10: Visualize your data

Graphical Analysis of Biological Data

This web page is available as a PDF file

Reading and Resources

Read Look at Data, introduction and section 1.1, from *Data Visualization* by Kieran Healy. You should also skim section 1.2 to learn what makes figures bad.

Skim A protocol for data exploration to avoid common statistical problems (free access) by Zuur et al. 2010. Focus on Figures 1-3, 5, 10-11, and the text associated with them. You will use this paper as a guide for homework 10.

Visualize your data

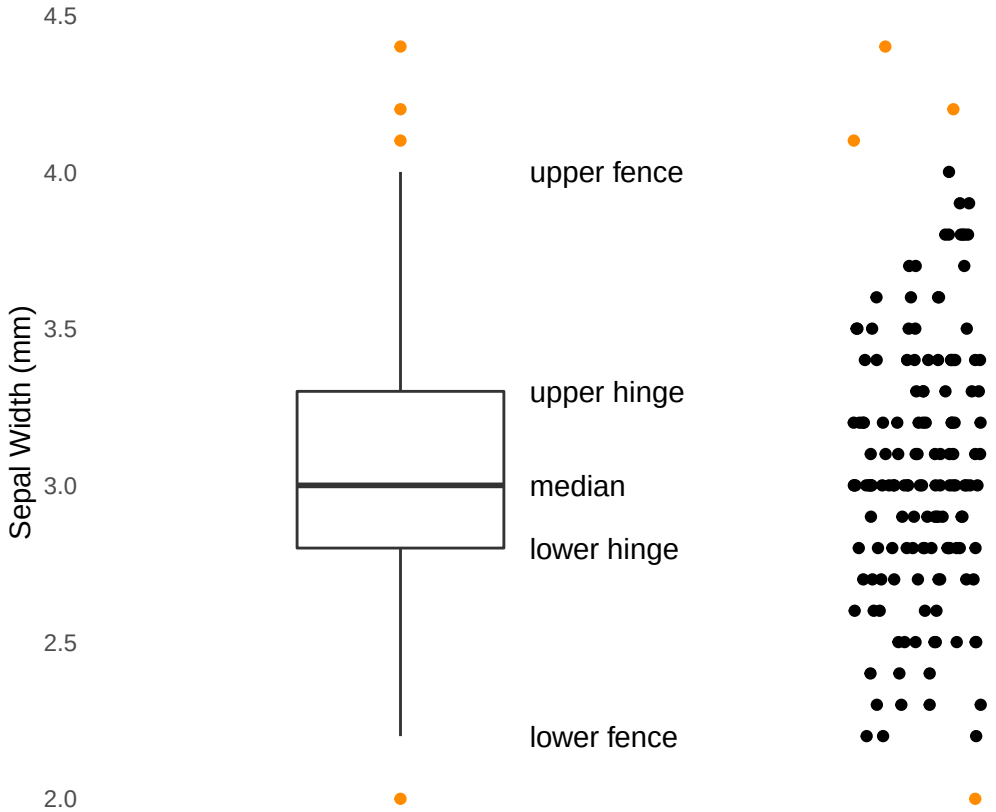
The quality of your data determines the reliability of your statistical analyses. If your data include "outliers" or your data do not meet assumptions of distribution assumed by the statistical analyses you use, then your results will be unreliable at best and unusable at worst. Therefore, an important first step is to visualize your data.

You will recreate Figures 2, 3, 5, 10, and 11. The purpose of this exercise is to learn how you use data visualization to check for outliers, normal distributions, and associations between data (Figure 1, Zuur et al 2010). The techniques covered here are not exhaustive but they are an essential first step.

Outliers: boxplots

Read Visualizing distributions along the vertical axis from *Fundamentals of Data Visualization* by Claus O. Wilke.

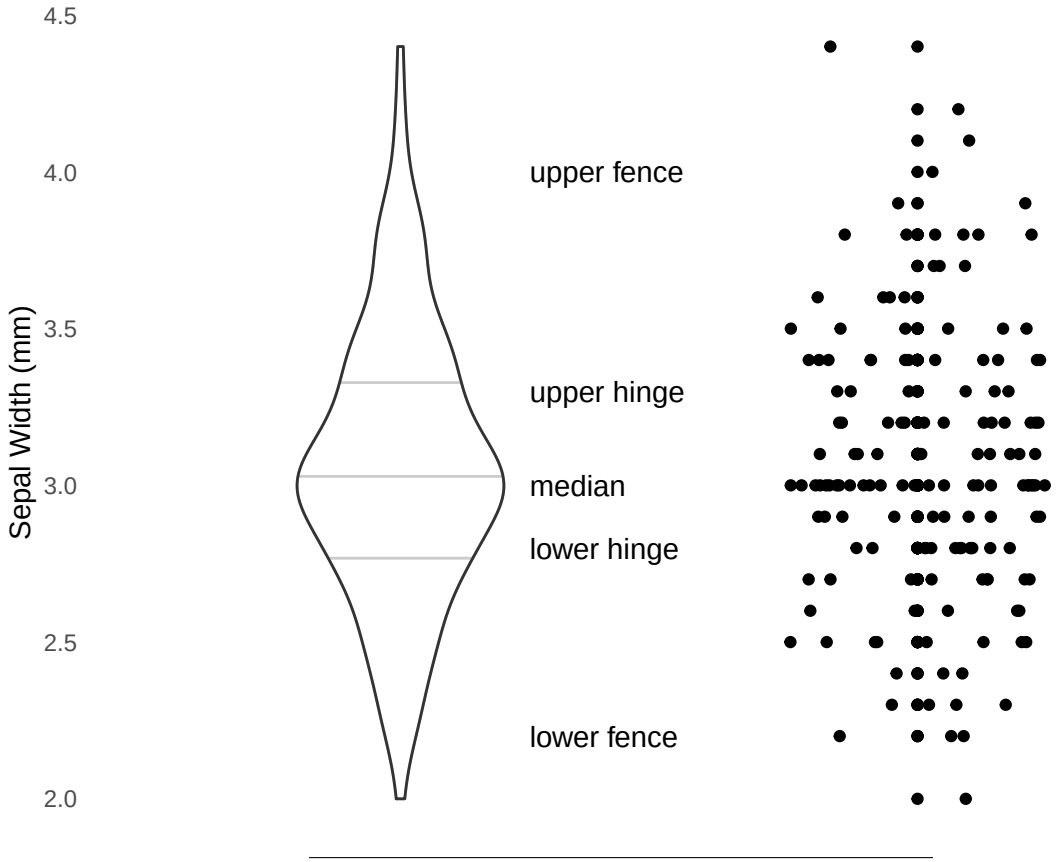
A good plot to use for each continuous variable in your data set is a *boxplot*, also called a box-and-whisker plot. Boxplots summarize the distribution of a variable and identify *possible* outliers. The figure below shows a boxplot on the left and the actual points on the right, for comparison.



A typical boxplot has several key features. The **median** is the middle value of the data set. 50% of the data are less than the median and 50% of the data are greater than the median. 25% of the data fall between the median and lower boundary of the box(**lower hinge**), and 25% of the data fall between the median and the upper box boundary (**upper hinge**). In other words, the height of the box contains 50% of the data. The box height is known as the **interquartile range** (IQR). The whiskers are the thin lines that extend below and above the box. The whiskers extend to the minumum and maximum values in the data *unless* some values fall outside of the IQR * 1.5, known as the "fence." Values outside of the fences are possible **outliers** and identified as individual points, here colored orange.

As you will see in the assignment, outliers identified by boxplots are *possible* outliers but must be explored in more depth.

An alternative to a boxplot is a violin plot. A violin plot combines concepts from a boxplot together with a density plot (see below). Here are the same data as a violin plot (`geom_violin`). Normally, violin plots are not drawn with the median, hinges, or fences, but I include them here for reference. *Do not confuse the width of the jittered plot on the right with the width of the violin along its length.* The jittered plot intentially spreads out the points to make them more visible. You can see there are many more points near the middle of the jitter plot than near the top and bottom. That is reflected in the width of the violin plot.



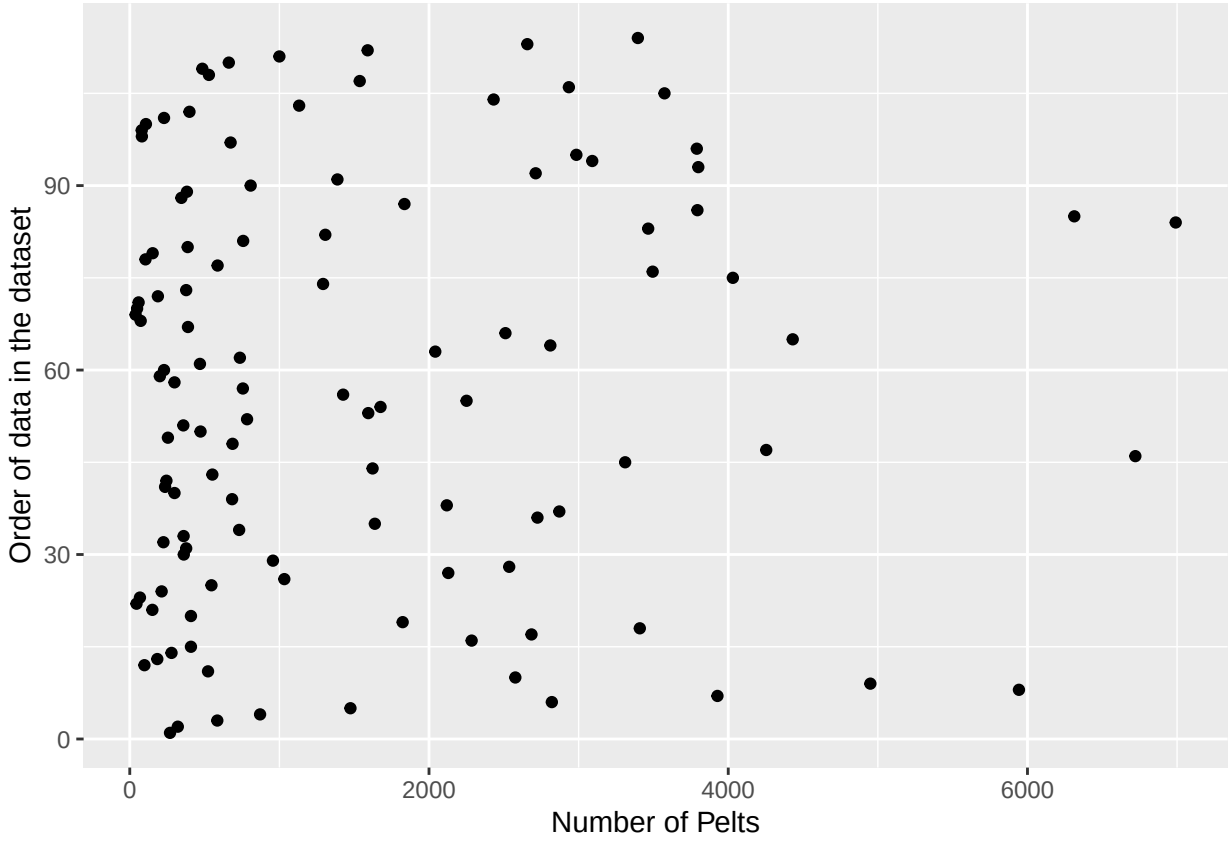
Outliers: Cleveland plots

Cleveland dot plots, or simply Cleveland plots, are named for William Cleveland, who developed them as an alternative to the all-too-common column and bar charts. Like column charts, Cleveland plots show the distribution of a continuous variable among nominal categories. But, by using dots instead of filled columns, Cleveland plots have a higher data-ink ratio. A Cleveland plot can superficially resemble a scatterplot but scatterplots (more below) plot the relationship between two continuous variables.

This Cleveland plot shows the number of harvested lynx pelts per year, in the row order of the data set. You can see that fewer than 1000 pelts were harvested most years, and it was uncommon but not rare to harvest more than 4000 pelts.

```
lynx_pelts <- tibble(pelts = as.numeric(lynx))

lynx_pelts %>%
  mutate(row_order = 1:nrow(.)) %>%
  ggplot() +
  geom_point(aes(y = row_order,
                 x = pelts)) +
  labs(x = "Number of Pelts",
       y = "Order of data in the dataset")
```



We'll do more with Cleveland-style plots in a future assignment. *Cleveland plots are one of the best plot types for exploring your data.*

Data distribution: histograms

Read Visualizing distributions: Histograms and density plots from *Fundamentals of Data Visualization* by Claus O. Wilke.

Many statistical analyses assume that your data are normally distributed, with a particular distribution around the average of your sample. You should always determine whether your data approximate a normal distribution. You can do this with statistical tests but you should begin by plotting your data. One useful plot to see if your data are normally distributed is a **histogram.** Histograms show the number of individuals (sometimes proportions or frequencies) that occur in categories that you specify.

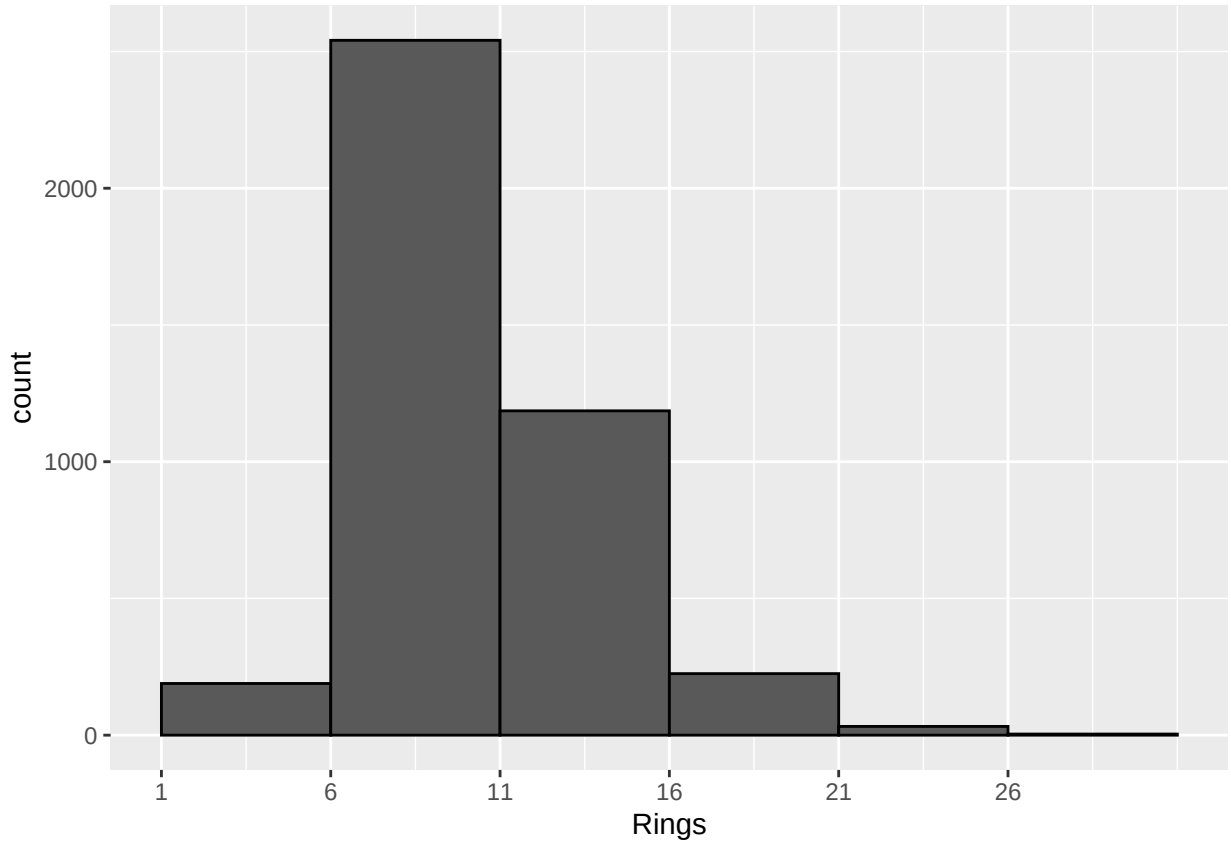
Age	N
01-05	189
06-10	2541
11-15	1186
16-20	225
21-25	32
26-30	4

A histogram-like plot is a grade distribution that shows the number of students that earned A, B, C, D, or F. Each grade is a distinct category. Nominal data like these are often shown as a column chart. But, column charts cannot be used (at least practically) for continuous data, like the data shown at right, taken from the abalone data you used previously. The age categories, or *bins* was specified by me. I used 5-year bins but I could have used 2-year, 3-year, etc. N represents the number of individuals in each age group. The data in the table clearly show that most individuals sampled are between 6-15 years old.

A simple summary like this is not too difficult to do using the `mutate` and `summarize()` functions from the `dplyr` library but I had to specify each category by hand. If I decide the 5-year bins do not provide enough resolution, then I

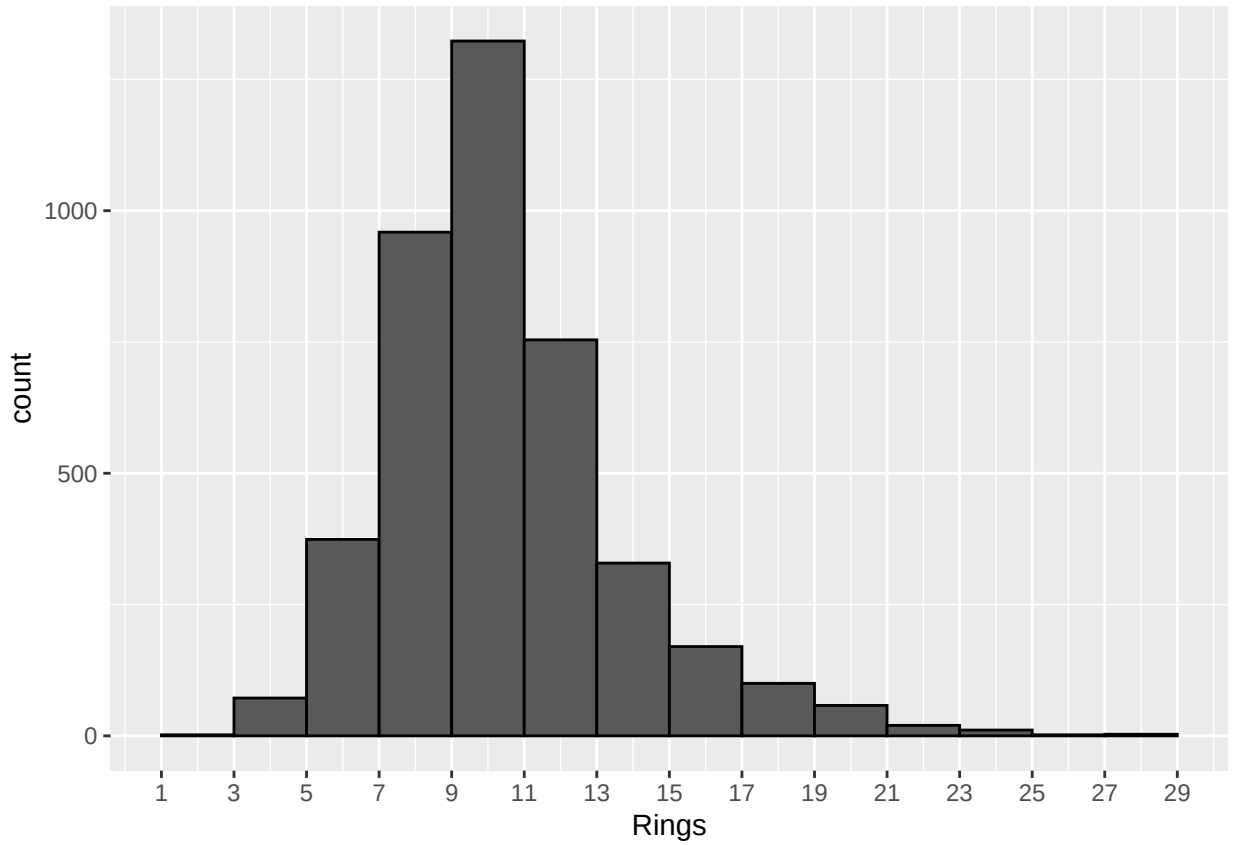
have to manually edit the code. Instead of summarizing the data this way, at least initially, I can use `geom_histogram` in ggplot2, as shown here.

```
# Data are already imported
bin_width <- 5
abalone %>%
  ggplot() +
  geom_histogram(aes(x = Rings),
                 boundary = TRUE,
                 binwidth = bin_width,
                 closed = "left",
                 color = "black") +
  scale_x_continuous(breaks = seq(1,30, by = bin_width))
```



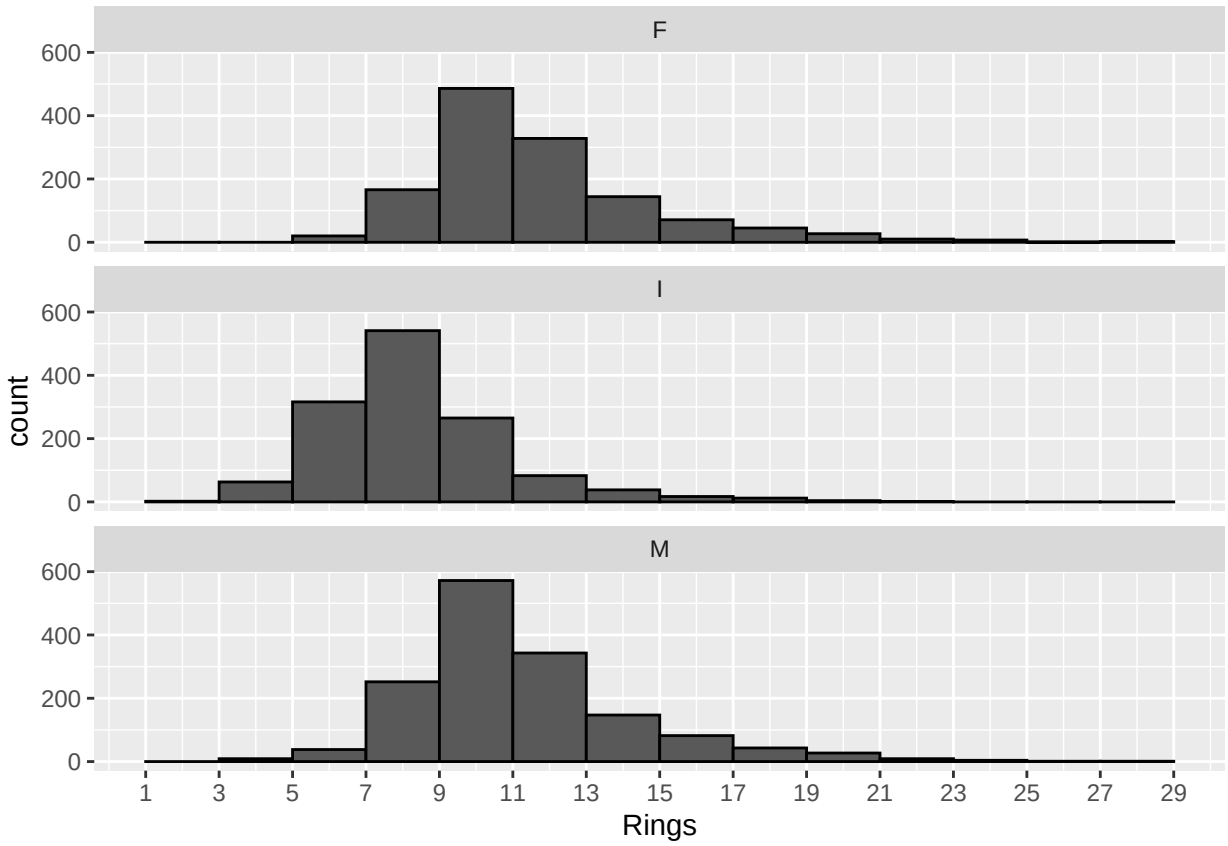
Compare the values in the table above right to the height of the bars in the histogram. Now, if I decide I want to change the age groups, I just change my `bin_width` variable to a different value, like 2. I can now see that most individuals in the sample were 9-10 years old. Also, I can see that the distribution of the the rings is approximately normal.

```
bin_width <- 2
abalone %>%
  ggplot() +
  geom_histogram(aes(x = Rings),
                 #boundary = FALSE,
                 binwidth = bin_width,
                 closed = "left",
                 color = "black") +
  scale_x_continuous(breaks = seq(1,30, by = bin_width))
```



You can apply `facet_wrap()` to histograms, just like any other plot in `ggplot2`. Here are the same data, but faceeted for the three types (Female, Immature, Male).

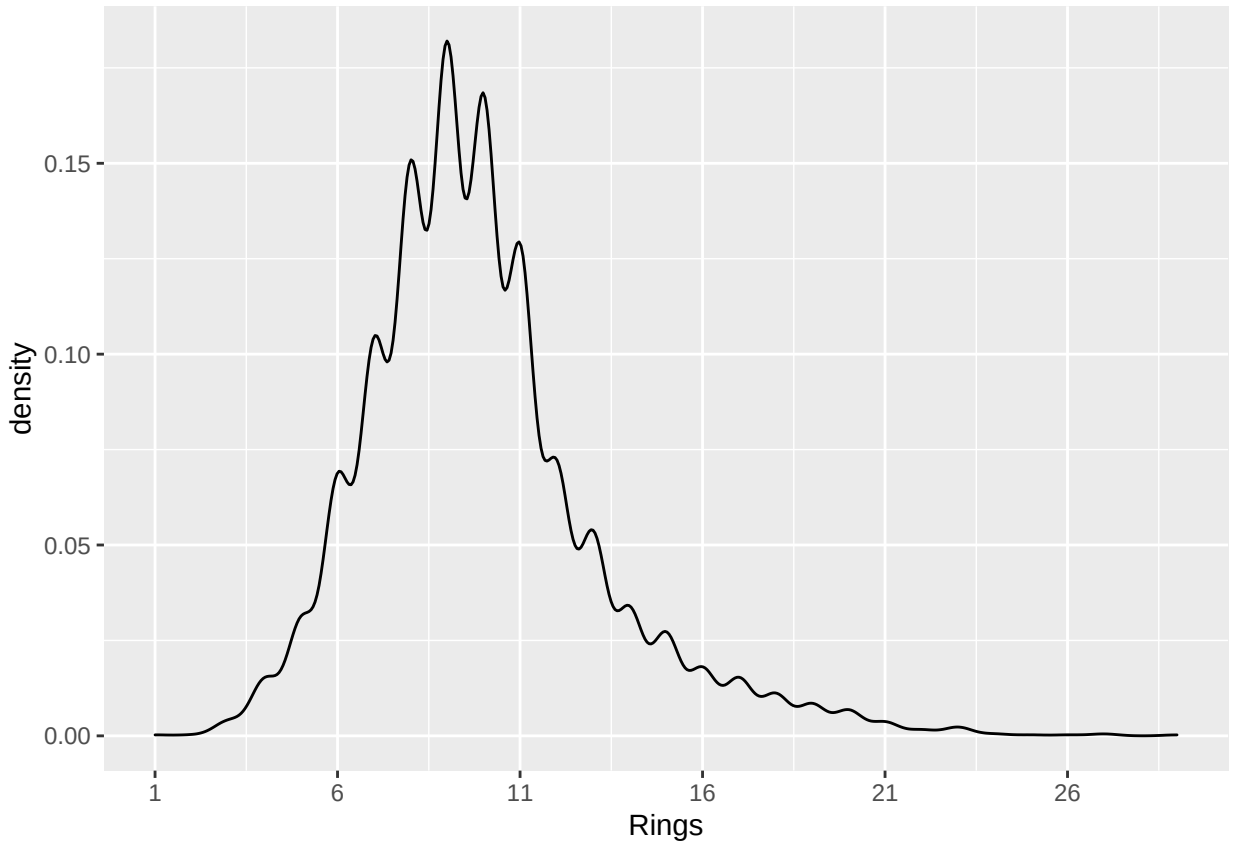
```
bin_width <- 2
abalone %>%
  ggplot() +
  geom_histogram(aes(x = Rings),
                 #boundary = FALSE,
                 binwidth = bin_width,
                 closed = "left",
                 color = "black") +
  scale_x_continuous(breaks = seq(1,30, by = bin_width)) +
  facet_wrap(~ Type, ncol = 1)
```



Data distribution: density plots

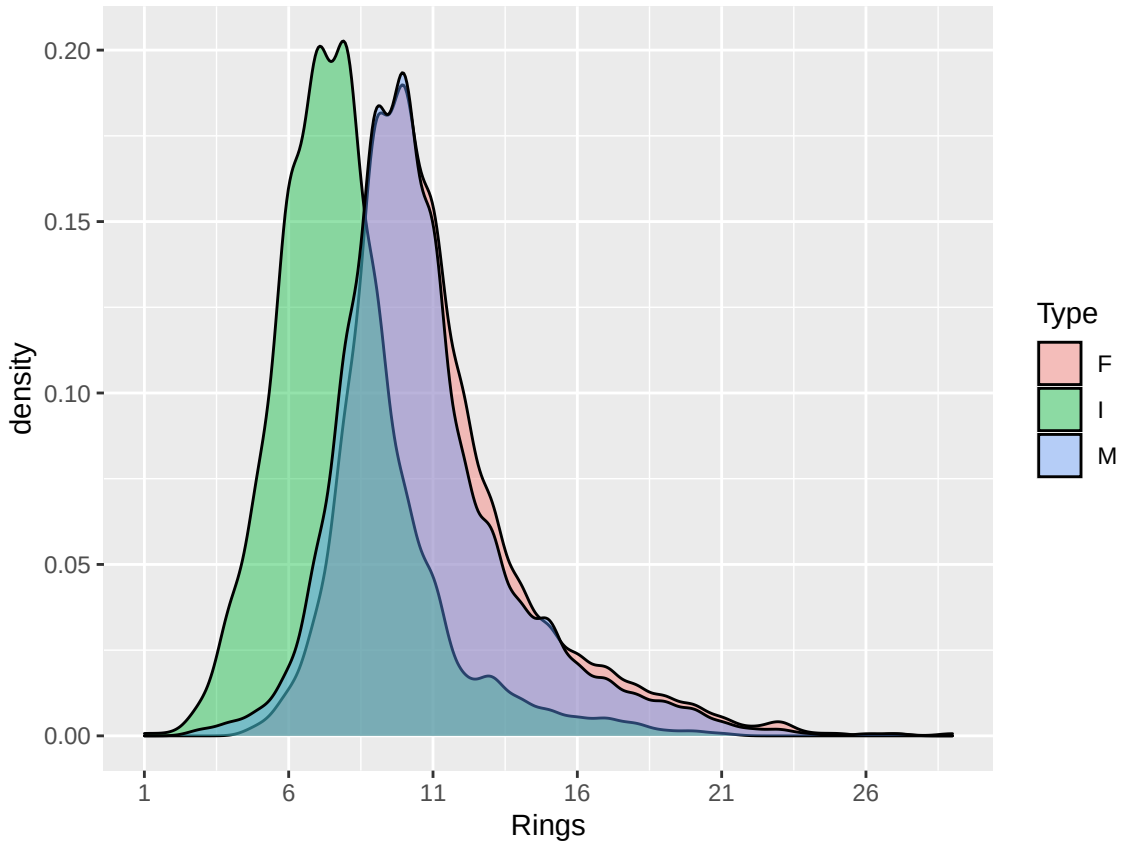
A similar but *arguably* more effective plot is a **density plot.** Density plots also show the distribution of continuous data but without arbitrary breaks in the data. Density plots show a curve that represents the probability of sampling any value. The result looks like a "normal curve" or "bell curve" that you might be familiar with. You make a density plot with `geom_density`, as shown here with the same data.

```
abalone %>%
  ggplot() +
  geom_density(aes(x = Rings)) +
  scale_x_continuous(breaks = seq(1,30, by = 5))
```



You can facet the data but you can instead use the `group` or `fill` argument to plot multiple distributions. Here, I use `fill = Type` to color-code the three abalone types. `alpha` sets the transparency of the fill color, so you can identify all three density plots. `alpha` can range from 0 (no fill) to 1 (solid color; default) but middle values are most useful. I used 0.42 for non-arbitrary reasons.

```
abalone %>%
  ggplot() +
  geom_density(aes(x = Rings,
                   fill = Type),
               alpha = 0.42) +
  scale_x_continuous(breaks = seq(1,30, by = 5))
```



Relationships: scatterplots

Read Scatter plots from *Fundamentals of Data Visualization* by Claus O. Wilke.

Scatterplots are useful to detect relationships between data, as you observed in previous assignments. If some variables in your data have a strong relationship, then using all of the related variables will bias your analyses. You would then choose carefully which variables to discard and which to use.

For assignment 10 you will use the same dataset used by Zuur et al. (2010) to generate some of their figures. You will then use these techniques to explore another dataset in assignment 11.