Introduction to programming in R: visualization

Erik Spence

SciNet HPC Consortium

15 March 2023

Today's slides

Today's slides can be found here. Go to the "Introduction to programming in R" page, under Lectures, "Visualization".

`https://scinet.courses/1277`

Today's Lecture

Today's class will explore visualization in R.

- ggplot,
- modifying aesthetics,
- professional plotting guidelines.

Visualization in R

R has built-in capabilities to graphically present your data. However, there are a number of high-quality visualization packages also available in R.

- 'ggplot2', a powerful layer system that makes it easy to combine different sources of data.
- 'Lattice', very strong at visualizing multi-variate data.
- 'Plotly', makes interactive plots.
- 'highcharter', used for dynamic charting.
- 'Leaflet', builds interactive maps.

Today we will explore the capabilities of ggplot.

Installing ggplot2

To install 'ggplot2', the package we'll use for this class, run the following command in your R prompt.

```
>
> install.packages('ggplot2')
Installing package into "/Users/ejspence/R/x86_64-pc-linux-gnu-library/4.2"
...
* DONE (ggplot2)
>
> library(ggplot2)
>
```

Now we're ready to go.

Visualization using ggplot2

The 'gg' in 'ggplot2' stands for Grammar of Graphics, a graphics concept which describes plots by using a "grammar". Basically a grammar of graphics is a framework which follows a layered approach to describe and construct visualizations in a structured manner.

According to the 'ggplot2' approach, a plot can be divided into different fundamental parts:

Plot = Data + Aesthetics + Geometry.

The principal components of every plot can be defined as follows:

- Data is a data frame,
- Aesthetics is used to indicate 'x' and 'y' variables. It can also be used to control the colour, the size or the shape of points, the height of bars, etc.
- Geometry corresponds to the type of graphics (histogram, box plot, line plot, density plot, scatter plot, etc.)

Visualization using ggplot2, continued

The 'ggplot2' package allows us to create really nice plots.

- 'ggplot()' function is more flexible and robust than ‘qplot‘ for building a plot piece by piece.
- 'qplot()' is a quick plot function which is easy to use for simple plots. This is now deprecated, and should not be used.

The generated plot can be kept as a variable and then printed at any time using the function 'print()'.

You can save your plot in the current working directory using the command 'ggsave'.

```
>
> ggsave("plot.png", width = 5, height = 5)
>
```

Box plot

The box plot displays the distribution of a continuous variable. It visualizes the following statistics:

- the median;
- first and third quartiles (the 25th and 75th percentiles) and all "outlying" points individually;
- whiskers extend 1.5 * IQR from the first or third quartiles (where IQR is the inter-quartile range, or distance between the first and third quartiles), and
- outlying points.

In particular, box plots are used to display continuous variables that are in different categories.

Box plot, example

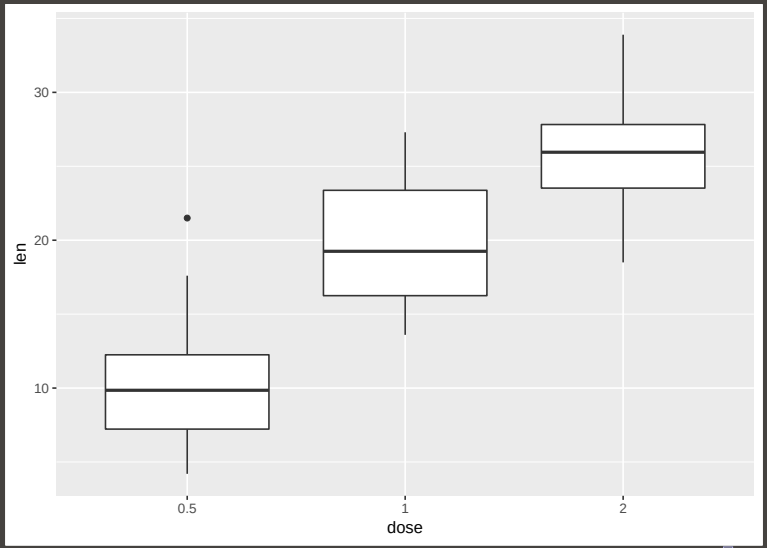
We use ggplot to build our image.

- We need to make sure that the dose column of our data is a factor.
- The name of the data frame is specified as the data.
- The Aesthetics are indicated using the 'aes' function. This is usually specified within the ggplot call.
- We assign a 'handle' to the plot, 'p'.
- We can then continue to add whatever pieces we want to image. In this case we add the Geometry to the plot.

```
> library(ggplot2)
>
> my.data <- ToothGrowth
>
> my.data$dose <- as.factor(my.data$dose)
>
> p <- ggplot(data = my.data,
+             aes(x = dose, y = len))
>
> p <- p + geom_boxplot()
>
> p
>
```

To use ggplot you almost always need to have your data in a data frame.

Box plot, example, continued



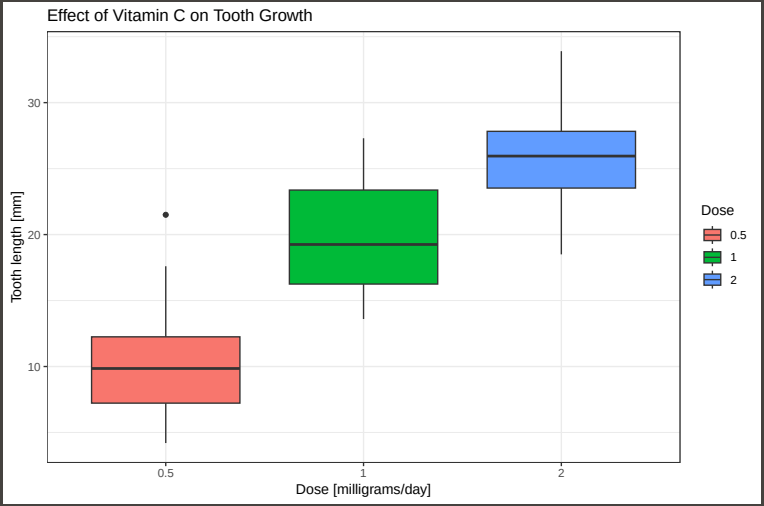
Improving our box plot

Let's add some labels.

- The aesthetic mappings, like 'color', 'shape' and 'fill' map to categorical variables.
- The 'labs' specifies plot labels.
  - 'x' and 'y' are the axis labels.
  - 'title' is the plot title
  - 'fill' is the title of the legend.
- theme_bw removes the grey background and puts in a black-and-white one.

```
>
> ggplot(data = my.data,
+        aes(x = dose, y = len, fill = dose)) +
+    geom_boxplot() +
+    labs(x = "Dose [milligrams/day]",
+         y = "Tooth length [mm]",
+         title = "Effect of Vitamin C on Tooth Growth",
+         fill = "Dose") +
+    theme_bw()
>
```

Box plot, example, continued



Improving our box plot more

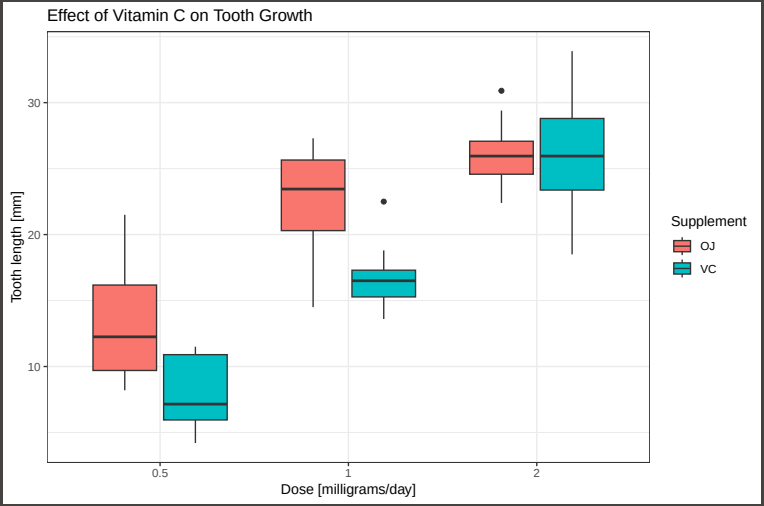
We can add additional information on our plot by specifying that we want to fill our boxes based on the supplement type, rather than the dose.

This means indicating whether the supplement was orange juice (OJ) versus ascorbic acid (VC).

The 'supp' column is already a factor.

```
>
> ggplot(data = my.data,
+        aes(x = dose, y = len, fill = supp)) +
+    geom_boxplot() +
+    labs(x = "Dose [milligrams/day]",
+         y = "Tooth length [mm]",
+         title = "Effect of Vitamin C on Tooth Growth",
+         fill = "Supplement") +
+    theme_bw()
>
```

Box plot, example, continued



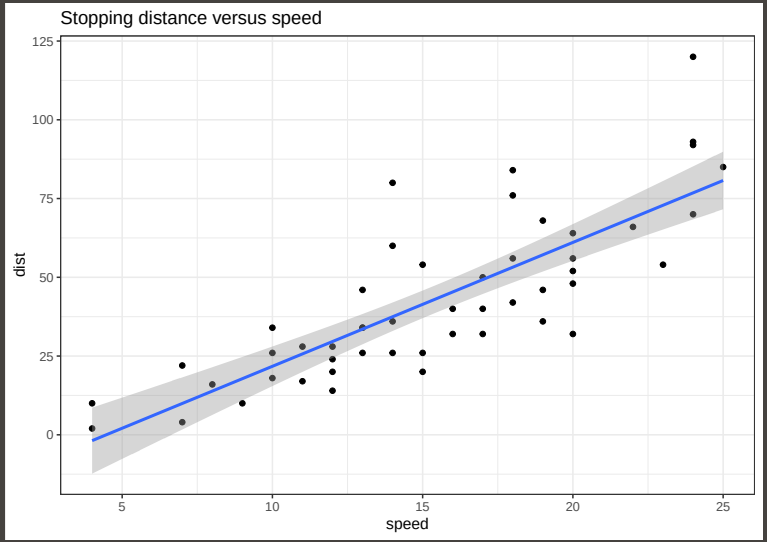
Linear model example

The cars data set is built into R. Here we will overlay a linear model onto our data.

- geom_point: draws data points on the plot,
- geom_smooth: adds some sort of smoothing. Here we indicate that we want a linear model.

```
>
> ggplot(data = cars,
+        aes(x = speed, y = dist)) +
+    geom_point() +
+    geom_smooth(method = 'lm') +
+    ggtitle("Stopping distance versus speed") +
+    theme_bw()
>
```

Linear model example, continued



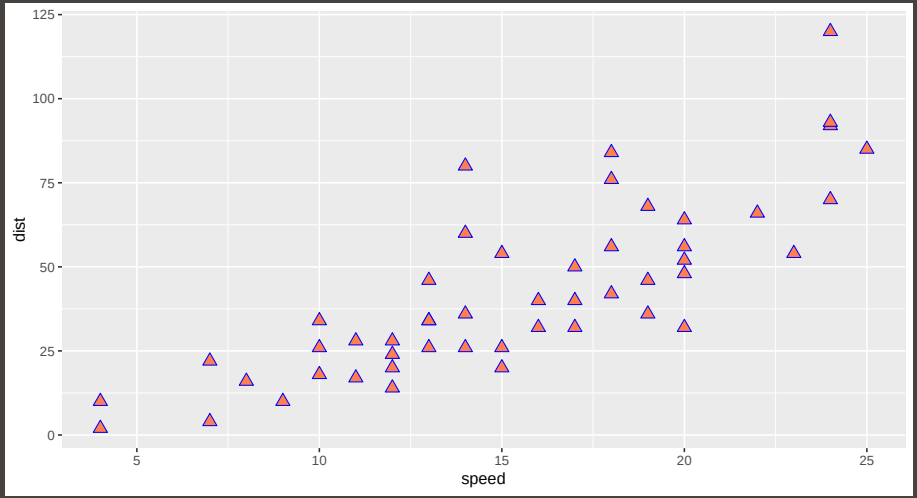
Scatter plot example

The geom_point function takes a number of different optional arguments.

- size: the size of the data points.
- color: the colour of either the border, or the whole symbol.
- fill: the colour to fill in the symbol. This appears to only work with certain symbols.
- shape: the shape of your data points.

```
>
> ggplot(data = cars,
+        aes(x = speed, y = dist)) +
+    geom_point(size = 3, color = "blue",
+               fill = "coral",
+               shape = 24)
>
```

Scatter plot example, continued



Scatter plot example 2

If you specify an aesthetic, the geom_point function can also

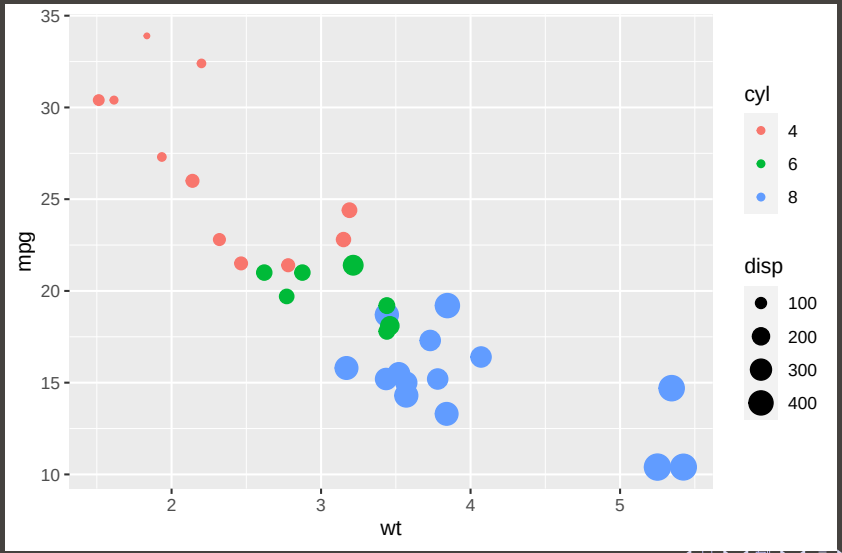
- colour the points by column value, either continuous or categorical.
- change the shapes of points based on category.
- change the size based on column value.

If you put the 'aes' call under ggplot it will apply to the whole plot. If you want the aesthetic change to only apply to a part of the plot, put it in the specific section you're trying to change.

```
>
> my.data <- mtcars
>
> my.data$cyl <- as.factor(my.data$cyl)
>
> ggplot(data = my.data,
+        aes(x = wt, y = mpg)) +
+    geom_point(aes(size = disp,
+                   color = cyl))
>
```

Note that the 'disp' column is a continuous value.

Scatter plot example 2, continued



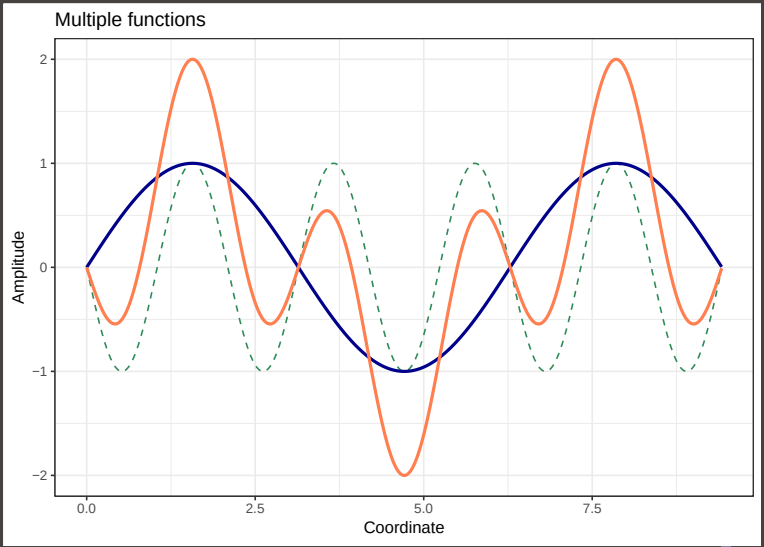
Line plot

The geom_line function is used to create line plots.

- It takes the usual aesthetic arguments. It also takes
- 'linetype', which changes the type of line displayed.
- 'linewidth', which changes the line's thickness.

```
> x <- seq(0, 3 * pi, len = 100)
> fast.wave <- sin(3 * x - pi)
> slow.wave <- sin(x)
> total.wave <- fast.wave + slow.wave
> my.data <- data.frame(x, fast.wave, slow.wave,
+                       total.wave)
>
> ggplot(data = my.data, aes(x = x)) +
+    geom_line(aes(y = fast.wave), col = "seagreen",
+              linetype = 2) +
+    geom_line(aes(y = slow.wave), col = "darkblue",
+              linetype = 1, linewidth = 1) +
+    geom_line(aes(y = total.wave), col = "coral",
+              linewidth = 1) +
+    labs(x = "Coordinate", y = "Amplitude",
+         title = "Multiple functions") +
+    theme_bw()
```

Line plot, continued



Other types of plot

The ggplot2 package also supports many other types of plots. You can use the following functions combined with the command 'ggplot':

- geom_histogram(), to plot histograms,
- geom_bar(), to display bar plots,
- geom_line(), to draw lines,
- geom_errorbar, for error bars,
- geom_contour(), for contour plots,
- and many others.

You can find most of ggplot2 commands on the following cheat sheet:
`https://github.com/rstudio/cheatsheets/blob/master/data-visualization-2.1.pdf`

Professional plotting guidelines

Scientists must make their work presentable. That means making a serious effort to make your results easy for your audience to understand. Suggestions:

- DO label everything: axes, lines, data.
- DO put units on your axis labels, including colour bars.
- DO use legends, where appropriate.
- DO adjust the font size of axis and tick labels so that they can actually be read.
- DO NOT put a title label over your plot if using a caption (for talks and papers).
- DO NOT use colours that cannot be read on a white background (yellow, orange, light green, cyan). This is especially important for figures used in talks.
- DO make your data fill the plot.

Professional plotting guidelines, continued

It's also important to consider file types and sizes. More suggestions:

- DO set the image size and resolution to that requested by the journal you're submitting to.
- DO NOT use bitmap or stroke fonts for your plot. These cannot be rescaled properly, which is often needed for publication. Use vector (the default for R) or TrueType fonts.
- If possible, DO NOT use image file types that cannot be scaled (bitmap, jpeg). Use EPS or PDF.
- DO NOT leave a bunch of white space around the outside of your plots.
- DO make a script (in R or whatever language), whose sole purpose is to make that plot for your paper.

Summary

A review of today's class.

- The ggplot2 library is very powerful. As a general rule it's used over base R's builtin plotting functionality.
- To use ggplot you specify the Data + Aesthetics + Geometry.
- Sometimes the aesthetics are added to the geometry as well, for changing plotting shapes, colours or sizes.
- We've barely scratched the surface of what you can do with ggplot. There are many examples online.
- Always be sure to create professional-quality plots.