

Probabilidade e Estatística com R

Fernando Náufel

(versão de 10/11/2021)

Sumário

Apresentação	2
Exercício	3
1 O Que É Estatística?	4
1.1 Vídeo 1	4
1.2 Exercícios	4
1.3 Vídeo 2	7
1.4 Exercícios	7
2 Introdução a R	8
2.1 Vídeo 1	8
2.2 Vídeo 2	8
2.3 Exercícios	8
3 Visualização com ggplot2	10
3.1 Vídeo 1	10
3.2 Componentes de um gráfico ggplot2	10
3.3 Conjunto de dados	13
3.4 Gráficos de dispersão (<i>scatter plots</i>)	19
3.5 Histogramas e cia.	33
3.6 Ogiva (frequência acumulada)	36
3.7 Ramos e folhas	37
3.8 Exercícios	38

Apresentação

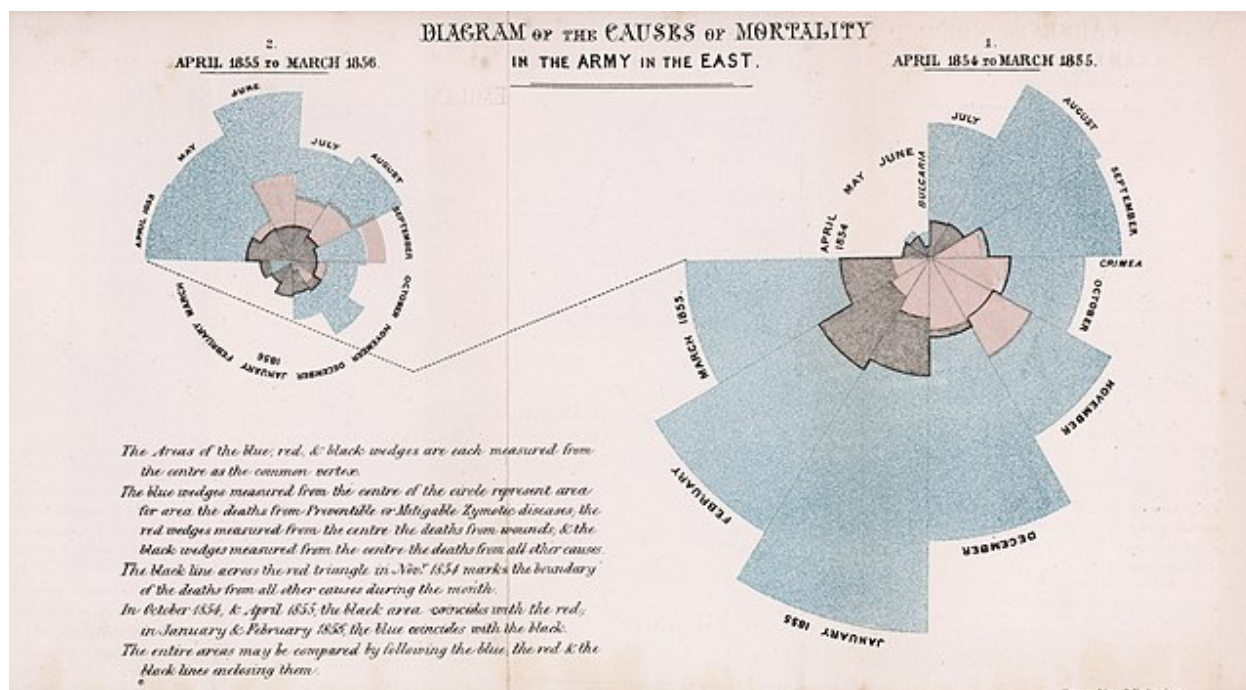
Atenção



Este material ainda está em construção.

Pode haver mudanças a qualquer momento.

Verifique, no rodapé da página web ou na capa do arquivo pdf, a data desta versão.



Este livro/site foi iniciado em 2020, durante a pandemia de COVID-19, quando a Universidade Federal Fluminense (UFF) funcionou em regime de ensino remoto durante mais de

um ano.

Para atender os alunos do curso de Probabilidade e Estatística do curso de graduação em Ciência da Computação da UFF, decidi gravar aulas em vídeo e disponibilizar os arquivos usados nelas. Foram esses arquivos que deram origem a este livro/site.

Para tirar o máximo proveito deste material, você deve fazer o seguinte:

1. Assistir aos vídeos contidos em cada capítulo. A *playlist* completa está em <https://www.youtube.com/playlist?list=PL7SRLwLs7ocaV-Y1vrVU3W7mZnnS0qkWV>.
2. Instalar o R no seu computador ou abrir uma conta no RStudio Cloud, para poder usar o R *online*. Você encontra instruções para fazer isto no [capítulo de introdução a R](#).
3. Seguir os *links* para outras fontes *online* que abordam assuntos que não são cobertos em detalhes neste curso.
4. Fazer os exercícios. Ao longo do tempo, acrescentarei *links* para vídeos explicando as soluções.

O código-fonte deste livro/site pode ser encontrado neste repositório do Github¹.

Se você preferir ler este livro em pdf, ou se quiser imprimi-lo, faça o download do arquivo [aqui](#)².

Exercício

1. Pesquise sobre a imagem do início deste capítulo. Ela foi criada em 1858 por Florence Nightingale.

¹<https://github.com/fnaufel/probestr>

²<https://github.com/fnaufel/probestr/blob/master/docs/probestr.pdf>

O Que É Estatística?

1.1

Vídeo 1

https://youtu.be/6Q_XSoLCIpc

1.2

Exercícios

1. Você está interessado em estimar a altura de todos os homens da sua faculdade. Para isso, você decide medir as alturas de todos os homens da sua turma de Estatística.
 - Qual é a amostra?
 - Qual é a população?
2. Um instituto de pesquisa entrevista um grupo de 1000 pessoas, perguntando a cada uma se ela vai votar a favor do candidato A na próxima eleição. Dos entrevistados, 600 responderam que sim. A proporção 0,6 (ou 60%) é uma estatística ou um parâmetro?
3. Você vê alguma diferença entre as cinco situações abaixo? Quais das situações são equivalentes em termos da probabilidade de conseguir 5 cartas do mesmo naipe?
 - a. Usando um baralho normal, você retira 10 cartas e registra as cartas retiradas.

- b. Usando um baralho normal, você repete a seguinte sequência de ações 10 vezes: retirar uma carta do baralho, registrar a carta retirada e repor a carta no baralho.
 - c. Usando uma caixa contendo todas as cartas de 1 milhão de baralhos reunidos, você retira 10 cartas e registra as cartas retiradas.
 - d. Usando uma caixa contendo todas as cartas de 1 milhão de baralhos reunidos, você repete a seguinte sequência de ações 10 vezes: retirar uma carta da caixa, registrar a carta retirada e repor a carta na caixa.
 - e. Usando um baralho *infinito*, você retira 10 cartas e registra as cartas retiradas.
 - f. Usando um baralho *infinito*, você repete a seguinte sequência de ações 10 vezes: retirar uma carta do baralho, registrar a carta retirada e repor a carta no baralho.
4. Qual a graça dos quadrinhos na Figura 1.1, que também aparecem no vídeo¹?



Figura 1.1: <http://xkcd.com/552/>

5. Qual a graça dos quadrinhos na Figura 1.2?
6. Veja este vídeo sobre o cavalo Hans:

<https://youtu.be/G3VkCmdUfZE>

Qual a relação entre esta história e a necessidade de duplo cegamento?

¹https://youtu.be/6Q_XSoLCIpc?t=1385



Figura 1.2: <http://xkcd.com/1462/>

1.3

Vídeo 2

<https://youtu.be/492VASxIDRo>

1.4

Exercícios

1. Por que não faz sentido calcular a média dos CEPs de um grupo de pessoas?
2. Uma temperatura de -40 graus Celsius é igual a uma temperatura de -40 graus Fahrenheit?
3. Uma temperatura de zero graus Celsius é igual a uma temperatura de zero graus Fahrenheit?
4. Uma variação de temperatura de 1 grau Celsius é igual a uma variação de temperatura de 1 grau Fahrenheit?
5. Um saldo bancário de zero reais é igual a um saldo bancário de zero dólares?
6. Um produto de 1 milhão de reais custa o mesmo que um produto de 1 milhão de dólares?
7. Meses representados por números de 1 a 12 são dados de que nível?

Introdução a R

2.1

Vídeo 1

<https://youtu.be/1kXQDNqm41c>

2.2

Vídeo 2

<https://youtu.be/3GEc1oiKDrU>

2.3

Exercícios

1. Para criar sua conta no RStudio Cloud, acesse <https://rstudio.cloud/>.
2. Se você preferir instalar o R no seu computador, acesse
 - <https://cran.r-project.org/> para baixar e instalar o R, e
 - <https://rstudio.com/products/rstudio/download/> para baixar e instalar o RStudio, um IDE específico para R.

3. Abra o RStudio Cloud ou o seu RStudio instalado localmente.
4. Crie um novo projeto. Sempre trabalhe em projetos para ter seus arquivos organizados.

5. Para instalar o `swirl` (pacote do R para exercícios interativos)¹, execute o seguinte comando no console do RStudio:

```
install.packages("swirl")
```

6. Para instalar os exercícios de introdução a R, execute os seguintes comandos no console do RStudio:

```
library(swirl)  
install_course_github('fnaufel', 'introR')
```

7. Mude o idioma para português e execute o `swirl`.

```
select_language('portuguese', append_rprofile = TRUE)  
swirl()
```

8. Na primeira execução, você vai precisar se identificar (qualquer nome serve). Com essa identificação, o `swirl` vai registrar o seu progresso nas lições.
9. No `swirl`, as perguntas são mostradas no console. Você também deve responder no console.
10. Às vezes, um *script* será aberto no editor de textos para que você complete um programa. Quando seu programa estiver pronto, salve o arquivo e digite `submit()` no console para o `swirl` processar o *script*.
11. O `swirl` dá instruções claras no console. Na dúvida, digite `info()` no *prompt* do R (`>`).
12. Se, em vez do *prompt* do R, o console mostrar reticências (`...`), tecele *Enter*.
13. Se nada funcionar, tecele *ESC*.
14. Para sair do `swirl()`, digite `bye()` no *prompt* do R.
15. Para voltar para os exercícios, digite

```
library(swirl)  
swirl()
```

¹<https://swirlstats.com/>

Visualização com ggplot2

3.1

Vídeo 1

<https://youtu.be/OBpNjqIIyhI>

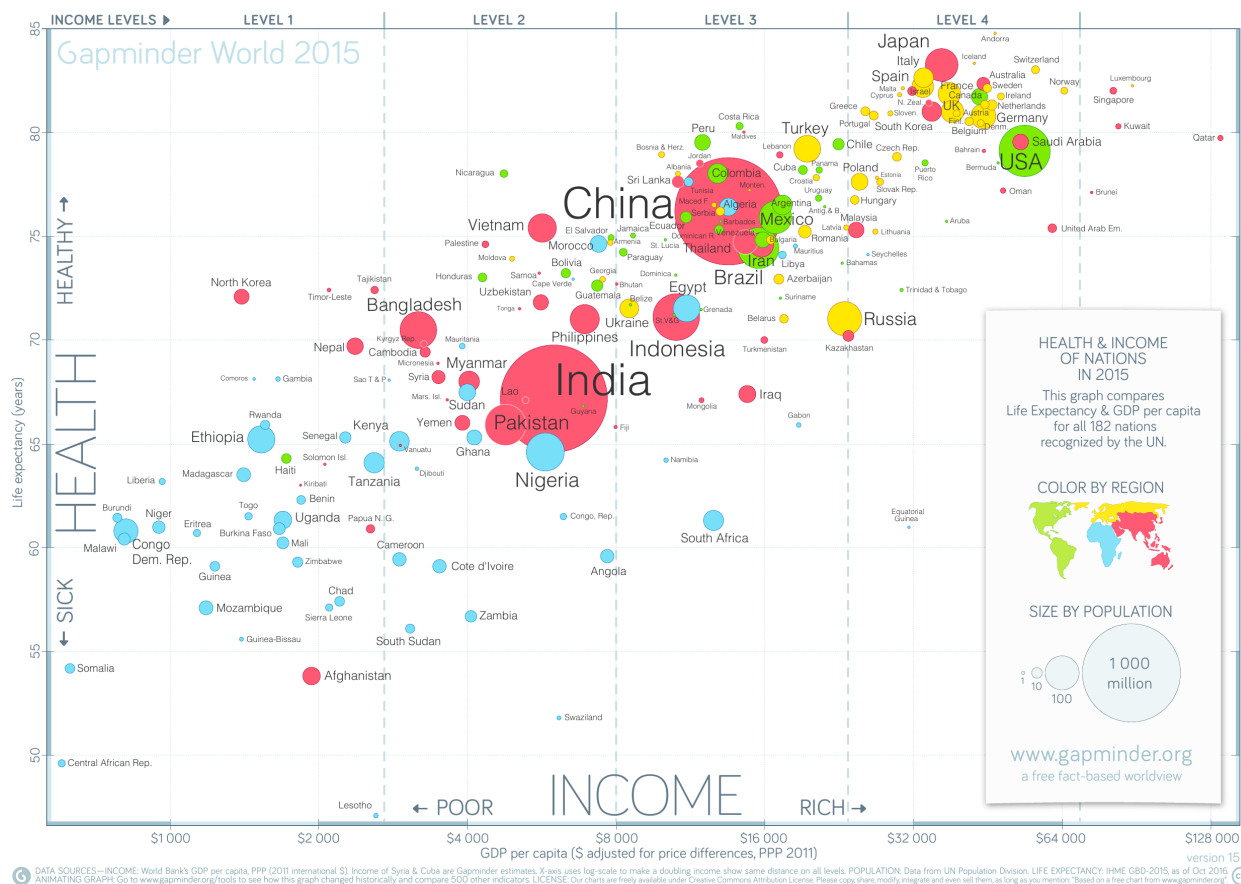
3.2

Componentes de um gráfico ggplot2

3.2.1

Geometrias e mapeamentos estéticos (*mappings*)

- Observe o gráfico abaixo, obtido de <https://www.gapminder.org/downloads/updated-gapminder-world-poster-2015/>.



- O gráfico mostra como, em cada país, a saúde (mais precisamente, a expectativa de vida) se relaciona com a riqueza (mais precisamente, o PIB *per capita*).
- Além da expectativa de vida e o do PIB *per capita*, o gráfico traz mais informações sobre cada país.
- Cada país é representado por um ponto (a **geometria**).
- Informações sobre cada país são representadas por características do ponto correspondente (as **estéticas**):

Variável	Geometria	Estética
PIB <i>per capita</i>	ponto	posição x
Expectativa de vida	ponto	posição y
População	ponto	tamanho
Continente	ponto	cor

- Você pode usar outras estéticas para representar informações:
 - Cor de preenchimento.
 - Cor do traço.
 - Tipo do traço (sólido, pontilhado, tracejado etc.).
 - Forma (círculo, quadrado, triângulo etc.).

- Opacidade.
- etc.
- Você pode usar outras geometrias:
 - Linhas.
 - Barras ou colunas.
 - Caixas.
 - etc.

3.2.2

Escalas (*scales*)

- As escalas controlam os detalhes da aparência da geometria e do mapeamento (eixos, cores etc.).
- Os eixos do gráfico acima são escalas **contínuas**, com valores reais.
- Observe o eixo horizontal. Os valores não aumentam linearmente, mas sim exponencialmente: cada passo à direita equivale a *dobrar* o valor do PIB. O eixo horizontal segue uma **escala logarítmica**.
- Os tamanhos dos pontos formam uma escala **discreta**, com 4 valores possíveis (veja a legenda no canto inferior direito do gráfico).
- As cores também formam uma escala discreta.

3.2.3

Rótulos (*labels*)

- O gráfico também representa informação na forma de texto.
- Além de rótulos (por exemplo, o texto que identifica cada eixo), **o texto também pode, ele mesmo, ser uma geometria, com suas próprias estéticas**: observe como o nome de cada país é escrito em um tamanho proporcional à sua população.

3.2.4

Outros componentes

- Coordenadas:
 - Este gráfico usa **coordenadas cartesianas**, com eixos x e y .
 - Existem gráficos que usam um sistema de **coordenadas polares**.
- Temas:
 - Incluem todos os elementos “decorativos”: cor de fundo, linhas de grade, etc. Ajudam a facilitar a leitura e a interpretação.
 - No gráfico acima, um detalhe interessante do tema é a divisão de cada eixo em segmentos claros e segmentos escuros.

- Legendas (*guides*).
- Facetas:
 - Às vezes, um gráfico é composto por múltiplos subgráficos.
 - Cada subgráfico é uma **faceta**.
 - Facetas evitam que informações demais sejam apresentadas no mesmo lugar.

3.3

Conjunto de dados

- Nossos exemplos de gráficos vão usar dados sobre o sono de diversos mamíferos.
- O conjunto de dados se chama `msleep` e está incluído no pacote `ggplot2`.
- Para ver a documentação, digite

```
library(ggplot2)
?msleep
```

- Vamos atribuir o conjunto de dados à variável `df`:

```
df <- msleep
df
## # A tibble: 83 x 11
##   name      genus vore  order conservation sleep_total sleep_rem sleep_cycle
##   <chr>    <chr> <chr> <chr> <chr>          <dbl>    <dbl>    <dbl>
## 1 Cheet~ Acin~ carni Carn~ lc              12.1      NA      NA
## 2 Owl m~ Aotus omni  Prim~ <NA>          17        1.8     NA
## 3 Mount~ Aplo~ herbi Rode~ nt          14.4      2.4     NA
## 4 Great~ Blar~ omni  Sori~ lc          14.9      2.3     0.133
## 5 Cow     Bos   herbi Arti~ domesticated    4        0.7     0.667
## 6 Three~ Brad~ herbi Pilo~ <NA>          14.4      2.2     0.767
## # ... with 77 more rows, and 3 more variables: awake <dbl>,
## #   brainwt <dbl>, bodywt <dbl>
```

- Vamos examinar a estrutura — usando R base:

```
str(df)
## tibble [83 x 11] (S3: tbl_df/tbl/data.frame)
##  $ name      : chr [1:83] "Cheetah" "Owl monkey" "Mountain beaver" ...
##  $ genus      : chr [1:83] "Acinonyx" "Aotus" "Aplodontia" ...
##  $ vore       : chr [1:83] "carni" "omni" "herbi" ...
##  $ order      : chr [1:83] "Carnivora" "Primates" "Rodentia" ...
##  $ conservation: chr [1:83] "lc" NA "nt" ...
##  $ sleep_total : num [1:83] 12,1 17 14,4 14,9 4 14,4 8,7 7 ...
##  $ sleep_rem   : num [1:83] NA 1,8 2,4 2,3 0,7 2,2 1,4 NA ...
##  $ sleep_cycle : num [1:83] NA NA NA 0,133 ...
##  $ awake      : num [1:83] 11,9 7 9,6 9,1 20 9,6 15,3 17 ...
```

```
## $ brainwt      : num [1:83] NA 0,0155 NA 0,00029 0,423 NA NA NA ...
## $ bodywt       : num [1:83] 50 0,48 1,35 0,019 ...
```

- Podemos usar `glimpse`, uma função do `tidyverse`:

```
glimpse(df)
## Rows: 83
## Columns: 11
## $ name      <chr> "Cheetah", "Owl monkey", "Mountain beaver", "Great~
## $ genus     <chr> "Acinonyx", "Aotus", "Aplodontia", "Blarina", "Bos~
## $ vore      <chr> "carni", "omni", "herbi", "omni", "herbi", "herbi"~
## $ order     <chr> "Carnivora", "Primates", "Rodentia", "Soricomorpha~
## $ conservation <chr> "lc", NA, "nt", "lc", "domesticated", NA, "vu", NA~
## $ sleep_total <dbl> 12,1, 17,0, 14,4, 14,9, 4,0, 14,4, 8,7, 7,0, 10,1,~
## $ sleep_rem  <dbl> NA, 1,8, 2,4, 2,3, 0,7, 2,2, 1,4, NA, 2,9, NA, 0,6~
## $ sleep_cycle <dbl> NA, NA, NA, 0,1333333, 0,6666667, 0,7666667, 0,383~
## $ awake     <dbl> 11,9, 7,0, 9,6, 9,1, 20,0, 9,6, 15,3, 17,0, 13,9, ~
## $ brainwt   <dbl> NA, 0,01550, NA, 0,00029, 0,42300, NA, NA, NA, 0,0~
## $ bodywt    <dbl> 50,000, 0,480, 1,350, 0,019, 600,000, 3,850, 20,49~
```

- Para examinar só as primeiras linhas do *data frame*:

```
head(df)
## # A tibble: 6 x 11
##   name      genus vore  order conservation sleep_total sleep_rem sleep_cycle
##   <chr>    <chr> <chr> <chr> <chr>          <dbl>      <dbl>      <dbl>
## 1 Cheet~ Acin~ carni Carn~ lc              12.1        NA         NA
## 2 Owl m~ Aotus omni  Prim~ <NA>           17          1.8        NA
## 3 Mount~ Aplo~ herbi Rode~ nt             14.4        2.4        NA
## 4 Great~ Blar~ omni  Sori~ lc             14.9        2.3        0.133
## 5 Cow    Bos   herbi Arti~ domesticated      4          0.7        0.667
## 6 Three~ Brad~ herbi Pilo~ <NA>           14.4        2.2        0.767
## # ... with 3 more variables: awake <dbl>, brainwt <dbl>, bodywt <dbl>
```

- Para examinar o *data frame* interativamente:

```
view(df)
```

- Podemos produzir um sumário dos dados usando o pacote *summarytools* (que já foi carregado neste documento):

```
df %>% dfSummary() %>% print()
```

Variável	Estatísticas / Valores	Freqs (% de Válidos)	Faltante
name [character]	1. African elephant 2. African giant pouched rat 3. African striped mouse 4. Arctic fox 5. Arctic ground squirrel 6. Asian elephant 7. Baboon 8. Big brown bat 9. Bottle-nosed dolphin 10. Brazilian tapir [73 outros]	1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 73 (88,0%)	0 (0,0%)
genus [character]	1. Panthera 2. Sperophilus 3. Equus 4. Vulpes 5. Acinonyx 6. Aotus 7. Aplodontia 8. Blarina 9. Bos 10. Bradypus [67 outros]	3 (3,6%) 3 (3,6%) 2 (2,4%) 2 (2,4%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 67 (80,7%)	0 (0,0%)
vore [character]	1. carni 2. herbi 3. insecti 4. omni	19 (25,0%) 32 (42,1%) 5 (6,6%) 20 (26,3%)	7 (8,4%)
order [character]	1. Rodentia 2. Carnivora 3. Primates 4. Artiodactyla 5. Soricomorpha 6. Cetacea 7. Hyracoidea 8. Perissodactyla 9. Chiroptera 10. Cingulata [9 outros]	22 (26,5%) 12 (14,5%) 12 (14,5%) 6 (7,2%) 5 (6,0%) 3 (3,6%) 3 (3,6%) 3 (3,6%) 2 (2,4%) 2 (2,4%) 13 (15,7%)	0 (0,0%)
conservation [character]	1. cd 2. domesticated 3. en 4. lc 5. nt 6. vu	2 (3,7%) 10 (18,5%) 4 (7,4%) 27 (50,0%) 4 (7,4%) 7 (13,0%)	29 (34,9%)

Variável	Estatísticas / Valores	Freqs (% de Válidos)	Faltante
sleep_total [numeric]	Média (dp) : 10,4 (4,5) mín < mediana < máx: 1,9 < 10,1 < 19,9 IQE (CV) : 5,9 (0,4)	65 valores distintos	0 (0,0%)
sleep_rem [numeric]	Média (dp) : 1,9 (1,3) mín < mediana < máx: 0,1 < 1,5 < 6,6 IQE (CV) : 1,5 (0,7)	32 valores distintos	22 (26,5%)
sleep_cycle [numeric]	Média (dp) : 0,4 (0,4) mín < mediana < máx: 0,1 < 0,3 < 1,5 IQE (CV) : 0,4 (0,8)	22 valores distintos	51 (61,4%)
awake [numeric]	Média (dp) : 13,6 (4,5) mín < mediana < máx: 4,1 < 13,9 < 22,1 IQE (CV) : 5,9 (0,3)	65 valores distintos	0 (0,0%)
brainwt [numeric]	Média (dp) : 0,3 (1) mín < mediana < máx: 0 < 0 < 5,7 IQE (CV) : 0,1 (3,5)	53 valores distintos	27 (32,5%)
bodywt [numeric]	Média (dp) : 166,1 (786,8) mín < mediana < máx: 0 < 1,7 < 6654 IQE (CV) : 41,6 (4,7)	82 valores distintos	0 (0,0%)

- Vemos que há muitos NA em diversas variáveis. Para nossos exemplos simples de visualização, vamos usar as colunas

- name
- genus
- order
- sleep_total
- awake
- bodywt
- brainwt

- Mas... a coluna que mostra a dieta (vore) tem só 7 NA. Quais são?

```
df %>%
  filter(is.na(vore)) %>%
  select(name)
## # A tibble: 7 x 1
##   name
##   <chr>
## 1 Vesper mouse
## 2 Desert hedgehog
## 3 Deer mouse
```

```
## 4 Phalanger
## 5 Rock hyrax
## 6 Mole rat
## # ... with 1 more row
```

- OK. Vamos manter a coluna `vore` também, apesar dos `NA`. Quando formos usar esta variável, tomaremos cuidado.
- Também... a coluna `bodywt` tem 0 como valor mínimo. Como assim?

```
df %>%
  filter(bodywt < 1) %>%
  select(name, bodywt) %>%
  arrange(bodywt)
## # A tibble: 35 x 2
##   name                bodywt
##   <chr>              <dbl>
## 1 Lesser short-tailed shrew 0.005
## 2 Little brown bat        0.01
## 3 Greater short-tailed shrew 0.019
## 4 Deer mouse             0.021
## 5 House mouse            0.022
## 6 Big brown bat          0.023
## # ... with 29 more rows
```

- Ah, sem problema. A função `dfSummary` arredondou estes pesos para 0. Os valores de verdade ainda estão na *tibble*.
- Vamos criar uma *tibble* nova, só com as colunas que nos interessam:

```
sono <- df %>%
  select(
    name, order, genus, vore, bodywt,
    brainwt, awake, sleep_total
  )
```

- Vamos ver o sumário:

```
sono %>% dfSummary() %>% print()
```

Variável	Estatísticas / Valores	Freqs (% de Válidos)	Faltante
name [character]	1. African elephant 2. African giant pouched rat 3. African striped mouse 4. Arctic fox 5. Arctic ground squirrel 6. Asian elephant 7. Baboon 8. Big brown bat 9. Bottle-nosed dolphin 10. Brazilian tapir [73 outros]	1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 73 (88,0%)	0 (0,0%)
order [character]	1. Rodentia 2. Carnivora 3. Primates 4. Artiodactyla 5. Soricomorpha 6. Cetacea 7. Hyracoidea 8. Perissodactyla 9. Chiroptera 10. Cingulata [9 outros]	22 (26,5%) 12 (14,5%) 12 (14,5%) 6 (7,2%) 5 (6,0%) 3 (3,6%) 3 (3,6%) 3 (3,6%) 2 (2,4%) 2 (2,4%) 13 (15,7%)	0 (0,0%)
genus [character]	1. Panthera 2. Sperophilus 3. Equus 4. Vulpes 5. Acinonyx 6. Aotus 7. Aplodontia 8. Blarina 9. Bos 10. Bradypus [67 outros]	3 (3,6%) 3 (3,6%) 2 (2,4%) 2 (2,4%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 1 (1,2%) 67 (80,7%)	0 (0,0%)
vore [character]	1. carni 2. herbi 3. insecti 4. omni	19 (25,0%) 32 (42,1%) 5 (6,6%) 20 (26,3%)	7 (8,4%)
bodywt [numeric]	Média (dp) : 166,1 (786,8) mín < mediana < máx: 0 < 1,7 < 6654 IQE (CV) : 41,6 (4,7)	82 valores distintos	0 (0,0%)
brainwt [numeric]	Média (dp) : 0,3 (1) mín < mediana < máx: 0 < 0 < 5,7 IQE (CV) : 0,1 (3,5)	53 valores distintos	27 (32,5%)

Variável	Estatísticas / Valores	Freqs (% de Válidos)	Faltante
awake [numeric]	Média (dp) : 13,6 (4,5) mín < mediana < máx: 4,1 < 13,9 < 22,1 IQE (CV) : 5,9 (0,3)	65 valores distintos	0 (0,0%)
sleep_total [numeric]	Média (dp) : 10,4 (4,5) mín < mediana < máx: 1,9 < 10,1 < 19,9 IQE (CV) : 5,9 (0,4)	65 valores distintos	0 (0,0%)

3.4

Gráficos de dispersão (*scatter plots*)

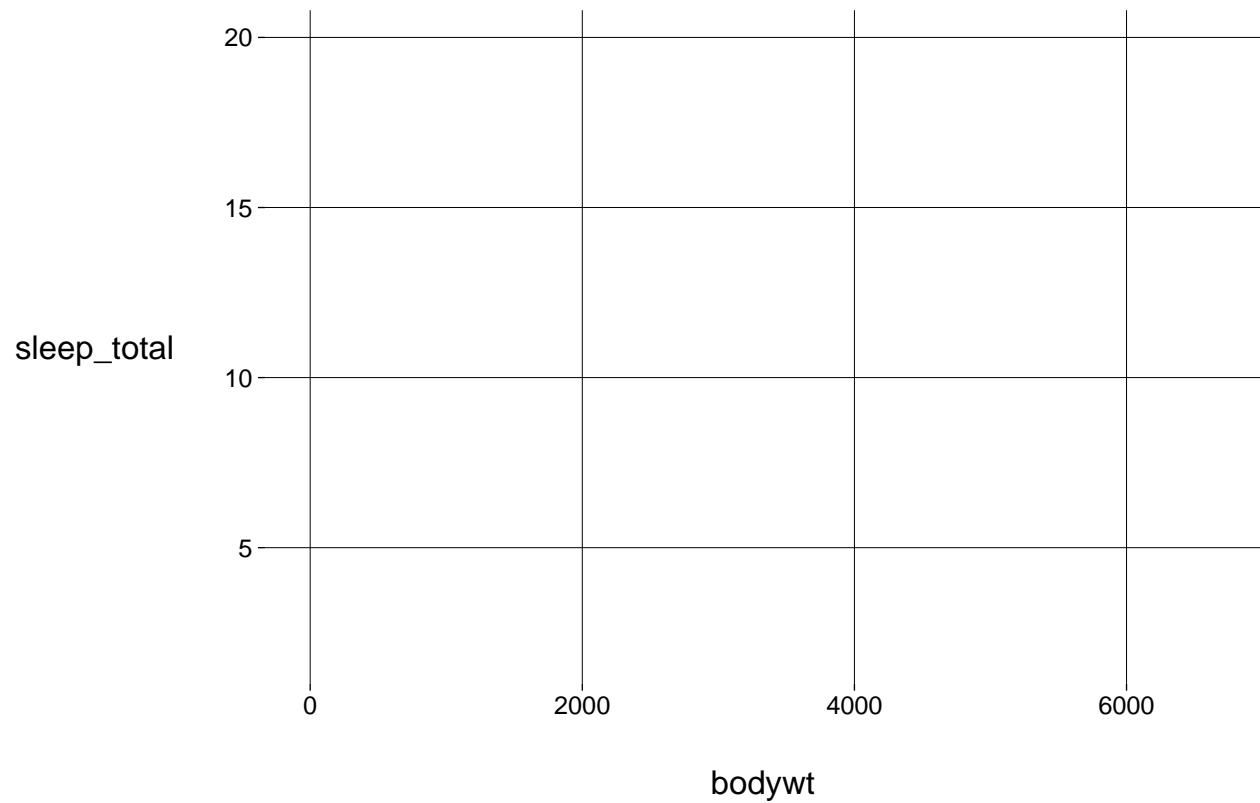
- Servem para visualizar a *relação* entre **duas variáveis quantitativas**.
- Essa relação *não* é necessariamente de causalidade.
- Isto é, a variável representada no eixo horizontal não determina, necessariamente, os valores da variável representada no eixo vertical.
- Pense em associação, correlação, não em causalidade.
- Troque as variáveis de eixo, se ajudar a deixar isto claro.

3.4.1

Horas de sono e peso corporal

Como as variáveis `sleep_total` e `bodywt` estão relacionadas?

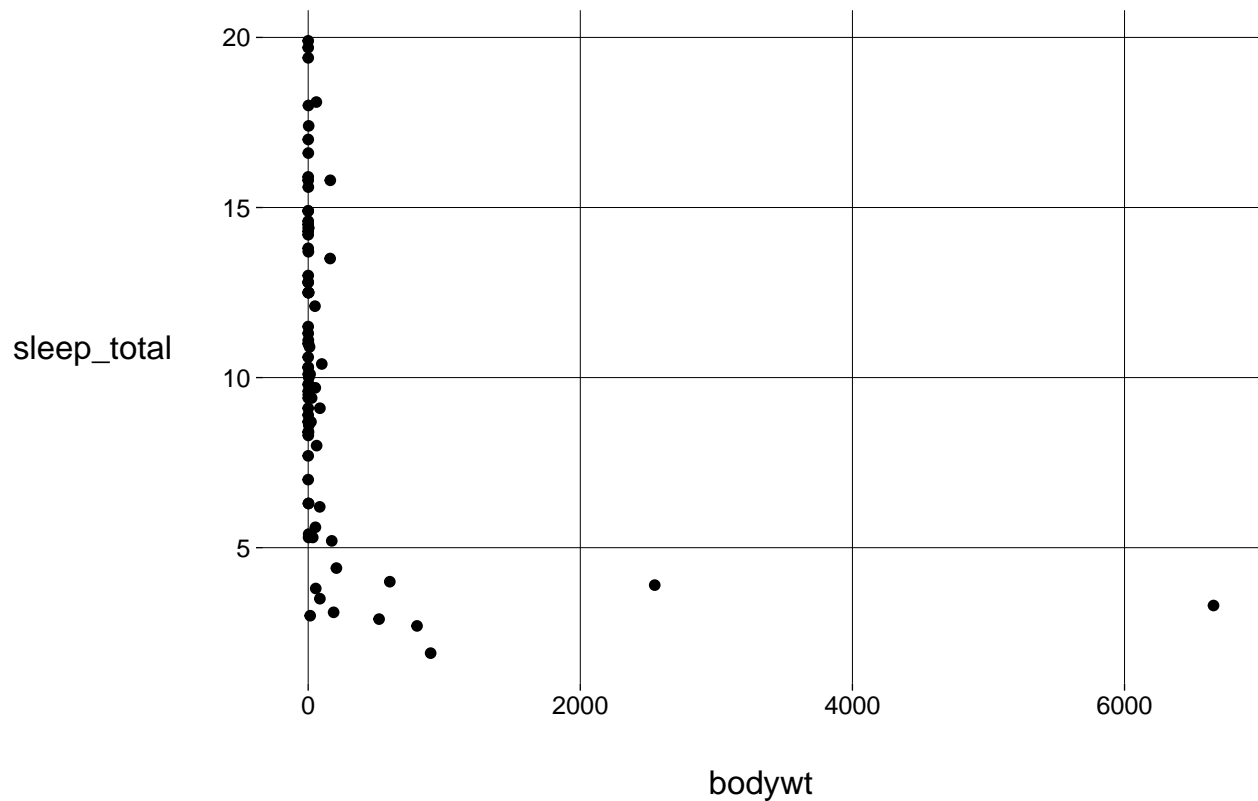
```
sono %>%
  ggplot(aes(x = bodywt, y = sleep_total))
```



O que houve? Cadê os pontos?

O problema foi que só especificamos o mapeamento estético. Faltou a geometria.

```
sono %>%  
  ggplot(aes(x = bodywt, y = sleep_total)) +  
  geom_point()
```



Que horror.

A única coisa que percebemos aqui é que os mamíferos muito pesados dormem menos de 5 horas por noite.

Estes animais muito pesados estão estragando a escala do eixo x .

Que animais são estes?

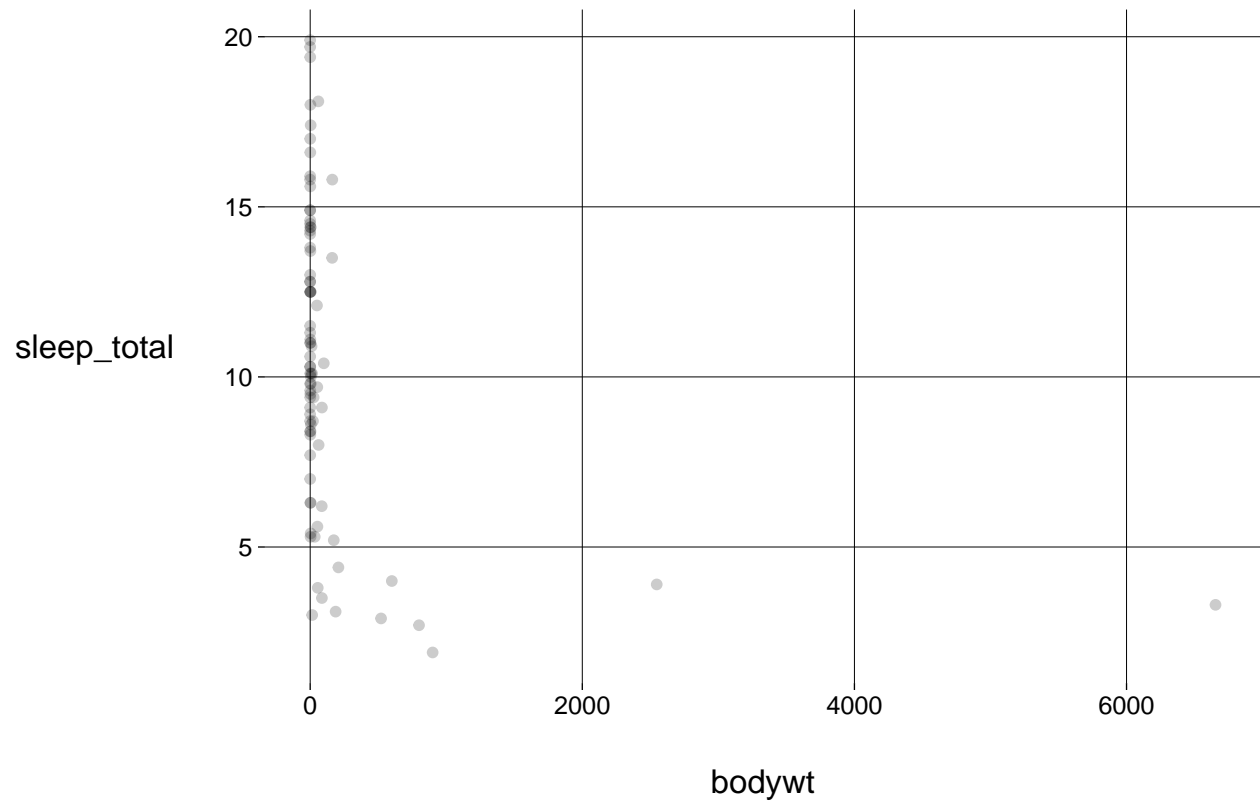
```
sono %>%
  filter(bodywt > 250) %>%
  select(name, bodywt) %>%
  arrange(bodywt)
## # A tibble: 6 x 2
##   name          bodywt
##   <chr>         <dbl>
## 1 Horse          521
## 2 Cow            600
## 3 Pilot whale    800
## 4 Giraffe        900.
## 5 Asian elephant 2547
## 6 African elephant 6654
```

Além disso, há muitos pontos sobrepostos. Em bom português, temos um problema de *overplotting*.

Existem diversas maneiras de lidar com isso.

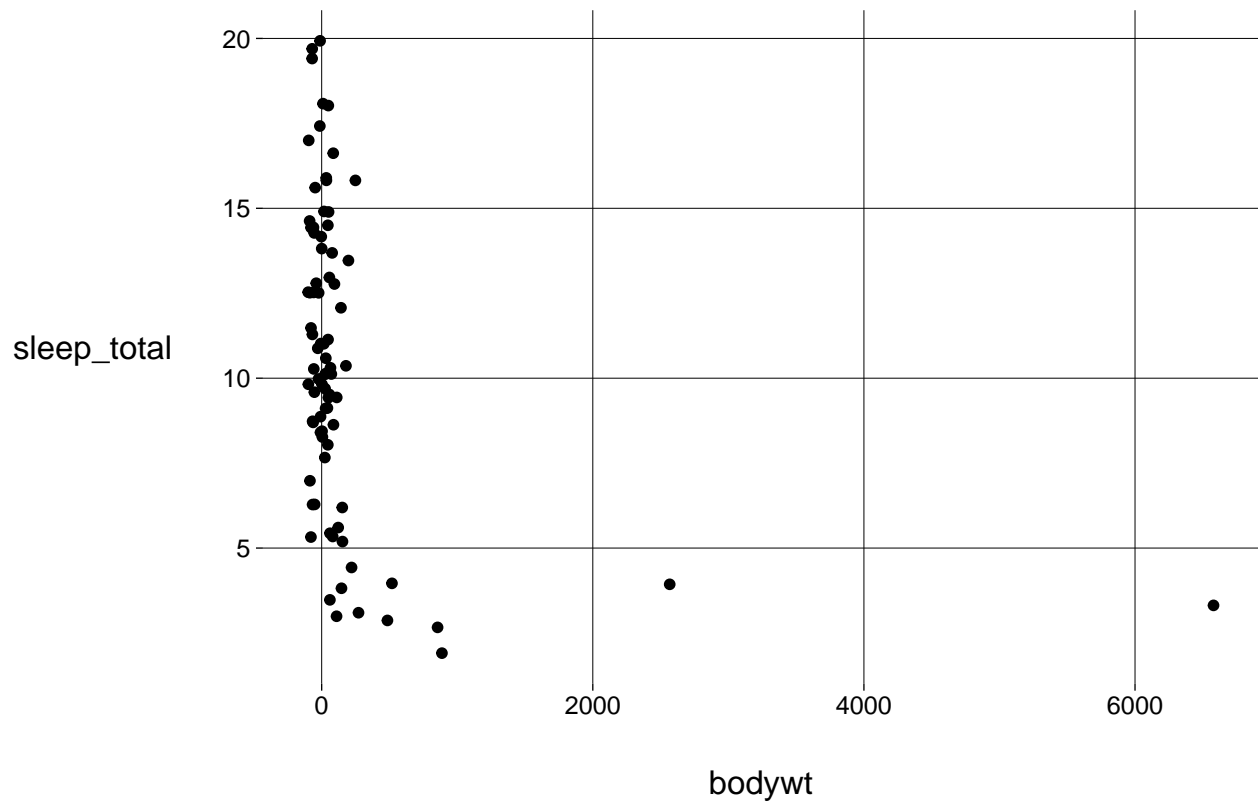
A primeira delas é alterando a opacidade dos pontos:

```
sono %>%  
  ggplot(aes(x = bodywt, y = sleep_total)) +  
    geom_point(alpha = 0.2)
```



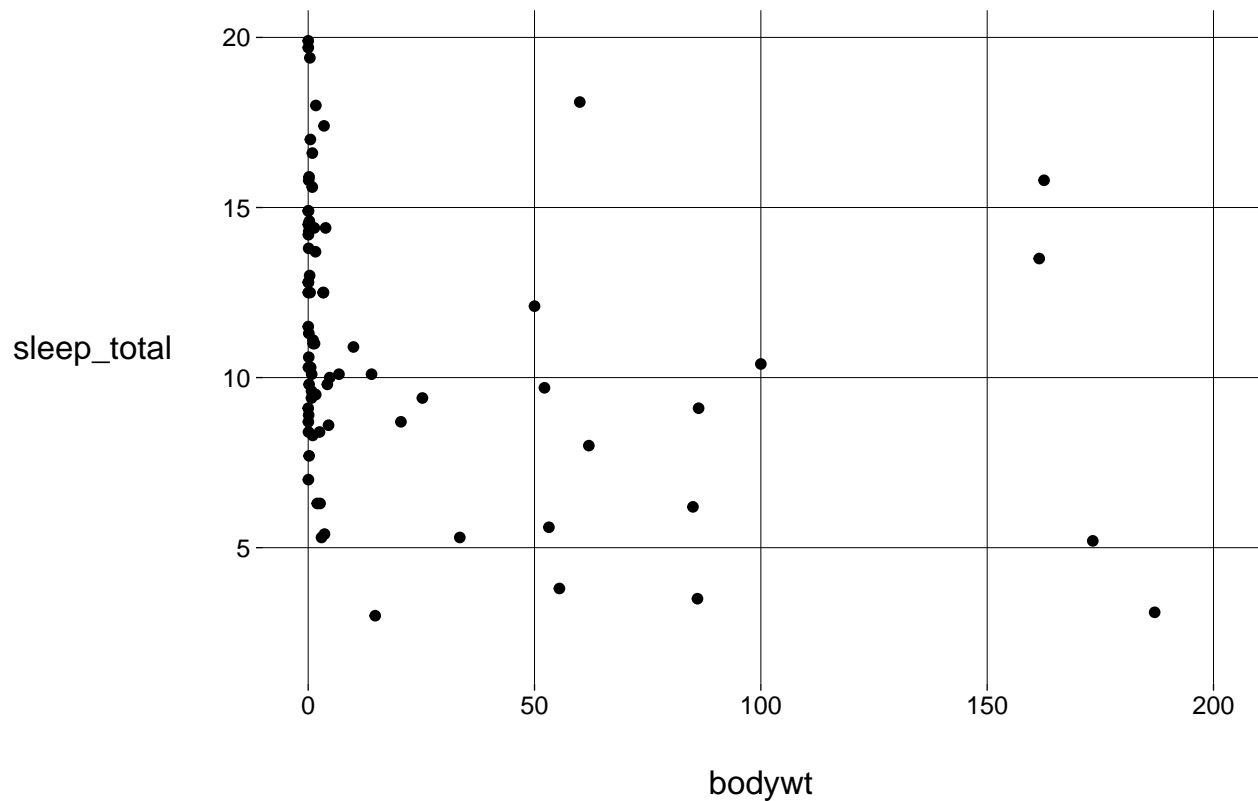
Outra maneira é usar `geom_jitter` em vez de `geom_point`:

```
sono %>%  
  ggplot(aes(x = bodywt, y = sleep_total)) +  
    geom_jitter(width = 100)
```



Vamos mudar a escala do gráfico para nos concentrarmos nos animais menos pesados:

```
sono %>%  
  ggplot(aes(x = bodywt, y = sleep_total)) +  
    geom_point() +  
    scale_x_continuous(limits = c(0, 200))  
## Warning: Removed 7 rows containing missing values (geom_point).
```

Nesta escala, a relação entre horas de sono e peso não é mais tão regular.

3.4.2

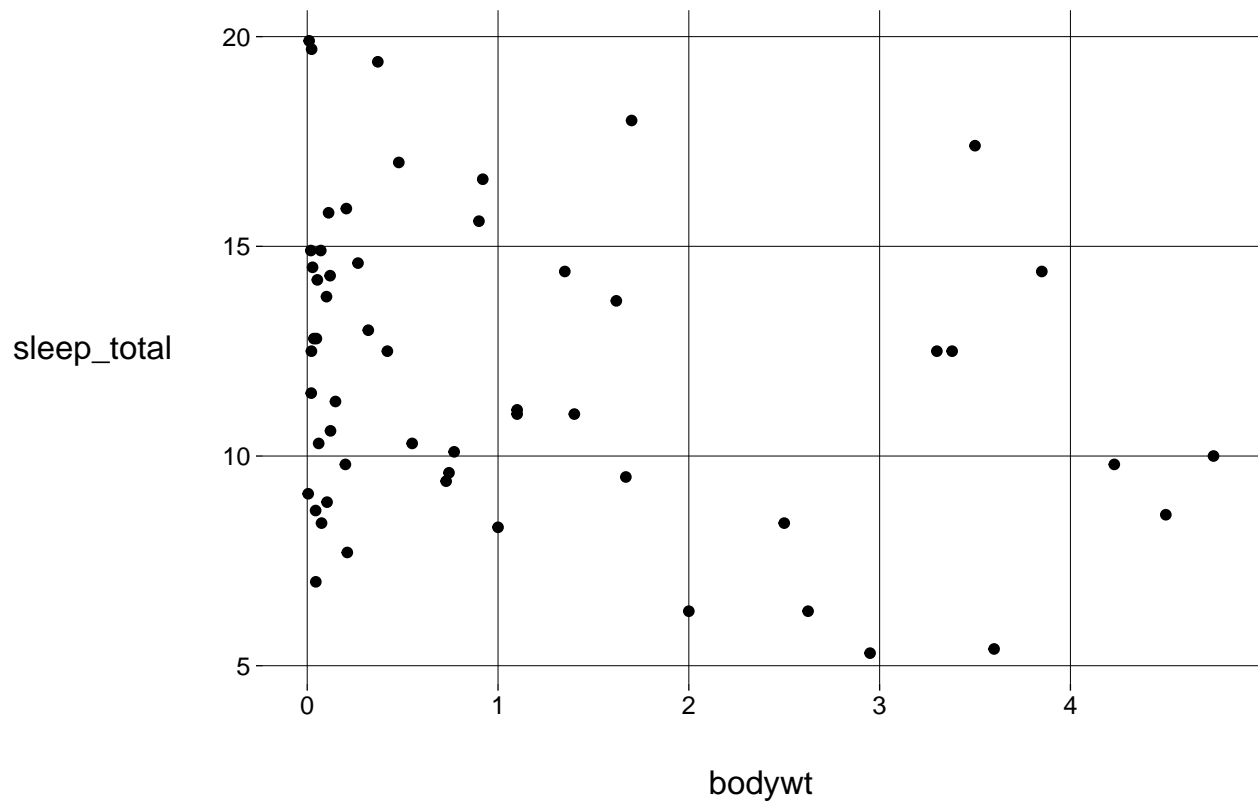
Horas de sono e peso corporal para animais pequenos

Vamos restringir o gráfico a animais com no máximo 5kg.

```
limite <- 5
```

Em vez de mudar a escala do gráfico, vamos filtrar as linhas do *data frame*:

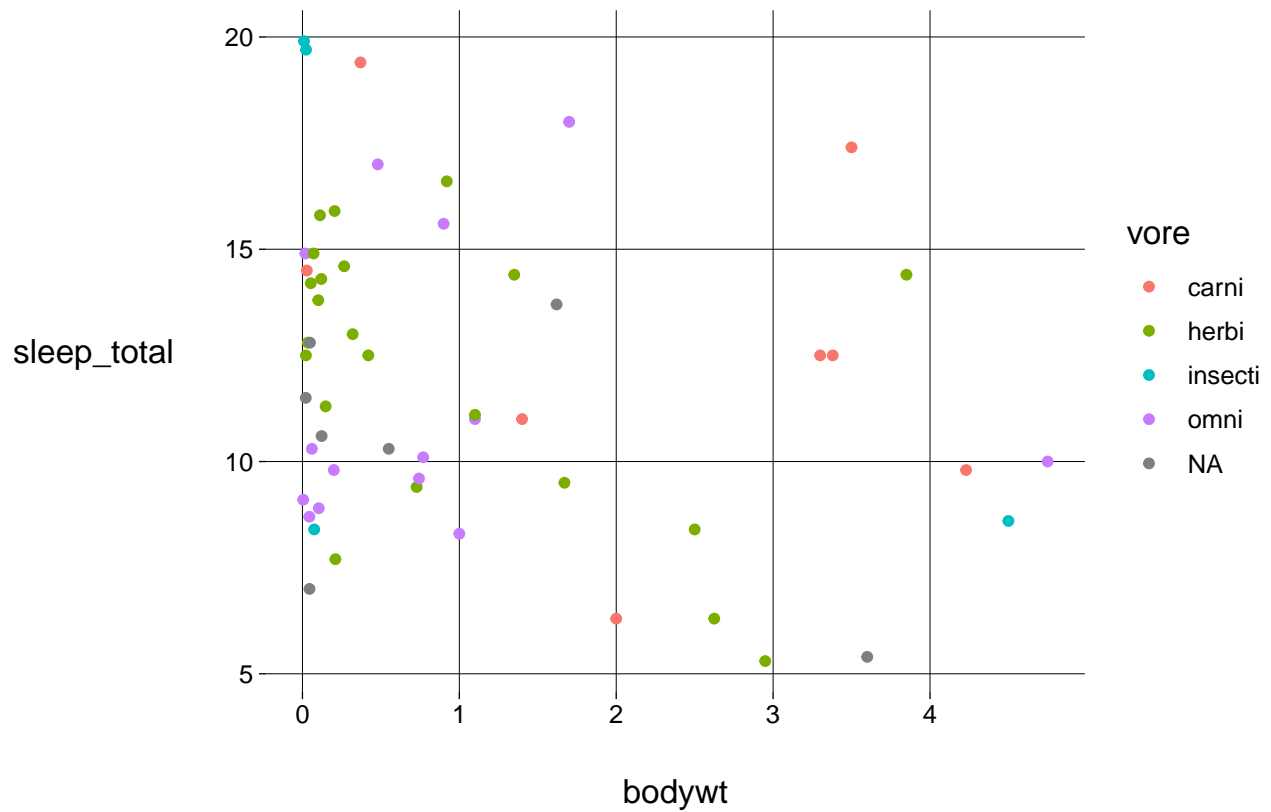
```
sono %>%
  filter(bodywt < limite) %>%
  ggplot(aes(x = bodywt, y = sleep_total)) +
  geom_point()
```



3.4.3

Includendo a dieta

```
sono %>%  
  filter(bodywt < limite) %>%  
  ggplot(aes(x = bodywt, y = sleep_total, color = vore)) +  
    geom_point()
```

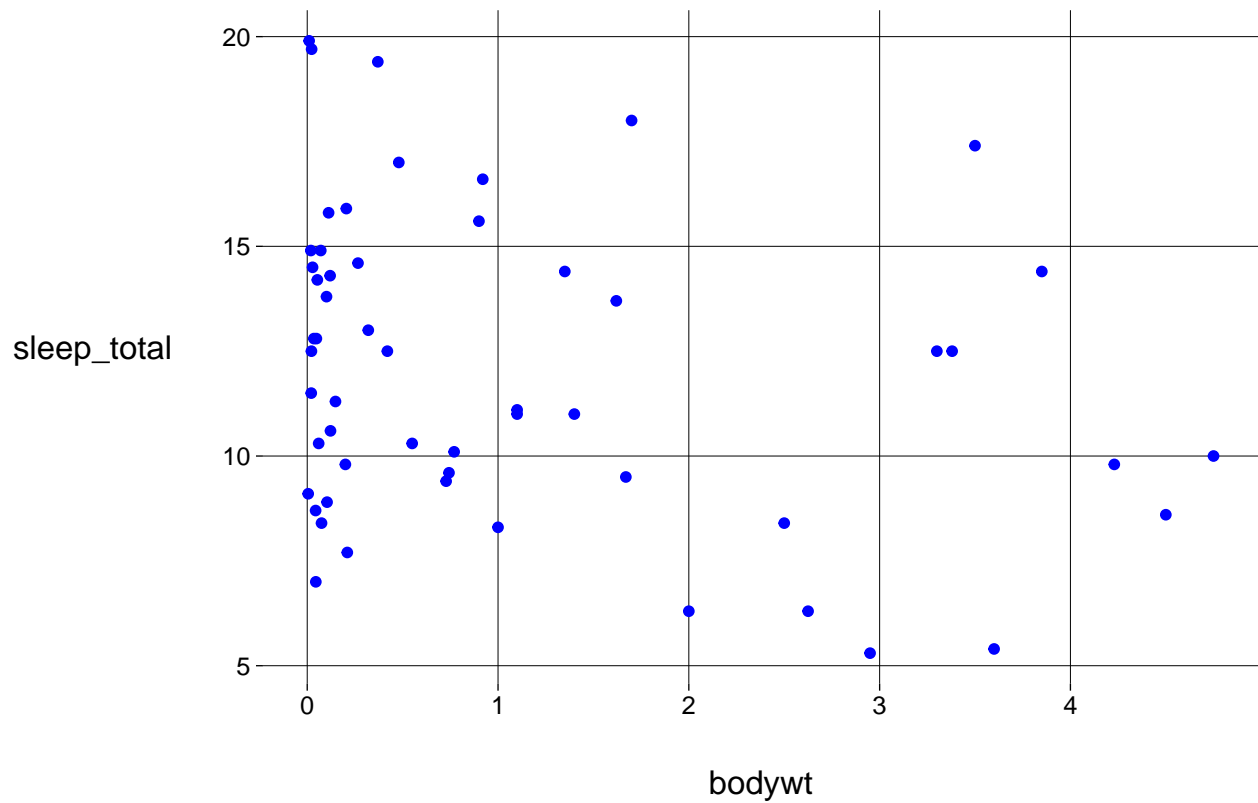



3.4.5

Estética fixa ou dependendo de variável?

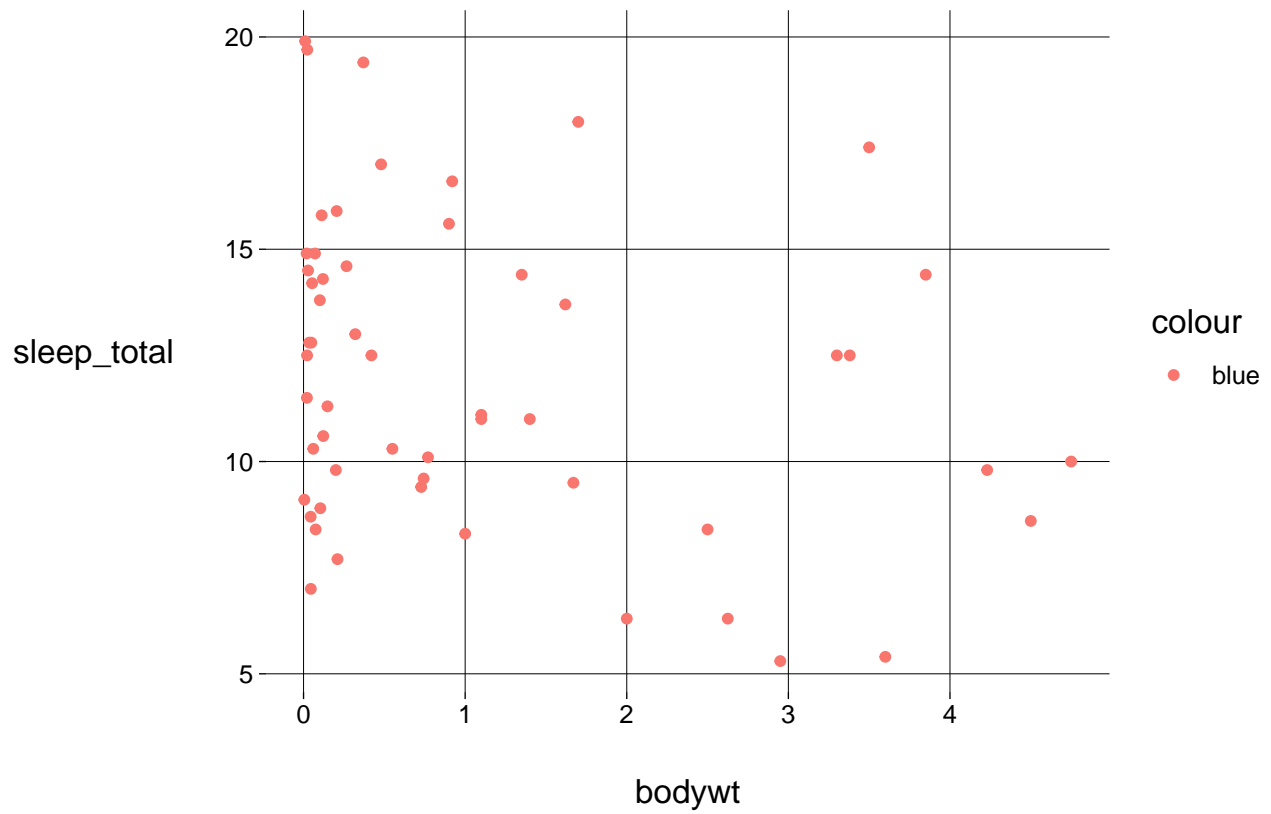
Compare o último *chunk* acima com:

```
sono %>%
  filter(bodywt < limite) %>%
  ggplot() +
    geom_point(aes(x = bodywt, y = sleep_total), color = 'blue')
```



Um erro comum:

```
sono %>%  
  filter(bodywt < limite) %>%  
  ggplot() +  
    geom_point(aes(x = bodywt, y = sleep_total, color = 'blue'))
```

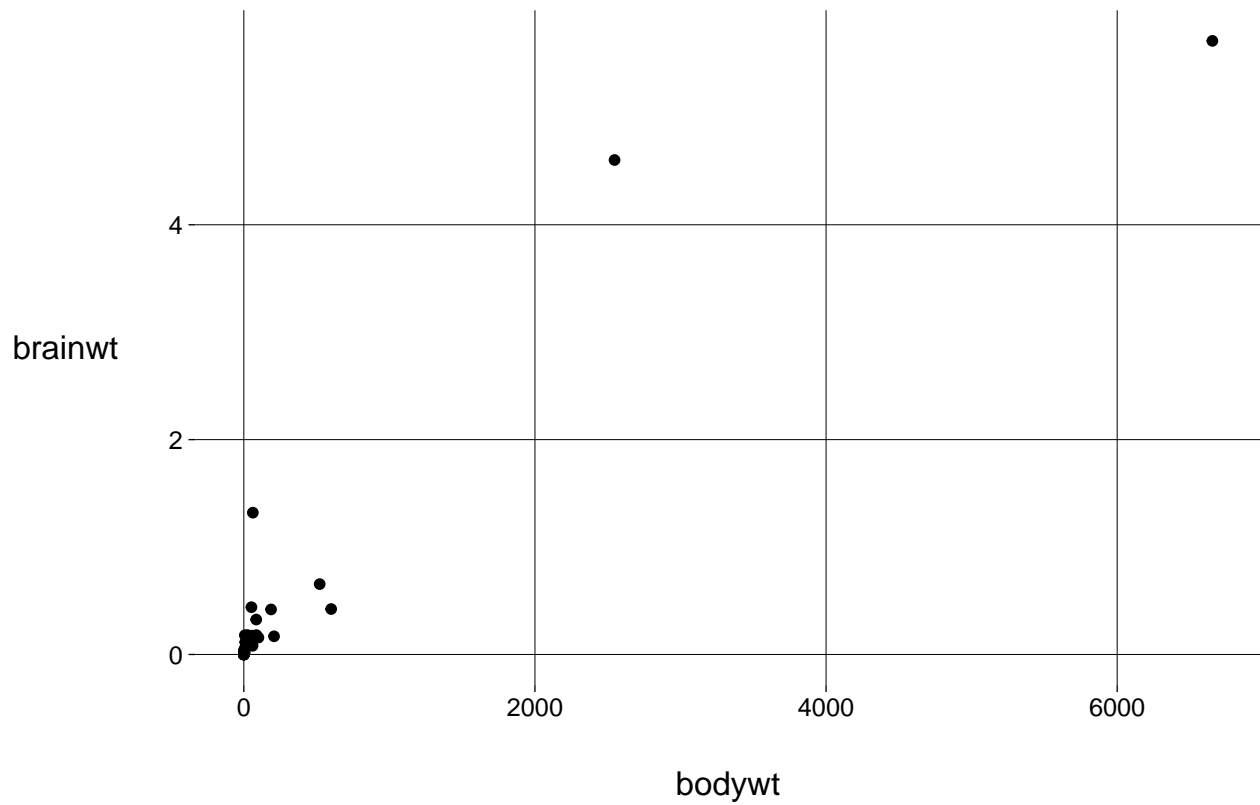


3.4.6

Uma correlação mais clara

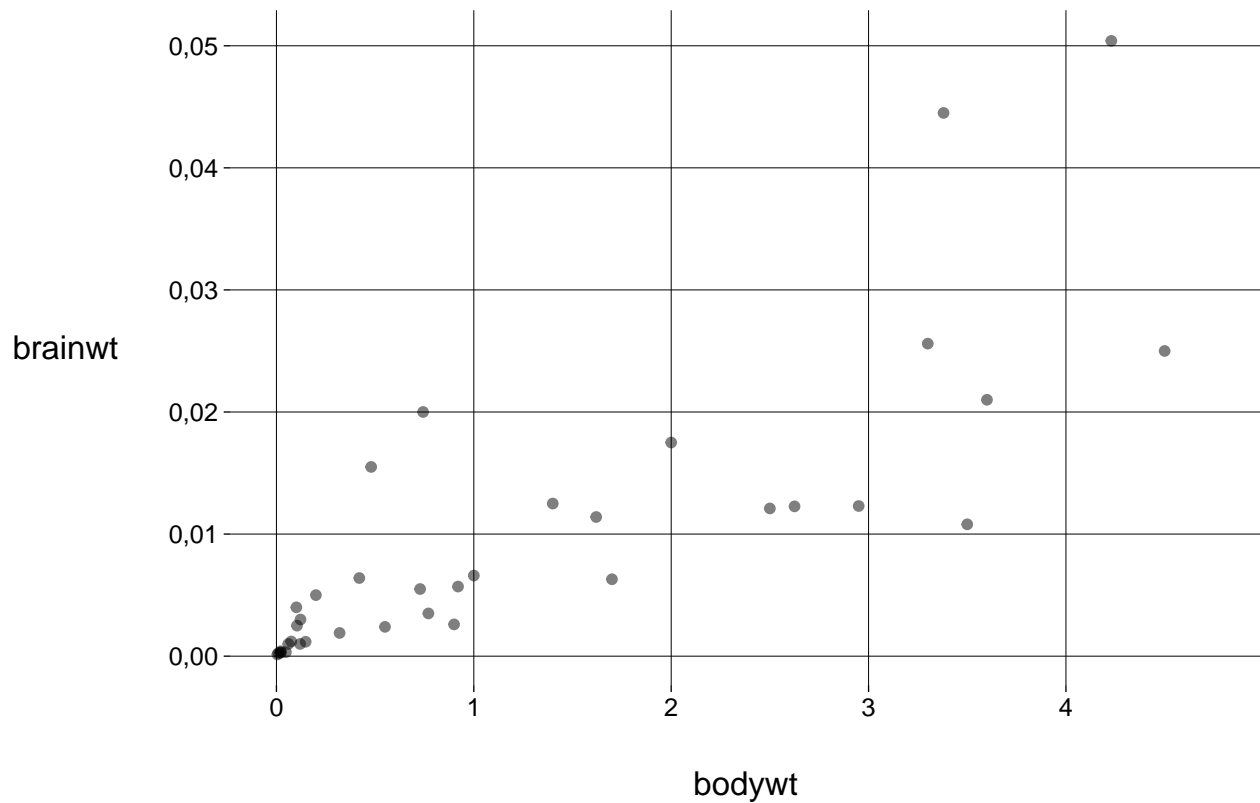
Peso cerebral versus peso corporal:

```
sono %>%  
  ggplot() +  
    geom_point(aes(x = bodywt, y = brainwt))  
## Warning: Removed 27 rows containing missing values (geom_point).
```



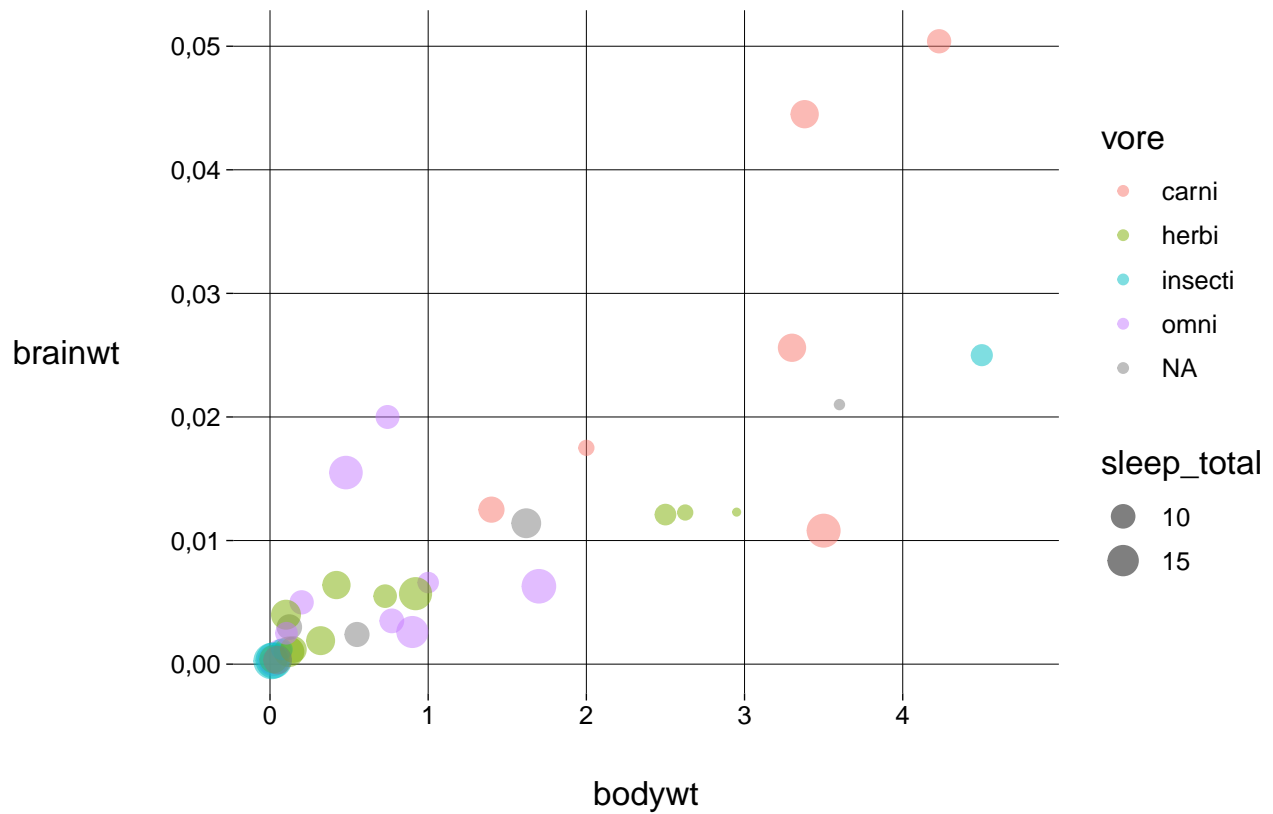
Vamos restringir aos animais mais leves e mudar a opacidade:

```
sono %>%  
  filter(bodywt < limite) %>%  
  ggplot() +  
    geom_point(aes(x = bodywt, y = brainwt), alpha = .5)  
## Warning: Removed 18 rows containing missing values (geom_point).
```



Vamos incluir horas de sono e dieta:

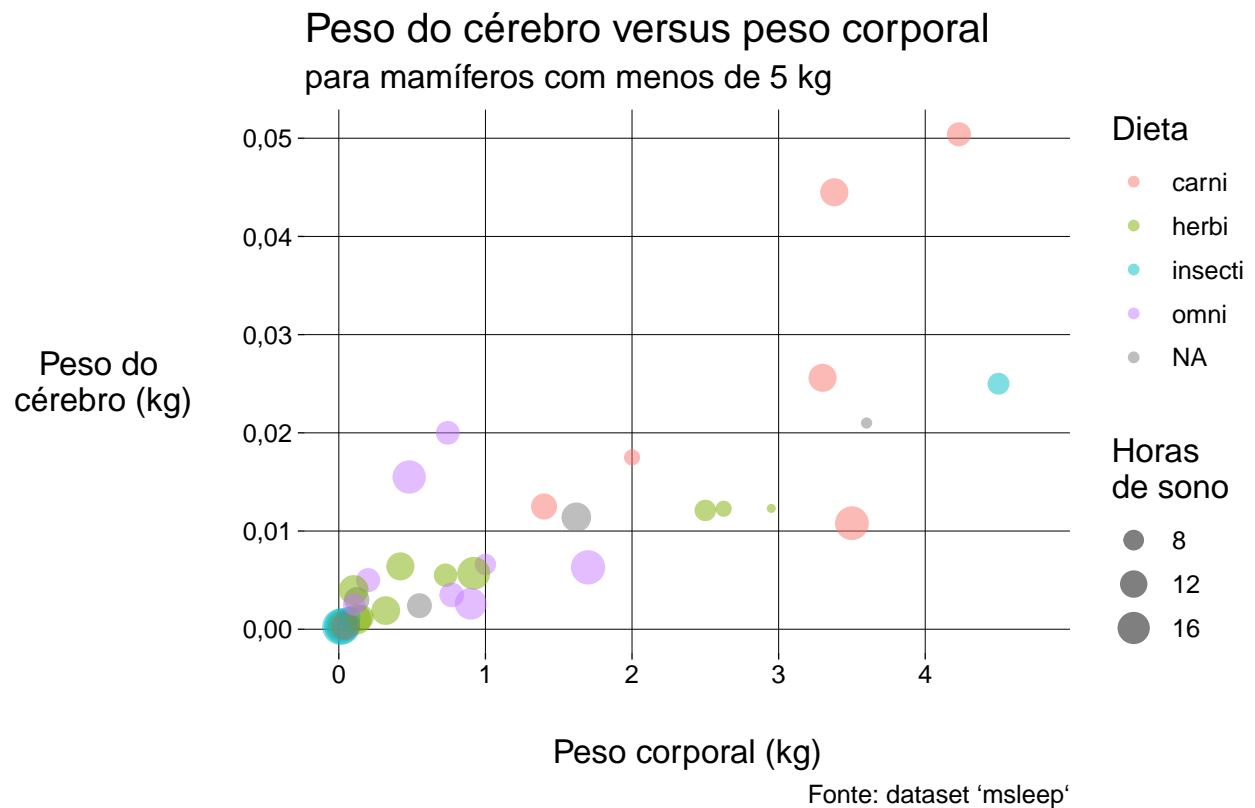
```
sono %>%
  filter(bodywt < limite) %>%
  ggplot() +
    geom_point(
      aes(
        x = bodywt,
        y = brainwt,
        size = sleep_total,
        color = vore
      ),
      alpha = .5
    )
## Warning: Removed 18 rows containing missing values (geom_point).
```

Mudar a escala dos tamanhos e incluir rótulos:

```
sono %>%
  filter(bodywt < limite) %>%
  ggplot() +
    geom_point(
      aes(
        x = bodywt,
        y = brainwt,
        size = sleep_total,
        color = vore
      ),
      alpha = .5
    ) +
    scale_size(
      breaks = seq(0, 24, 4)
    ) +
    labs(
      title = 'Peso do cérebro versus peso corporal',
      subtitle = paste0(
        'para mamíferos com menos de ',
        limite,
        ' kg'
      ),
      caption = 'Fonte: dataset `msleep`',
    )
```

```
x = 'Peso corporal (kg)',
y = 'Peso do\n cérebro (kg)',
color = 'Dieta',
size = 'Horas\nde sono'
)
## Warning: Removed 18 rows containing missing values (geom_point).
```



3.5

Histogramas e cia.

A idéia agora é agrupar indivíduos em classes, dependendo do valor de uma variável numérica.

Vamos nos concentrar nas horas de sono.

3.5.1

Distribuições de frequência

Em R base, é fácil fazer os agrupamentos:

```
sono$sleep_total
## [1] 12,1 17,0 14,4 14,9 4,0 14,4 8,7 7,0 10,1 3,0 5,3 9,4 10,0
## [14] 12,5 10,3 8,3 9,1 17,4 5,3 18,0 3,9 19,7 2,9 3,1 10,1 10,9
## [27] 14,9 12,5 9,8 1,9 2,7 6,2 6,3 8,0 9,5 3,3 19,4 10,1 14,2
## [40] 14,3 12,8 12,5 19,9 14,6 11,0 7,7 14,5 8,4 3,8 9,7 15,8 10,4
## [53] 13,5 9,4 10,3 11,0 11,5 13,7 3,5 5,6 11,1 18,1 5,4 13,0 8,7
## [66] 9,6 8,4 11,3 10,6 16,6 13,8 15,9 12,8 9,1 8,6 15,8 4,4 15,6
## [79] 8,9 5,2 6,3 12,5 9,8
```

```
sono$sleep_total %>% range()
## [1] 1,9 19,9
```

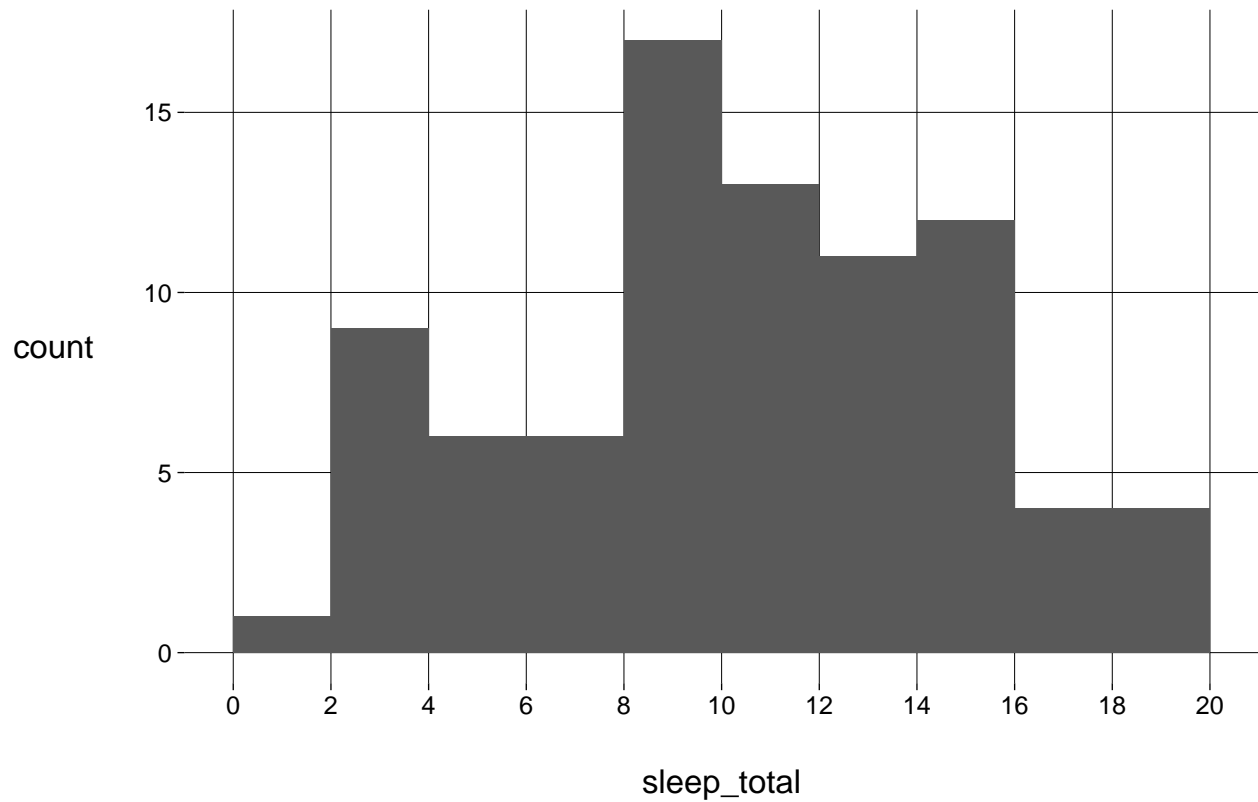
```
sono$sleep_total %>%
  cut(breaks = seq(0, 20, 2), right = FALSE)
## [1] [12,14) [16,18) [14,16) [14,16) [4,6) [14,16) [8,10) [6,8)
## [9] [10,12) [2,4) [4,6) [8,10) [10,12) [12,14) [10,12) [8,10)
## [17] [8,10) [16,18) [4,6) [18,20) [2,4) [18,20) [2,4) [2,4)
## [25] [10,12) [10,12) [14,16) [12,14) [8,10) [0,2) [2,4) [6,8)
## [33] [6,8) [8,10) [8,10) [2,4) [18,20) [10,12) [14,16) [14,16)
## [41] [12,14) [12,14) [18,20) [14,16) [10,12) [6,8) [14,16) [8,10)
## [49] [2,4) [8,10) [14,16) [10,12) [12,14) [8,10) [10,12) [10,12)
## [57] [10,12) [12,14) [2,4) [4,6) [10,12) [18,20) [4,6) [12,14)
## [65] [8,10) [8,10) [8,10) [10,12) [10,12) [16,18) [12,14) [14,16)
## [73] [12,14) [8,10) [8,10) [14,16) [4,6) [14,16) [8,10) [4,6)
## [81] [6,8) [12,14) [8,10)
## 10 Levels: [0,2) [2,4) [4,6) [6,8) [8,10) [10,12) [12,14) ... [18,20)
```

```
sono$sleep_total %>%
  cut(breaks = seq(0, 20, 2), right = FALSE) %>%
  table(dnn = 'Horas de sono') %>%
  as.data.frame()
## # A tibble: 10 x 2
##   Horas.de.sono Freq
##   <fct>         <int>
## 1 [0,2)         1
## 2 [2,4)         8
## 3 [4,6)         7
## 4 [6,8)         5
## 5 [8,10)        17
## 6 [10,12)       14
## # ... with 4 more rows
```

3.5.2

Histograma

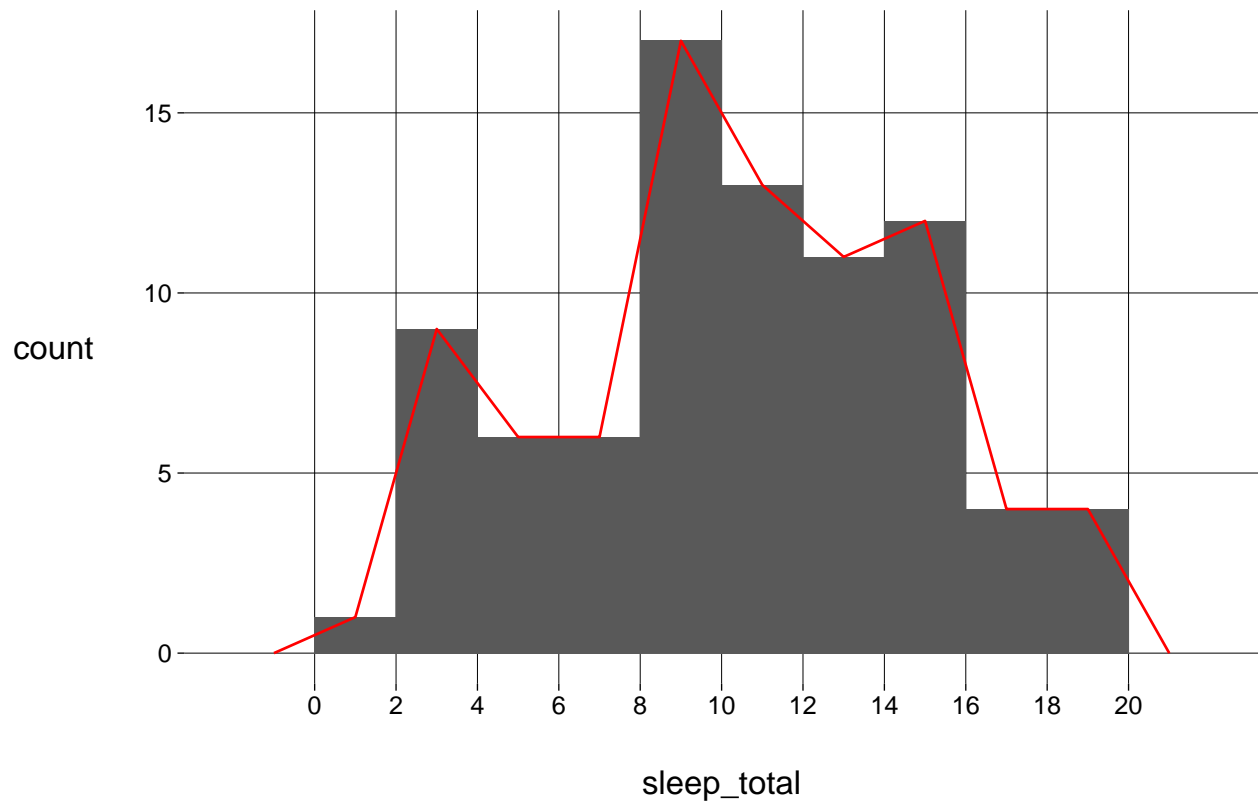
```
sono %>%  
  ggplot(aes(x = sleep_total)) +  
    geom_histogram(breaks = seq(0, 20, 2)) +  
    scale_x_continuous(breaks = seq(0, 20, 2))
```



3.5.3

Polígono de frequência

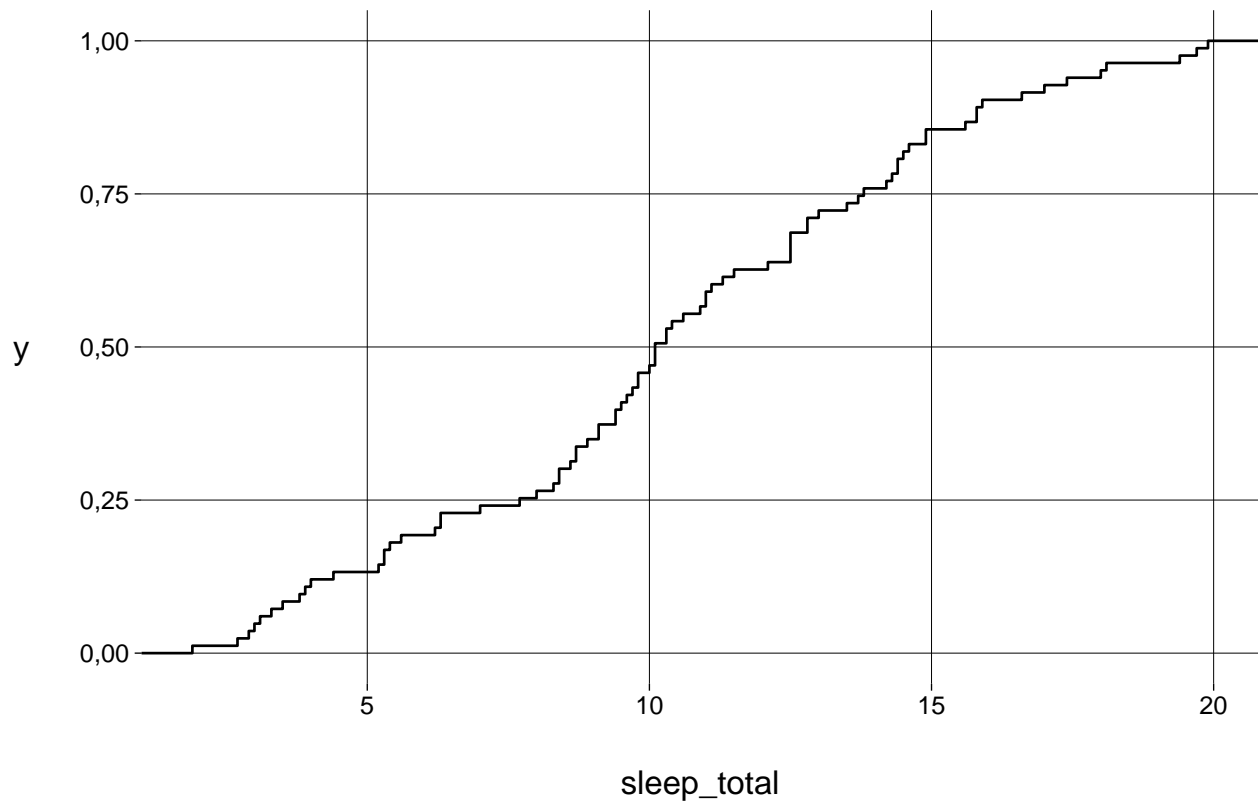
```
sono %>%  
  ggplot(aes(x = sleep_total)) +  
    geom_histogram(breaks = seq(0, 20, 2)) +  
    geom_freqpoly(breaks = seq(0, 20, 2), color = 'red') +  
    scale_x_continuous(breaks = seq(0, 20, 2))
```



3.6

Ogiva (frequência acumulada)

```
sono %>%  
  ggplot(aes(x = sleep_total)) +  
  geom_step(stat = 'ecdf')
```



3.7

Ramos e folhas

```
sono$sleep_total %>%
  stem()
##
##   The decimal point is at the |
##
##    0 | 9
##    2 | 79013589
##    4 | 0423346
##    6 | 23307
##    8 | 03446779114456788
##   10 | 01113346900135
##   12 | 15555880578
##   14 | 234456996889
##   16 | 604
##   18 | 01479
```

3.8

Exercícios

1. Construa um histograma da variável `brainwt`. Escolha o número de classes que você achar melhor. O que acontece com os valores `NA`?
2. Construa um *scatter plot* de horas de sono versus peso do cérebro. Você percebe alguma correlação entre estas variáveis?