# Data handling and visualisation in R

Marine Ecosystem Dynamics

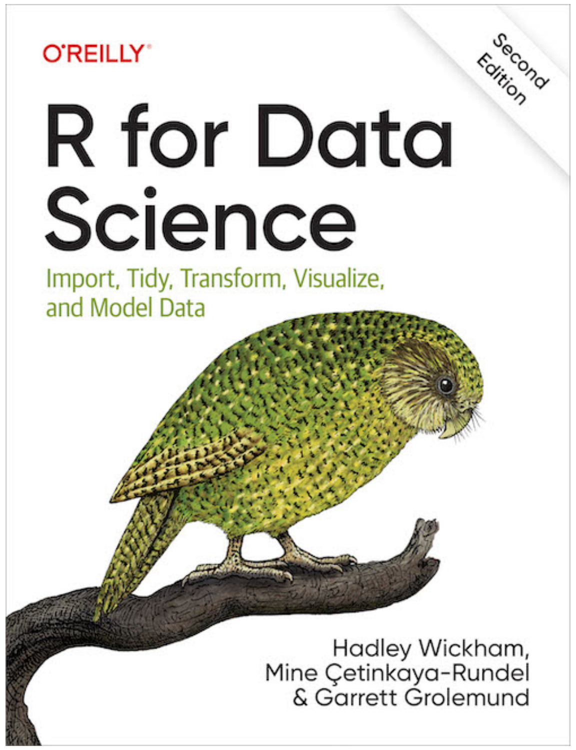
Kinlan M.G. Jan

Stockholm University

# Plan for today’s lecture

- Introduction to `tidyverse`
- Pipe the data using `magrittr`
- Clean the data using `tidyr`
- Arrange the data using `dplyr`
- Plot using `ggplot2`

# Tidyverse


O'REILLY
Second Edition
R for Data Science
Import, Tidy, Transform, Visualize, and Model Data
Hadley Wickham, Mine Çetinkaya-Rundel & Garrett Grolemund

- `tidyverse` is a collection of packages
- It is now a standard in data analysis
- It is easier to read and keep track of what is happening with the pipe operator `%>%`

The book is available online

# Pipe the data using `magrittr`

`%>%` takes the data from the left and place it to the right

- `x %>% function()` = `function(x)`

Without the pipe operator:

```r
iris_subset <- iris[, c("Species", "Sepal.
iris_subset$Sepal_Ratio <- iris_subset$Sep

aggregate(Sepal_Ratio ~ Species, data = ir
                     FUN = function(x) c(Av
                                          st
```

With the pipe operator:

```r
library(magrittr)
iris %>%
  dplyr::select(Species, Sepal.Length, Sep
  dplyr::mutate(Sepal_Ratio = Sepal.Lengtl
  dplyr::group_by(Species)%>%
  dplyr::summarise(Average_ratio = mean(Se
                   standard_deviation = sc
```

R 4.1.0 introduced a native pipe operator, `|>`

Stockholm University

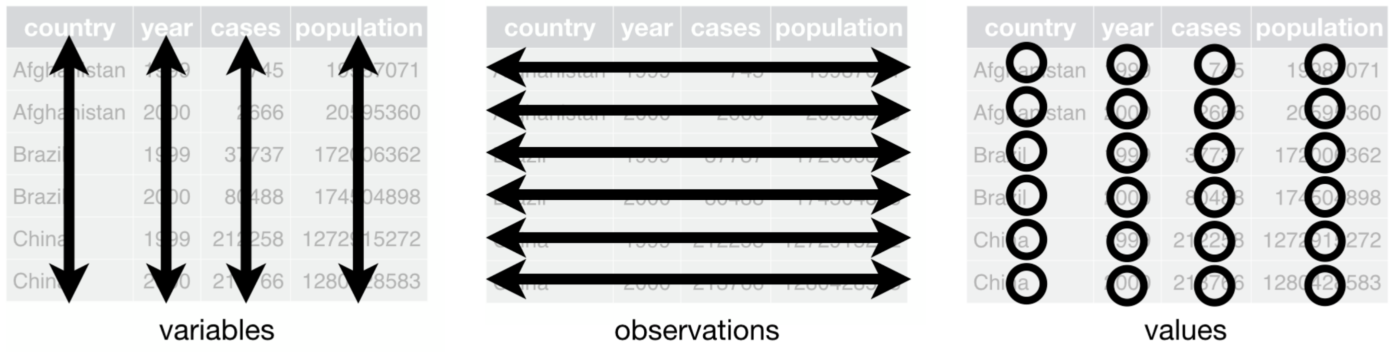
# Tidy the data with `tidyr`

A table is tidy if:

- Each variable is in its own column
- Each observation is in its own row

Key functions:

- `pivot_longer`
- `pivot_wider`
- `unite`
- `separate`



source: https://r4ds.had.co.nz/tidy-data.html

Stockholm University

## Tidy the data with `tidyr` - `iris` example

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species | id |
|---|---|---|---|---|---|
| 5.1 | 3.5 | 1.4 | 0.2 | setosa | 1 |
| 4.9 | 3.0 | 1.4 | 0.2 | setosa | 2 |
| 4.7 | 3.2 | 1.3 | 0.2 | setosa | 3 |
| 4.6 | 3.1 | 1.5 | 0.2 | setosa | 4 |

```r
iris %<>% dplyr::mutate(id = 1:150)
iris |> head(4)
```

## Tidy the data with tidyr - pivot_longer

| Species | id | Parameter | Size |
|---|---|---|---|
| setosa | 1 | Sepal.Length | 5.1 |
| setosa | 1 | Sepal.Width | 3.5 |
| setosa | 1 | Petal.Length | 1.4 |
| setosa | 1 | Petal.Width | 0.2 |

```r
long_iris <- iris |>
  tidyr::pivot_longer(1:4,
                      names_to = "Parameter",
                      values_to = "Size")
long_iris |> head(4)
```

## Tidy the data with `tidyr` - `separate`

| Species | id | Organ | Measure | Size |
|---|---|---|---|---|
| setosa | 1 | Sepal | Length | 5.1 |
| setosa | 1 | Sepal | Width | 3.5 |
| setosa | 1 | Petal | Length | 1.4 |
| setosa | 1 | Petal | Width | 0.2 |

```r
sep_iris <- long_iris |>
  tidyr::separate(Parameter, into = c("Organ", "Measure"))
sep_iris |> head(4)
```

## Tidy the data with `tidyr` - `pivot_wider`

| Species | id | Measure | Sepal | Petal |
|---|---|---|---|---|
| setosa | 1 | Length | 5.1 | 1.4 |
| setosa | 1 | Width | 3.5 | 0.2 |
| setosa | 2 | Length | 4.9 | 1.4 |
| setosa | 2 | Width | 3.0 | 0.2 |

```r
wide_iris <- sep_iris |>
  tidyr::pivot_wider(names_from = "Organ",
                     values_from = "Size")
wide_iris |>  head(4)
```

## Tidy the data with `tidyr` - `unite`

| Species | id | Measure | Sepal/ Petal |
|---|---|---|---|
| setosa | 1 | Length | 5.1/1.4 |
| setosa | 1 | Width | 3.5/0.2 |
| setosa | 2 | Length | 4.9/1.4 |
| setosa | 2 | Width | 3/0.2 |

```r
unite_iris <- wide_iris |>
  tidyr::unite(col = "Sepal/Petal", c(Sepal, Petal), sep = "/")
unite_iris |>  head(4)
```

## Arrange the data with `dyplr`

`dplyr` simplifies the data manipulation with self-explanatory functions:

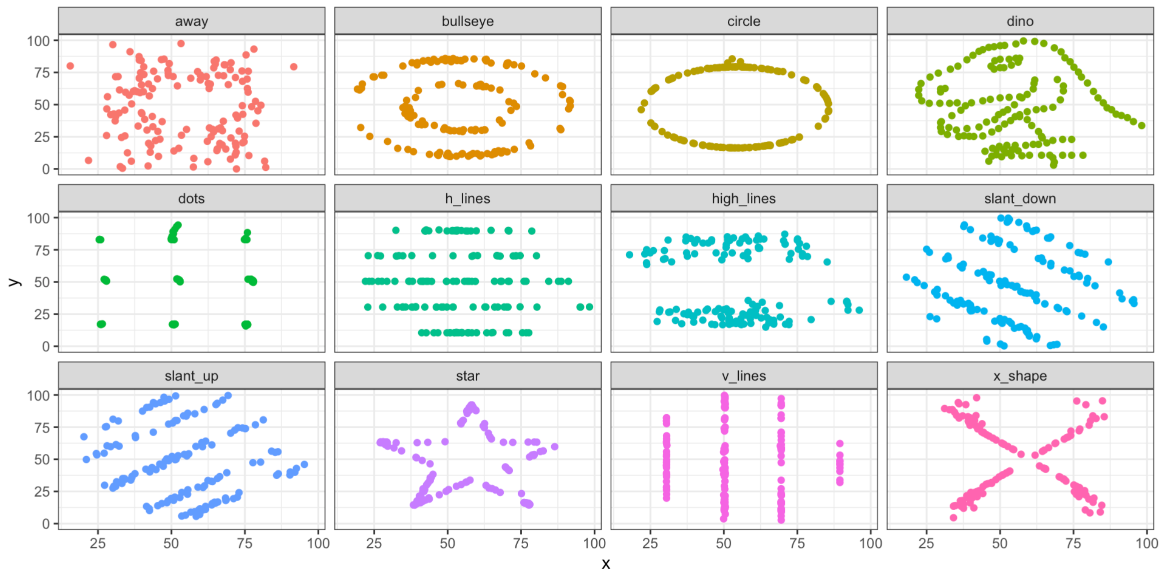
- `filter` observations based on their values
- `mutate` a new column as a function of others
- `select` variables based on their names
- `group_by` variable
- `summarise` the data

```r
iris %>%
  dplyr::filter(Petal.Length >= 1.4) %>%
  dplyr::mutate(Sepal_Ratio = Sepal.Length/Sepal.Width) %>%
  dplyr::select(Species, Sepal_Ratio) %>% # equaivalent to select(-c(Sepal.Length, Sepal.Wi
  dplyr::group_by(Species) %>%
  dplyr::summarise(Average_ratio = mean(Sepal_Ratio),
                   standard_deviation = sd(Sepal_Ratio))
```

Stockholm University

# Data visulalisation

It is very important to look at the data.
Totally different data might have similar statistics…

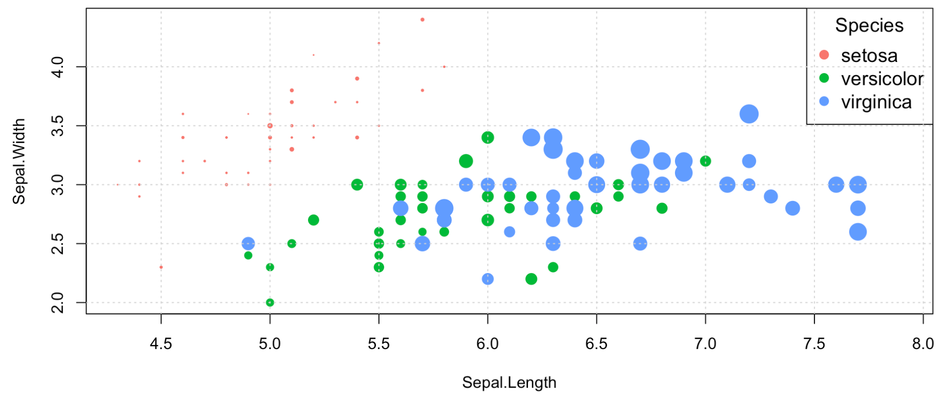


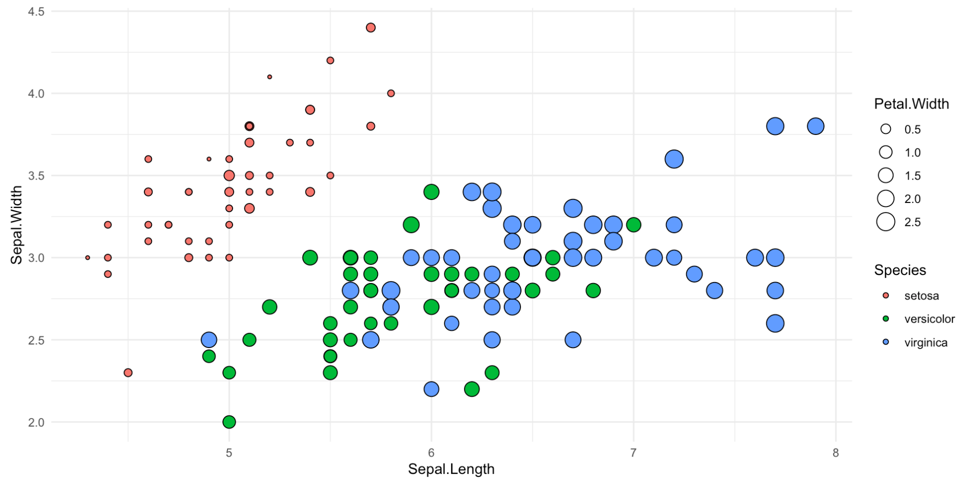
| statistics | value |
|---|---|
| mean_x | 54.27 |
| mean_y | 47.83 |
| sd_x | 16.77 |
| sd_y | 26.94 |

source: datasauRus

Stockholm University

# Visualise the data with ggplot2





```r
species_palette <- c("#F8766D", "#00BA38",
size_vector <- iris$Petal.Width
plot(x = iris$Sepal.Length,
     y = iris$Sepal.Width,
     col = species_palette[iris$Species],
     bg = species_palette[iris$Species],
     pch = 21,
     cex = size_vector,
     xlim = c(min(iris$Sepal.Length), max(
     ylim = c(min(iris$Sepal.Width), max(i
     xlab = "Sepal.Length",
     ylab = "Sepal.Width")
legend("topright", legend = levels(iris$Sp
grid(lwd = 1, lty = "dotted")
```

```r
library(ggplot2)
ggplot(iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     fill = Species,
                     size = Petal.Width))
  geom_point(shape = 21,
             col = 1) +
  theme_minimal()
```

Stockholm University

# Grammar of graphic (gg)

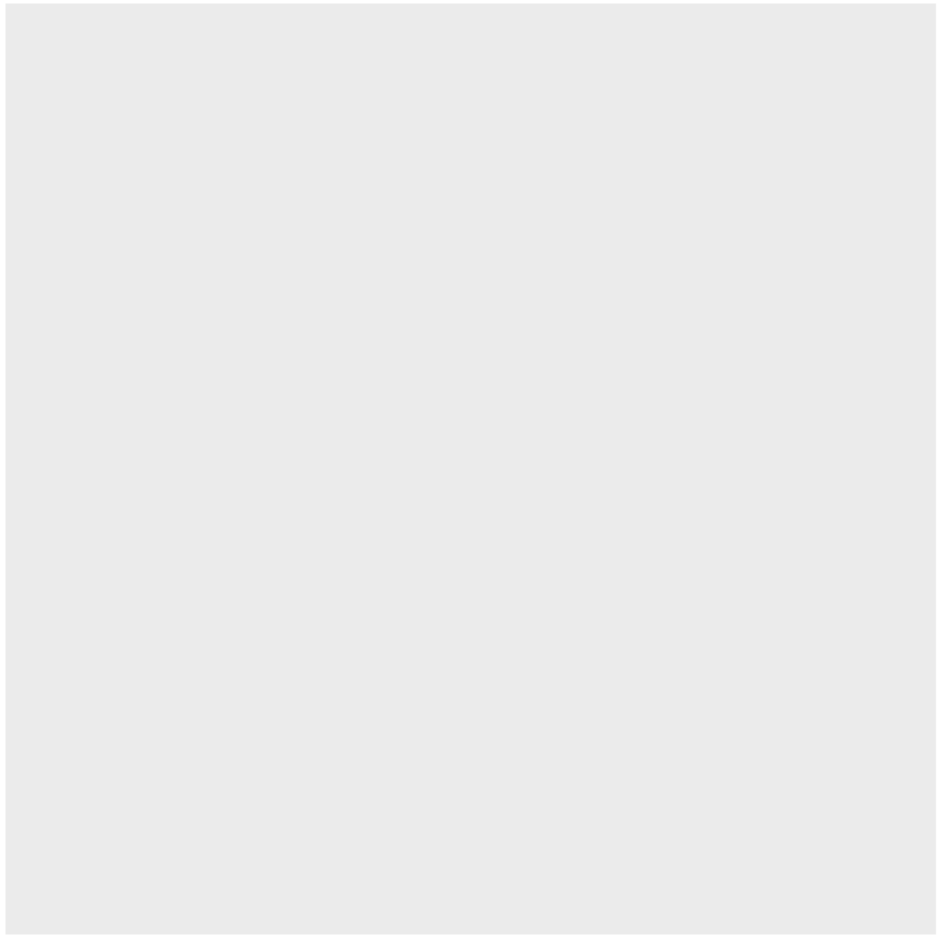
- Data
- Aesthetics - Visual characteristics (e.g., x, y, size)
- Geometry - How to represent the data (e.g., lines, point, boxplot)

- Statistics - What statistics to show
- Facets - Split the data
- Coordinates - Position of the geometry
- Themes - Visual changes

```
ggplot(data = <DATA>,
       mapping = aes(<AESTHETICS>)) +
  geom_<GEOMETRY>()
```

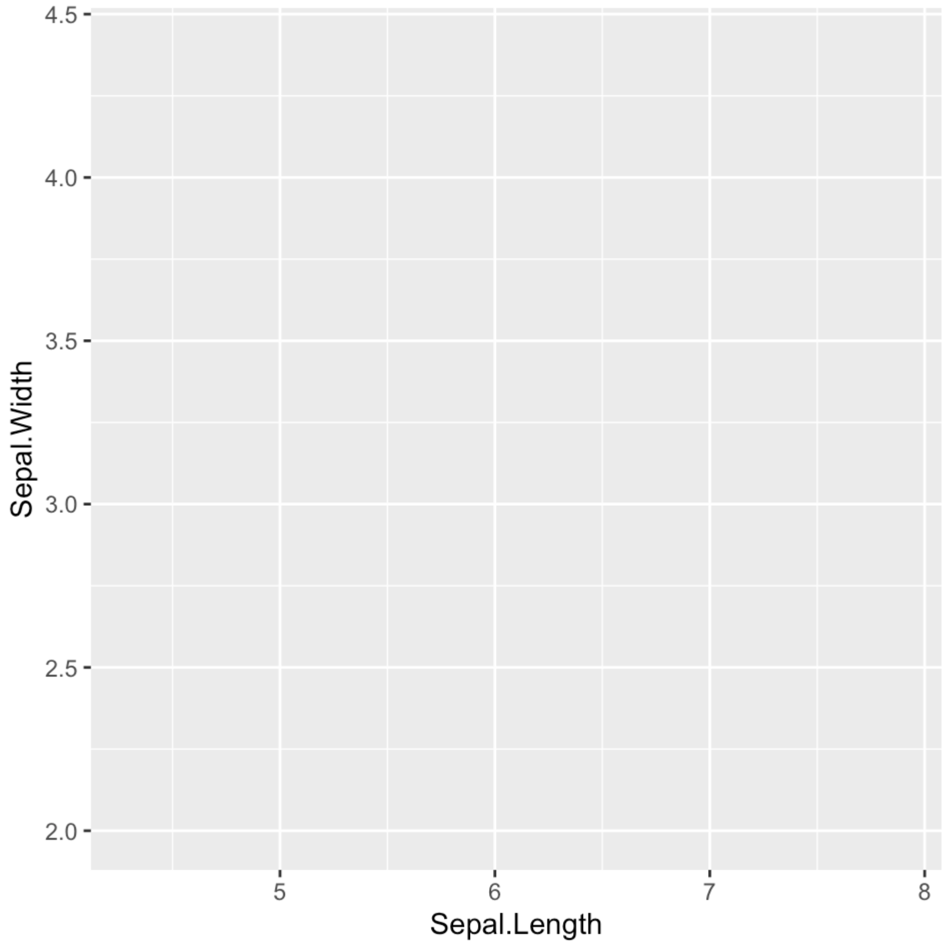
## Let's plot using `ggplot2` - *Data*

```r
ggplot(data = iris)
```



## Let’s plot using `ggplot2` - *Aesthetics*

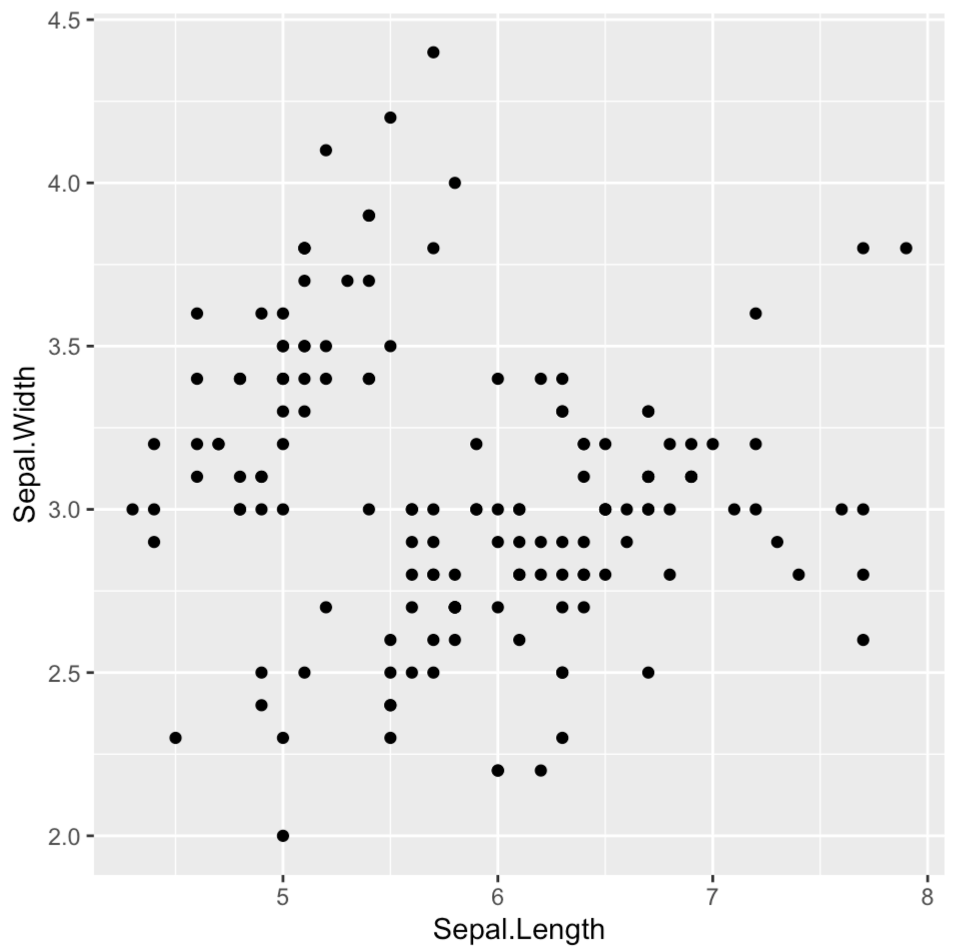
```
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width))
```



Stockholm University

# Let’s plot using ggplot2 - *Geometry*

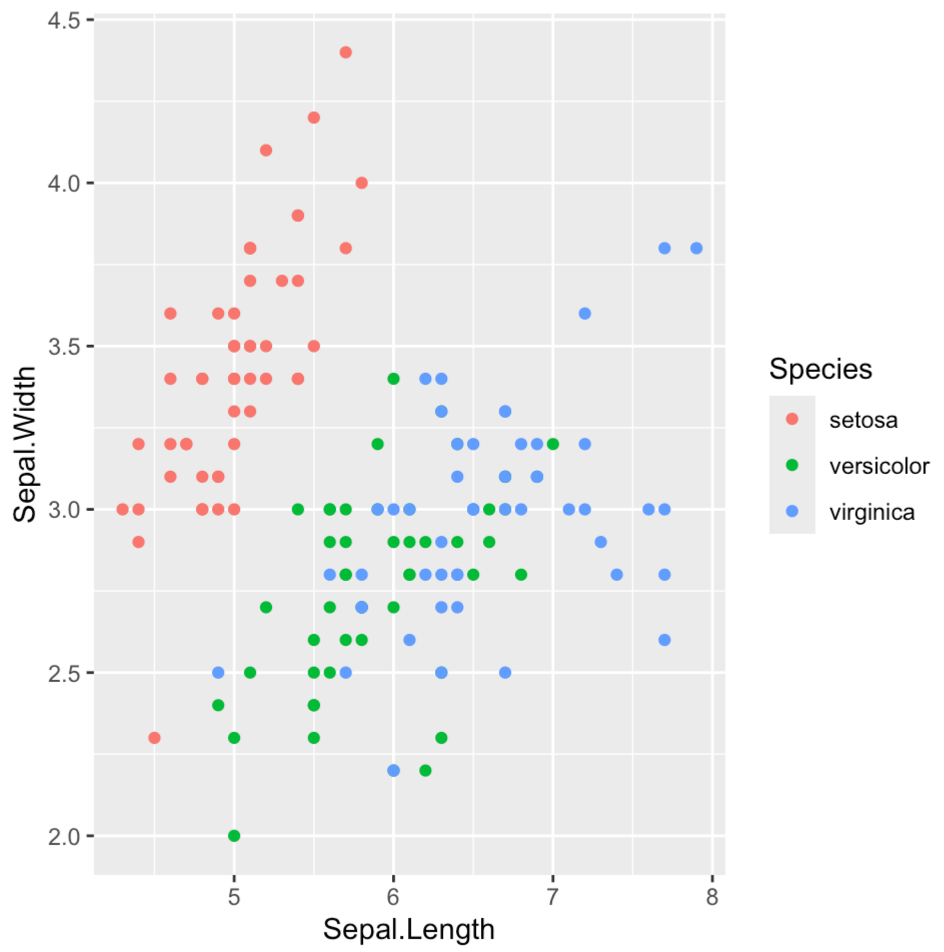
```r
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width)) +
  geom_point()
```



Stockholm University

# Let’s plot using ggplot2 - *Geometry*

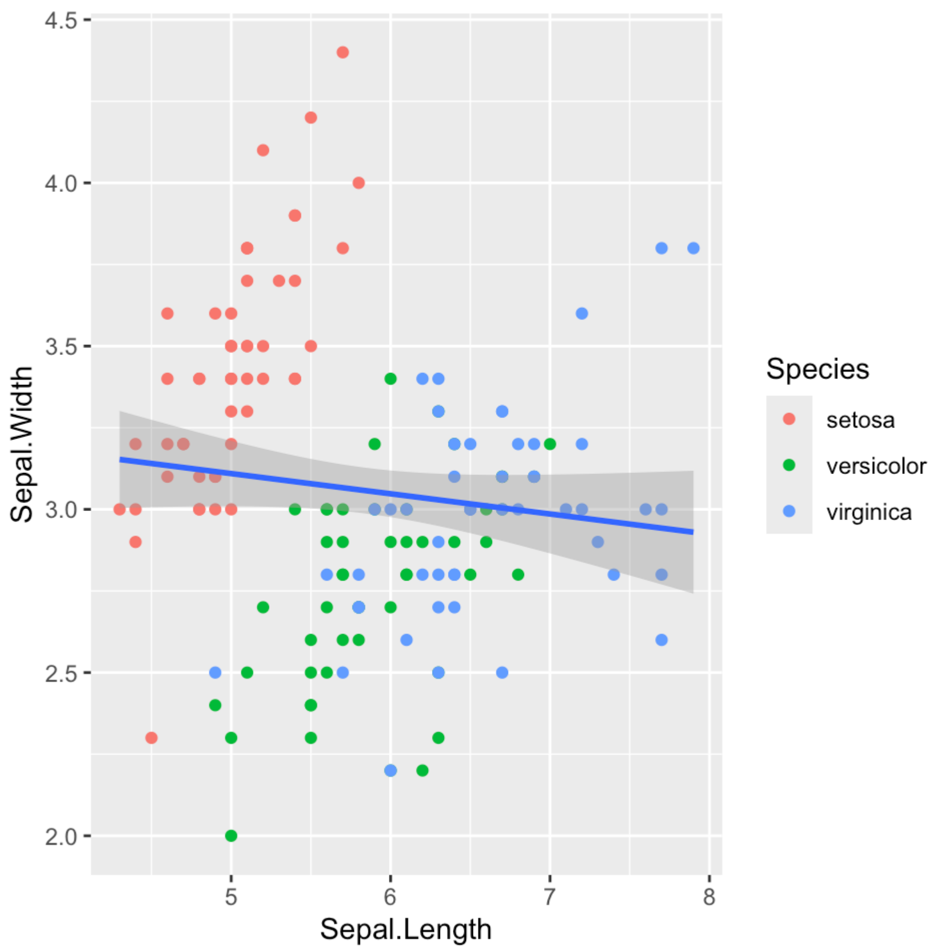
```
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width)) +
  geom_point(mapping = aes(col = Species))
```



Stockholm University

# Let’s plot using **ggplot2** - *Statistics*

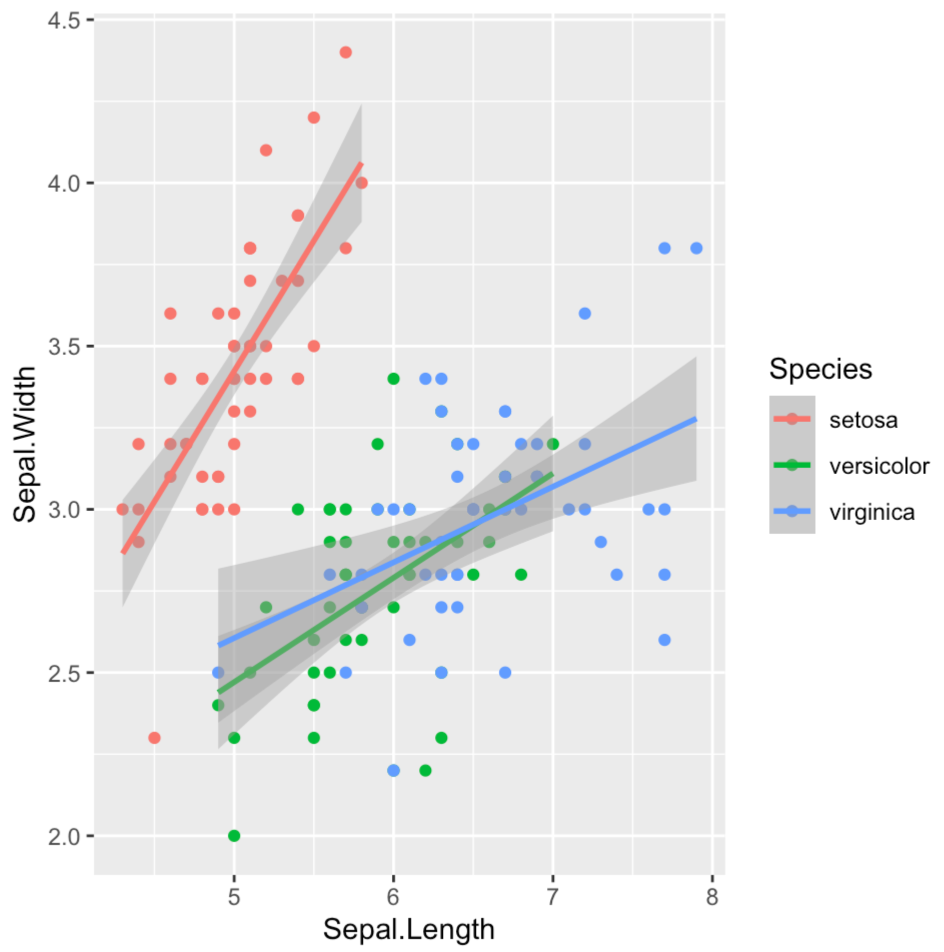
```r
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width)) +
  geom_point(mapping = aes(col = Species))
  stat_smooth(method = "lm")
```



Stockholm University

# Let’s plot using `ggplot2` - *Statistics*

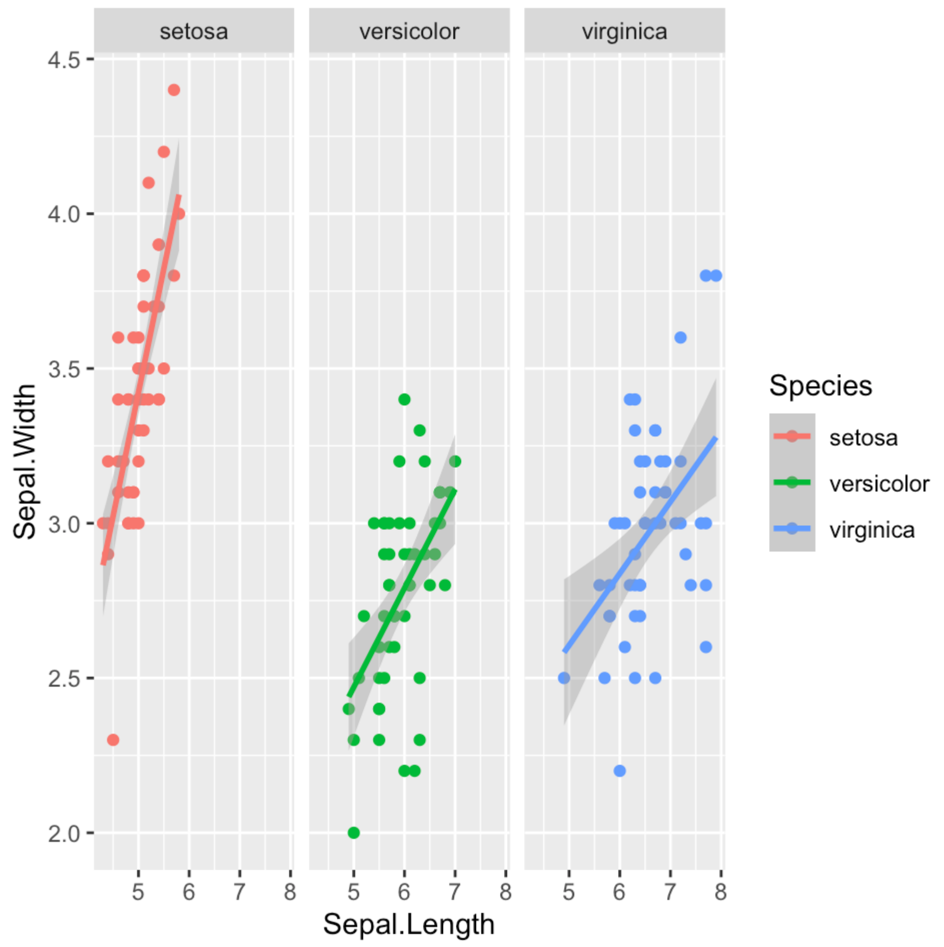
```r
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     col = Species)) +
  geom_point() +
  stat_smooth(method = "lm")
```



Stockholm University

# Let's plot using `ggplot2` - *Facets*

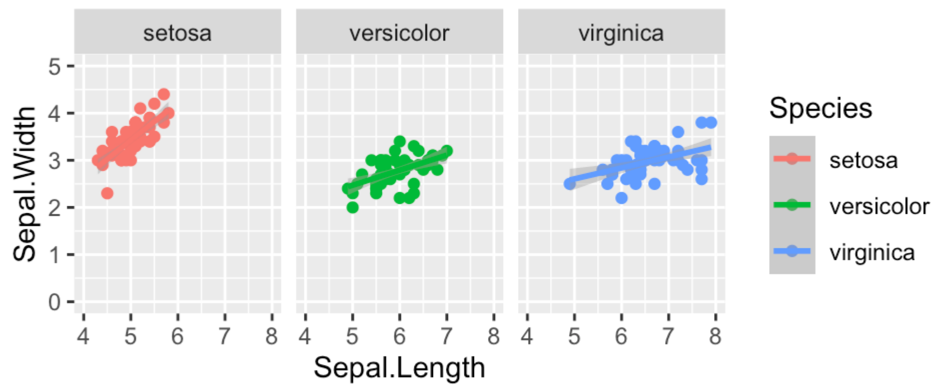
```r
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     col = Species)) +
  geom_point() +
  stat_smooth(method = "lm") +
  facet_wrap(~Species)
```



Stockholm University

# Let’s plot using ggplot2 - *Coordinates*

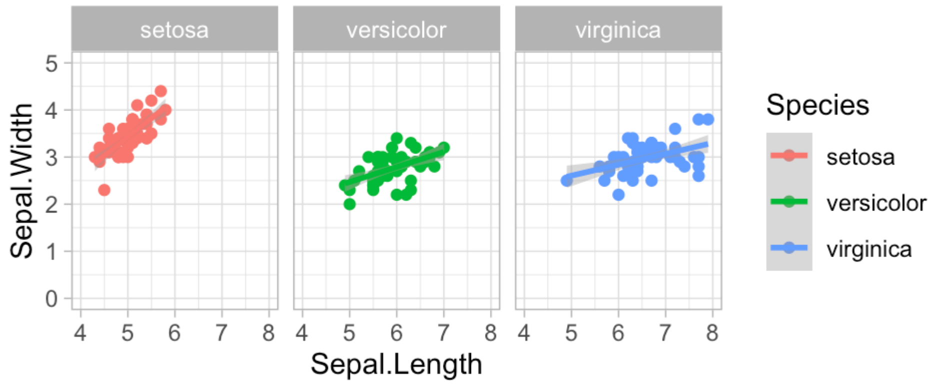
```r
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     col = Species)) +
  geom_point() +
  stat_smooth(method = "lm") +
  facet_wrap(~Species)+
  coord_fixed(xlim = c(4,8), ylim = c(0,5
```



Stockholm University

# Let’s plot using ggplot2 - *Themes*

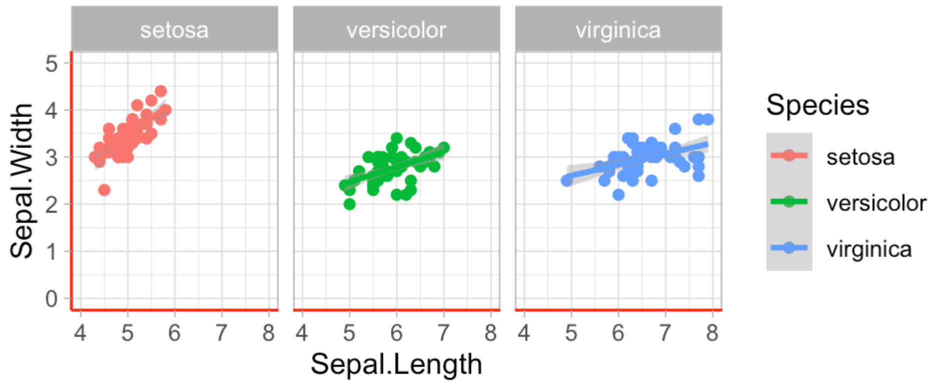
```r
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     col = Species)) +
  geom_point() +
  stat_smooth(method = "lm") +
  facet_wrap(~Species) +
  coord_fixed(xlim = c(4,8), ylim = c(0,5)
  theme_light()
```



Stockholm University

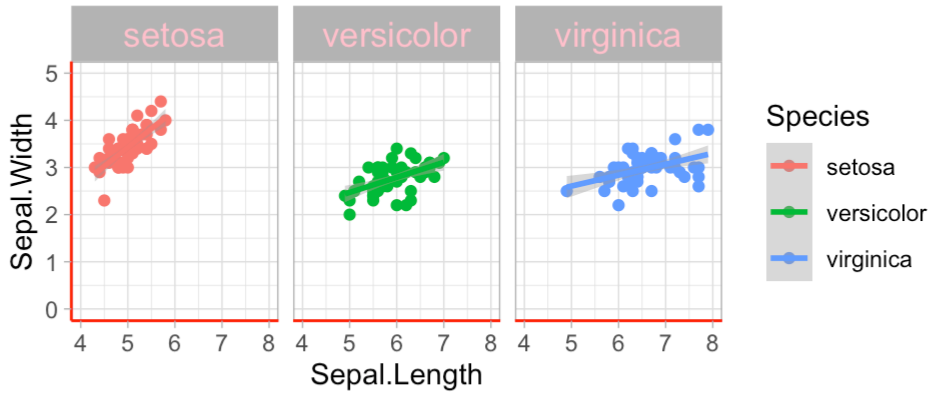
# Let’s plot using ggplot2 - *Themes*

```
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     col = Species)) +
  geom_point() +
  stat_smooth(method = "lm") +
  facet_wrap(~Species) +
  coord_fixed(xlim = c(4,8), ylim = c(0,5
  theme_light() +
  theme(axis.line = element_line(color = '
```



# Let’s plot using ggplot2 - *Themes*

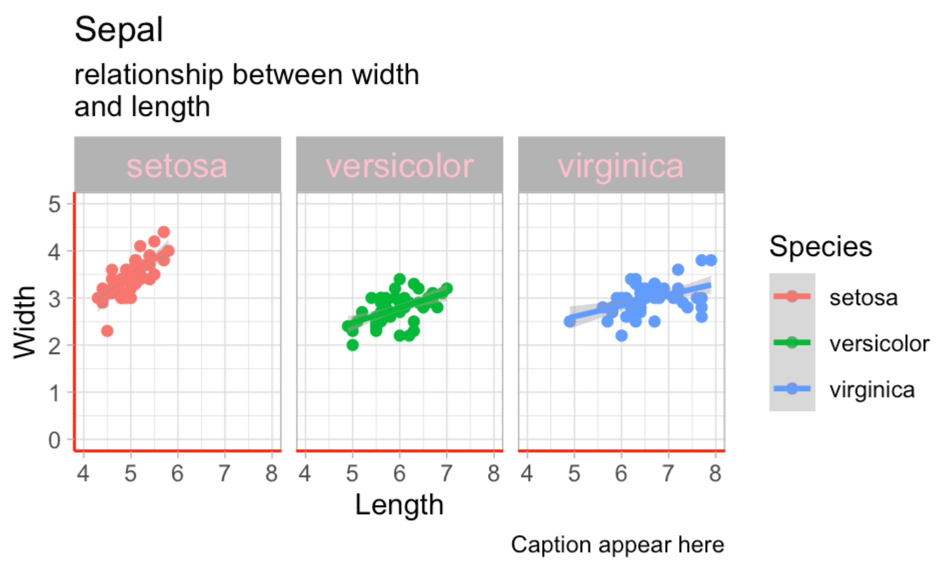
```
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     col = Species)) +
  geom_point() +
  stat_smooth(method = "lm") +
  facet_wrap(~Species) +
  coord_fixed(xlim = c(4,8), ylim = c(0,5)
  theme_light() +
  theme(axis.line = element_line(color = '
        strip.text = element_text(size = 1
```



Stockholm University

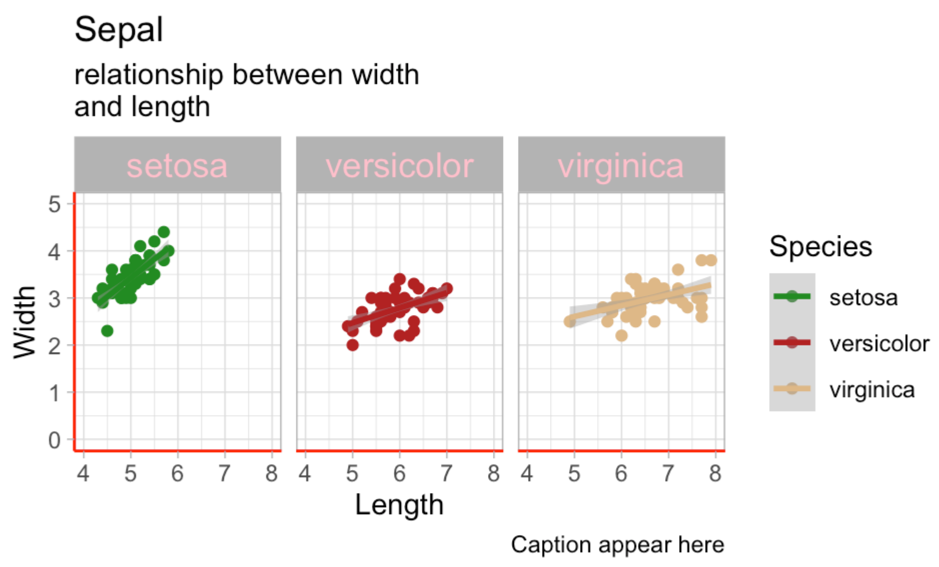
# Let’s plot using ggplot2 - *Extra tips*

```
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     col = Species)) +
  geom_point() +
  stat_smooth(method = "lm") +
  facet_wrap(~Species)+
  coord_fixed(xlim = c(4,8), ylim = c(0,5
  theme_light() +
  theme(axis.line = element_line(color = '
        strip.text = element_text(size = 1
  labs(title = "Sepal", x = "Length" , y =
```



# Let's plot using ggplot2 - *Extra tips*

```
ggplot(data = iris,
       mapping = aes(x = Sepal.Length,
                     y = Sepal.Width,
                     col = Species)) +
  geom_point() +
  stat_smooth(method = "lm") +
  facet_wrap(~Species)+
  coord_fixed(xlim = c(4,8), ylim = c(0,5
  theme_light() +
  theme(axis.line = element_line(color = '
        strip.text = element_text(size = 1
  labs(title = "Sepal", x = "Length" , y =
  scale_color_manual(values = c("forestgre
```


Sepal
relationship between width and length
setosa
versicolor
virginica
Width
Length
Species
setosa
versicolor
virginica
Caption appear here