Downloads").



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:

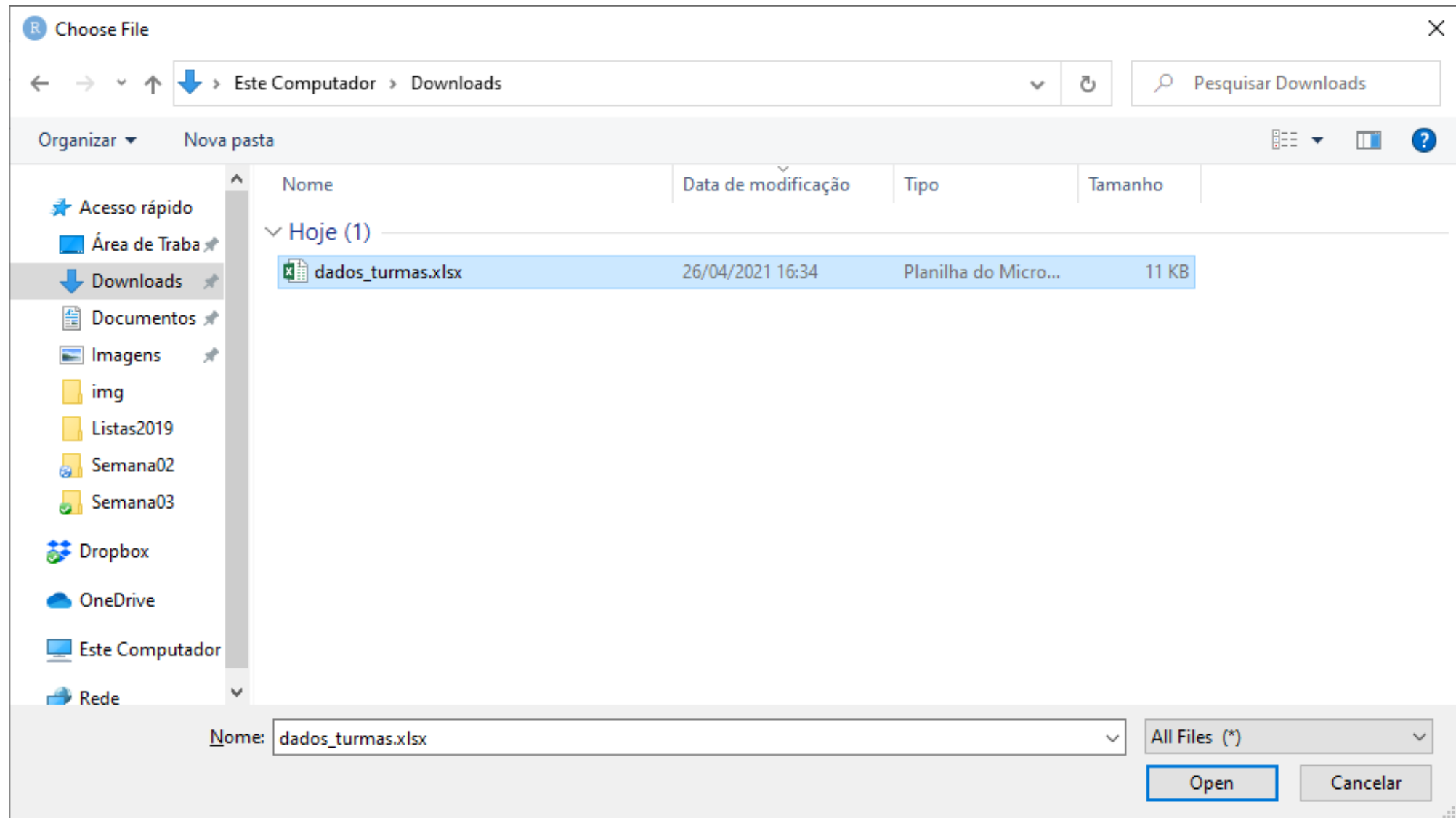
Code Preview:

```
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 **Downloads**. Selecione o arquivo e clique em **Open**.



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

Import Excel Data

File/URL:
C:/Users/Usuario/Downloads/dados_turmas.xlsx Browse...

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	socialmente	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	socialmente	1.76	21
12	M	CE	socialmente	1.91	21
13	M	CC	socialmente	1.68	21
14	M	CE	pouco	1.70	18
15	M	CE	socialmente	1.76	19
16	F	CE	pouco	1.65	20
17	F	CC	socialmente	1.58	19
18	M	CE	socialmente	1.87	19
19	M	CE	socialmente	1.75	18
20	F	C	mais_social	1.69	22
21	M	C	socialmente	1.74	19

Previewing first 50 entries.

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

Import Cancel

? Reading Excel files using readxl

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 código `install.packages("readxl")`.

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


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) # não esqueça de instalar o tidyverse para ter o glimpse
glimpse(dados_turmas) # vislumbre, resumo rápido dos dados
```

```
#> Rows: 49
#> Columns: 6
#> $ id      <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
#> $ sexo    <chr> "F", "F", "F", "F", "M", "M", "M", "M", "M", "M", "M", "M", "M", "M", "M", "M", "F", "F",
#> $ cor_cabelo <chr> "CC", "CE", "CC", "CE", "CE", "CC", "L", "CE", "CC", "C", "CC", "CE", "CC", "CE", "
#> $ GA      <chr> "mais_social", "nao_consomme", "pouco", "socialmente", "mais_social", "pouco", "nao_
```

Exemplo da base de dados das turmas

Construir uma tabela de frequências para a variável `sexo` contendo as frequências absolutas e relativas e a porcentagem para as categoria existentes. Após isso, realizar uma visualização de dados com gráficos de Colunas, Barras e Setores (Pizza oi *Pie*).

Para isso vamos utilizar o R para contar as frequências absolutas das classes presentes na variável qualitativa nominal `sexo`. Precisaremos fazer algumas operações nos dados das turmas e para isso 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())  
tab
```

```
#> # A tibble: 2 x 2  
#>   sexo      ni  
#>   <chr> <int>  
#> 1 F      17  
#> 2 M      32
```

- `group_by()` a grupa as categorias da variável `sexo`.
- `summarise` vai criar o resumo dos dados, ou seja, contará o valor de cada categoria e mostrará em `ni`.

Frequência Absoluta (n_i)

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

Então temos $n_F = 17$ e $n_M = 32$, cuja soma é $n = 49$

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).

Frequência Relativa (f_i) ou proporção

É definida como a proporção de cada realização 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

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

```
#> # A tibble: 2 x 3  
#>   sexo    ni    fi  
#>   <chr> <int> <dbl>  
#> 1 F      17 0.347  
#> 2 M      32 0.653
```

- `mutate()` permitirá criarmos mais uma coluna na nossa tabela de resumo, no caso a frequência relativa

Portanto: $f_F = \frac{17}{49} = 0,347$ e $f_M = \frac{32}{49} = 0,653$

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

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

onde k é o número de categorias da variável `sexo`, ou seja, 2.

Porcentagem de frequência (*perc* ou %)

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

```
tab<-dados_turmas %>%  
  group_by(sexo) %>%  
  summarise(ni=n()) %>%      # Frequência Absoluta  
  mutate(fi = ni/sum(ni),    # Frequência relativa  
          perc = fi*100)     # Porcentagem de frequência  
tab
```

```
#> # A tibble: 2 x 4  
#>   sexo    ni    fi  perc  
#>   <chr> <int> <dbl> <dbl>  
#> 1 F      17 0.347  34.7  
#> 2 M      32 0.653  65.3
```

OBS: A soma das Porcentagens de frequência é igual a 100%.

Agora, a partir das tabelas de frequências, poderemos criar representações gráficas que nos auxiliarão na apresentação e interpretação do comportamento dos dados da variável estudada.

Essa etapa é a **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 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`, e serão construídos a partir da tabela `tab` anterior.

O `ggplot` funciona com camadas gráficas de representação e formatação, que são adicionadas à base gráfica por meio do operador de adição `+` digitado ao final da linha.

Gráfico de Colunas para Sexo

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

```
tab %>%  
  ggplot(aes(x=sexo, y=fi)) +  
  geom_col()
```

Gráfico de Colunas para Sexo

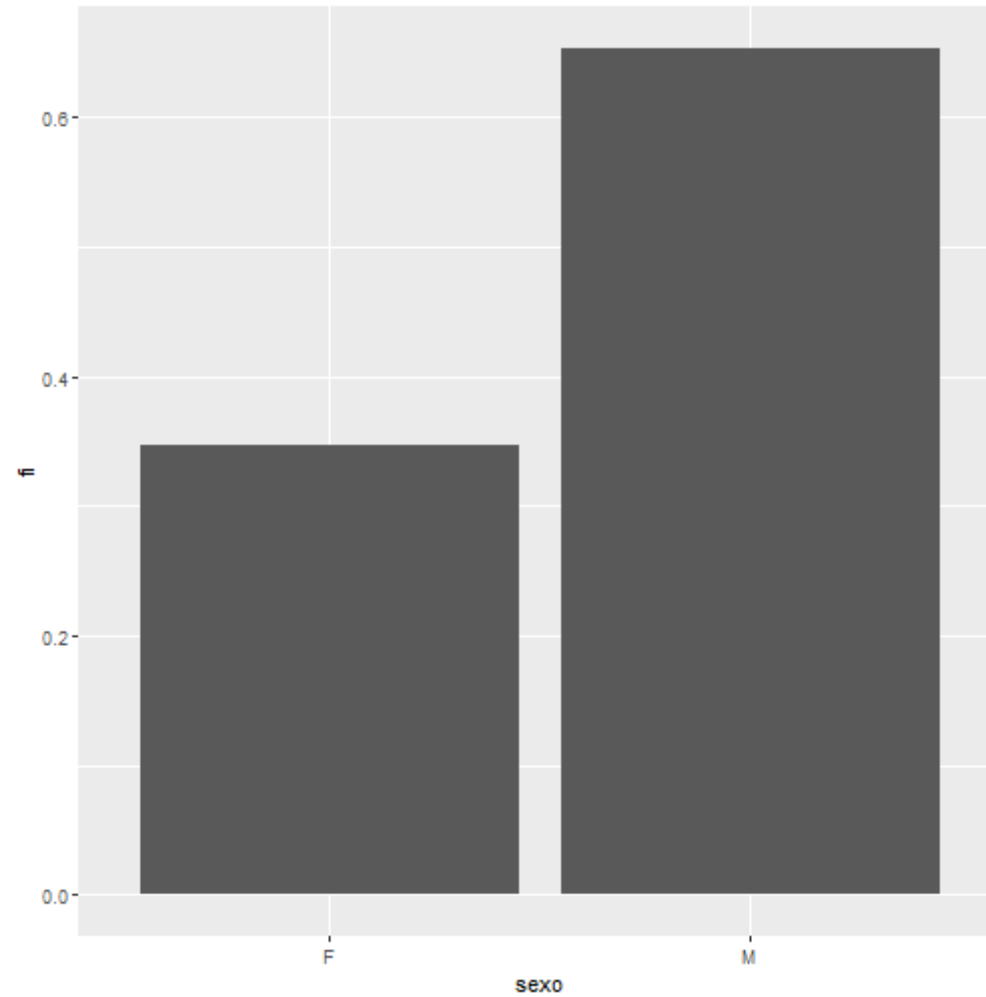


Gráfico de Barras para Sexo

Semelhante ao gráfico de colunas, contudo, com as barras na horizontal, facilita a leitura do nome das categorias, pois muitas vezes esses podem ser extensos.

```
tab %>%  
  ggplot(aes(x=fi,y=sexo)) +  
  geom_col(fill="aquamarine4")
```

- o argumento `fill = "aquamarine4"` permite que possamos alterar a cor do preenchimento (*fill*) da barra para a cor "aquamarine4", outras cores são possíveis tente algumas **das cores do R**.

Gráfico de Setores para Sexo

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

```
tab %>%  
  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.

Tabela de Frequência e para a variável Cor de cabelo

Vamos, mais uma vez, utilizar o R para conseguirmos as tabelas e a representação gráfica. Observe que esse é o mesmo código daquele apresentado no slide 13, mas com o nome da coluna `cor_cabelo` ao invés de `sexo` dentro da função `group_by()`

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

```
#> # A tibble: 5 x 4  
#>   cor_cabelo    ni      fi  perc  
#>   <chr>      <int>  <dbl> <dbl>  
#> 1 C          6 0.122  12.2  
#> 2 CC         10 0.204  20.4  
#> 3 CE         25 0.510  51.0  
#> 4 L          4 0.0816  8.16  
#> 5 P          4 0.0816  8.16
```

Gráfico de Colunas para Cor de cabelo

```
tab %>%  
  ggplot(aes(x=cor_cabelo,y=fi,  
             fill=cor_cabelo)) +  
  geom_col()
```

- 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.

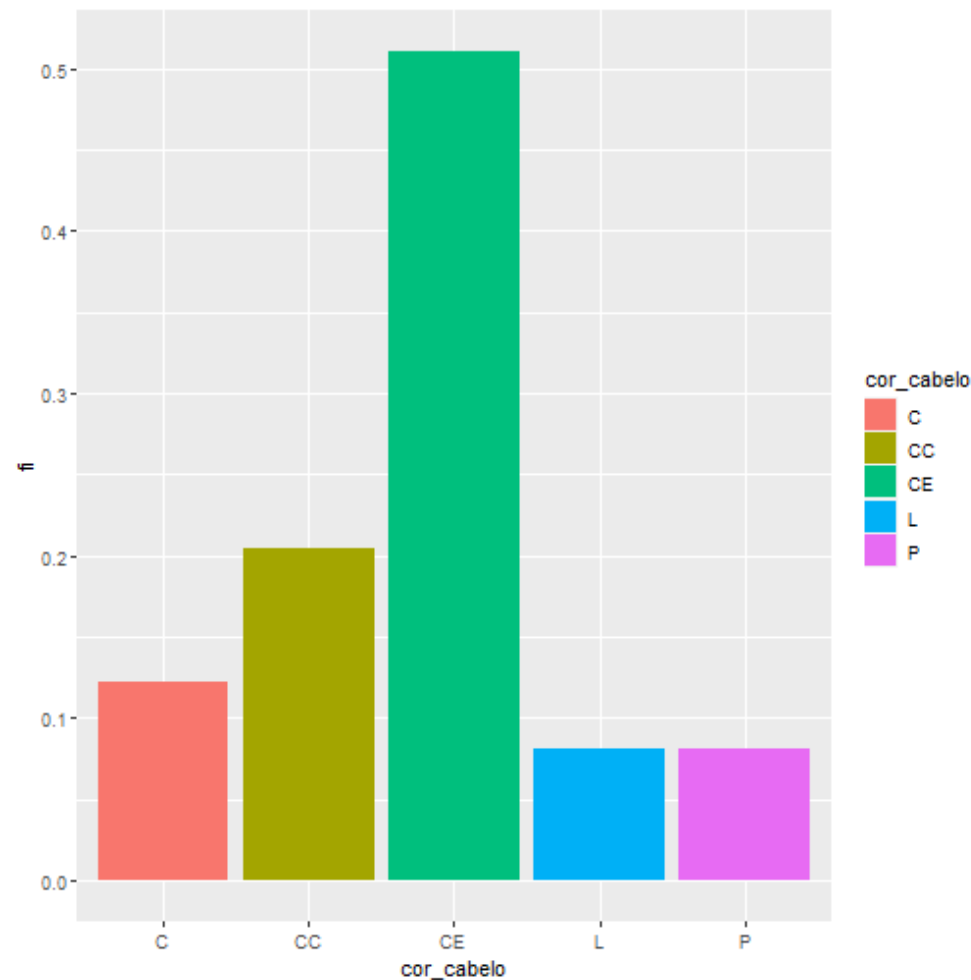


Gráfico de Barras para Cor de cabelo

```
tab %>%  
  ggplot(aes(x=fi, y=cor_cabelo,  
             fill=cor_cabelo)) +  
  geom_col()
```

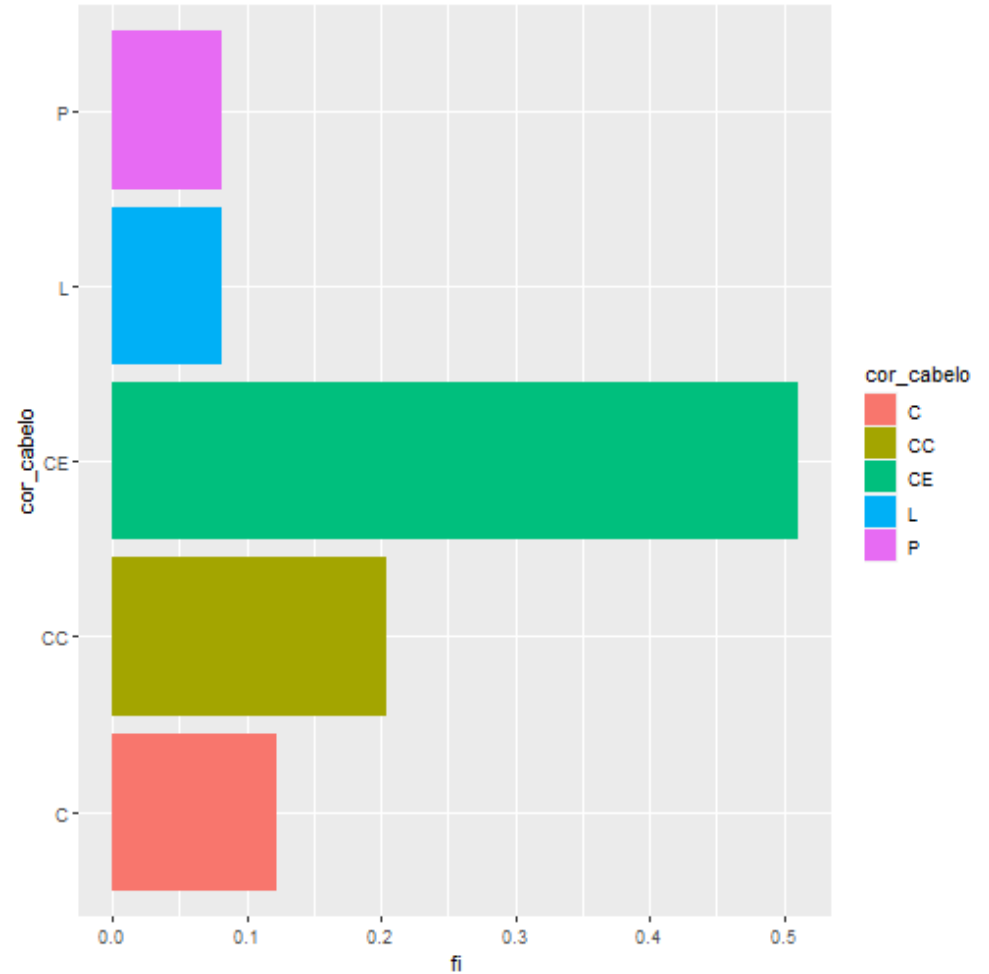
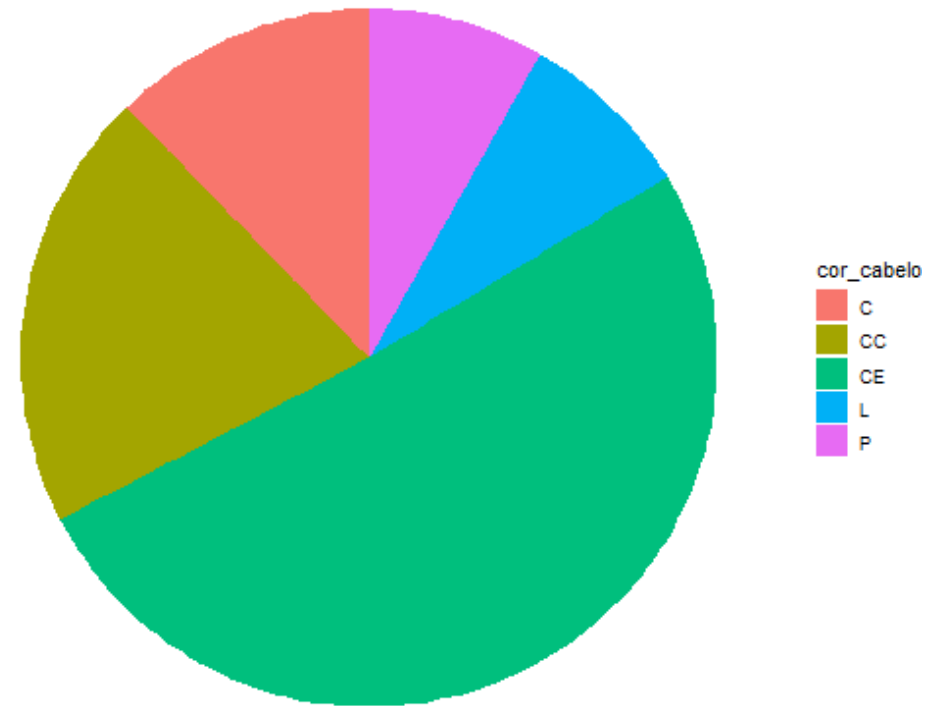


Gráfico de Setores para cor_cabelo

```
tab %>%  
  ggplot(aes(x="", y=fi,  
             fill=cor_cabelo)) +  
  geom_bar(stat="identity") +  
  coord_polar("y", start=0) +  
  theme_void()
```



Visualização dos dados

(variáveis quantitativas)

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

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

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

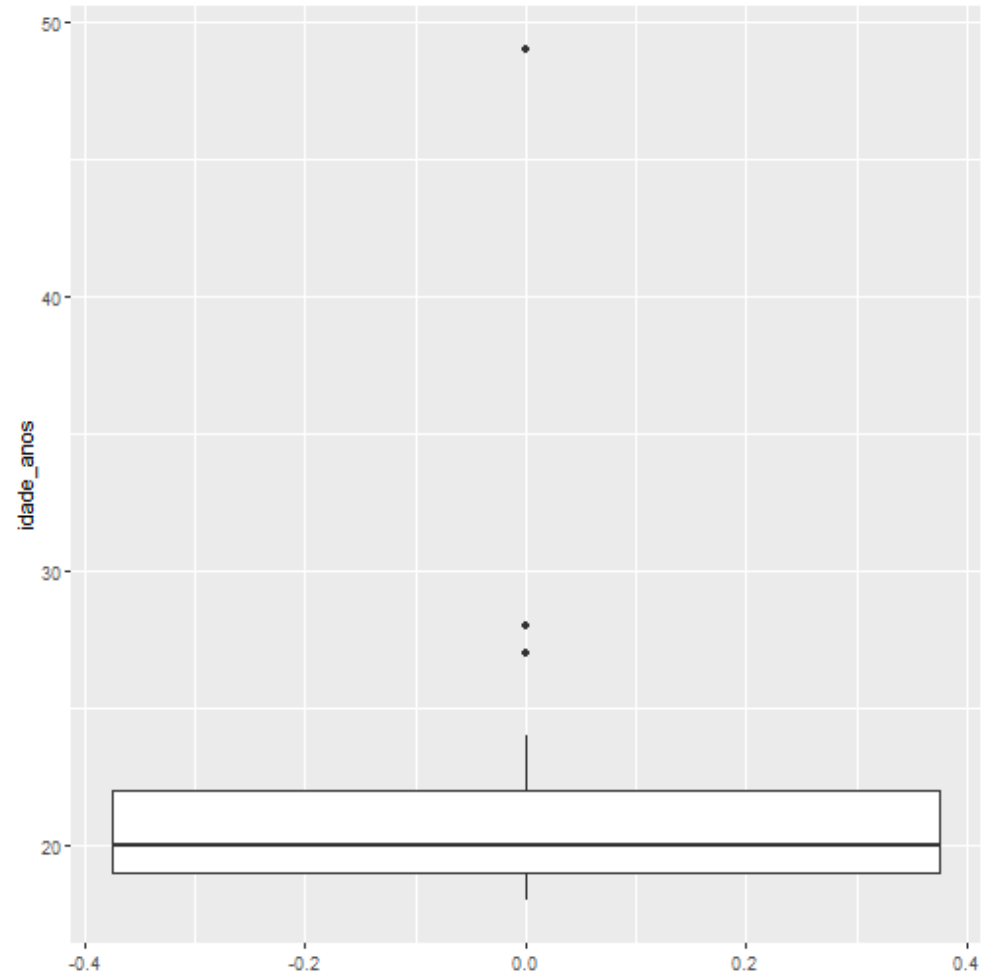
idade_anos	ni	fi	perc
18	8	0.1632653	16.326531
19	14	0.2857143	28.571429
20	11	0.2244898	22.448980
21	3	0.0612245	6.122449
22	3	0.0612245	6.122449
23	6	0.1224490	12.244898
24	1	0.0204082	2.040816
27	1	0.0204082	2.040816
28	1	0.0204082	2.040816
49	1	0.0204082	2.040816

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`.

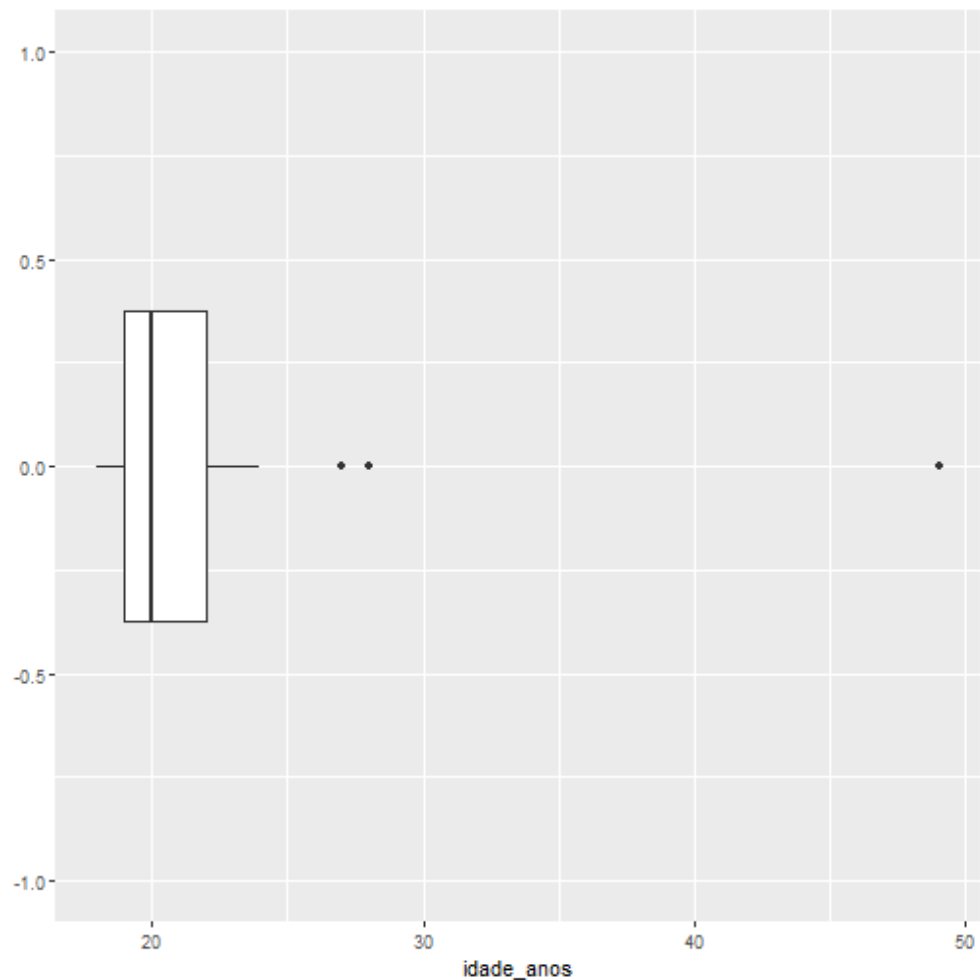
```
dados_turmas %>%  
  ggplot(aes(y=idade_anos)) +  
  geom_boxplot()
```



- 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 vez o IIQ de cada limite da caixa. Esses pontos fora da curva, denominados **outliers** são incomuns, então são plotados individualmente.
- Uma linha (ou bigode de gato, daí o nome *Box and Whiskers*) que se estende de cada lado da caixa e vai até o ponto mais distante da distribuição que não seja um valor incomum **outlier**.

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))`.

```
dados_turmas %>%  
  ggplot(aes(x=idade_anos)) +  
  geom_boxplot()+  
  coord_cartesian(ylim=c(-1,1))
```



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

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

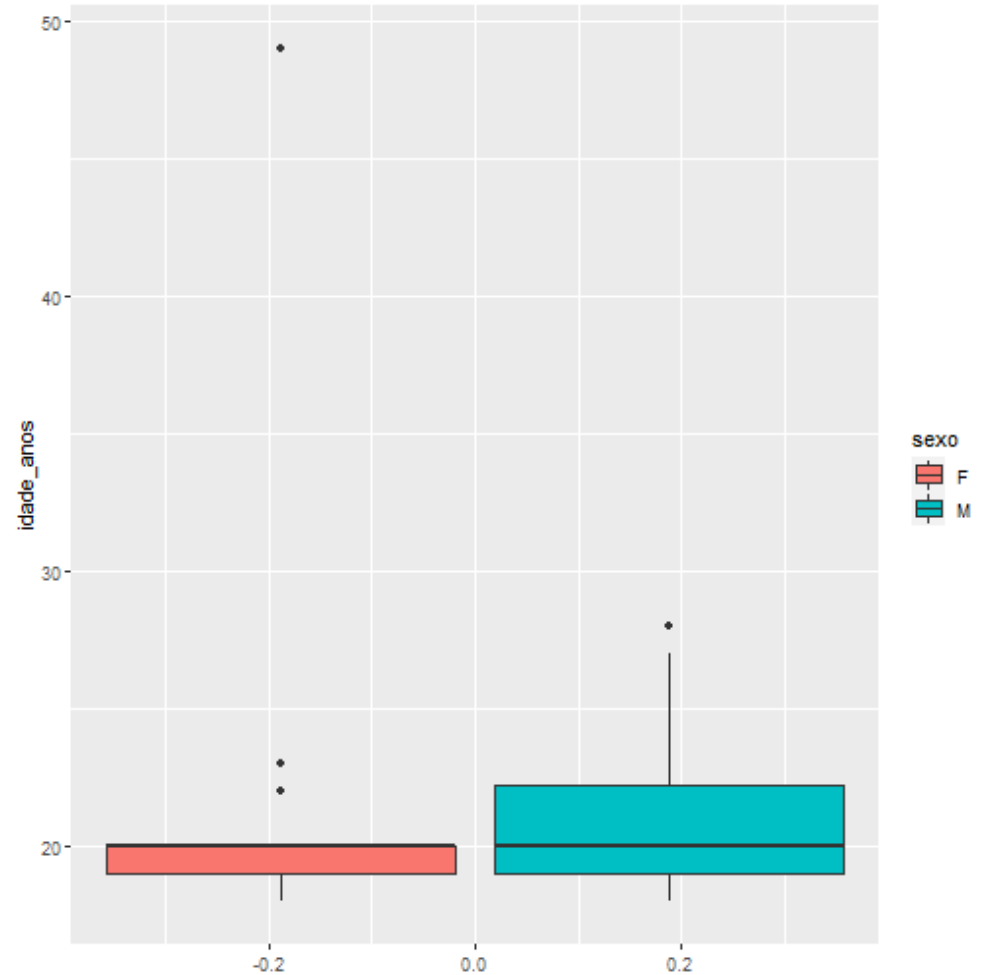


Tabela de frequência e visualização para Altura em m (contínua)

Quando a variável quantitativa for contínua, recomendamos a utilização dos gráficos boxplot e o histograma.

Devemos, inicialmente construir a tabela de frequência da variável. Devemos lembrar que seus valores não se repetem, pois não sabemos o valor real da observação e sim temos apenas uma aproximação dada pelo instrumento de medida.

Assim, 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
(1.5,1.58]	5	0.1020408	10.20408
(1.58,1.66]	8	0.1632653	16.32653
(1.66,1.75]	12	0.2448980	24.48980
(1.75,1.83]	17	0.3469388	34.69388
(1.83,1.91]	7	0.1428571	14.28571

Amplitude Total

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

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

Para os dados de altura temos:

$$\Delta = \text{Máximo} - \text{Mínimo} = 1,91 \text{ m} - 1,50 \text{ m} = 0,41 \text{ m}$$

No R podemos calcular a amplitude total com as funções do pacote base, para isso deveremos, 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
```

Número de intervalos de classes (k)

Definiremos k como sendo o número de **subintervalos** 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 (Δ_i)

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

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

Para os dados de altura:

$$\Delta_i = \frac{\Delta}{k} = \frac{0,41 \text{ m}}{5} = 0,082 \text{ m}$$

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

```
#> [1] 0.082
```

Cada um dos 5 intervalos terá uma amplitude de 0,082 m. Ou seja, o cálculo dos limites das classes é feito a partir da adição ao valor Mínimo o valor de Δ_i k vezes:

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

```
#> [1] 1.500 1.582 1.664 1.746 1.828 1.910
```

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.

classes_altura	ni	fi	perc
(1.5,1.58]	5	0.1020408	10.20408
(1.58,1.66]	8	0.1632653	16.32653
(1.66,1.75]	12	0.2448980	24.48980
(1.75,1.83]	17	0.3469388	34.69388
(1.83,1.91]	7	0.1428571	14.28571

```
#> [1] 1.500 1.582 1.664 1.746 1.828 1.910
```

Agora podemos criar um gráfico de barras para cada uma dessas classes, denominado hitograma:

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.

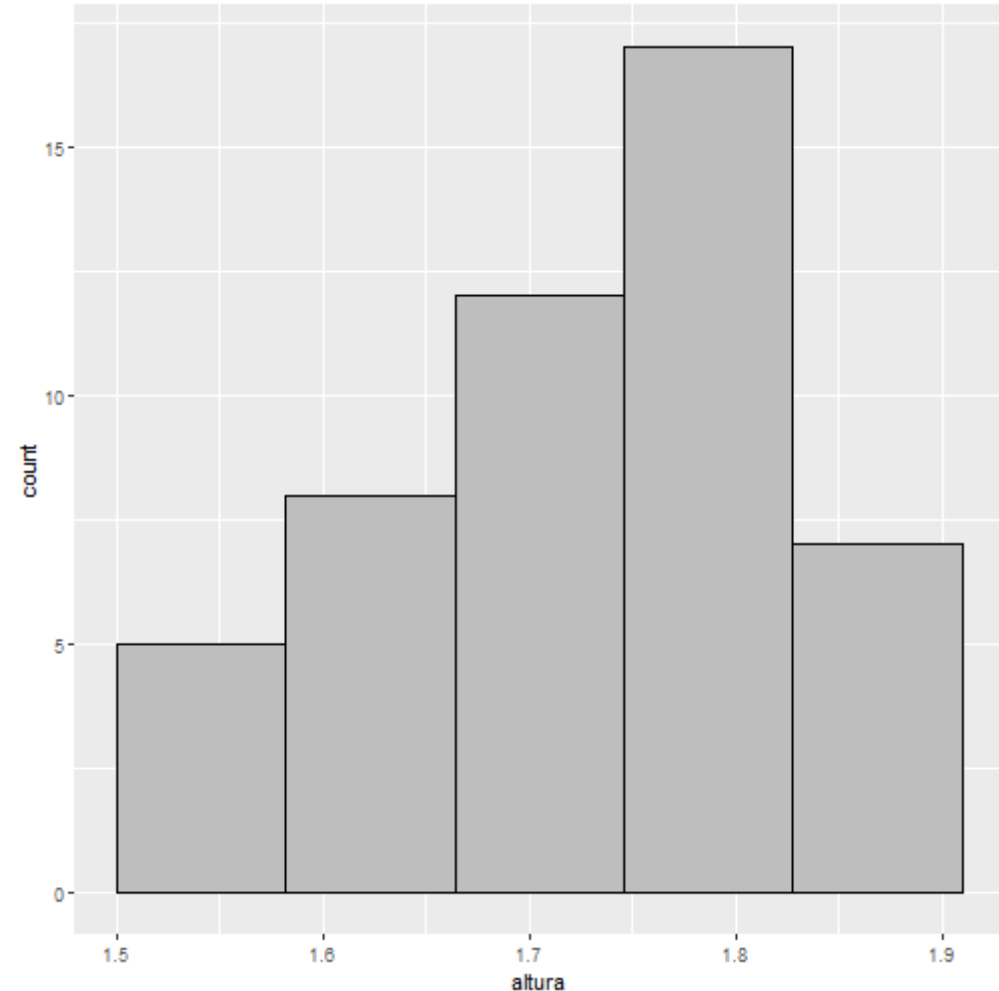


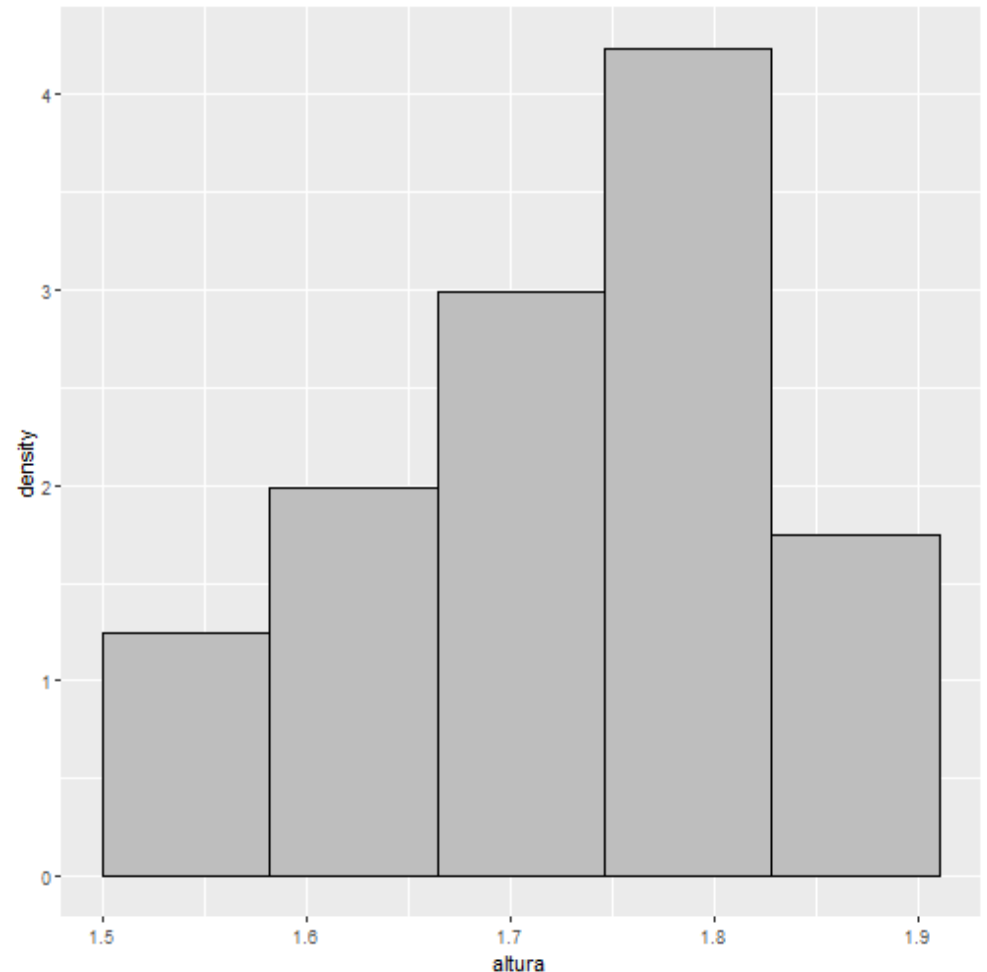
Gráfico histograma (frequências relativas)

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/Δ_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)
```

classes_altura	ni	fi	perc	di
(1.5,1.58]	5	0.1020408	10.20408	1.244400
(1.58,1.66]	8	0.1632653	16.32653	1.991040
(1.66,1.75]	12	0.2448980	24.48980	2.986560
(1.75,1.83]	17	0.3469388	34.69388	4.230961
(1.83,1.91]	7	0.1428571	14.28571	1.742160