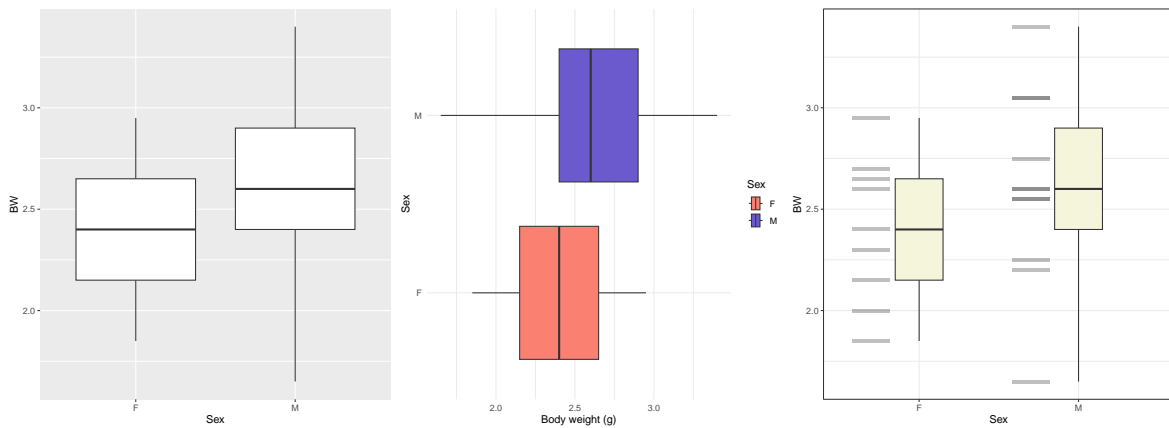


Boxplots in R with ggplot2

Cheatsheet

2024-07-17

This work was developed using resources that are available under a [Creative Commons Attribution 4.0 International License](#), made available on the [SOLES Open Educational Resources](#) repository by the School of Life and Environmental Sciences, The University of Sydney.



About

The **boxplot** is a visual representation of a dataset's distribution, showing the median, quartiles, and outliers. It is useful for comparing distributions between groups and identifying outliers within a single group.

Assumed knowledge

- You know how to install and load packages in R.
- You know how to import data into R.
- You recognise data frames and vectors.

💡 Data structure

Your data should be **structured** in a way that makes it *easy* to plot. The ideal structure is **long**, i.e. one where each column represents a variable and each row an observation (Figure 1). You can either reshape your data in R or **move cells manually** in a spreadsheet program to achieve the desired structure. For boxplots comparing more than one group of data, a **categorical variable** representing the group should be present in the data.

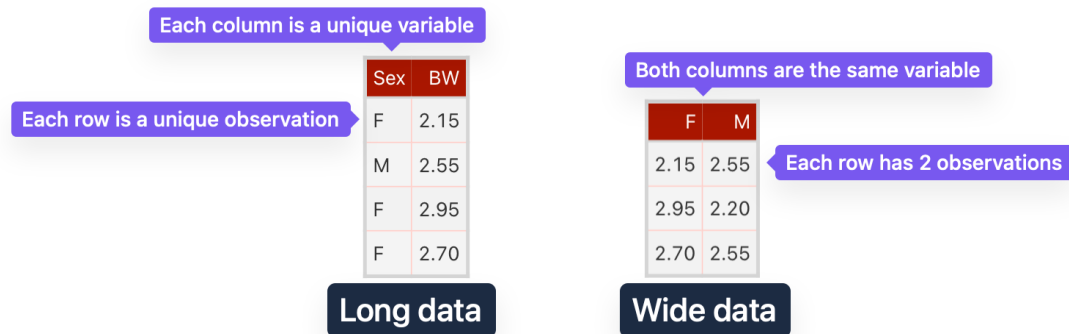


Figure 1: Long data (left) where each column is a different variable – e.g. **Sex** is categorical and **BW** is the measured, continuous response – is preferred over wide data (right), as it makes it easier to manipulate data when plotting.

Data

For this cheatsheet we will use part of the possums dataset used in [BIOL2022](#) labs.

Import data

```
library(readxl) # load the readxl package
possums <- read_excel("possum_bw.xlsx") # read file, store as "possums" object
```

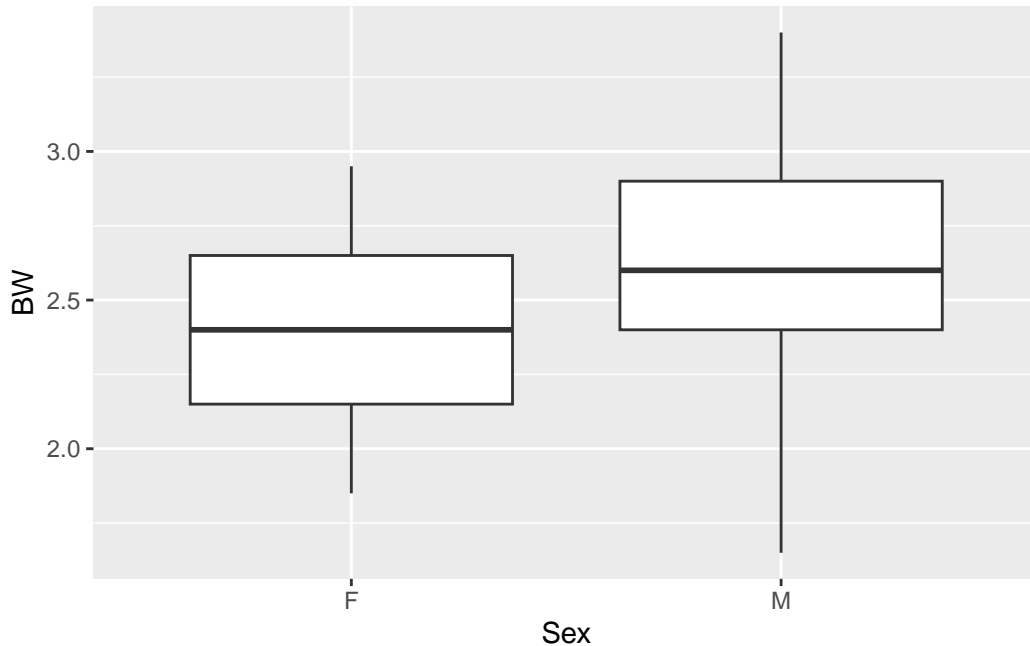
Plot

Below are multiple versions of a boxplot comparing the body weight, BW, of possums between two groups defined by the **Sex** variable. Use the code snippets and their different implementations to understand how to customise your boxplot.

Version 1

```
library(ggplot2) ①  
ggplot(possums, aes(x = Sex, y = BW)) + ②  
  geom_boxplot() ③
```

- ① The `library()` function loads a package. Here, we load the `ggplot2` package to enable the functions required to create the plot.
- ② The `ggplot()` function creates a plot canvas. The `aes()` function specifies the aesthetic mappings, i.e. which variables are mapped to the x and y axes.
- ③ Once the canvas is defined, the data can be added automatically using `geom_*()` functions. Here, `geom_boxplot()` adds the boxplot to the canvas, structured according to the aesthetic mappings.



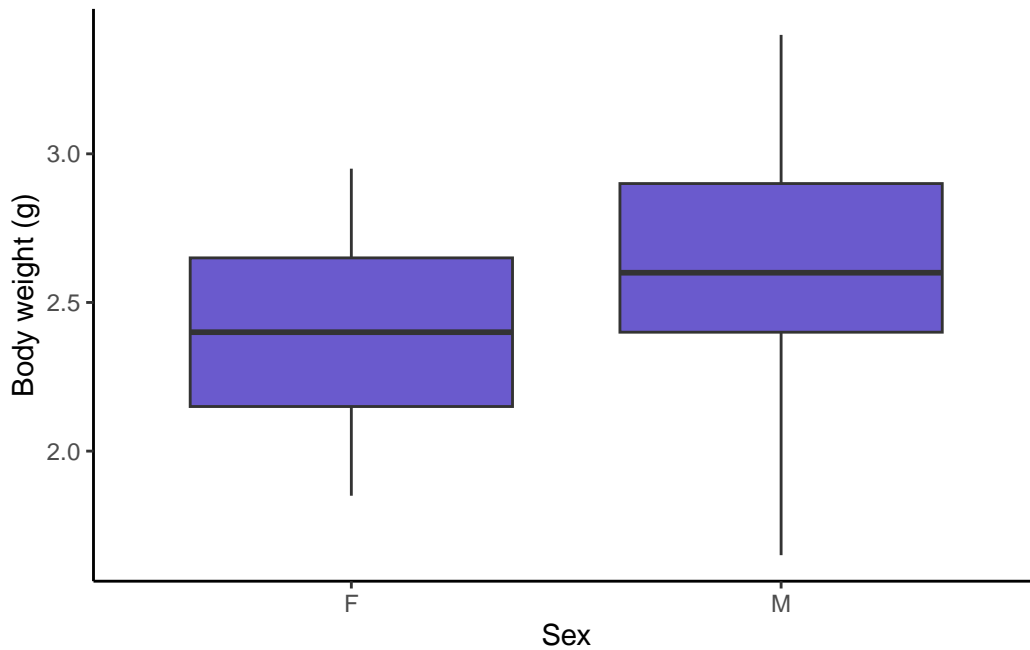
Version 2

```
library(ggplot2)  
ggplot(possums, aes(x = Sex, y = BW)) +  
  geom_boxplot(fill = "slateblue") +  
  xlab("Sex") +
```

```
ylab("Body weight (g)") +  
theme_classic()
```

③

- ① Adding a `fill` argument to the `geom_boxplot()` function changes the colour of the boxplot.
- ② `xlab()` and `ylab()` add labels to the x and y axes, respectively.
- ③ An optional step, `theme_classic()` changes the plot's appearance without needing to specify complex customisations.



Version 3

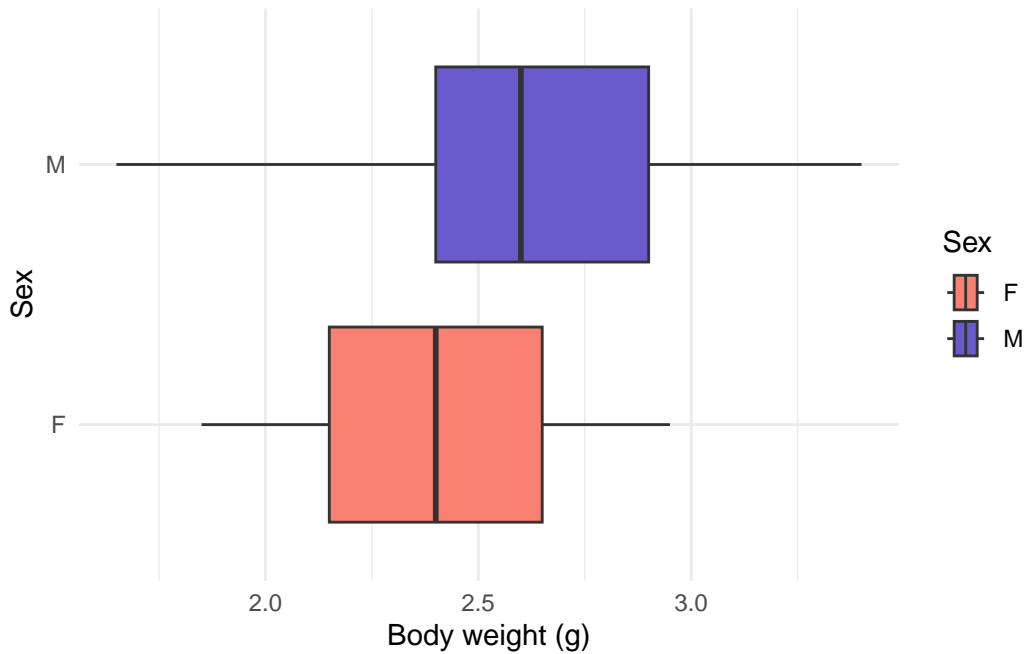
```
library(ggplot2)  
ggplot(possums, aes(x = BW, y = Sex, fill = Sex)) +  
  geom_boxplot() +  
  xlab("Body weight (g)") +  
  ylab("Sex") +  
  theme_minimal() +  
  scale_fill_manual(values = c("salmon", "slateblue"))
```

①

②

- ① The `Sex` variable is mapped to the y-axis and the `BW` variable to the x-axis. The `fill` aesthetic is used to colour the boxplots by the `Sex` variable.

- ② The `scale_fill_manual()` function allows you to manually set the colours of the boxplots defined by the `fill` aesthetic in the `aes()` function above. There must be one colour for each level of the `Sex` variable.



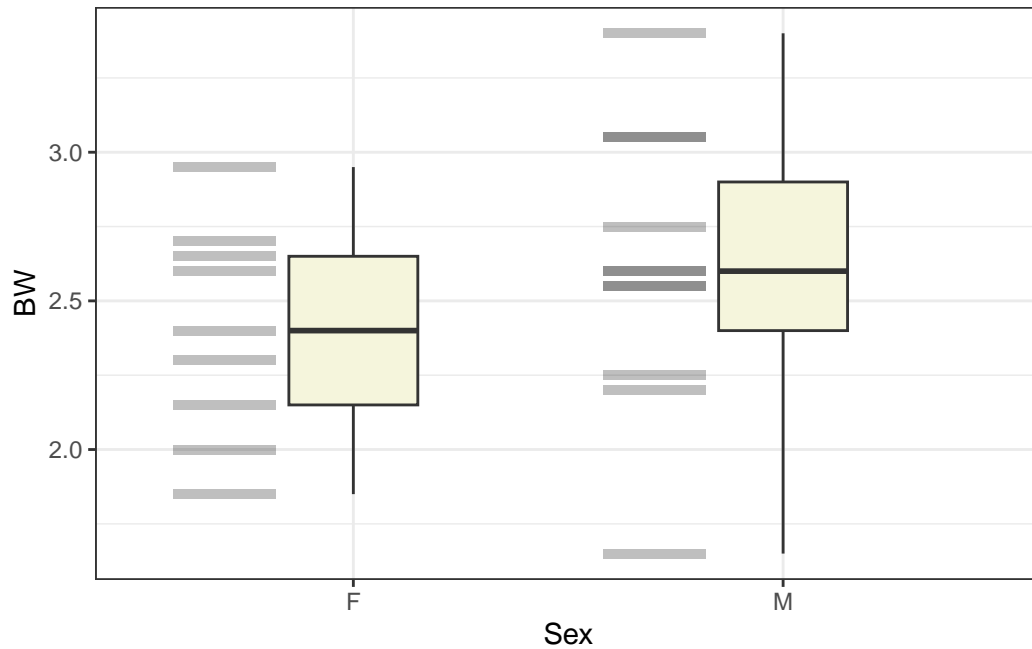
Version 4

```
library(ggplot2)
ggplot(possums) +
  aes(x = Sex, y = BW) +
  geom_boxplot(width = .3, fill = "beige") +
  geom_point(
    position = position_nudge(x = -.3),
    shape = 95, size = 24, alpha = .25
  ) +
  theme_bw()
```

①
②
③
④
⑤

- ① The `aes()` function is placed outside the `ggplot()` function, allowing the aesthetic mappings to be used across multiple `geom_*()` functions.
- ② `geom_boxplot()` can be customised further using the `width` argument to change the width of the boxplots.
- ③ `geom_point()` adds points to the plot.
- ④ The `position_nudge()` function moves the points to the left of the boxplots by -0.3 units.

- ⑤ The **shape**, **size**, and **alpha** arguments customise the appearance of the points, resulting in a different visual representation of the data “points”.



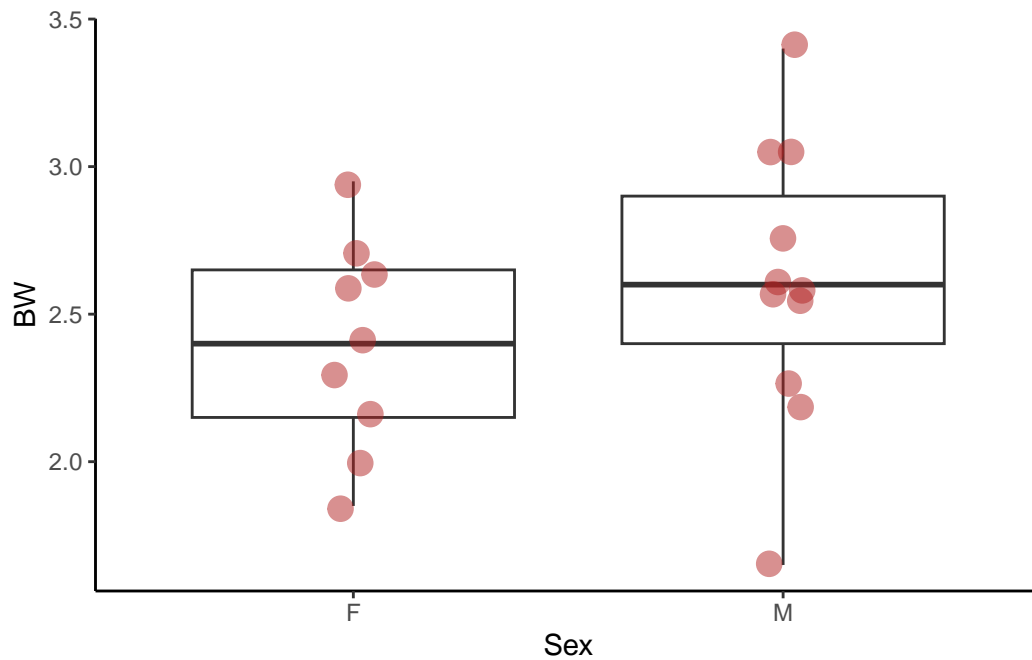
Version 5

```
library(ggplot2)
plot1 <-
  ggplot(possums) +
  aes(x = Sex, y = BW) ①

plot1 + ②
  geom_boxplot() +
  geom_point(
    position = position_jitter(width = .05, seed = 0), ③
    size = 4, alpha = .5,
    colour = "firebrick"
  ) +
  theme_classic()
```

- ① It is possible to save current work on a plot for later use by assigning it to an object, e.g. plot1.

- ② To continue working on the plot, use the `+` operator on the saved object and continue adding layers.
- ③ The `position_jitter()` function adds a small amount of random noise to the points, preventing them from overlapping. The `seed` argument ensures the noise is consistent across multiple plots.



More resources

- [R colors](#) – a good resource for choosing colours using words in R.
- [Beyond bar and box plots](#) – alternative visualisation methods in R for comparing groups.
- [Boxplot – the R Graph Gallery](#) – a gallery of boxplot examples in R.