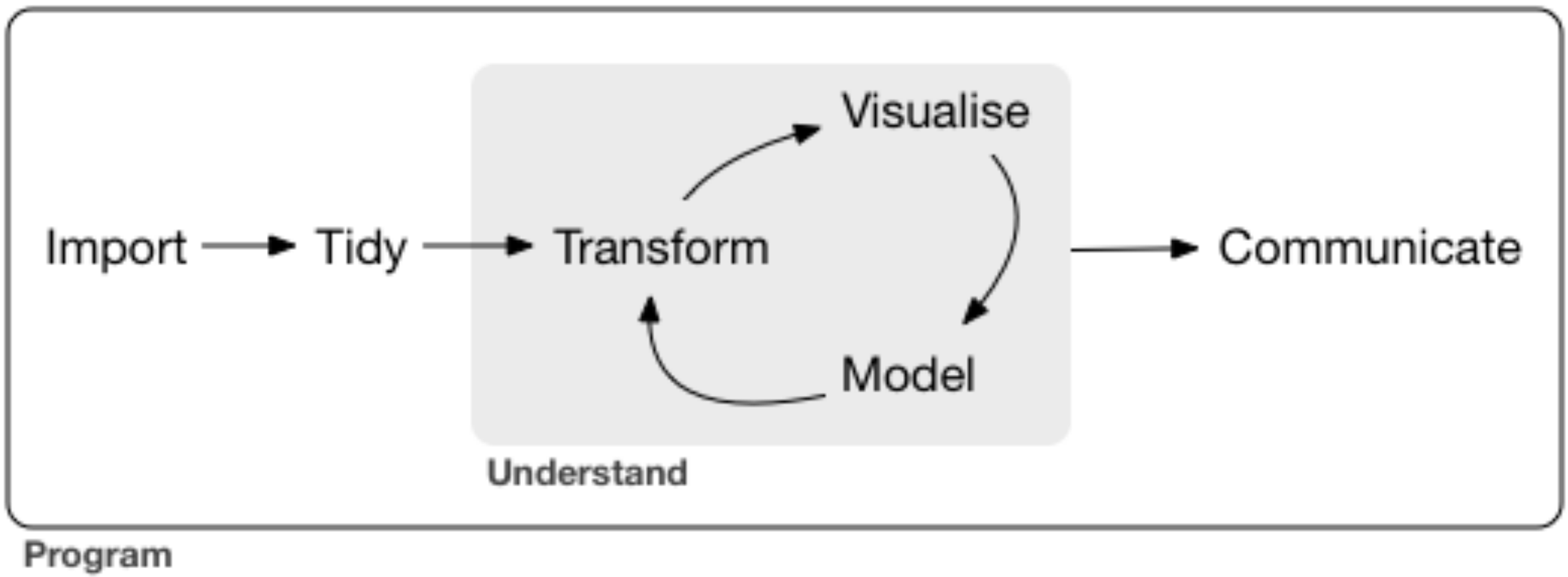
Introduction to R & data

Part II: Modern R with tidyverse

Data science workflow



Tidy data ensures that further processing can be done efficiently, and reproducibly.

Tidy data is easy to manipulate, model, and visualize.

Tidy data

- Each **variable** is a column and contains **values**
- Each **observation** is a row
- Each type of **observational unit** forms a table

Patient	Temp	Med_A	Temp_A	Med_B	Temp_B
122030	37.0	300	37.1	NA	NA
122021	38.2	200	38.1	85	36.5
124500	38.1	300	38.0	NA	NA
126098	39.1	NA	NA	100	36.8

Tidy data

- Each **variable** is a column and contains **values**
- Each **observation** is a row
- Each type of **observational unit** forms a table

values in column names

Patient	Temp	Med_A	Temp_A	Med_B	Temp_B
122030	37.0	300	37.1	NA	NA
122021	38.2	200	38.1	85	36.5
124500	38.1	300	38.0	NA	NA
126098	39.1	NA	NA	100	36.8

Tidy data

- Each **variable** is a column and contains **values**
- Each **observation** is a row
- Each type of **observational unit** forms a table

multiple observations per row

values in column names

Patient	Temp	Med_A	Temp_A	Med_B	Temp_B
122030	37.0	300	37.1	NA	NA
122021	38.2	200	38.1	85	36.5
124500	38.1	300	38.0	NA	NA
126098	39.1	NA	NA	100	36.8

Tidy data

ID no longer unique per row
(but can still be used to collect all info on single patient)

Time series (order) **lost**
(so make sure this is explicit)

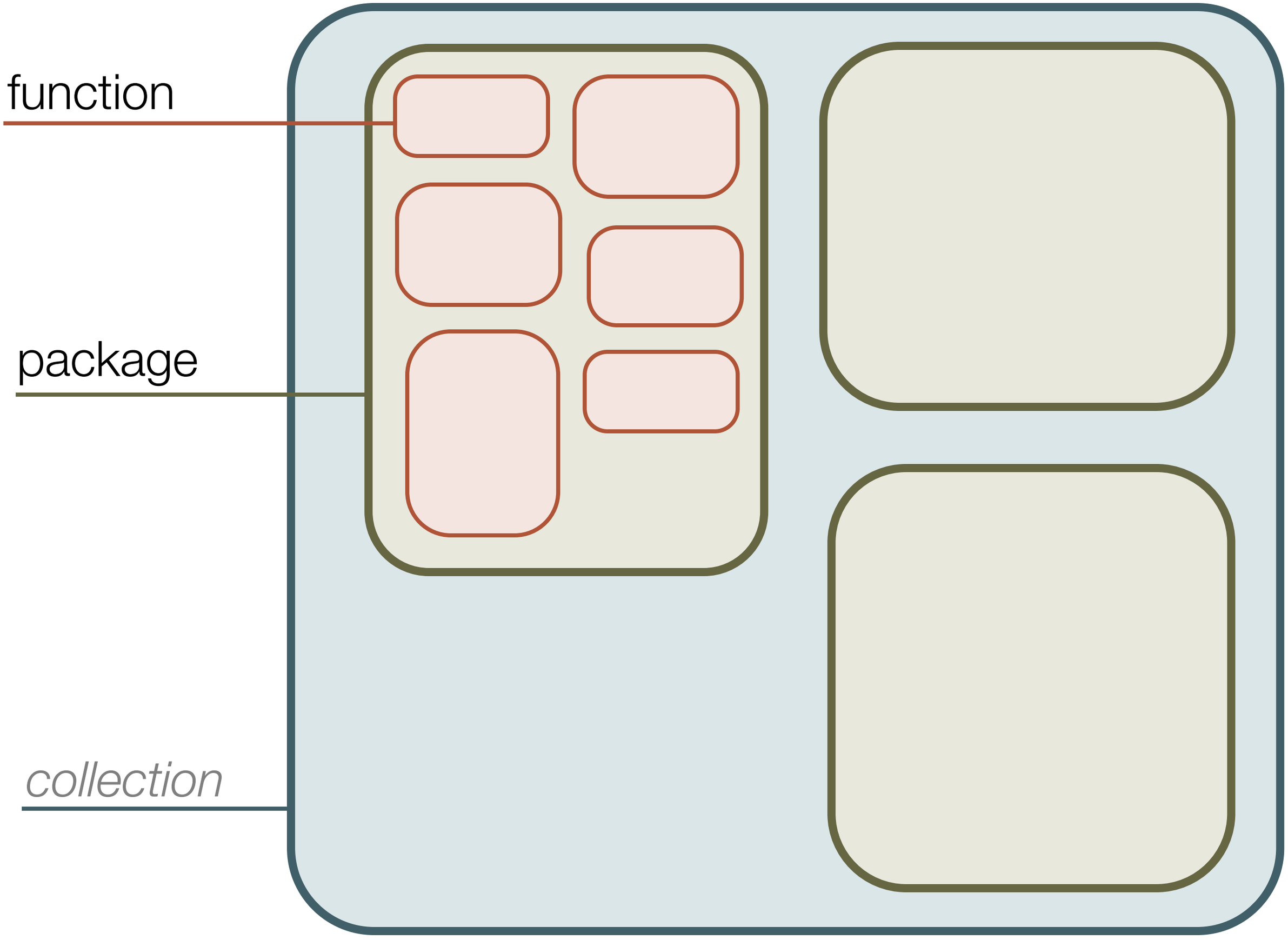
Patient	Temp	Treatment	Dose
122030	37.0	None	NA
122030	37.1	A	300
122021	38.2	None	NA
122021	38.1	A	200
122021	36.5	B	85
124500	38.1	None	NA
124500	38.0	A	300
126098	39.1	None	NA
126098	36.8	B	100

Tidyverse

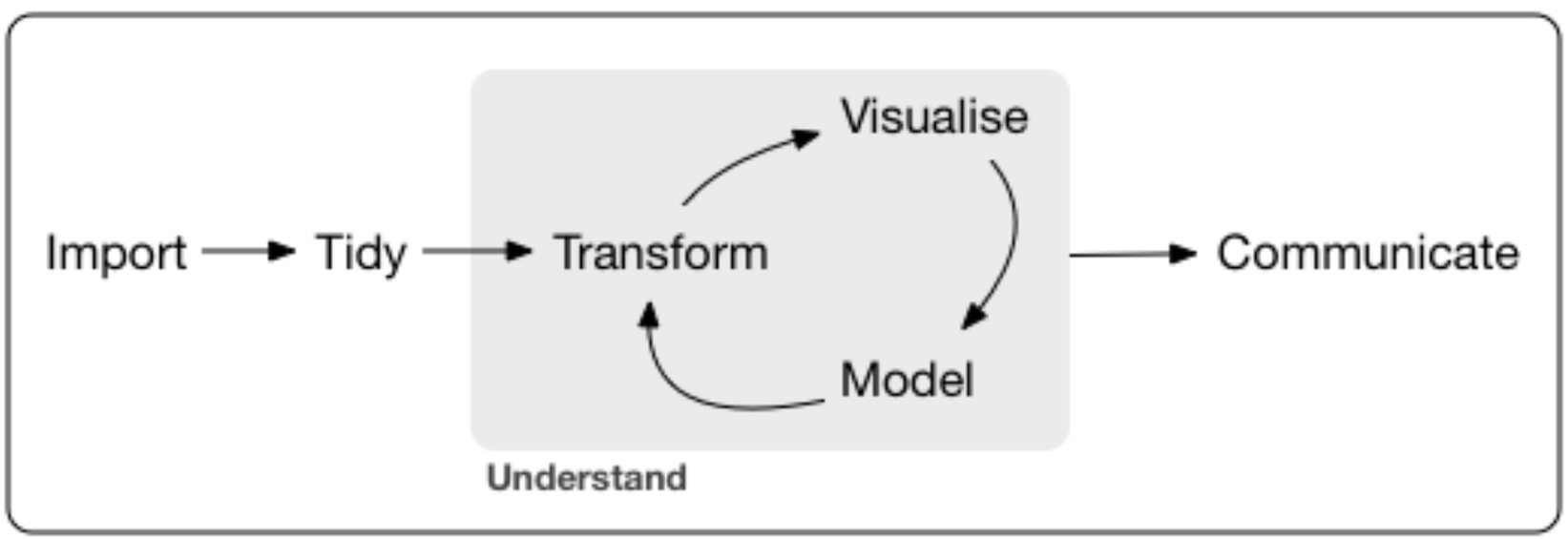
Modern R for data science

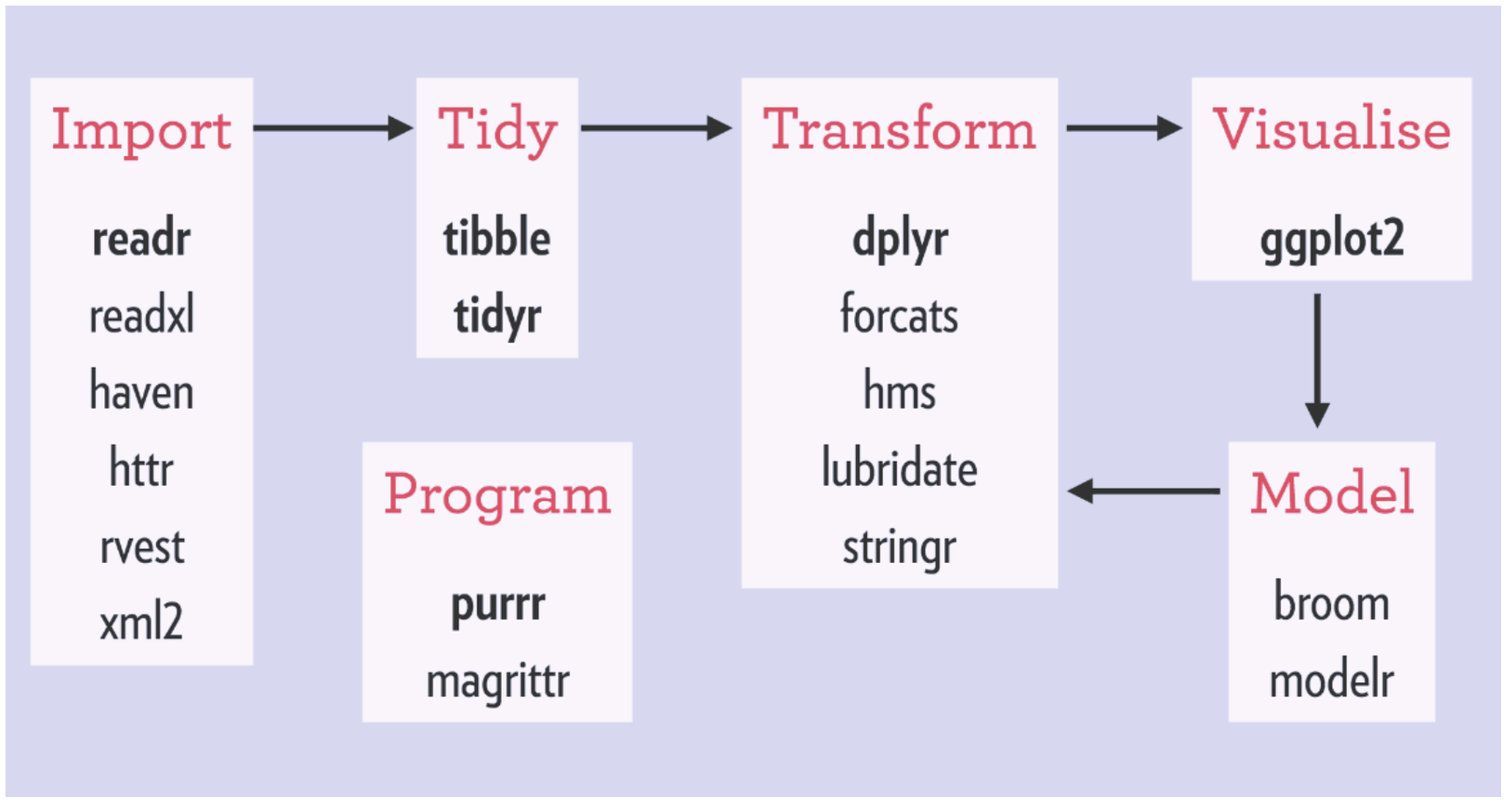
"The tidyverse is an opinionated collection of R packages **designed for data science**. All packages **share an underlying design** philosophy, grammar, and data structures."

– tidyverse.org (2018)


function
package
collection

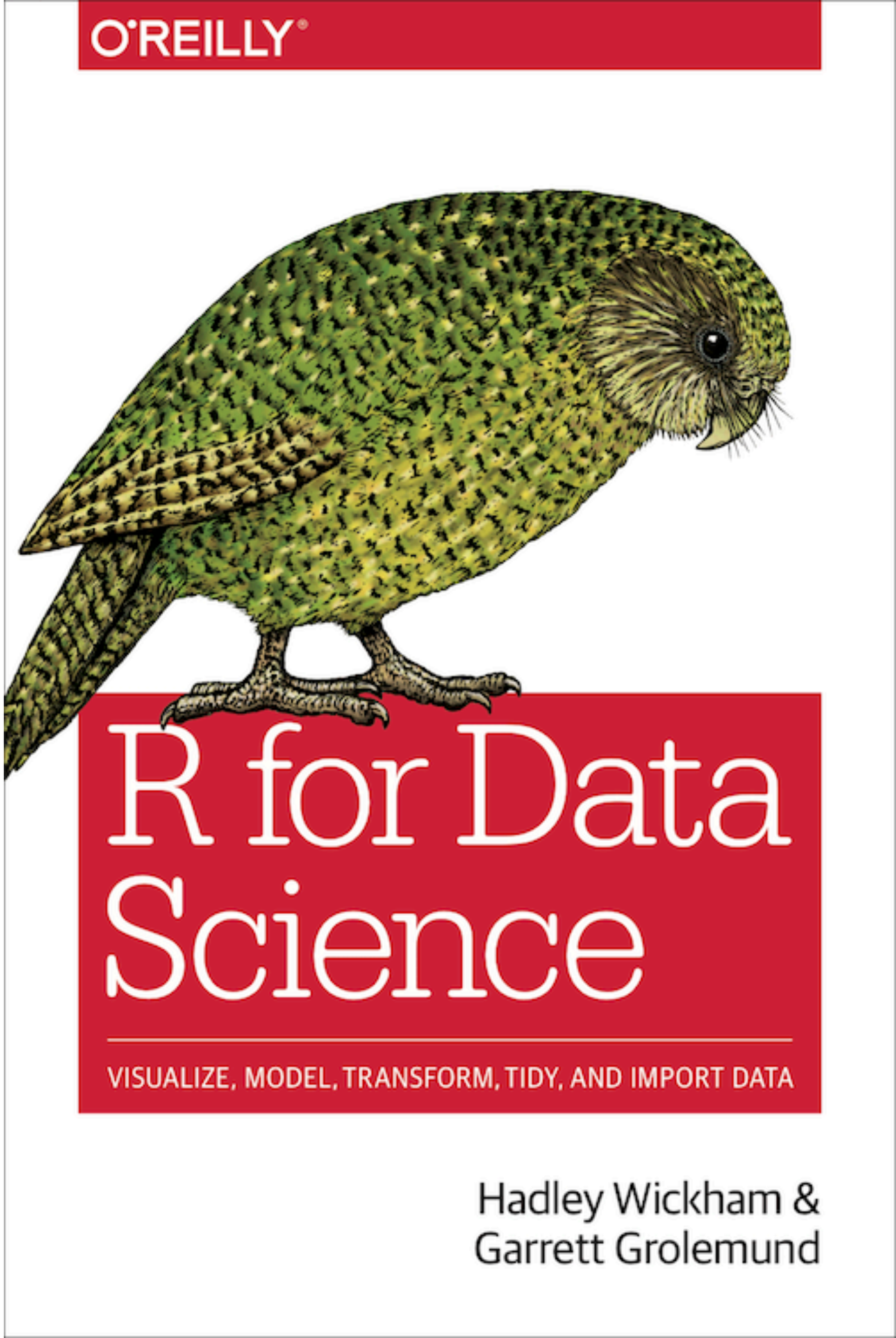
Tidyverse: connecting packages for every step





Learn tidyverse

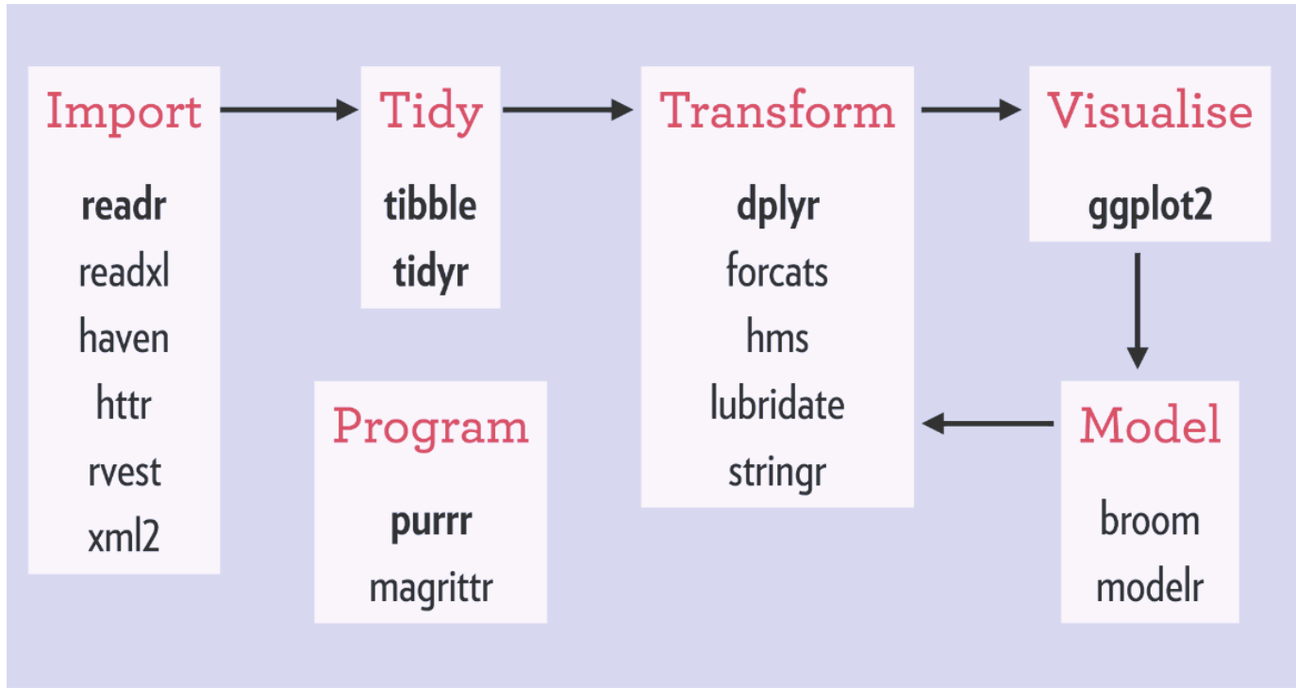
- R for Data Science (book); freely available on [r4ds.had.co.nz/](r4ds.had.co.nz/)
- ggplot2 (book)
- cheatsheets www.rstudio.com/resources/cheatsheets/



Your afternoon roadmap

- A short explanation on screen (~15 minutes)
- Exercises in a document on your computer (~45 minutes)
  - 2-3 basic exercises
  - optional exercises, for further practice
  - recommended reading, for some extra insight into Tidyverse
- We will work with a new data set
- The exercises are presented in Rmarkdown format.

Your afternoon roadmap



- Introduction to **tidyverse**, **Rmarkdown**, and **tibble**.
- Load and save data with **readr**
- [break]
- Data visualisation with **ggplot**
- [break]
- Data transformation with **tidyr** and **dplyr**

Console, Scripts and Rmarkdown

Console	Code execution	-
Script	Code	Extension: .R (example_script.R)
Rmarkdown	Code + Narrative	Extension: .Rmd (example_report.Rmd)

From code to document

- **Combine code with narrative (R with markdown)**

- Turn your analyses into high quality documents, reports, presentations, papers...

- … or your note book.

Rmarkdown: example

Untitled.Rmd

Knit

```
---
title: "Demo"
author: "S. Tudent"
date: "6/12/2019"
output: html_document
---

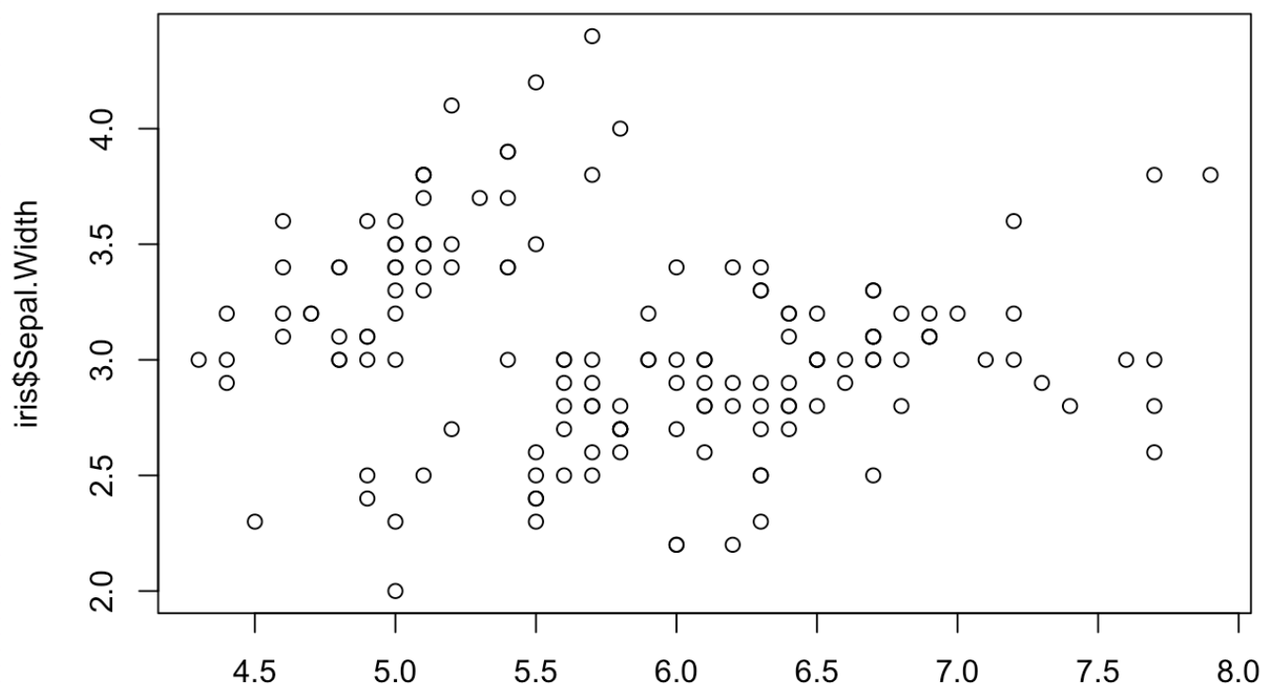
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

Title

Some text about your project

Section

More text about your project

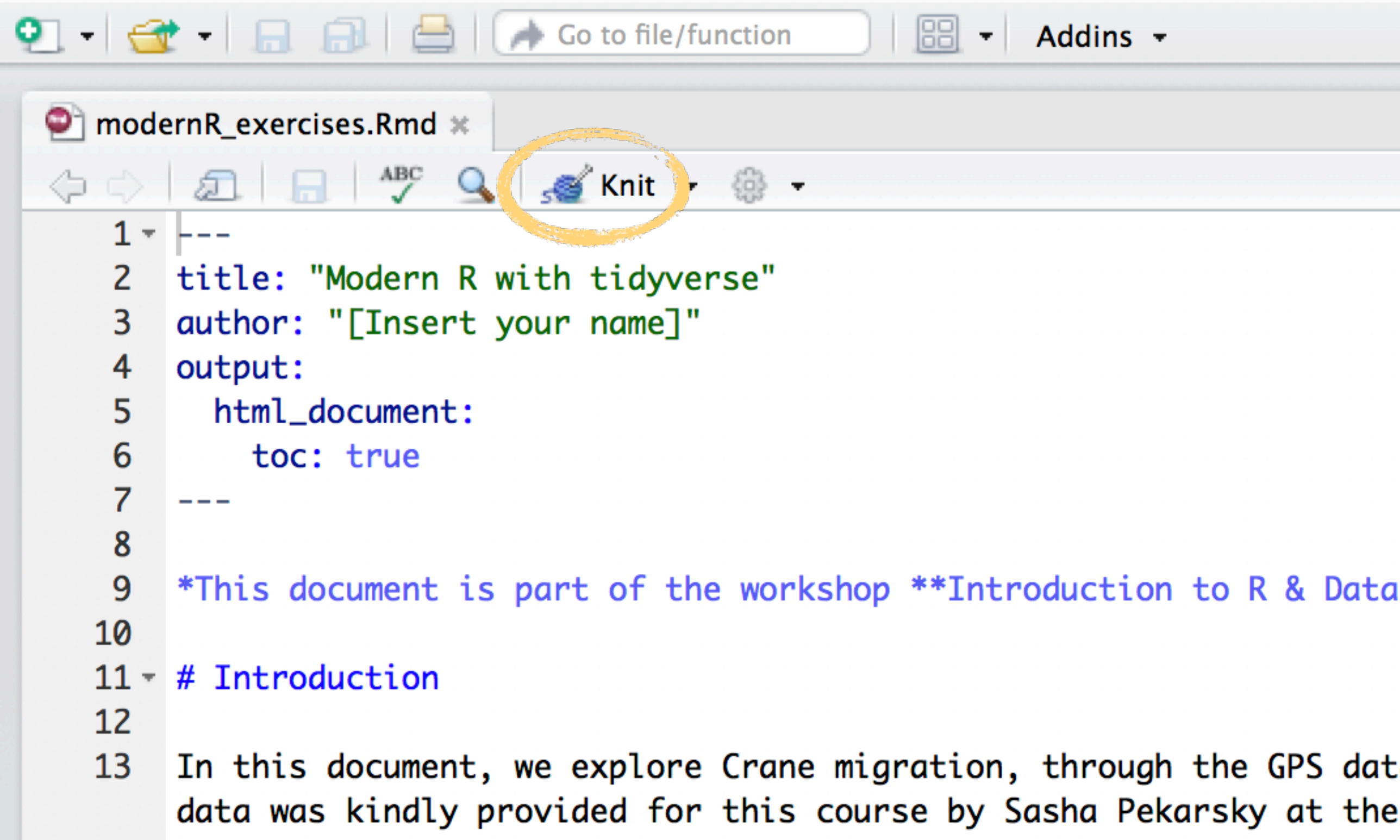
```{r}
max(iris$Petal.Length)
plot(iris$Sepal.Length,iris$Sepal.Width)
```
```

24:1 Section R Markdown

Demo

S. Tudent

6/12/2019

Title

Some text about your project

Section

More text about your project

```
max(iris$Petal.Length)
```

```
## [1] 6.9
```

```
plot(iris$Sepal.Length,iris$Sepal.Width)
```



Exercise: open the exercise Rmarkdown

1. RStudio > File > Open Project > introduction-to-R-and-data-github.Rproj

2. From the ‘files’ menu (bottom right), select modernR_exercises.Rmd

3. Update the header information with your name!

Run the first code chunk!

```
## Technical requirements

This project depends on the `tidyverse` package. Load tidyverse into your work
environment.

```{r}
library(tidyverse)
```
```

```
> library(tidyverse)
── Attaching packages ─────────────────────────────────── tidyverse 1.2.1 ──
   ggplot2 3.1.0       purrr   0.3.0
   tibble  2.0.1       dplyr   0.7.8
   tidyr   0.8.2       stringr 1.4.0
   readr   1.3.1       forcats 0.3.0
── Conflicts ────────────────────────────────────── tidyverse_conflicts() ──
   dplyr::filter() masks stats::filter()
   dplyr::lag()    masks stats::lag()
```

All done? Time to knit!



package: Tibble

"Tibbles are data frames, but they tweak some older behaviours to make life a little easier."

- Hadley Wickham

Tibbles

```
> tibble(a = 1:26, b = letters)
# A tibble: 26 x 2
       a b
   <int> <chr>
 1     1 a
 2     2 b
 3     3 c
 4     4 d
 5     5 e
 6     6 f
 7     7 g
 8     8 h
 9     9 i
10    10 j
# ... with 16 more rows
```

Tibbles: from data.frame to tibble

```
> as_tibble(iris)
# A tibble: 150 x 5
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
          <dbl>       <dbl>        <dbl>       <dbl> <fct>
 1          5.10        3.50         1.40       0.200 setosa
 2          4.90        3.00         1.40       0.200 setosa
 3          4.70        3.20         1.30       0.200 setosa
 4          4.60        3.10         1.50       0.200 setosa
 5          5.00        3.60         1.40       0.200 setosa
 6          5.40        3.90         1.70       0.400 setosa
 7          4.60        3.40         1.40       0.300 setosa
 8          5.00        3.40         1.50       0.200 setosa
 9          4.40        2.90         1.40       0.200 setosa
10          4.90        3.10         1.50       0.100 setosa
# ... with 140 more rows
```

Tibbles: from tibble to data.frame

```
> iris_tibble <- as_tibble(iris)
> as.data.frame(iris_tibble)
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
3            4.7         3.2          1.3         0.2     setosa
4            4.6         3.1          1.5         0.2     setosa
5            5.0         3.6          1.4         0.2     setosa
6            5.4         3.9          1.7         0.4     setosa
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
9            4.4         2.9          1.4         0.2     setosa
10           4.9         3.1          1.5         0.1     setosa
11           5.4         3.7          1.5         0.2     setosa
12           4.8         3.4          1.6         0.2     setosa
13           4.8         3.0          1.4         0.1     setosa
14           4.3         3.0          1.1         0.1     setosa
15           5.8         4.0          1.2         0.2     setosa
16           5.7         4.4          1.5         0.4     setosa
```

Load and save data

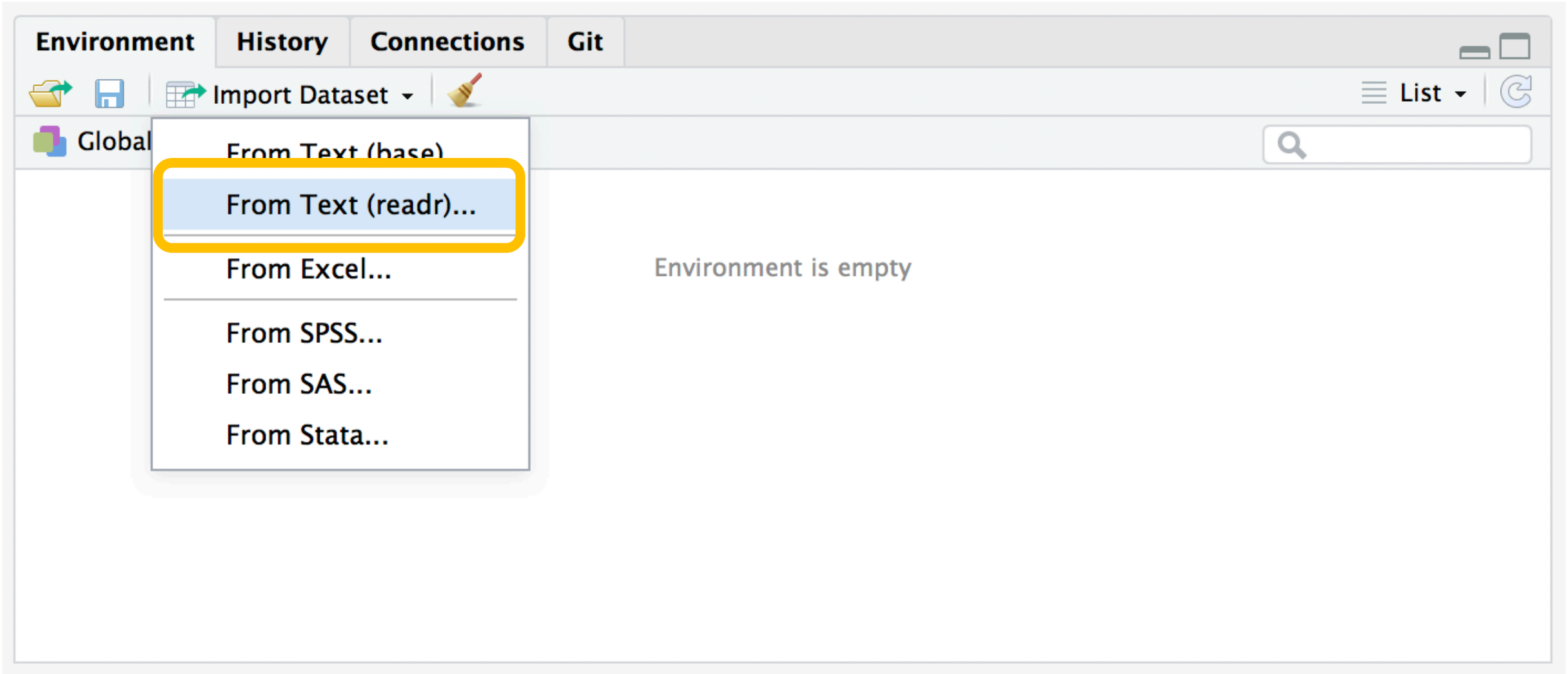
The basics of readr

Flat data files

- Most common format: Comma-separated values (CSV)
- Delimiter-separated values (also stored as CSV)
- Common delimiters:
  - comma ,
  - semicolon ;
  - tab \t

```
mpg;cyl;disp;hp;drat;wt;qsec;v
21;6;160;110;3.9;2.62;16.46;0;
21;6;160;110;3.9;2.875;17.02;0
22.8;4;108;93;3.85;2.32;18.61;
21.4;6;258;110;3.08;3.215;19.4
18.7;8;360;175;3.15;3.44;17.02
18.1;6;225;105;2.76;3.46;20.22
14.3;8;360;245;3.21;3.57;15.84
24.4;4;146.7;62;3.69;3.19;20;1
22.8;4;140.8;95;3.92;3.15;22.9
19.2;6;167.6;123;3.92;3.44;18.
17.8;6;167.6;123;3.92;3.44;18.
16.4;8;275.8;180;3.07;4.07;17.
17.3;8;275.8;180;3.07;3.73;17.
15.2;8;275.8;180;3.07;3.78;18;
10.4;8;472;205;2.93;5.25;17.98
10.4;8;460;215;3;5.424;17.82;0
14.7;8;440;230;3.23;5.345;17.4
32.4;4;78.7;66;4.08;2.2;19.47;
30.4;4;75.7;52;4.93;1.615;18.5
33.9;4;71.1;65;4.22;1.835;19.9
21.5;4;120.1;97;3.7;2.465;20.0
```

Load data by using buttons



Load data by using buttons

Import Text Data

File/Url:

~/surfdrive/Projects/presentations/workshops/introduction-to-R-and-data-github/data/iris.csv

Browse...

Data Preview:

Sepal.Length *(double)*	Sepal.Width *(double)*	Petal.Length *(double)*	Petal.Width *(double)*	Species *(character)*
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa

Previewing first 50 entries.

Import Options:

Name: iris

Skip: 0

First Row as Names

Trim Spaces

Open Data Viewer

Delimiter: Comma

Quotes: Default

Locale: Configure...

Escape: None

Comment: Default

NA: Default

Code Preview:

```
library(readr)
iris <- read_csv("~/surfdrive/Projects/presentations/workshops/introduction-to-R-and-data-github/data/iris.csv")
View(iris)
```

Import

Cancel

Read flat files

```
> library(readr)
> data_iris <- read_delim("data/iris.csv", delim=',')
```

Read flat files

```
> library(readr)
> data_iris <- read_delim("data/iris.csv", delim=',')
Parsed with column specification:
cols(
  Sepal.Length = col_double(),
  Sepal.Width = col_double(),
  Petal.Length = col_double(),
  Petal.Width = col_double(),
  Species = col_character()
)
```

Read flat files

```
> library(readr)
> data_iris <- read_csv("data/iris.csv")
Parsed with column specification:
cols(
  Sepal.Length = col_double(),
  Sepal.Width = col_double(),
  Petal.Length = col_double(),
  Petal.Width = col_double(),
  Species = col_character()
)
```

Exercise dataset

- GPS data from 39 tagged individual cranes during fall migration of 2017

- Data: longitude, latitude, speed, tracker status…

- Courtesy of Sasha Pekarsky at the Hebrew University of Jerusalem, Israel

- In the ‘data’ folder of your course materials.

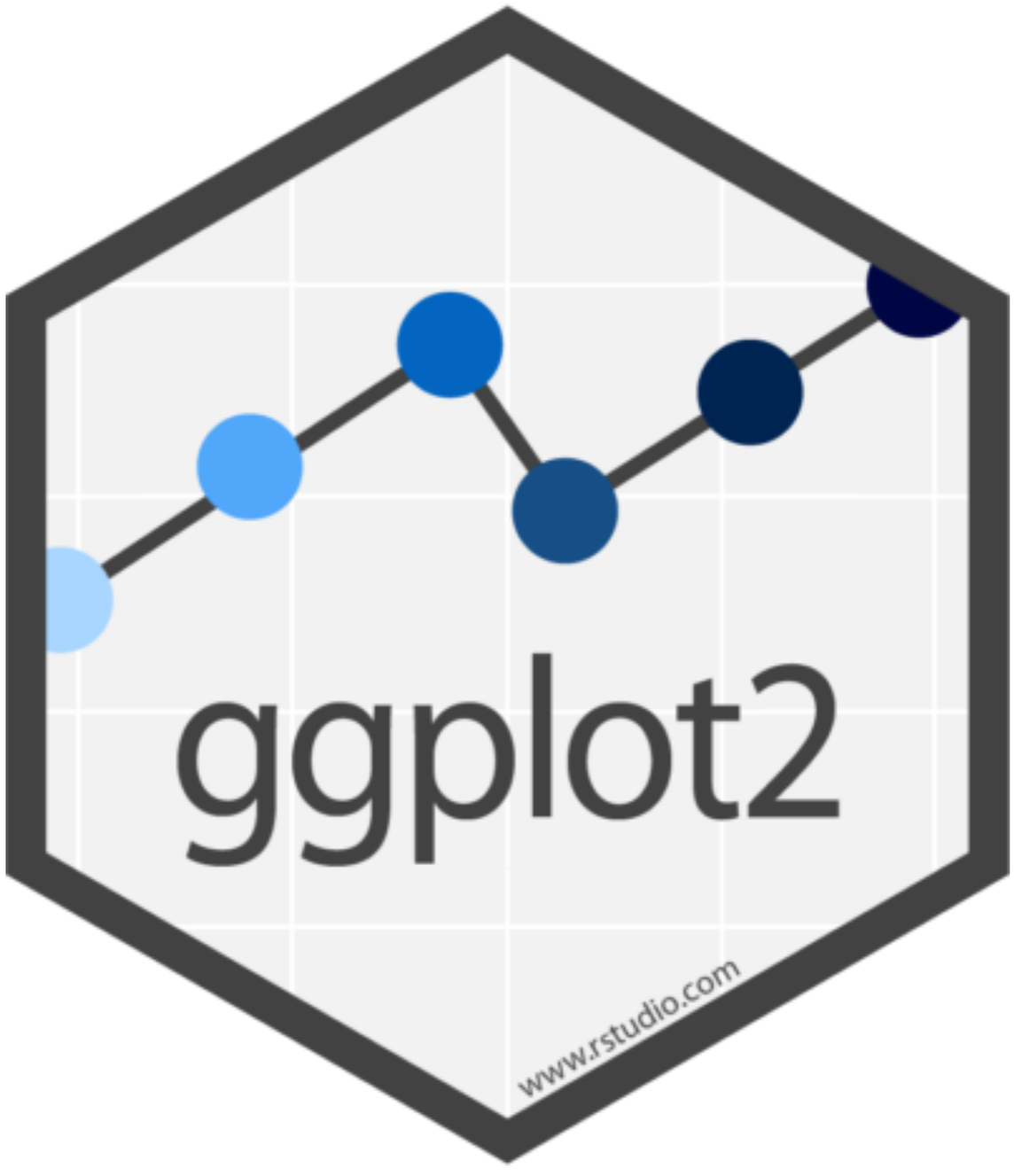
Exercises

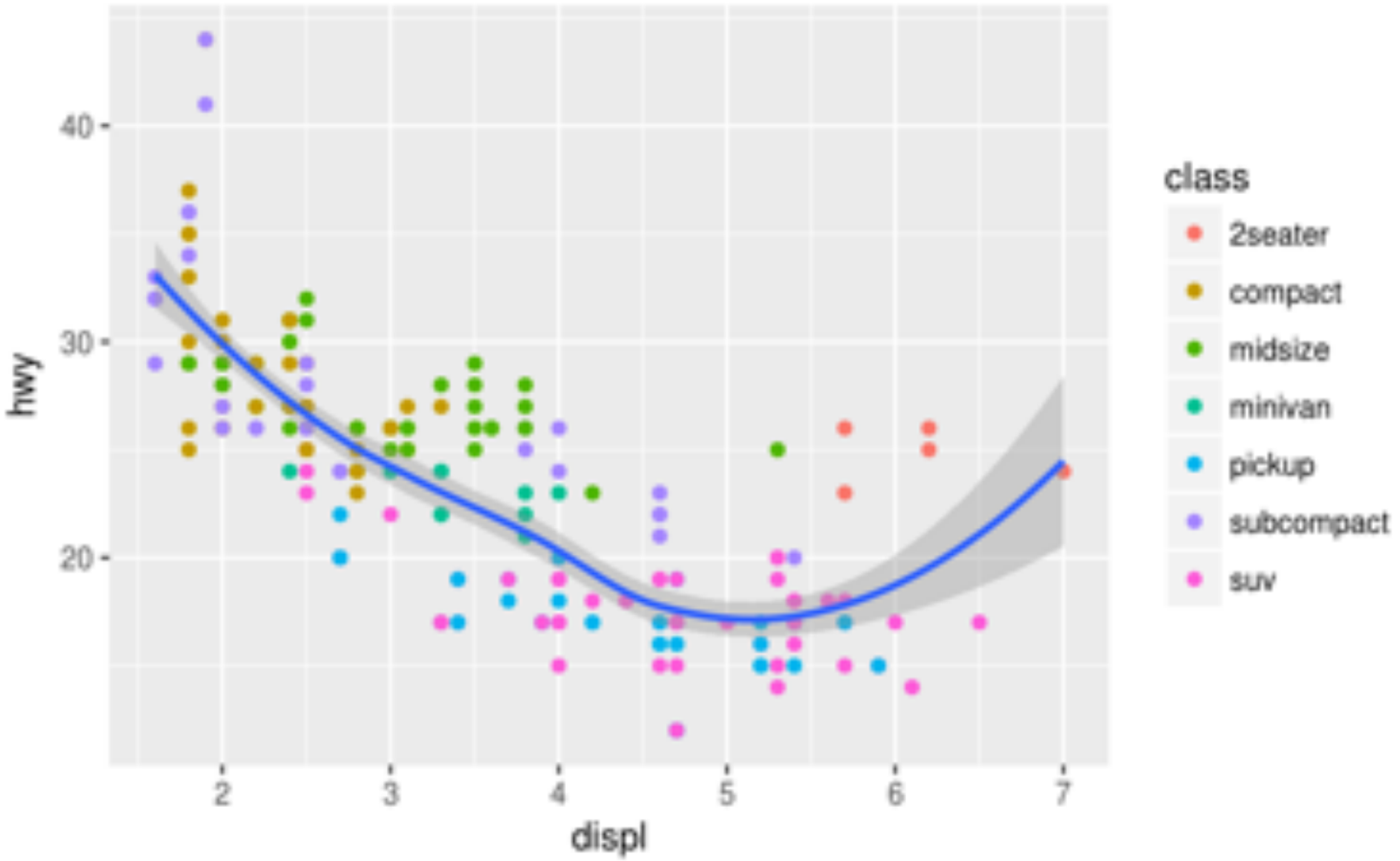
Please do basic exercises 1-I and 1-II.

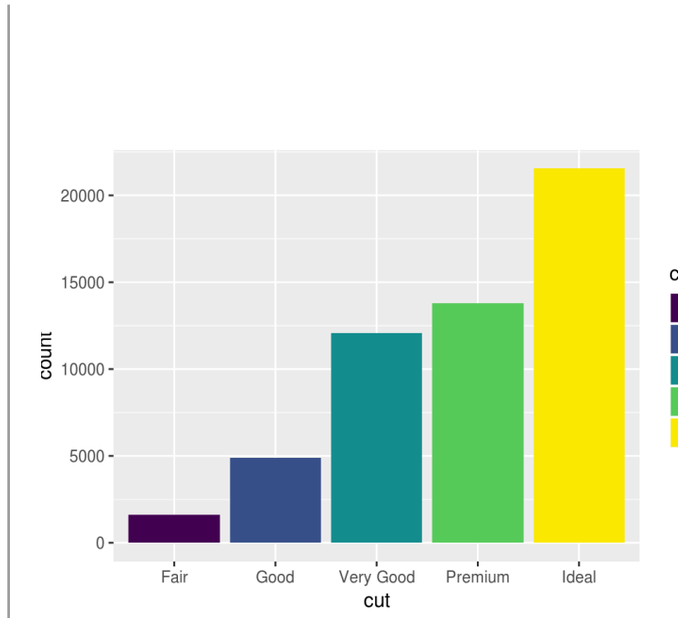
Time left? Opt for an optional exercise, or explore the recommended reading!

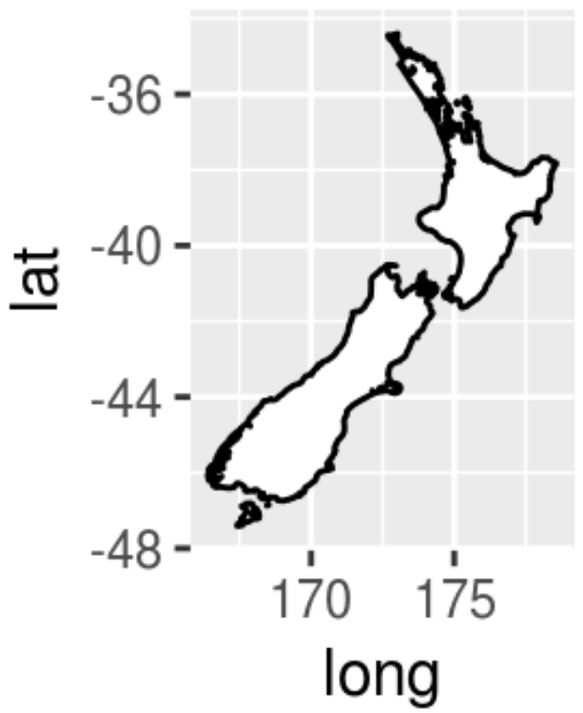
Data visualisation

Create eye candy with ggplot2







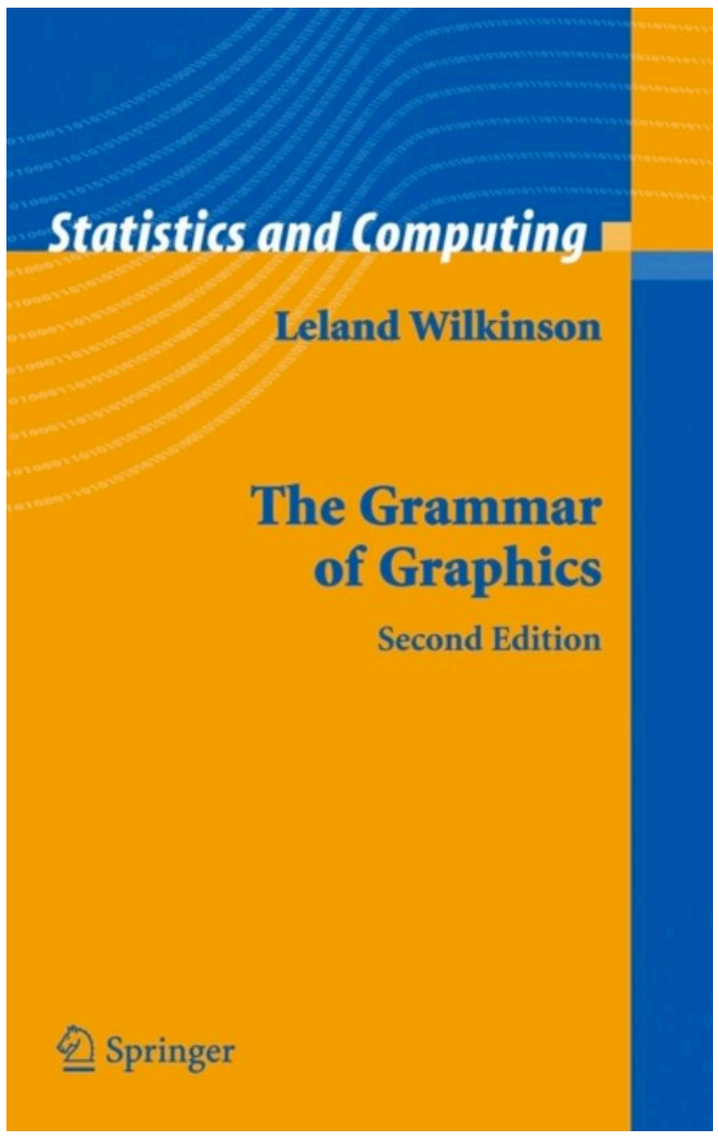


Examples of plots with R (and ggplot2)

Data visualisation with ggplot2

- **ggplot2** is an extremely popular data visualisation package for R
- Simple syntax, easy to learn, nice plots
- Developed and maintained by Hadley Wickham
- Based on the book: The Grammar of Graphics (Wilkinson, 2005)

The grammar of graphics

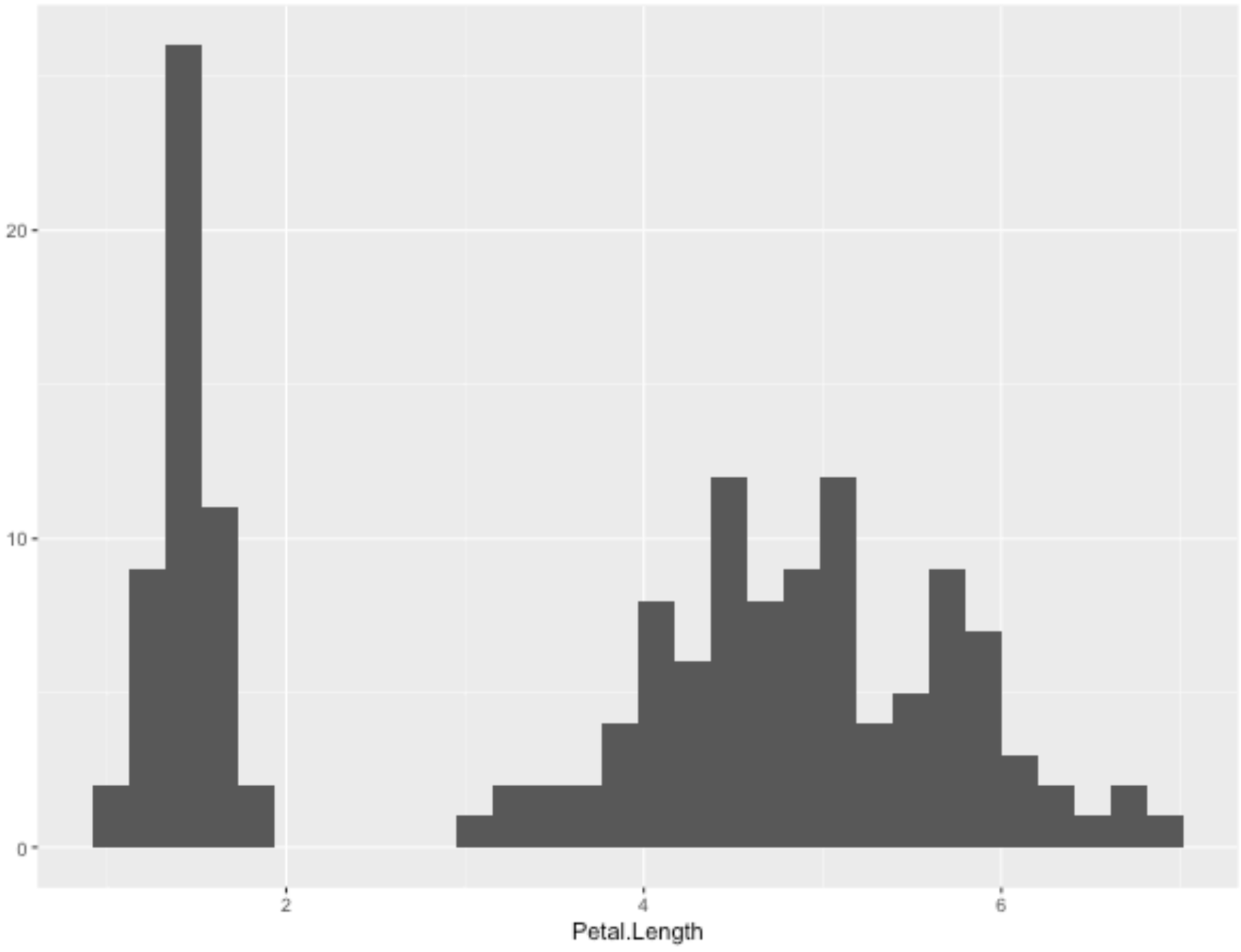


Data	The variables in a tibble or data.frame
Aesthetics	x- and y-axis, colour, size, alpha, shape
Geometries	point, line, bar, histogram

Source: https://www.youtube.com/watch?v=uiTc55clwuA

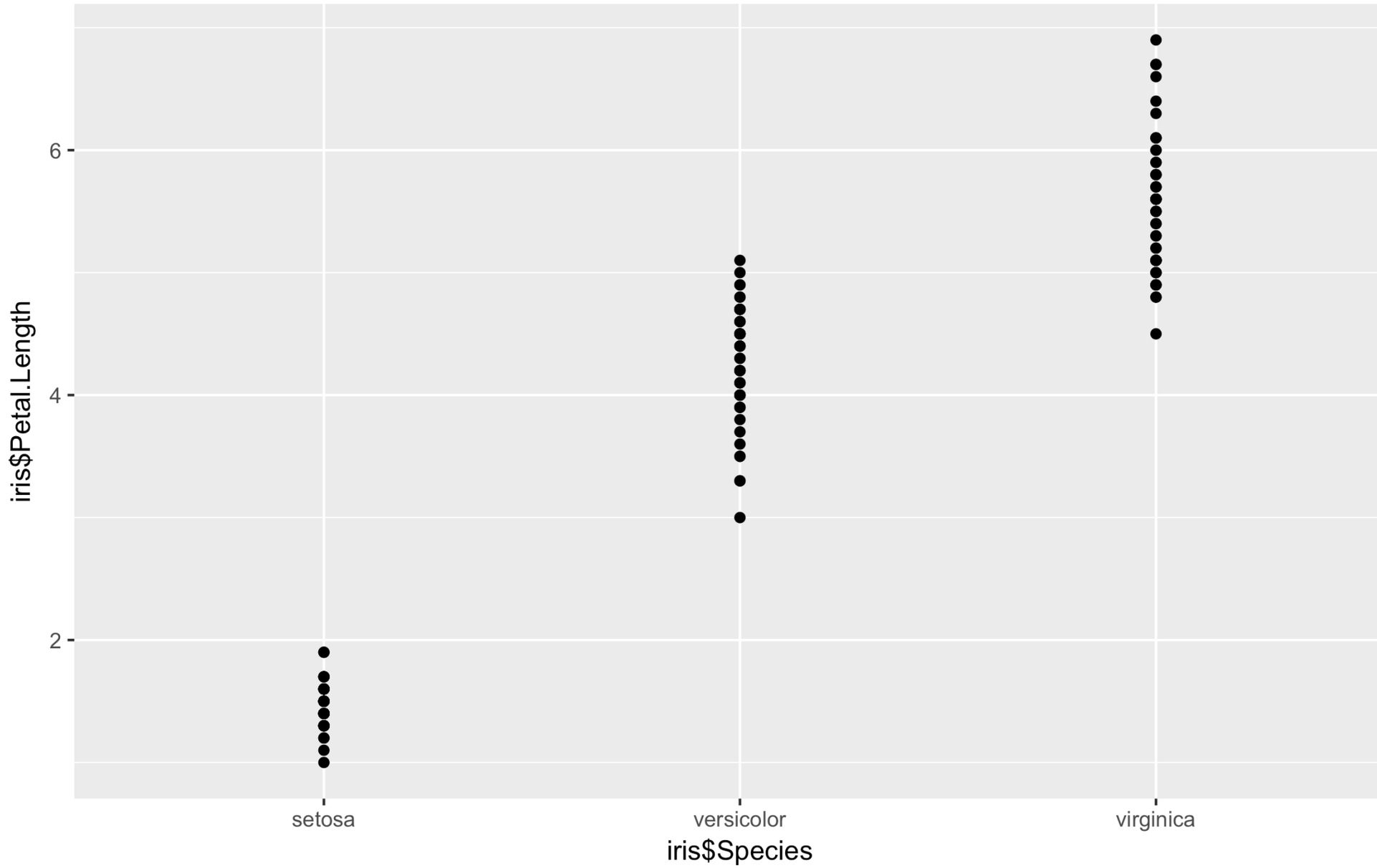
Quick plots

```
> qplot(Petal.Length, data = iris)
```



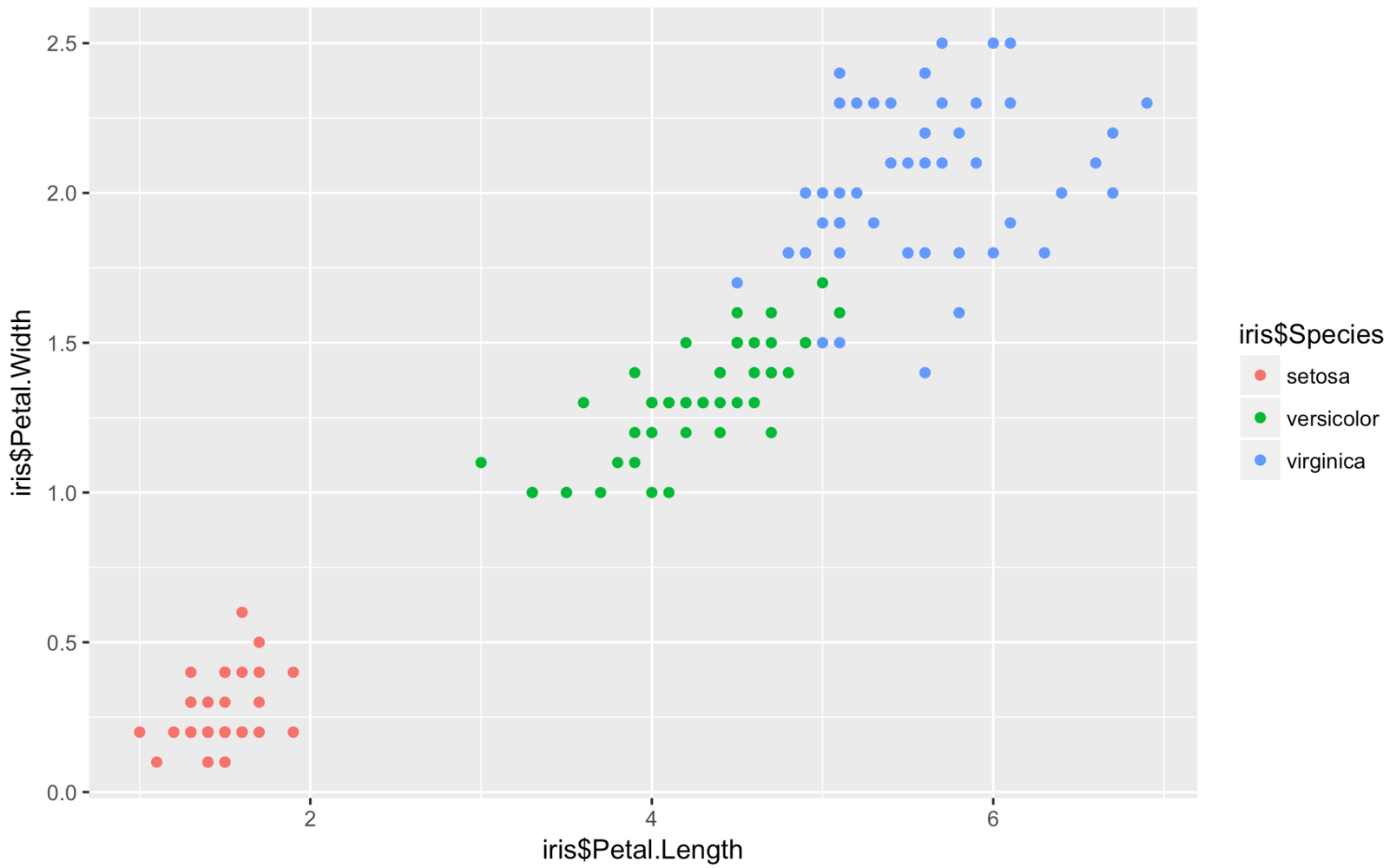
Quick plots

```
> qplot(Petal.Length, data = iris)
> qplot(Species, Petal.Length, data = iris)
```



Quick plots

```
> qplot(Petal.Length, data = iris)
> qplot(Species, Petal.Length, data = iris)
> qplot(Petal.Length, Petal.Width, data = iris, colour=Species)
```



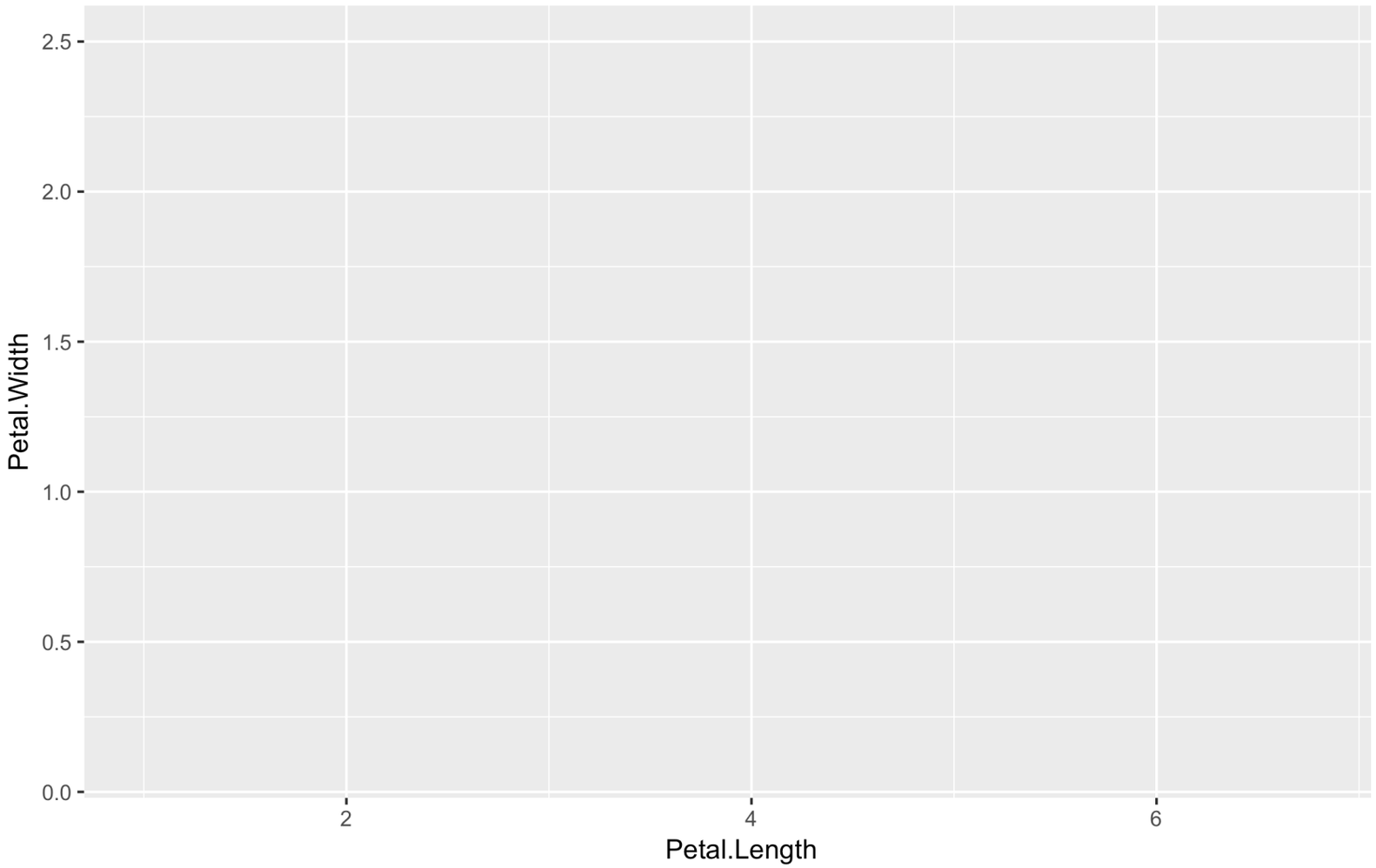
ggplot (grammar of graphics)

ggplot: Data, aesthetics, geometrics

```
> ggplot(iris)
```

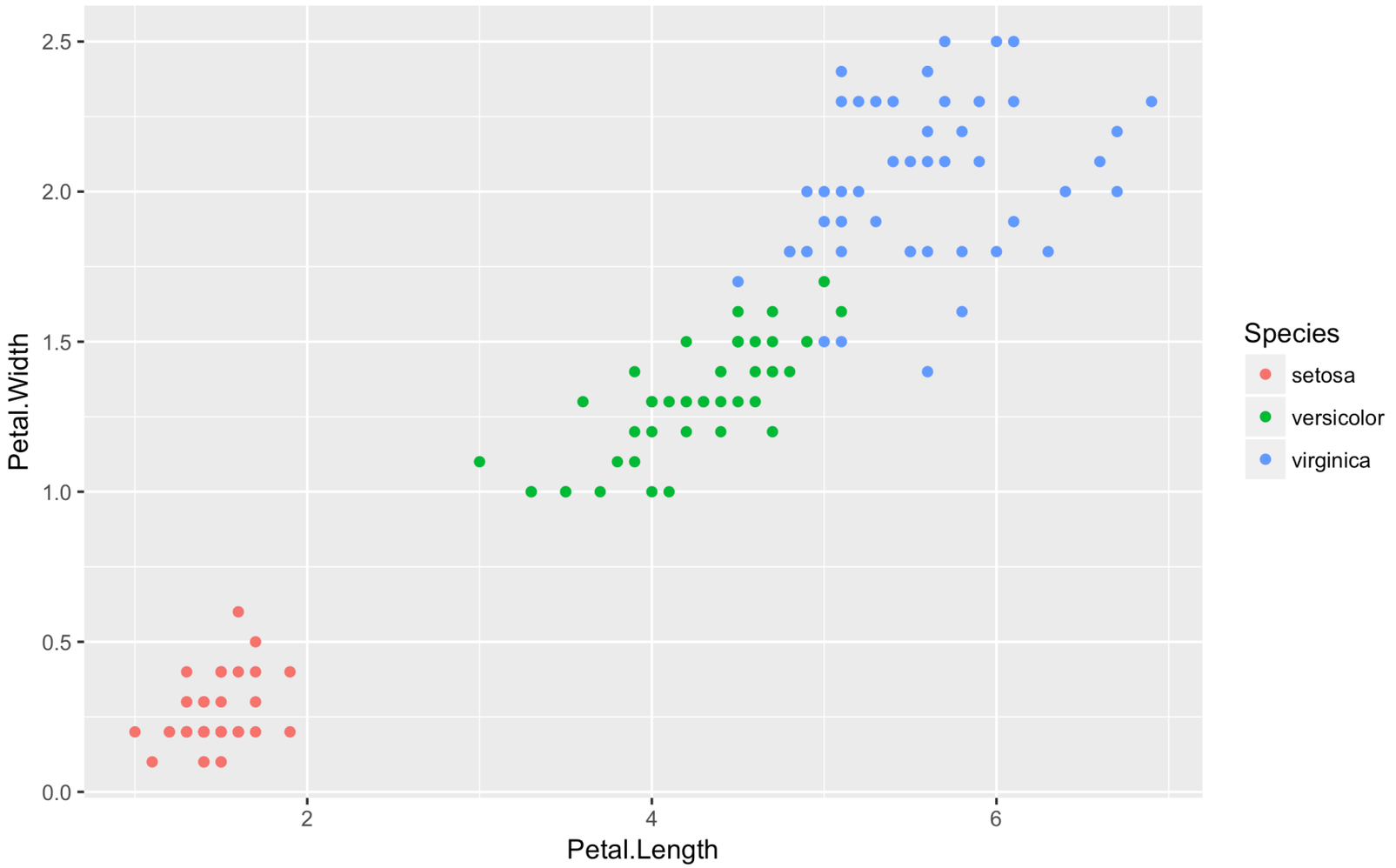
ggplot: Data, aesthetics, geometrics

```
> ggplot(iris)
> ggplot(iris, aes(x = Petal.Length, y = Petal.Width, colour=Species))
```



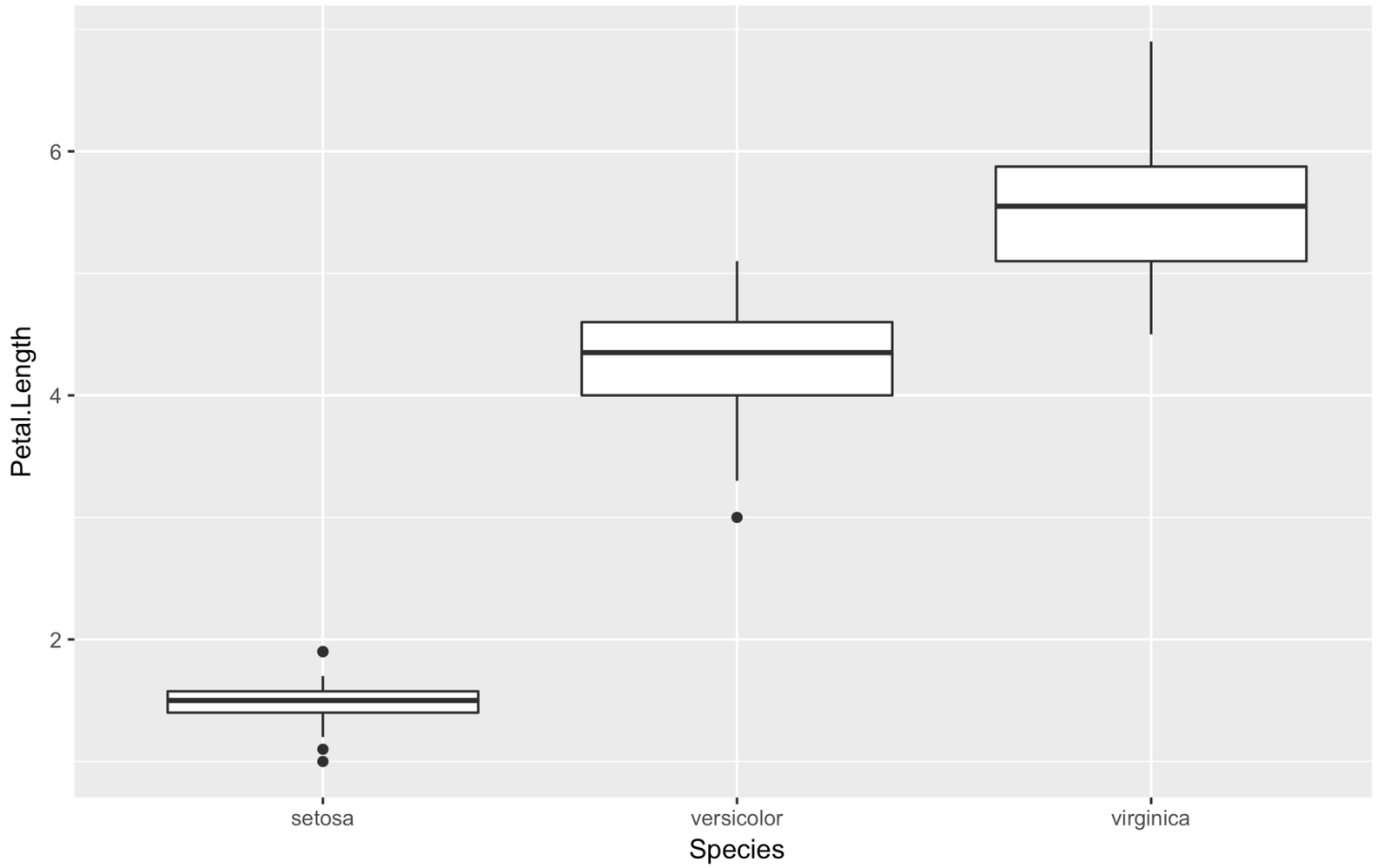
ggplot: Data, aesthetics, geometrics

```
> ggplot(iris)
> ggplot(iris, aes(x = Petal.Length, y = Petal.Width, colour=Species))
> ggplot(iris, aes(x = Petal.Length, y = Petal.Width, colour=Species)) +
    geom_point()
```



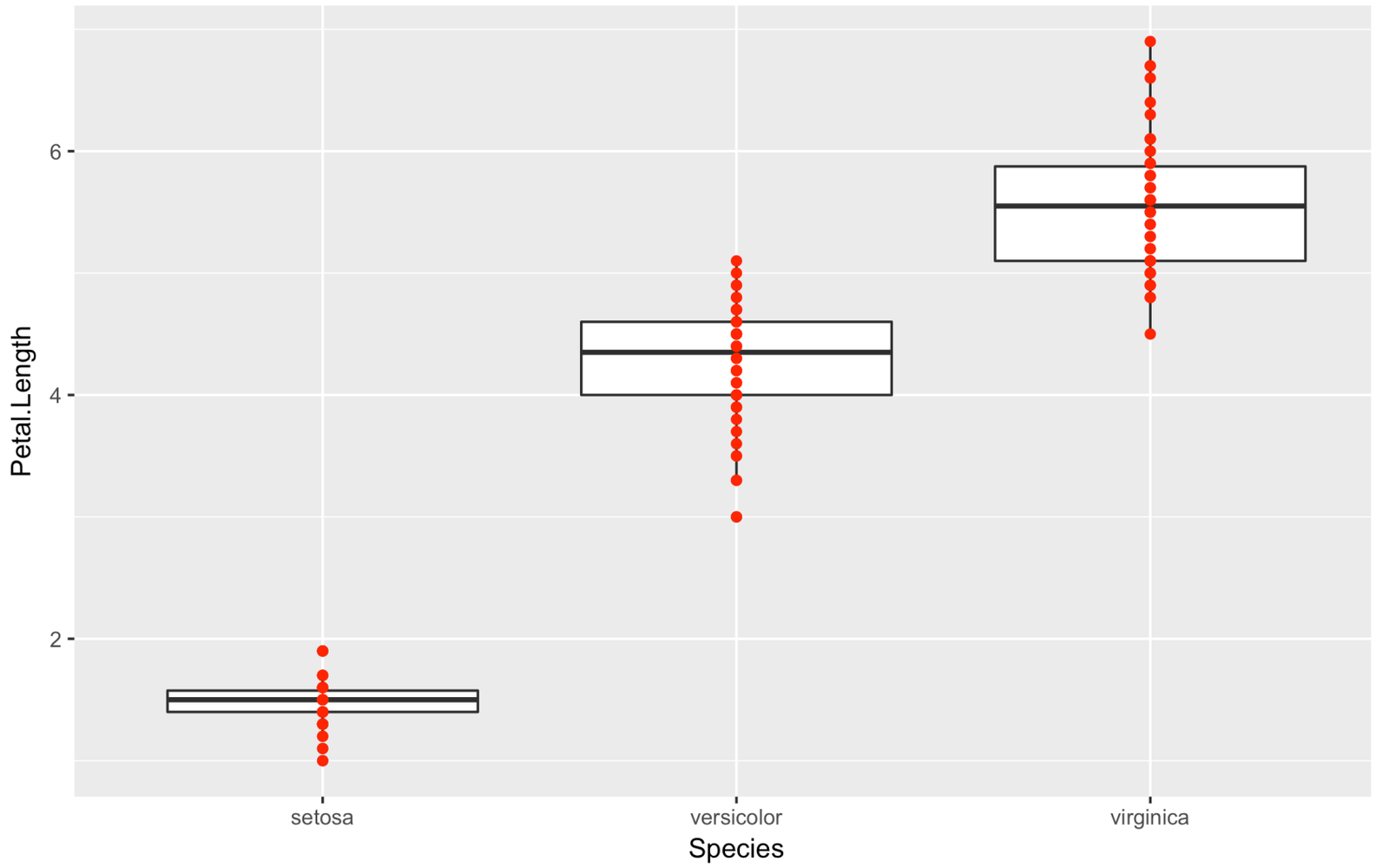
ggplot: Multiple geometric layers

```
> ggplot(iris, aes(x = Species, y = Petal.Length)) +
      geom_boxplot()
```



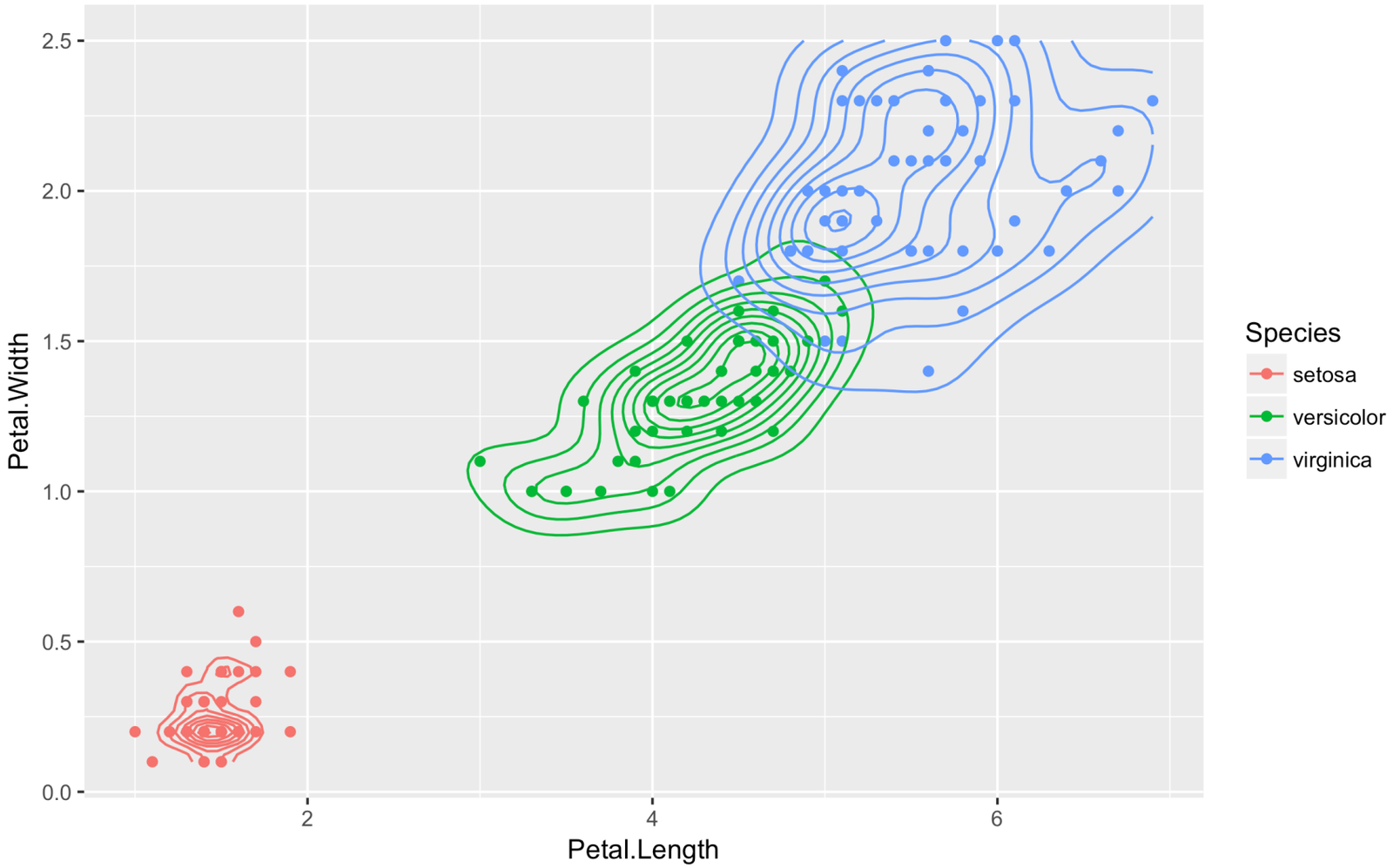
ggplot: Multiple geometric layers

```
> ggplot(iris, aes(x = Species, y = Petal.Length)) +
      geom_boxplot() +
      geom_point(colour='red')
```



ggplot: Statistical layers

```
> ggplot(iris, aes(x = Petal.Length, y = Petal.Width, colour=Species)) +
    geom_point() +
    stat_density_2d()
```



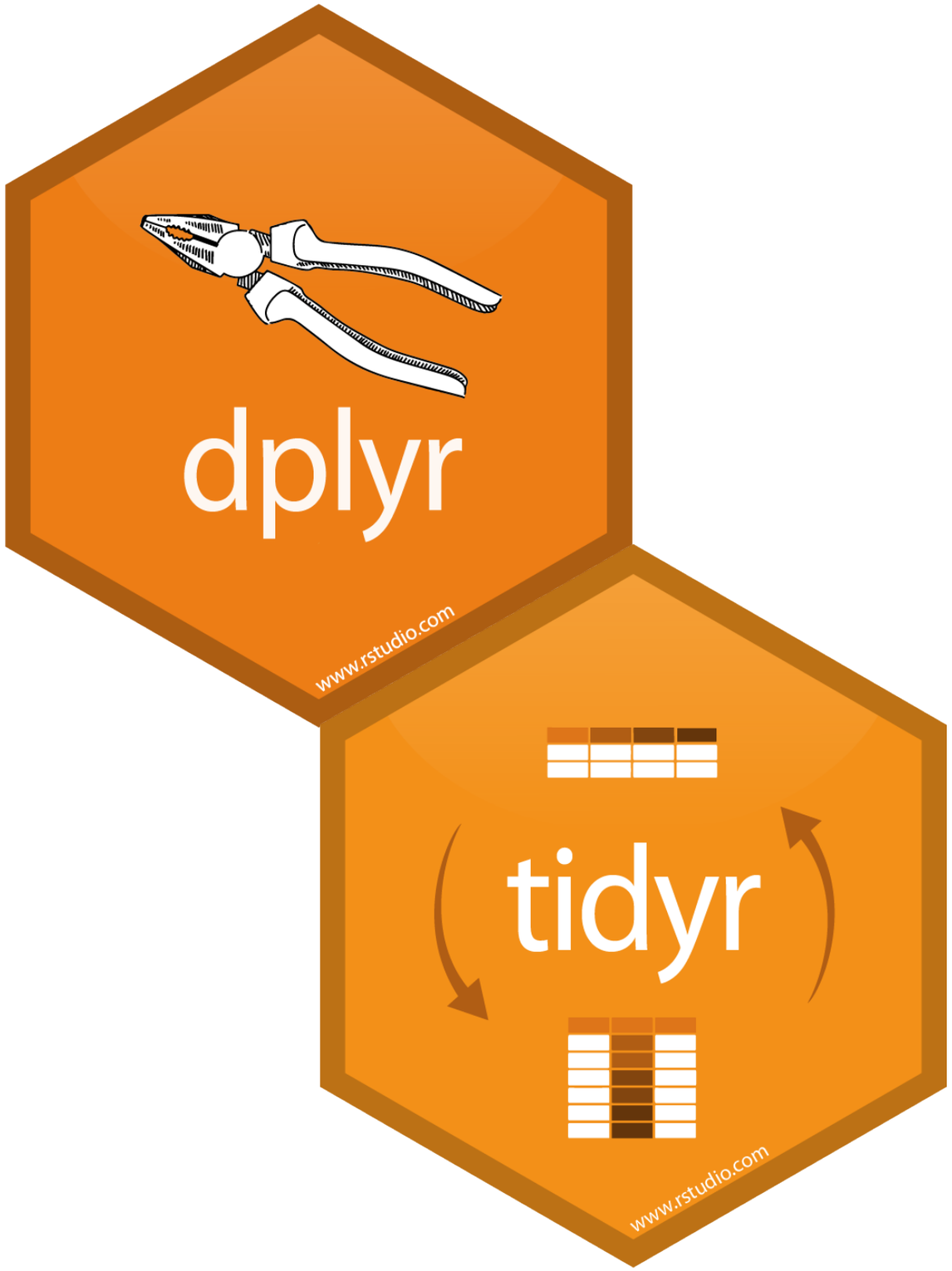
Exercises

Please do basic exercises 2-I and 2-II.

Time left? Opt for an optional exercise, or explore the recommended reading!

Data transformation

Explore the power of dplyr and tidyr



filter()

Function: filter()

```
> filter(iris, Species=="virginica")
```

Function: filter()

```
> filter(iris, Species=="virginica")
# A tibble: 50 x 5
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
          <dbl>       <dbl>        <dbl>       <dbl> <fct>
 1          6.30        3.30         6.00        2.50 virginica
 2          5.80        2.70         5.10        1.90 virginica
 3          7.10        3.00         5.90        2.10 virginica
 4          6.30        2.90         5.60        1.80 virginica
 5          6.50        3.00         5.80        2.20 virginica
 6          7.60        3.00         6.60        2.10 virginica
 7          4.90        2.50         4.50        1.70 virginica
 8          7.30        2.90         6.30        1.80 virginica
 9          6.70        2.50         5.80        1.80 virginica
10          7.20        3.60         6.10        2.50 virginica
# ... with 40 more rows
```

Function: filter()

```
> filter(iris, Species=="virginica", Sepal.Length>= 7.5)
```

Function: filter()

```
> filter(iris, Species=="virginica", Sepal.Length>= 7.5)
# A tibble: 6 x 5
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
         <dbl>       <dbl>        <dbl>       <dbl> <fct>
1          7.60        3.00         6.60        2.10 virginica
2          7.70        3.80         6.70        2.20 virginica
3          7.70        2.60         6.90        2.30 virginica
4          7.70        2.80         6.70        2.00 virginica
5          7.90        3.80         6.40        2.00 virginica
6          7.70        3.00         6.10        2.30 virginica
```

Function: filter()

```
> filter(iris, Species=="virginica", Sepal.Length>= 7.5)
# A tibble: 6 x 5
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
         <dbl>       <dbl>        <dbl>       <dbl> <fct>
1          7.60        3.00         6.60        2.10 virginica
2          7.70        3.80         6.70        2.20 virginica
3          7.70        2.60         6.90        2.30 virginica
4          7.70        2.80         6.70        2.00 virginica
5          7.90        3.80         6.40        2.00 virginica
6          7.70        3.00         6.10        2.30 virginica
```

Same as:

```
> filter(iris, Species=="virginica" & Sepal.Length>= 7.5)
```

select()

Subset Variables (Columns)

Function: select()

```
> select(iris, Sepal.Length, Sepal.Width, Species)
```

Function: select()

```
> select(iris, Sepal.Length, Sepal.Width, Species)
# A tibble: 150 x 3
   Sepal.Length Sepal.Width Species
          <dbl>       <dbl> <fct>
 1          5.10        3.50 setosa
 2          4.90        3.00 setosa
 3          4.70        3.20 setosa
 4          4.60        3.10 setosa
 5          5.00        3.60 setosa
 6          5.40        3.90 setosa
 7          4.60        3.40 setosa
 8          5.00        3.40 setosa
 9          4.40        2.90 setosa
10          4.90        3.10 setosa
# ... with 140 more rows
```

Function: select()

```
> select(iris, Sepal.Length, Sepal.Width, Species)
> select(iris, starts_with("Sepal"), Species)
```

Function: select()

```
> select(iris, Sepal.Length, Sepal.Width, Species)
> select(iris, starts_with("Sepal"), Species)
# A tibble: 150 x 3
   Sepal.Length Sepal.Width Species
          <dbl>       <dbl> <fct>
 1          5.10        3.50 setosa
 2          4.90        3.00 setosa
 3          4.70        3.20 setosa
 4          4.60        3.10 setosa
 5          5.00        3.60 setosa
 6          5.40        3.90 setosa
 7          4.60        3.40 setosa
 8          5.00        3.40 setosa
 9          4.40        2.90 setosa
10          4.90        3.10 setosa
# ... with 140 more rows
```

Function: select()

```
> select(iris, Sepal.Length, Sepal.Width, Species)
> select(iris, starts_with("Sepal"), Species)
> select(iris, -starts_with("Petal"))
```

Function: select()

```
> select(iris, Sepal.Length, Sepal.Width, Species)
> select(iris, starts_with("Sepal"), Species)
> select(iris, -starts_with("Petal"))
# A tibble: 150 x 3
   Sepal.Length Sepal.Width Species
          <dbl>       <dbl> <fct>
 1          5.10        3.50 setosa
 2          4.90        3.00 setosa
 3          4.70        3.20 setosa
 4          4.60        3.10 setosa
 5          5.00        3.60 setosa
 6          5.40        3.90 setosa
 7          4.60        3.40 setosa
 8          5.00        3.40 setosa
 9          4.40        2.90 setosa
10          4.90        3.10 setosa
# ... with 140 more rows
```

Combining functions: the power of piping

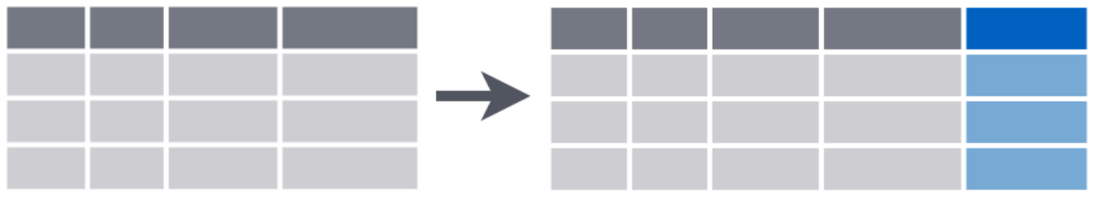
%>%


```
iris %>% filter(Species=="setosa") %>%
   select(starts_with("Sepal"), Species) %>%
   head()
```

```
  Sepal.Length Sepal.Width Species
1          5.1         3.5  setosa
2          4.9         3.0  setosa
3          4.7         3.2  setosa
4          4.6         3.1  setosa
5          5.0         3.6  setosa
6          5.4         3.9  setosa
```

mutate()

Make New Variables



Function: mutate()

```
> mutate(iris,
         petal_area = pi * Petal.Length/2 * Petal.Width/2)
```

Function: mutate()

```
> mutate(iris,
         petal_area = pi * Petal.Length/2 * Petal.Width/2)
# A tibble: 150 x 6
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species petal_area
          <dbl>       <dbl>        <dbl>       <dbl> <fct>        <dbl>
 1          5.1         3.5          1.4         0.2 setosa       0.220
 2          4.9         3            1.4         0.2 setosa       0.220
 3          4.7         3.2          1.3         0.2 setosa       0.204
 4          4.6         3.1          1.5         0.2 setosa       0.236
 5          5           3.6          1.4         0.2 setosa       0.220
 6          5.4         3.9          1.7         0.4 setosa       0.534
 7          4.6         3.4          1.4         0.3 setosa       0.330
 8          5           3.4          1.5         0.2 setosa       0.236
 9          4.4         2.9          1.4         0.2 setosa       0.220
10          4.9         3.1          1.5         0.1 setosa       0.118
# ... with 140 more rows
```

Function: mutate()

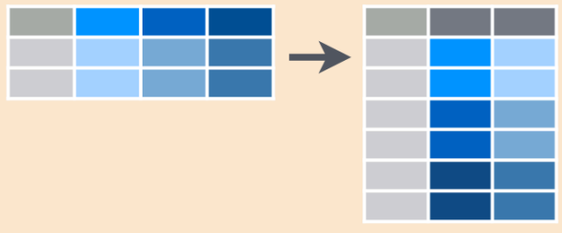
```
> mutate(iris,
         # compute the area of a single petal
         petal_area = pi * Petal.Length/2 * Petal.Width/2,
         # abbreviate the name of the species
         Species_abbr = substring(Species, 1, 3)
)
```

Function: mutate()

```
> mutate(iris,
         # compute the area of a single petal
         petal_area = pi * Petal.Length/2 * Petal.Width/2,
         # abbreviate the name of the species
         Species_abbr = substring(Species, 1, 3)
)
```

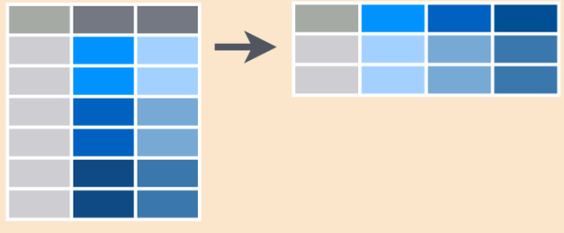
```
# A tibble: 150 x 7
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species petal_area Species_abbr
          <dbl>       <dbl>        <dbl>       <dbl> <fct>        <dbl> <chr>
 1          5.1         3.5          1.4         0.2 setosa       0.220 set
 2          4.9         3            1.4         0.2 setosa       0.220 set
 3          4.7         3.2          1.3         0.2 setosa       0.204 set
 4          4.6         3.1          1.5         0.2 setosa       0.236 set
 5          5           3.6          1.4         0.2 setosa       0.220 set
 6          5.4         3.9          1.7         0.4 setosa       0.534 set
 7          4.6         3.4          1.4         0.3 setosa       0.330 set
 8          5           3.4          1.5         0.2 setosa       0.236 set
 9          4.4         2.9          1.4         0.2 setosa       0.220 set
10          4.9         3.1          1.5         0.1 setosa       0.118 set
# ... with 140 more rows
```

gather() and spread()



tidyr::**gather(cases, "year", "n", 2:4)**

Gather columns into rows.



tidyr::**spread(pollution, size, amount)**

Spread rows into columns.

Wide and long data

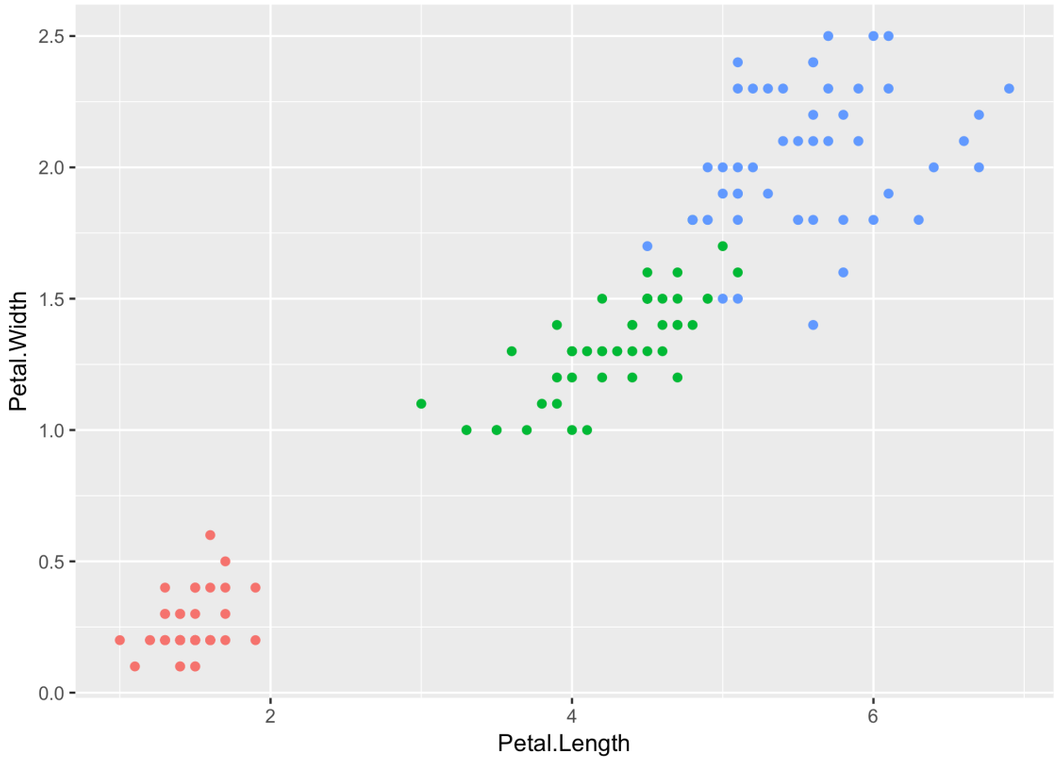
```
# A tibble: 150 x 6
   Species observation Petal.Length Petal.Width Sepal.Length Sepal.Width
   <chr>         <int>        <dbl>       <dbl>        <dbl>       <dbl>
 1 setosa            1         1.40       0.200         5.10        3.50
 2 setosa            2         1.40       0.200         4.90        3.00
 3 setosa            3         1.30       0.200         4.70        3.20
 4 setosa            4         1.50       0.200         4.60        3.10
 5 setosa            5         1.40       0.200         5.00        3.60
 6 setosa            6         1.70       0.400         5.40        3.90
 7 setosa            7         1.40       0.300         4.60        3.40
 8 setosa            8         1.50       0.200         5.00        3.40
 9 setosa            9         1.40       0.200         4.40        2.90
10 setosa           10         1.50       0.100         4.90        3.10
# ... with 140 more rows
```

- More information per row
- Combines all measurements on a single individual
- Necessary to plot matching measurements

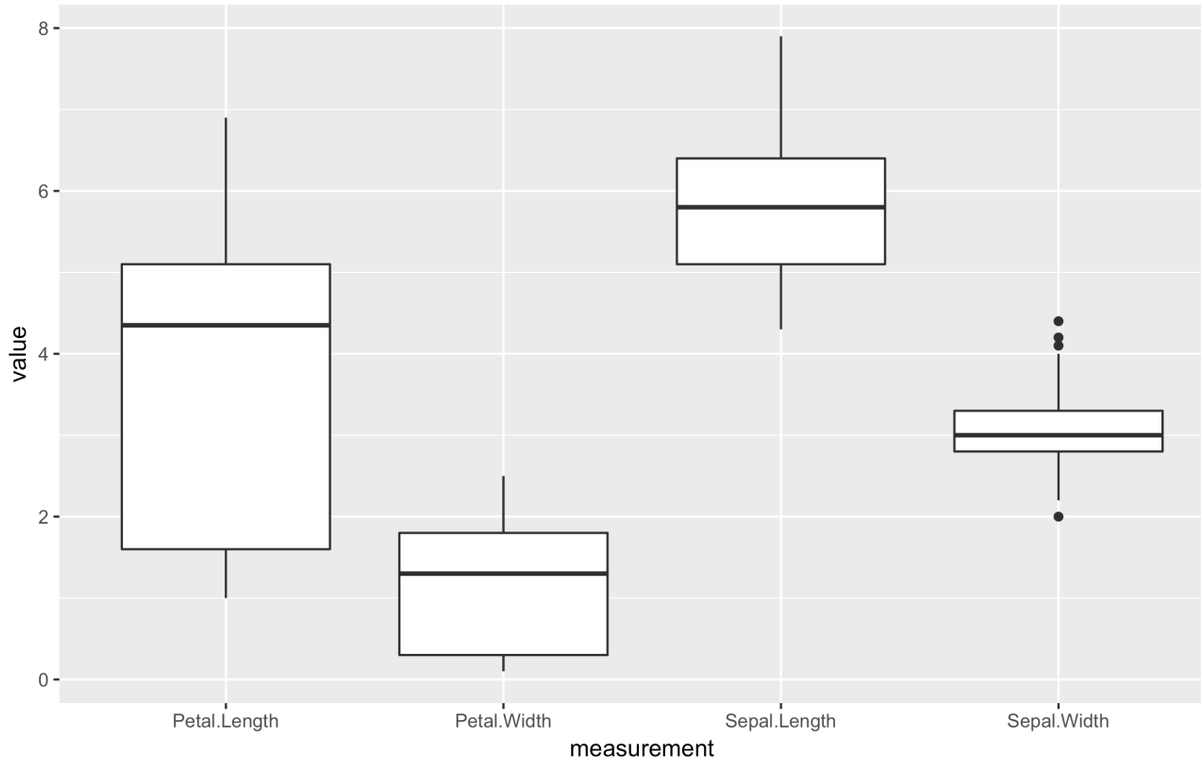
```
# A tibble: 600 x 4
   Species observation measurement  value
   <chr>         <int> <chr>        <dbl>
 1 setosa            1 Sepal.Length 5.10
 2 setosa            1 Sepal.Width  3.50
 3 setosa            1 Petal.Length 1.40
 4 setosa            1 Petal.Width  0.200
 5 setosa            2 Sepal.Length 4.90
 6 setosa            2 Sepal.Width  3.00
 7 setosa            2 Petal.Length 1.40
 8 setosa            2 Petal.Width  0.200
 9 setosa            3 Sepal.Length 4.70
10 setosa            3 Sepal.Width  3.20
# ... with 590 more rows
```

- More information per column
- No values as column headers (tidy)
- Single observation in single row (tidy)
- Necessary to plot large amounts of data in a single plot

Wide vs. Long



- More information per row
- Combines all measurements on a single individual
- **Necessary to plot matching measurements**



- More information per column
- No values as column headers (tidy)
- Single observation in single row (tidy)
- **Necessary to plot large amounts of data in a single plot**

Function: gather()

```
> iris_obs <- mutate(iris, observation = 1:n())
> iris_obs
# A tibble: 150 x 6
```

headers

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species	observation
	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<int>
1	5.10	3.50	1.40	0.200	setosa	1
2	4.90	3.00	1.40	0.200	setosa	2
3	4.70	3.20	1.30	0.200	setosa	3
4	4.60	3.10	1.50	0.200	setosa	4
5	5.00	3.60	1.40	0.200	setosa	5
6	5.40	3.90	1.70	0.400	setosa	6
7	4.60	3.40	1.40	0.300	setosa	7
8	5.00	3.40	1.50	0.200	setosa	8
9	4.40	2.90	1.40	0.200	setosa	9
10	4.90	3.10	1.50	0.100	setosa	10

```
# ... with 140 more rows
```

content

Function: gather()

```
> iris_obs <- mutate(iris, observation = 1:n())
> iris_obs
> iris_long <- gather(iris_obs, measurement, value, Sepal.Length,
Sepal.Width, Petal.Length, Petal.Width)
> iris_long
# A tibble: 600 x 4
   Species observation measurement value
   <chr>         <int> <chr>       <dbl>
 1 setosa            1 Sepal.Length  5.10
 2 setosa            1 Sepal.Width   3.50
 3 setosa            1 Petal.Length  1.40
 4 setosa            1 Petal.Width   0.200
 5 setosa            2 Sepal.Length  4.90
 6 setosa            2 Sepal.Width   3.00
 7 setosa            2 Petal.Length  1.40
 8 setosa            2 Petal.Width   0.200
 9 setosa            3 Sepal.Length  4.70
10 setosa            3 Sepal.Width   3.20
# ... with 590 more rows
```

headers

content

Function: spread()

column to headers

column to be split as **content**

```
> iris_wide <- spread(iris_long, measurement, value)
> iris_wide
# A tibble: 150 x 6
```

Species	observation	Petal.Length	Petal.Width	Sepal.Length	Sepal.Width
<chr>	<int>	<dbl>	<dbl>	<dbl>	<dbl>
1 setosa	1	1.40	0.200	5.10	3.50
2 setosa	2	1.40	0.200	4.90	3.00
3 setosa	3	1.30	0.200	4.70	3.20
4 setosa	4	1.50	0.200	4.60	3.10
5 setosa	5	1.40	0.200	5.00	3.60
6 setosa	6	1.70	0.400	5.40	3.90
7 setosa	7	1.40	0.300	4.60	3.40
8 setosa	8	1.50	0.200	5.00	3.40
9 setosa	9	1.40	0.200	4.40	2.90
10 setosa	10	1.50	0.100	4.90	3.10

```
# ... with 140 more rows
```

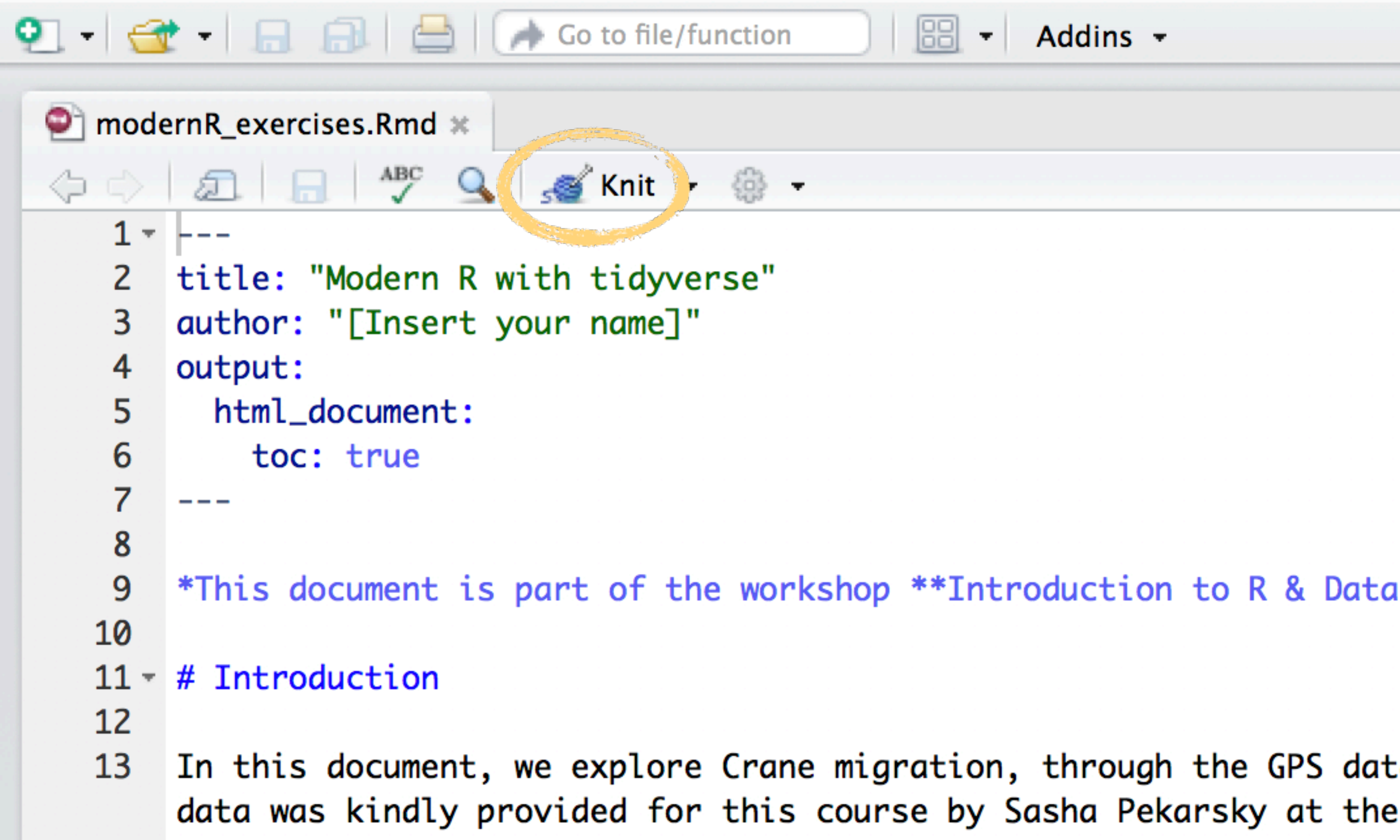
necessary for grouping!

Exercises

Please do basic exercises 3-I and 3-II.

Time left? Opt for an optional exercise, or explore the recommended reading!

All done? Time to knit!



Introduction to R & data

Before you leave…

Other RDM workshops

- uu.nl/rdm > Training & workshops
- Learn to write your Data Management Plan (online course)
- Quickstart to Research Data Management

R Cafe

Monthly event for R programmers at UU/UMCU. Be part of the R community! Work on your own project, and ask (or answer!) questions.

More info at:

tinyurl.com/RcafeUU

or:

github.com/UtrechtUniversity/R-data-cafe

```
useR <- function(){
    print("Good luck and see you!")
}
useR()
```