Exploring the tidyverse

H2 Data Science

Introduction to the *tidyverse*

Throughout much of your coursework, you will receive clean, compact datasets that contain all of the data and only the data that you need. This, of course, does not reflect the experience of research. In reality, a typical data scientist spends the majority of their time wrangling data. Inevitably, your data will come from several sources, be collected by different people, entered into a spreadsheet by even different people, and arrive on your computer in multiple arrangements. There may be missing or incorrect values, or multiple labels for the same treatment.

Even with perfectly cleaned datasets, you should spend a good amount of time *looking* at your data in different ways. This is known as Exploratory Data Analysis (EDA). These quick, messy plots and summaries are a great way to identify potential errors in your data and will help you develop an intuitive sense of the structure of and patterns within complex datasets.

The *tidyverse* is "an opinionated collection of R packages designed for data science" that is commonly used in the R-based data science world. They share a core design philosophy with a distinct coding grammar and distinct data structures. These packages work collaboratively to make data wrangling, analysis, and visualization easier, cleaner, and more reproducible.

We will just dip our toes into the *tidyverse* – there is much, much more that we will not cover.

What makes a dataset tidy vs. messy?

Hadley Wickham (R legend and person largely behind the *tidyverse*) has said that "tidy datasets are all alike, but every messy dataset is messy in its own way." Within the *tidyverse* framework, a tidy dataset will have the following characteristics:

1. Each variable is a single column.
2. Each observation is a single row.
3. Each type of observational unit is a single table.

If you have multiple spreadsheets, each should have a column with the same name that allows them to be joined or merged appropriately.

Guidelines for data entry

When entering your data, try to keep to the following guidelines:

1. Be consistent

Stick with a single way of entering variables (e.g., use 'M'/'F' or 'male'/'female', but don't mix them)

2. Choose good names for things

Avoid spaces where possible (use underscores or somethingLikeCamelCase). Names should be descriptive but not overly long. In R, objects can't start with numbers or special characters, so try to avoid that in your variable levels as well.

3. Write dates as YYYY-MM-DD

This is the global ISO 8601 standard. It sorts well and keeps Excel from seriously mangling your data.

4. No empty cells

Enter NA instead.

5. Put just one thing in a cell

For example, have a column `weight_kg` filled with numbers rather than a column called `weight` filled with values like "5.5 kg".

6. Don't use font color or highlighting as data

Sometimes used to flag values. If needed, create a new column to mark observations needing extra attention.

7. Save the data as plain text files

Proprietary formats like .xlsx can be handy, but plain text (.csv, .txt, .tsv, etc) does not require special software and can be read by anyone on any machine.

Key packages in the *tidyverse*

There are many packages associated with the tidyverse. We will cover **aspects** of only these select few:

- `dplyr`: data manipulation
- `tidyr`: tidying data
- `tibble`: a re-imagining of the dataframe
- `stringr`: working with strings
- `lubridate`: working with dates and date-times
- `ggplot2`: visualization

When you install the *tidyverse* package, all of these are installed (and more).

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

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

But first: Piping

Piping is a useful technique to keep your code legible. The 'pipe operator' allows you to string together functions by taking the output of one and sending it as input to the next. Many functions in the *tidyverse* are designed to work nicely in pipes.

The native R pipe is `|>` and was introduced in R 4.1.0. Prior to this, piping required the *magrittr* package. You will still see the *magrittr* pipe `%>%`. There are a few differences and the *magrittr* pipe has a few extra capabilities. We will stick with the native `|>` here.

```
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
## 
```

```
iris |>
  summary()
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
## 
```

Tibbles

Tibbles are dataframes, but with a few extra tweaks. See the Tibbles chapter in R for Data Science for a more comprehensive overview.

```
data("msleep")
?msleep
```

```
## starting httpd help server ... done
```

```
class(msleep)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
msleep
```

```
## # A tibble: 83 x 11
##    name   genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr>  <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 Cheet~ Acin~ carni Carn~ lc                  12.1      NA        NA      11.9
##  2 Owl m~ Aotus omni  Prim~ <NA>                17         1.8      NA       7
##  3 Mount~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA       9.6
##  4 Great~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133   9.1
##  5 Cow    Bos   herbi Arti~ domesticated         4         0.7       0.667  20
##  6 Three~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767   9.6
##  7 North~ Call~ carni Carn~ vu                   8.7       1.4       0.383  15.3
##  8 Vespe~ Calo~ <NA>  Rode~ <NA>                 7        NA        NA      17
##  9 Dog    Canis carni Carn~ domesticated        10.1       2.9       0.333  13.9
## 10 Roe d~ Capr~ herbi Arti~ lc                   3        NA        NA      21
## # i 73 more rows
## # i 2 more variables: brainwt <dbl>, bodywt <dbl>
```

The *tidyverse* alternative to `read.csv()` is `read_csv()`. It is basically the same, but with a few different defaults and switches, and it outputs a tibble.

```
fieldData.df <- read_csv("fieldData.csv")

class(fieldData.df)

fieldData.df

str(fieldData.df)
glimpse(fieldData.df)
```

Munging data

It is best to keep your data file as it is, and do all of your cleaning, filtering, transforming, etc. in R for full transparency and reproducibility. Editing by hand (e.g., in Excel) leaves no clear trail of what you did and also makes mistakes easy to make but difficult to find.

The dplyr package provides a set of functions that make munging your data simple and easy. See the Data Transformation chapter in R for Data Science for a more comprehensive overview.

Key functions:

- `filter()` filters **rows** based on a `TRUE/FALSE` logical statement
- `select()` selects **columns** by name, number, or a *tidyselect* helper function
- `arrange()` reorders the tibble according to the column(s) provided
- `mutate()` creates new columns
- `group_by()` groups the tibble by the column(s) provided
- `summarise()` creates summaries, either for the whole tibble or by group

```
# filter()
msleep |>
  filter(vore=="herbi")
```

```
## # A tibble: 32 x 11
##    name   genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr>  <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 Mount~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA       9.6
##  2 Cow    Bos   herbi Arti~ domesticated         4         0.7       0.667  20
##  3 Three~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767   9.6
##  4 Roe d~ Capr~ herbi Arti~ lc                   3        NA        NA      21
##  5 Goat   Capri herbi Arti~ lc                   5.3       0.6      NA      18.7
##  6 Guine~ Cavis herbi Rode~ domesticated         9.4       0.8       0.217  14.6
##  7 Chinc~ Chin~ herbi Rode~ domesticated        12.5       1.5       0.117  11.5
##  8 Tree ~ Dend~ herbi Hyra~ lc                   5.3       0.5      NA      18.7
##  9 Asian~ Elep~ herbi Prob~ en                   3.9      NA        NA      20.1
## 10 Horse  Equus herbi Peri~ domesticated         2.9       0.6       1      21.1
## # i 22 more rows
## # i 2 more variables: brainwt <dbl>, bodywt <dbl>
```

```
# select()
msleep |>
  select(-conservation)
```

```
## # A tibble: 83 x 10
##    name       genus vore  order sleep_total sleep_rem sleep_cycle awake  brainwt
##    <chr>      <chr> <chr> <chr>       <dbl>     <dbl>       <dbl> <dbl>    <dbl>
##  1 Cheetah    Acin~ carni Carn~        12.1      NA        NA      11.9 NA
##  2 Owl monkey Aotus omni  Prim~        17         1.8      NA       7    0.0155
##  3 Mountain ~ Aplo~ herbi Rode~        14.4       2.4      NA       9.6 NA
##  4 Greater s~ Blar~ omni  Sori~        14.9       2.3       0.133   9.1  0.00029
##  5 Cow        Bos   herbi Arti~         4         0.7       0.667  20    0.423
##  6 Three-toe~ Brad~ herbi Pilo~        14.4       2.2       0.767   9.6 NA
##  7 Northern ~ Call~ carni Carn~         8.7       1.4       0.383  15.3 NA
##  8 Vesper mo~ Calo~ <NA>  Rode~         7        NA        NA      17   NA
##  9 Dog        Canis carni Carn~        10.1       2.9       0.333  13.9  0.07
## 10 Roe deer   Capr~ herbi Arti~         3        NA        NA      21    0.0982
## # i 73 more rows
## # i 1 more variable: bodywt <dbl>
```

```
msleep |>
  select(1, 9:11)
```

```
## # A tibble: 83 x 4
##    name                       awake  brainwt  bodywt
##    <chr>                          <dbl>    <dbl>   <dbl>
##  1 Cheetah                         11.9 NA        50
##  2 Owl monkey                       7    0.0155    0.48
##  3 Mountain beaver                  9.6 NA         1.35
##  4 Greater short-tailed shrew       9.1  0.00029   0.019
##  5 Cow                             20    0.423   600
##  6 Three-toed sloth                 9.6 NA         3.85
##  7 Northern fur seal               15.3 NA        20.5
##  8 Vesper mouse                    17   NA         0.045
##  9 Dog                             13.9  0.07     14
## 10 Roe deer                        21    0.0982   14.8
## # i 73 more rows
```

```
msleep |>
  select(name, order, starts_with("sleep"))
```

```
## # A tibble: 83 x 5
##    name                       order        sleep_total sleep_rem sleep_cycle
##    <chr>                      <chr>              <dbl>     <dbl>       <dbl>
##  1 Cheetah                    Carnivora           12.1      NA        NA
##  2 Owl monkey                 Primates            17         1.8      NA
##  3 Mountain beaver            Rodentia            14.4       2.4      NA
##  4 Greater short-tailed shrew Soricomorpha        14.9       2.3       0.133
##  5 Cow                        Artiodactyla         4         0.7       0.667
##  6 Three-toed sloth           Pilosa              14.4       2.2       0.767
##  7 Northern fur seal          Carnivora            8.7       1.4       0.383
##  8 Vesper mouse               Rodentia             7        NA        NA
##  9 Dog                        Carnivora           10.1       2.9       0.333
## 10 Roe deer                   Artiodactyla         3        NA        NA
## # i 73 more rows
```

```
# arrange()
msleep |>
  filter(vore=="herbi") |>
  arrange(bodywt)
```

```
## # A tibble: 32 x 11
##    name   genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr>  <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 "Hous~ Mus   herbi Rode~ nt                  12.5       1.4       0.183  11.5
##  2 "Vole~ Micr~ herbi Rode~ <NA>                12.8      NA        NA      11.2
##  3 "Mong~ Meri~ herbi Rode~ lc                  14.2       1.9      NA       9.8
##  4 "West~ Euta~ herbi Rode~ <NA>                14.9      NA        NA       9.1
##  5 "Thir~ Sper~ herbi Rode~ lc                  13.8       3.4       0.217  10.2
##  6 "East~ Tami~ herbi Rode~ <NA>                15.8      NA        NA       8.2
##  7 "Gold~ Meso~ herbi Rode~ en                  14.3       3.1       0.2     9.7
##  8 "Cott~ Sigm~ herbi Rode~ <NA>                11.3       1.1       0.15   12.7
##  9 "Gold~ Sper~ herbi Rode~ lc                  15.9       3        NA       8.1
## 10 "Degu" Octo~ herbi Rode~ lc                   7.7       0.9      NA      16.3
## # i 22 more rows
## # i 2 more variables: brainwt <dbl>, bodywt <dbl>
```

```r
# mutate()
msleep |>
  filter(!is.na(brainwt) & !is.na(bodywt)) |>
  mutate(brain_pct=brainwt/bodywt*100) |>
  arrange(desc(brain_pct))
```

```
## # A tibble: 56 x 12
##    name   genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr>  <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 Thirt~ Sper~ herbi Rode~ lc                  13.8       3.4       0.217  10.2
##  2 Owl m~ Aotus omni  Prim~ <NA>                17         1.8      NA       7
##  3 Lesse~ Cryp~ omni  Sori~ lc                   9.1       1.4       0.15   14.9
##  4 Squir~ Saim~ omni  Prim~ <NA>                 9.6       1.4      NA      14.4
##  5 Macaq~ Maca~ omni  Prim~ <NA>                10.1       1.2       0.75   13.9
##  6 Galago Gala~ omni  Prim~ <NA>                 9.8       1.1       0.55   14.2
##  7 Littl~ Myot~ inse~ Chir~ <NA>                19.9       2         0.2     4.1
##  8 Mole ~ Spal~ <NA>  Rode~ <NA>                10.6       2.4      NA      13.4
##  9 Tree ~ Tupa~ omni  Scan~ <NA>                 8.9       2.6       0.233  15.1
## 10 Human  Homo  omni  Prim~ <NA>                 8         1.9       1.5    16
## # i 46 more rows
## # i 3 more variables: brainwt <dbl>, bodywt <dbl>, brain_pct <dbl>
```

```r
# summarise()
msleep |>
  filter(!is.na(awake)) |>
  summarise(nSpp=n_distinct(name),
            nGenera=n_distinct(genus),
            awake_mean=mean(awake),
            awake_sd=sd(awake))
```

```
## # A tibble: 1 x 4
##    nSpp nGenera awake_mean awake_sd
##   <int>   <int>      <dbl>    <dbl>
## 1    83      77       13.6     4.45
```

```r
msleep |>
  filter(!is.na(awake)) |>
  group_by(vore) |>
  summarise(nSpp=n_distinct(name),
            nGenera=n_distinct(genus),
            awake_mean=mean(awake),
            awake_sd=sd(awake))
```

```
## # A tibble: 5 x 5
##   vore     nSpp nGenera awake_mean awake_sd
##   <chr>   <int>   <int>      <dbl>    <dbl>
## 1 carni      19      16      13.6      4.68
## 2 herbi      32      29      14.5      4.88
## 3 insecti     5       5       9.06     5.92
## 4 omni       20      20      13.1      2.95
## 5 <NA>        7       7      13.8      3.00
```

Reshaping data

The tidyr package has many functions for tidying your datasets. We will cover just two that tend to be particularly useful.

Key functions:

- `pivot_longer()` takes a series of columns and stacks them into one
- `pivot_wider()` takes a single column and spreads them into several

```
# pivot_longer
msleep |>
  select(name, genus, order, brainwt, bodywt) |>
  pivot_longer(cols=contains("wt"), names_to="structure", values_to="weight")
```

```
## # A tibble: 166 x 5
##    name                       genus      order        structure    weight
##    <chr>                      <chr>      <chr>        <chr>         <dbl>
##  1 Cheetah                    Acinonyx   Carnivora    brainwt    NA
##  2 Cheetah                    Acinonyx   Carnivora    bodywt     50
##  3 Owl monkey                 Aotus      Primates     brainwt     0.0155
##  4 Owl monkey                 Aotus      Primates     bodywt      0.48
##  5 Mountain beaver            Aplodontia Rodentia     brainwt    NA
##  6 Mountain beaver            Aplodontia Rodentia     bodywt      1.35
##  7 Greater short-tailed shrew Blarina    Soricomorpha brainwt     0.00029
##  8 Greater short-tailed shrew Blarina    Soricomorpha bodywt      0.019
##  9 Cow                        Bos        Artiodactyla brainwt     0.423
## 10 Cow                        Bos        Artiodactyla bodywt    600
## # i 156 more rows
```

```
# pivot_wider
iris |>
  group_by(Species) |>
  mutate(Plant_id=row_number()) |>
  ungroup() |>
  select(Plant_id, Species, Petal.Width) |>
  pivot_wider(names_from="Species", values_from="Petal.Width")
```

```
## # A tibble: 50 x 4
##    Plant_id setosa versicolor virginica
##       <int>  <dbl>      <dbl>     <dbl>
##  1        1    0.2        1.4       2.5
##  2        2    0.2        1.5       1.9
##  3        3    0.2        1.5       2.1
##  4        4    0.2        1.3       1.8
##  5        5    0.2        1.5       2.2
##  6        6    0.4        1.3       2.1
##  7        7    0.3        1.6       1.7
##  8        8    0.2        1         1.8
##  9        9    0.2        1.3       1.8
## 10       10    0.1        1.4       2.5
## # i 40 more rows
```

Working with strings

The stringr package has many convenient functions for working with strings (text). They generally follow the naming form of `str_verb()`. See the Strings chapter in R for Data Science for a more comprehensive overview.

Key functions:

- `str_remove()` removes a *pattern* from each element in the vector
- `str_replace()` replaces a *pattern* from each element in the vector
- `str_sub()` extracts a subset of characters from each element in the vector
- `str_split()` splits each element in the vector based on a *pattern*

```
# str_remove / str_remove_all
msleep |>
  select(name, genus, order) |>
  mutate(nameNoSpace=str_remove(name, " "))
```

```
## # A tibble: 83 x 4
##    name                       genus       order        nameNoSpace
##    <chr>                      <chr>       <chr>        <chr>
##  1 Cheetah                    Acinonyx    Carnivora    Cheetah
##  2 Owl monkey                 Aotus       Primates     Owlmonkey
##  3 Mountain beaver            Aplodontia  Rodentia     Mountainbeaver
##  4 Greater short-tailed shrew Blarina     Soricomorpha Greatershort-tailed shrew
##  5 Cow                        Bos         Artiodactyla Cow
##  6 Three-toed sloth           Bradypus    Pilosa       Three-toedsloth
##  7 Northern fur seal          Callorhinus Carnivora    Northernfur seal
##  8 Vesper mouse               Calomys     Rodentia     Vespermouse
##  9 Dog                        Canis       Carnivora    Dog
## 10 Roe deer                   Capreolus   Artiodactyla Roedeer
## # i 73 more rows
```

```
msleep |>
  select(name, genus, order) |>
  mutate(nameNoSpace=str_remove_all(name, " "))
```

```
## # A tibble: 83 x 4
##    name                       genus       order        nameNoSpace
##    <chr>                      <chr>       <chr>        <chr>
##  1 Cheetah                    Acinonyx    Carnivora    Cheetah
##  2 Owl monkey                 Aotus       Primates     Owlmonkey
##  3 Mountain beaver            Aplodontia  Rodentia     Mountainbeaver
##  4 Greater short-tailed shrew Blarina     Soricomorpha Greatershort-tailedshrew
##  5 Cow                        Bos         Artiodactyla Cow
##  6 Three-toed sloth           Bradypus    Pilosa       Three-toedsloth
##  7 Northern fur seal          Callorhinus Carnivora    Northernfurseal
##  8 Vesper mouse               Calomys     Rodentia     Vespermouse
##  9 Dog                        Canis       Carnivora    Dog
## 10 Roe deer                   Capreolus   Artiodactyla Roedeer
## # i 73 more rows
```

```r
# str_replace / str_replace_all
msleep |>
  select(name, genus, order) |>
  mutate(nameNoSpace=str_replace(name, " ", "_"))
```

```
## # A tibble: 83 x 4
##    name                       genus       order        nameNoSpace
##    <chr>                      <chr>       <chr>        <chr>
##  1 Cheetah                    Acinonyx    Carnivora    Cheetah
##  2 Owl monkey                 Aotus       Primates     Owl_monkey
##  3 Mountain beaver            Aplodontia  Rodentia     Mountain_beaver
##  4 Greater short-tailed shrew Blarina     Soricomorpha Greater_short-tailed shr~
##  5 Cow                        Bos         Artiodactyla Cow
##  6 Three-toed sloth           Bradypus    Pilosa       Three-toed_sloth
##  7 Northern fur seal          Callorhinus Carnivora    Northern_fur seal
##  8 Vesper mouse               Calomys     Rodentia     Vesper_mouse
##  9 Dog                        Canis       Carnivora    Dog
## 10 Roe deer                   Capreolus   Artiodactyla Roe_deer
## # i 73 more rows
```

```r
msleep |>
  select(name, genus, order) |>
  mutate(nameNoSpace=str_replace_all(name, " ", "_"))
```

```
## # A tibble: 83 x 4
##    name                       genus       order        nameNoSpace
##    <chr>                      <chr>       <chr>        <chr>
##  1 Cheetah                    Acinonyx    Carnivora    Cheetah
##  2 Owl monkey                 Aotus       Primates     Owl_monkey
##  3 Mountain beaver            Aplodontia  Rodentia     Mountain_beaver
##  4 Greater short-tailed shrew Blarina     Soricomorpha Greater_short-tailed_shr~
##  5 Cow                        Bos         Artiodactyla Cow
##  6 Three-toed sloth           Bradypus    Pilosa       Three-toed_sloth
##  7 Northern fur seal          Callorhinus Carnivora    Northern_fur_seal
##  8 Vesper mouse               Calomys     Rodentia     Vesper_mouse
##  9 Dog                        Canis       Carnivora    Dog
## 10 Roe deer                   Capreolus   Artiodactyla Roe_deer
## # i 73 more rows
```

```r
# str_sub()
msleep |>
  select(name, genus, order) |>
  mutate(order_abbr=str_sub(order, 1, 4))
```

```
## # A tibble: 83 x 4
##    name                       genus       order        order_abbr
##    <chr>                      <chr>       <chr>        <chr>
##  1 Cheetah                    Acinonyx    Carnivora    Carn
##  2 Owl monkey                 Aotus       Primates     Prim
##  3 Mountain beaver            Aplodontia  Rodentia     Rode
##  4 Greater short-tailed shrew Blarina     Soricomorpha Sori
##  5 Cow                        Bos         Artiodactyla Arti
##  6 Three-toed sloth           Bradypus    Pilosa       Pilo
##  7 Northern fur seal          Callorhinus Carnivora    Carn
##  8 Vesper mouse               Calomys     Rodentia     Rode
##  9 Dog                        Canis       Carnivora    Carn
## 10 Roe deer                   Capreolus   Artiodactyla Arti
## # i 73 more rows
```

```
# str_split()
msleep |>
  select(name, genus, order) |>
  mutate(nameSplit=str_split(name, " "))
```

```
## # A tibble: 83 x 4
##    name                       genus       order        nameSplit
##    <chr>                      <chr>       <chr>        <list>
##  1 Cheetah                    Acinonyx    Carnivora    <chr [1]>
##  2 Owl monkey                 Aotus       Primates     <chr [2]>
##  3 Mountain beaver            Aplodontia  Rodentia     <chr [2]>
##  4 Greater short-tailed shrew Blarina     Soricomorpha <chr [3]>
##  5 Cow                        Bos         Artiodactyla <chr [1]>
##  6 Three-toed sloth           Bradypus    Pilosa       <chr [2]>
##  7 Northern fur seal          Callorhinus Carnivora    <chr [3]>
##  8 Vesper mouse               Calomys     Rodentia     <chr [2]>
##  9 Dog                        Canis       Carnivora    <chr [1]>
## 10 Roe deer                   Capreolus   Artiodactyla <chr [2]>
## # i 73 more rows
```

```
str_split(msleep$name, " ") |> head()
```

```
## [[1]]
## [1] "Cheetah"
##
## [[2]]
## [1] "Owl"    "monkey"
##
## [[3]]
## [1] "Mountain" "beaver"
##
## [[4]]
## [1] "Greater"      "short-tailed" "shrew"
##
## [[5]]
## [1] "Cow"
##
## [[6]]
## [1] "Three-toed" "sloth"
```

```
str_split_fixed(msleep$name, " ", 3) |> head()
```

```
##      [,1]         [,2]           [,3]
## [1,] "Cheetah"    ""             ""
## [2,] "Owl"        "monkey"       ""
## [3,] "Mountain"   "beaver"       ""
## [4,] "Greater"    "short-tailed" "shrew"
## [5,] "Cow"        ""             ""
## [6,] "Three-toed" "sloth"        ""
```

```
str_split_fixed(msleep$name, " ", 3)[,1]
```

```
##  [1] "Cheetah"       "Owl"           "Mountain"      "Greater"
##  [5] "Cow"           "Three-toed"    "Northern"      "Vesper"
##  [9] "Dog"           "Roe"           "Goat"          "Guinea"
## [13] "Grivet"        "Chinchilla"    "Star-nosed"    "African"
## [17] "Lesser"        "Long-nosed"    "Tree"          "North"
## [21] "Asian"         "Big"           "Horse"         "Donkey"
## [25] "European"      "Patas"         "Western"       "Domestic"
## [29] "Galago"        "Giraffe"       "Pilot"         "Gray"
## [33] "Gray"          "Human"         "Mongoose"      "African"
## [37] "Thick-tailed"  "Macaque"       "Mongolian"     "Golden"
## [41] "Vole"          "House"         "Little"        "Round-tailed"
## [45] "Slow"          "Degu"          "Northern"      "Rabbit"
## [49] "Sheep"         "Chimpanzee"    "Tiger"         "Jaguar"
## [53] "Lion"          "Baboon"        "Desert"        "Potto"
## [57] "Deer"          "Phalanger"     "Caspian"       "Common"
## [61] "Potoroo"       "Giant"         "Rock"          "Laboratory"
## [65] "African"       "Squirrel"      "Eastern"       "Cotton"
## [69] "Mole"          "Arctic"        "Thirteen-lined" "Golden-mantled"
## [73] "Musk"          "Pig"           "Short-nosed"   "Eastern"
## [77] "Brazilian"     "Tenrec"        "Tree"          "Bottle-nosed"
## [81] "Genet"         "Arctic"        "Red"
```

Working with dates

Date-time data are notoriously prickly, particularly in R. The lubridate package makes them much less so. See the Dates and Times chapter in R for Data Science for a more comprehensive overview.

Key functions:

- `ymd()`, `mdy()`, `dmy()` convert a string with **y**ear, **m**onth, and **d**ay into a date
- `ymd_hms()` converts a string to a date-time, including **h**our, **m**inutes, and **s**econds
- `year()`, `month()`, `mday()`, `yday()`, `week()` extract the corresponding component
- adding and subtracting work intuitively
- plotting with *ggplot2* is straightforward

```
data(storms)
glimpse(storms)
```

```
## Rows: 19,066
## Columns: 13
## $ name                         <chr> "Amy", "Amy", "Amy", "Amy", "Amy", "Amy",~
## $ year                         <dbl> 1975, 1975, 1975, 1975, 1975, 1975, 1975,~
## $ month                        <dbl> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,~
## $ day                          <int> 27, 27, 27, 27, 28, 28, 28, 28, 29, 29, 2~
## $ hour                         <dbl> 0, 6, 12, 18, 0, 6, 12, 18, 0, 6, 12, 18,~
## $ lat                          <dbl> 27.5, 28.5, 29.5, 30.5, 31.5, 32.4, 33.3,~
## $ long                         <dbl> -79.0, -79.0, -79.0, -79.0, -78.8, -78.7,~
## $ status                       <fct> tropical depression, tropical depression,~
## $ category                     <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ wind                         <int> 25, 25, 25, 25, 25, 25, 25, 30, 35, 40, 4~
## $ pressure                     <int> 1013, 1013, 1013, 1013, 1012, 1012, 1011,~
## $ tropicalstorm_force_diameter <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ hurricane_force_diameter     <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
```

```
storms_df <- storms |>
  mutate(date_chr=paste(year, month, day, sep="-"),
         date=ymd(date_chr),
         time=ymd_hms(paste0(date_chr, " ", hour, ":00:00"))) |>
  arrange(name, time) |>
  group_by(name) |>
  mutate(elapsedTime=time - first(time),
         elapsedHours=as.numeric(elapsedTime)/60/60,
         elapsedDays=difftime(time, first(time), units="days") |>
           as.numeric(),
         maxWind=max(wind)) |>
  ungroup()

glimpse(storms_df)
```

```
## Rows: 19,066
## Columns: 20
## $ name                         <chr> "AL011993", "AL011993", "AL011993", "AL01~
## $ year                         <dbl> 1993, 1993, 1993, 1993, 1993, 1993, 1993,~
## $ month                        <dbl> 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,~
## $ day                          <int> 31, 31, 1, 1, 1, 1, 2, 2, 2, 2, 3, 7, 8, ~
## $ hour                         <dbl> 12, 18, 0, 6, 12, 18, 0, 6, 12, 18, 0, 18~
## $ lat                          <dbl> 21.5, 22.3, 23.2, 24.5, 25.4, 26.1, 26.7,~
## $ long                         <dbl> -84.0, -82.0, -80.3, -79.0, -77.5, -75.8,~
## $ status                       <fct> tropical depression, tropical depression,~
## $ category                     <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ wind                         <int> 25, 25, 25, 25, 30, 30, 30, 30, 35, 35, 3~
## $ pressure                     <int> 1003, 1002, 1000, 1000, 999, 999, 999, 99~
## $ tropicalstorm_force_diameter <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ hurricane_force_diameter     <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ date_chr                     <chr> "1993-5-31", "1993-5-31", "1993-6-1", "19~
## $ date                         <date> 1993-05-31, 1993-05-31, 1993-06-01, 1993~
## $ time                         <dttm> 1993-05-31 12:00:00, 1993-05-31 18:00:00~
## $ elapsedTime                  <drtn> 0 secs, 21600 secs, 43200 secs, 64800 se~
## $ elapsedHours                 <dbl> 0, 6, 12, 18, 24, 30, 36, 42, 48, 54, 60,~
## $ elapsedDays                  <dbl> 0.0000000, 0.2500000, 0.5000000, 0.750000~
## $ maxWind                      <int> 35, 35, 35, 35, 35, 35, 35, 35, 35, 35, 3~
```

---

ggplot2

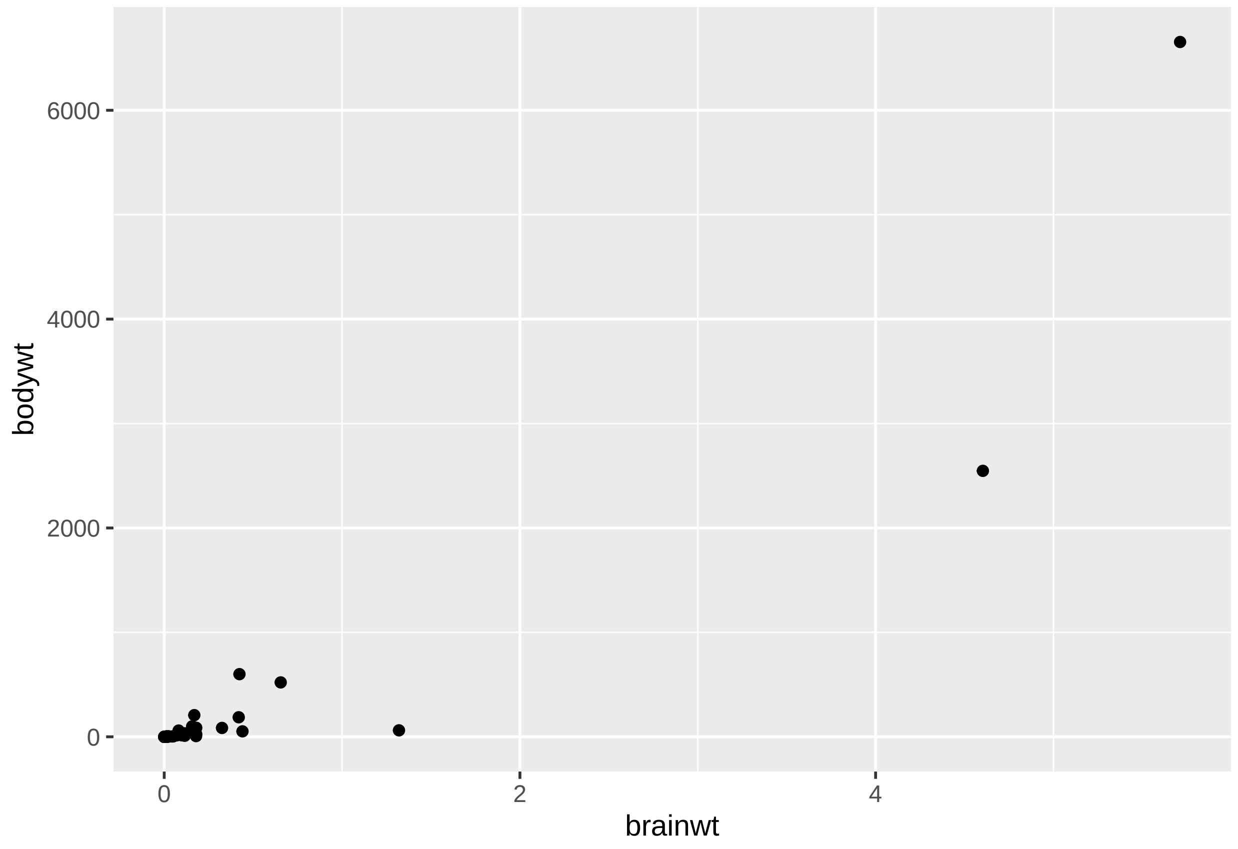
The *ggplot2* package is fantastic for exploratory data analysis.

Key functions:

- `ggplot()` initializes a plot object; always used first, followed by `+`
- `geom_*()` plot elements (e.g., `geom_point()`, `geom_line()`, etc)
- `stat_smooth()` calculates and plots a trend line
- `scale_colour_*()` specifies a custom colour palette
- `scale_fill_*()` specifies a custom fill palette
- `scale_x/y_*()` specifies a custom x or y axis
- `labs()` adds labels (x, y, title, subtitle)
- `facet_wrap()`, `facet_grid()` partition data into panels by the specified variable(s)

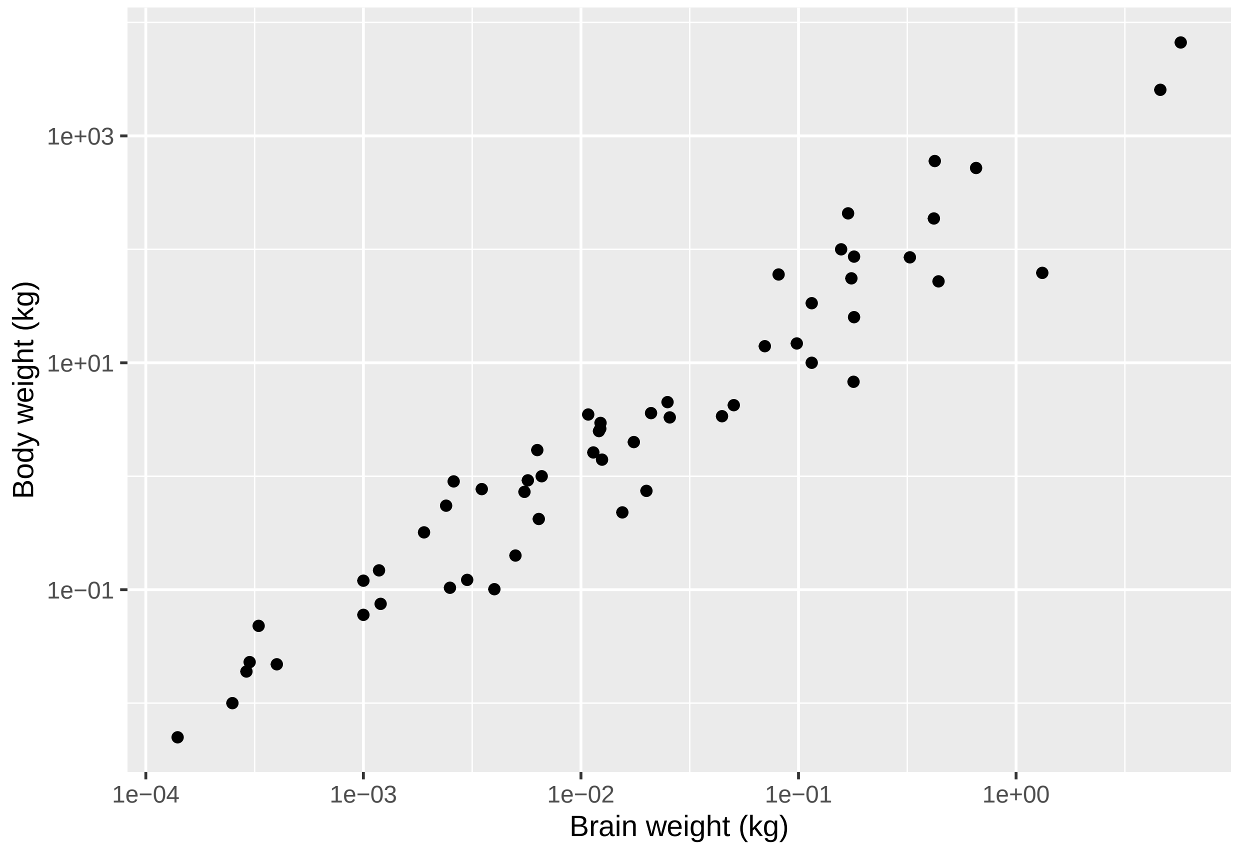
```
msleep |>
  ggplot(aes(brainwt, bodywt)) +
  geom_point()
```

```
## Warning: Removed 27 rows containing missing values (`geom_point()`).
```



```
msleep |>
  ggplot(aes(brainwt, bodywt)) +
  geom_point() +
  scale_x_log10("Brain weight (kg)") +
  scale_y_log10("Body weight (kg)")
```

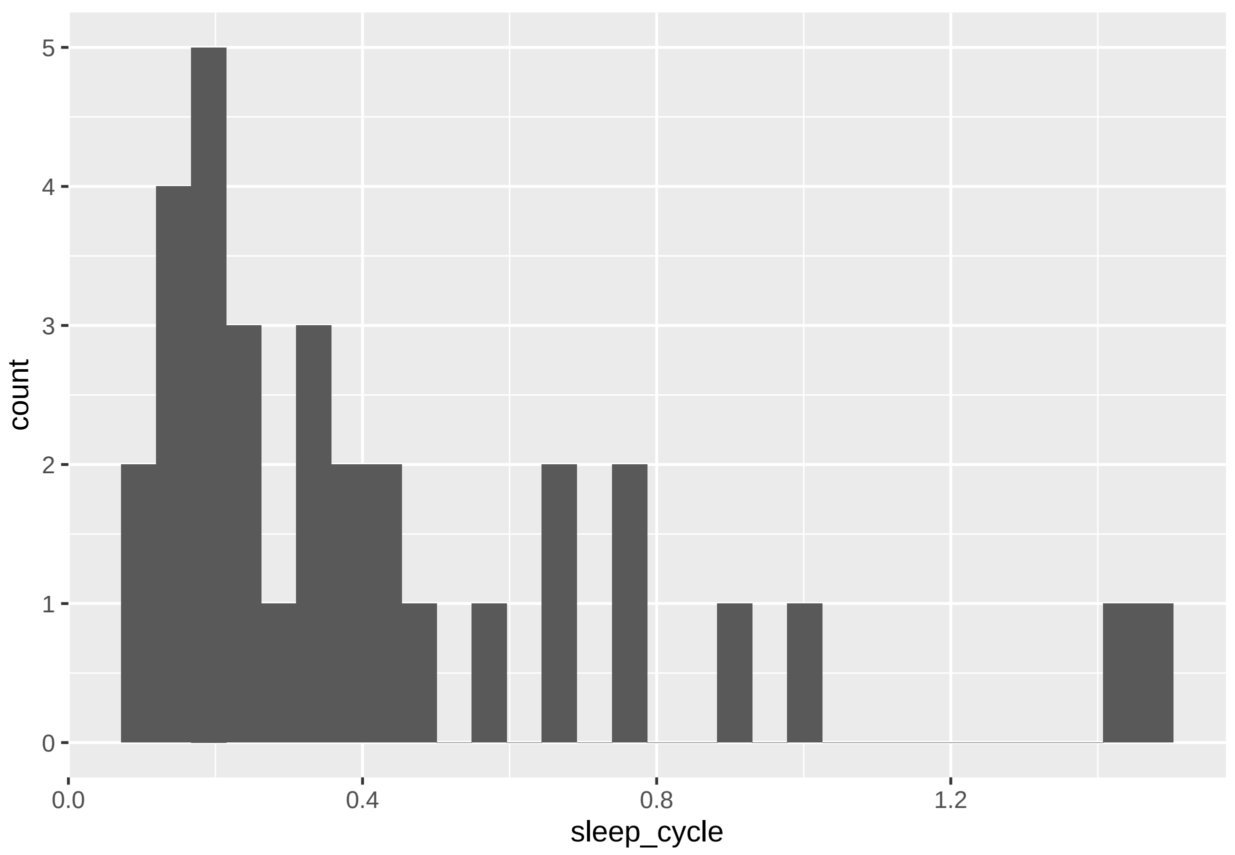
```
## Warning: Removed 27 rows containing missing values (`geom_point()`).
```



```
msleep |>
  ggplot(aes(sleep_cycle)) +
  geom_histogram()
```

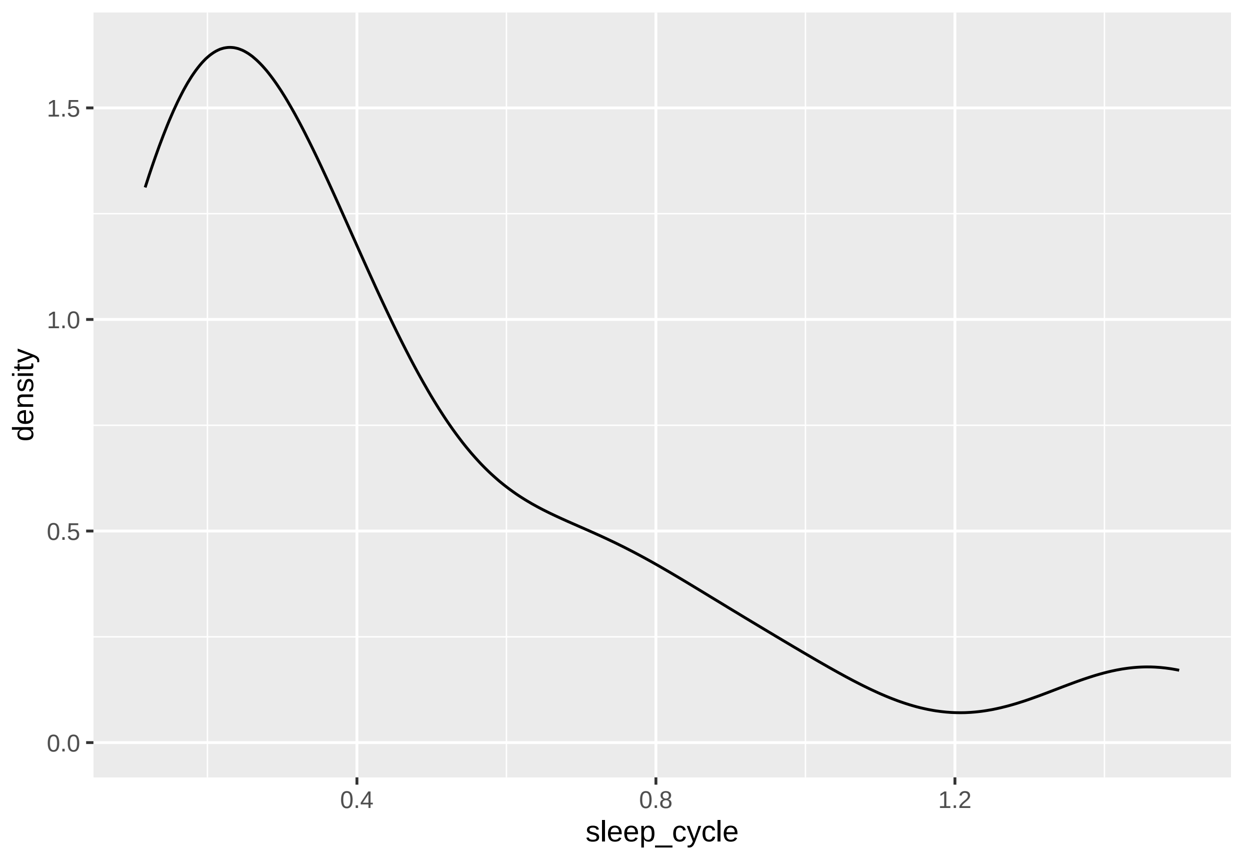
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 51 rows containing non-finite values (`stat_bin()`).
```



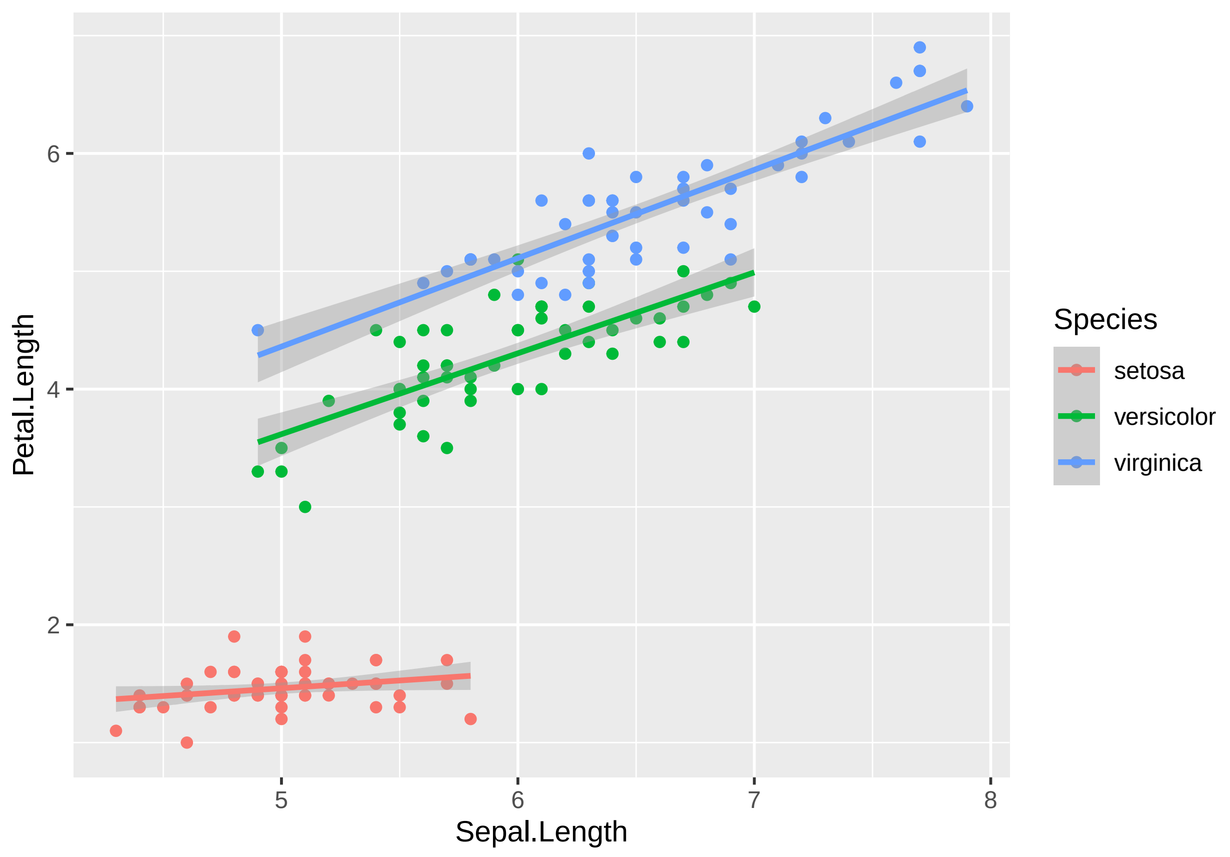
```
msleep |>
  ggplot(aes(sleep_cycle)) +
  geom_density()
```

```
## Warning: Removed 51 rows containing non-finite values (‘stat_density()‘).
```



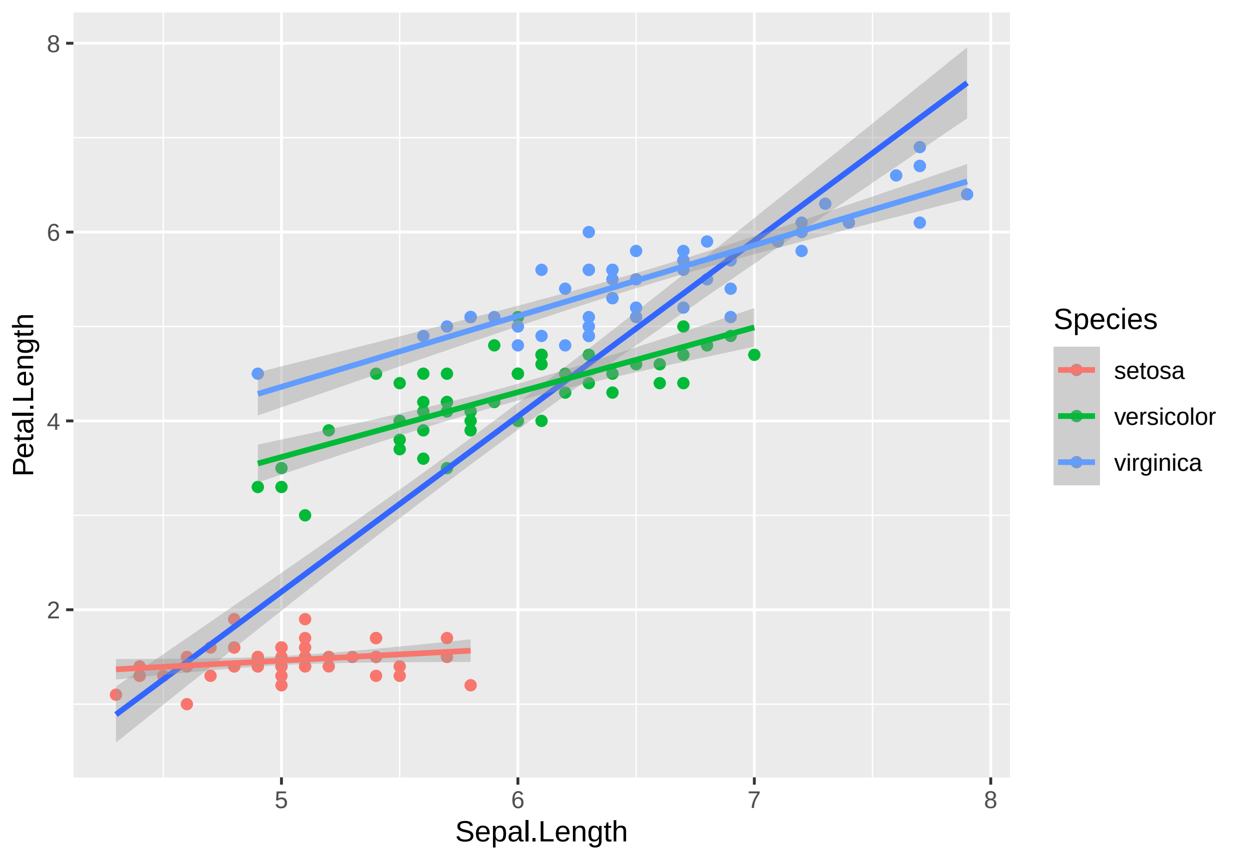
```
iris |>
  ggplot(aes(Sepal.Length, Petal.Length, colour=Species)) +
  geom_point() +
  stat_smooth(method="lm")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

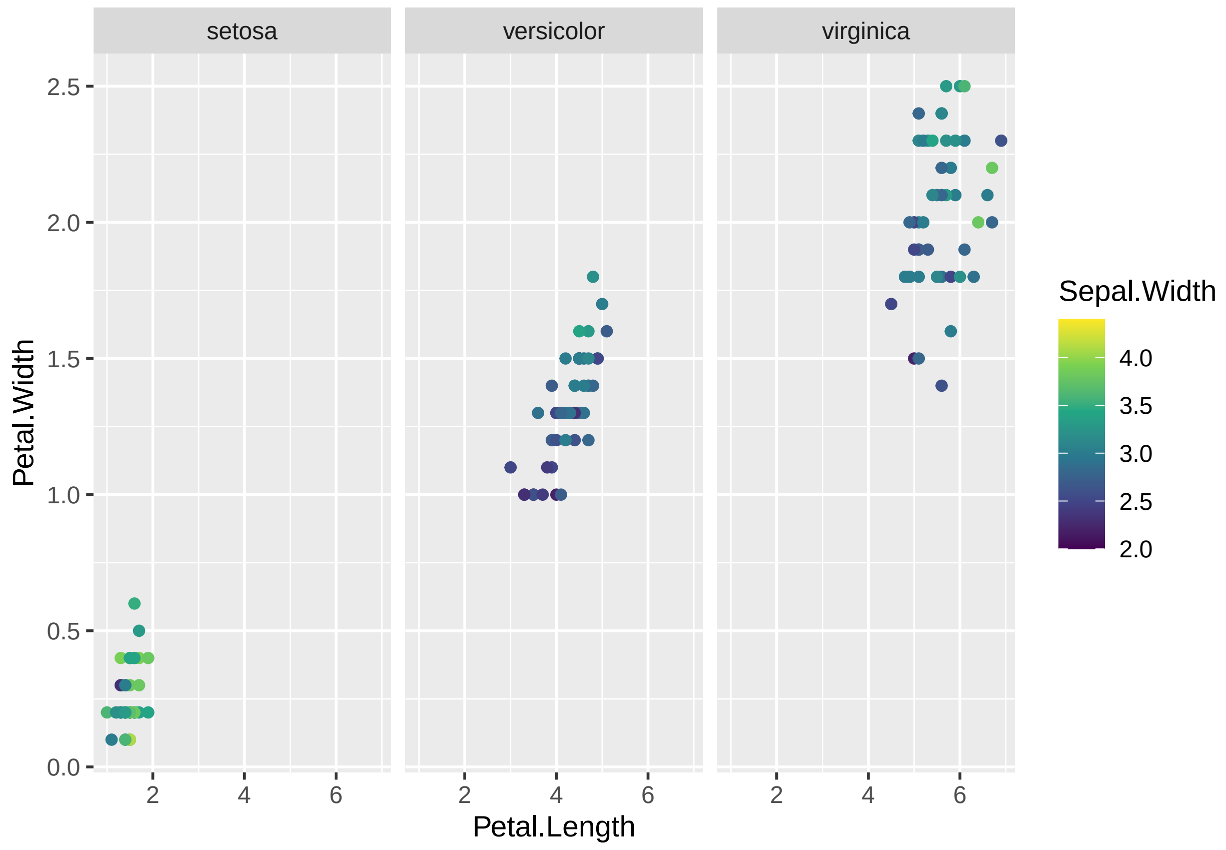


```r
iris |>
  ggplot(aes(Sepal.Length, Petal.Length)) +
  geom_point(aes(colour=Species)) +
  stat_smooth(method="lm") +
  stat_smooth(aes(colour=Species), method="lm")
```

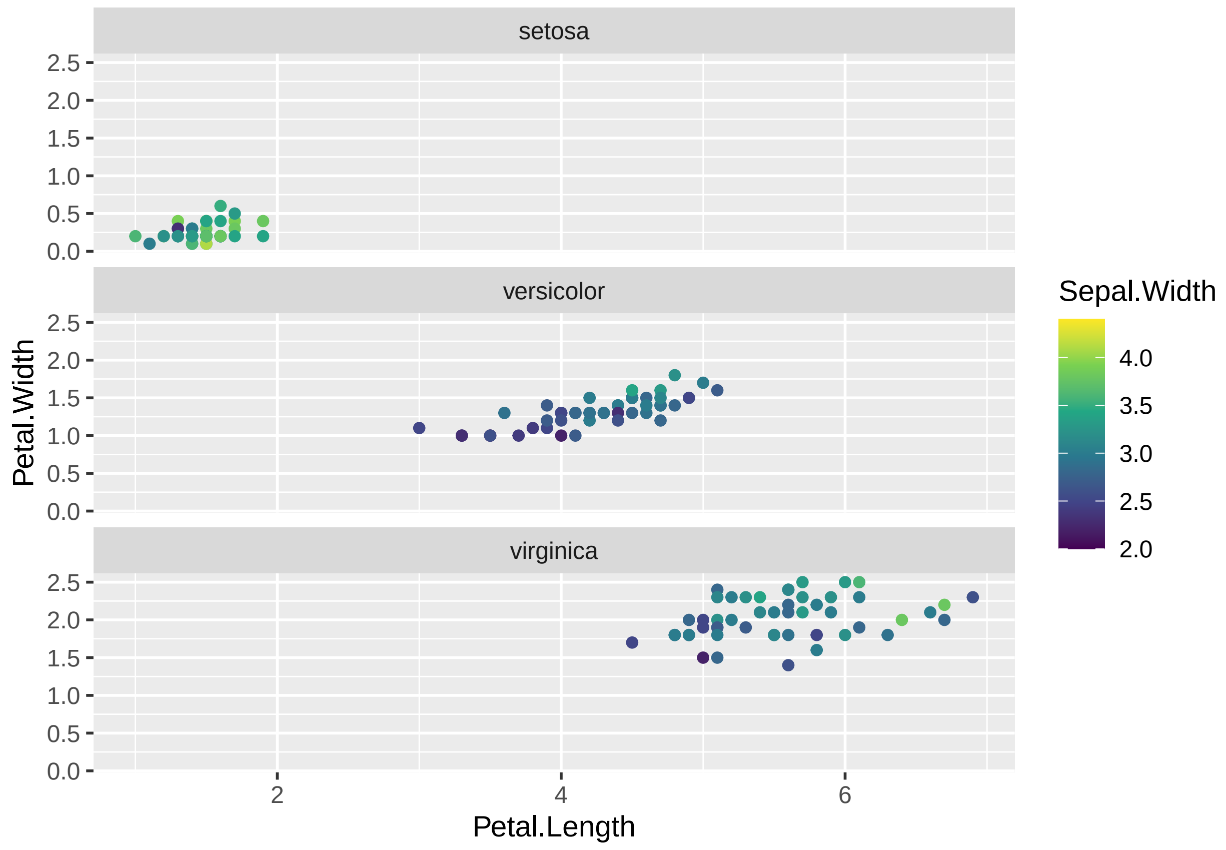
```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



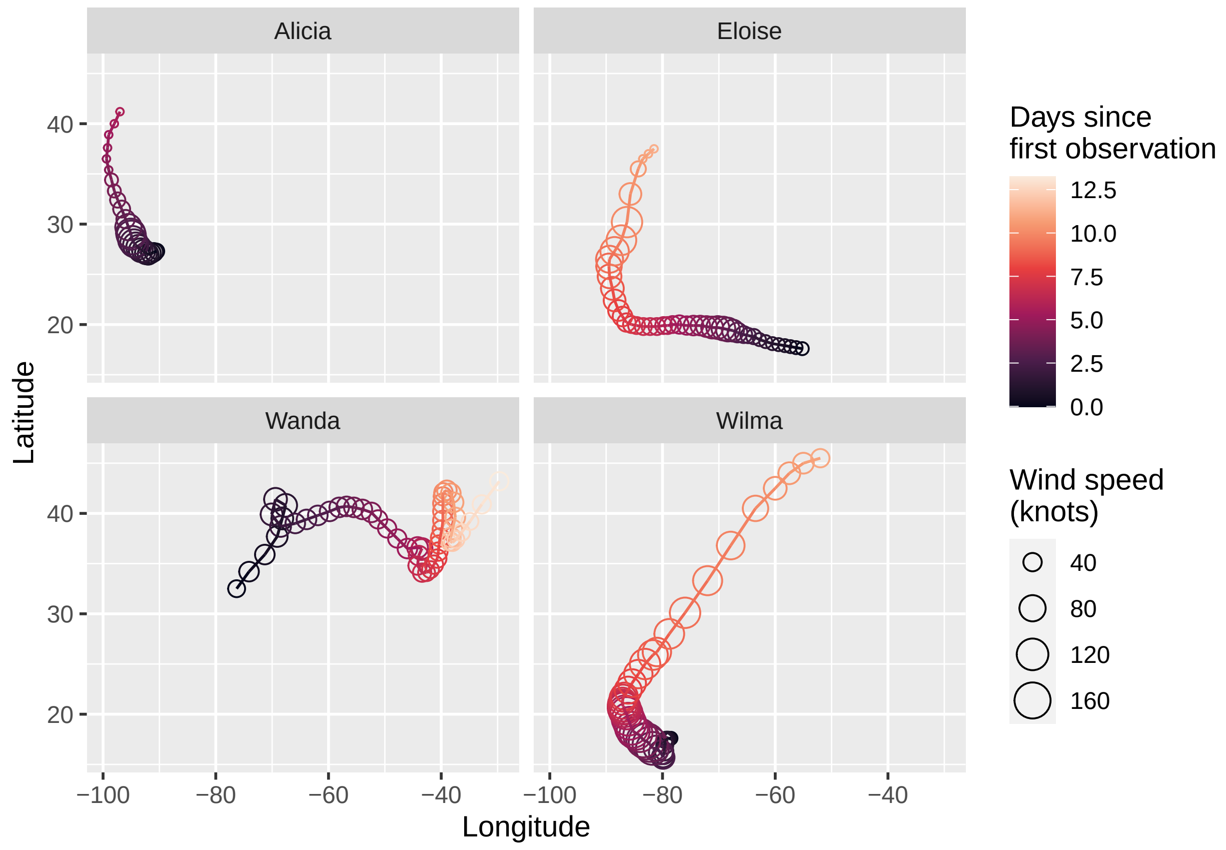
```
iris |>
  mutate(Sepal.Area=Sepal.Length*Sepal.Width) |>
  ggplot(aes(Petal.Length, Petal.Width, colour=Sepal.Width)) +
  geom_point() +
  scale_colour_viridis_c() +
  facet_wrap(~Species)
```



```
iris |>
  mutate(Sepal.Area=Sepal.Length*Sepal.Width) |>
  ggplot(aes(Petal.Length, Petal.Width, colour=Sepal.Width)) +
  geom_point() +
  scale_colour_viridis_c() +
  facet_wrap(~Species, ncol=1)
```



```
storms_df |>
  filter(name %in% c("Wanda", "Wilma", "Eloise", "Alicia")) |>
  ggplot(aes(long, lat, colour=elapsedDays)) +
  geom_point(aes(size=wind), shape=1) +
  geom_path() +
  scale_colour_viridis_c("Days since\nfirst observation", option="rocket") +
  scale_size_continuous("Wind speed\n(knots)") +
  labs(x="Longitude", y="Latitude") +
  facet_wrap(~name)
```


Alicia
Eloise
Wanda
Wilma
Latitude
Longitude
40
30
20
40
30
20
−100
−80
−60
−40
−100
−80
−60
−40
Days since first observation
12.5
10.0
7.5
5.0
2.5
0.0
Wind speed (knots)
40
80
120
160