Data visualization tutorial: exploring data and telling stories using ggplot2*

Peter Carbonetto *University of Chicago*

In this lesson, we will learn how to use ggplot2 to create simple yet effective data visualizations. The ggplot2 package is an incredibly powerful plotting interface that extends the base plotting functions in R. **This tutorial is also a Google doc.**

Some motivation

A good figure is an important part of an impactful research paper or presentation. A good figure is one that *tells an interesting story.*

Almost inevitably, creating a good figure takes iteration and refinement. You will rarely get it right on the first try.

For these reasons, taking the *programmatic approach* to creating plots is very powerful. It allows you to:

a. Create an endless variety of plots.
b. Reuse code to quickly create and revise plots.

In this tutorial we will explore the programmatic approach to plotting using **ggplot2**.

Setup

Do I have what I need? Download the tutorial materials from GitHub, and make sure you know where to find them.

If you have not already done so, install these packages:

```
install.packages("ggplot2")
install.packages("cowplot")
install.packages("ggrepel")
```

If you are running your code in a Jupyter notebook or in Google Colab, I recommend running this line of code so that the outputs look the same as they do in RStudio:

```
options(jupyter.rich_display = FALSE)
```

*This document is included as part of the Data Visualization tutorial packet for the BSD qBio Bootcamp, University of Chicago, 2024. **Current version**: August 09, 2024; **Corresponding author**: pcarbo@uchicago.edu. Thanks to Stefano Allesina, John Novembre, Stephanie Palmer and Matthew Stephens for their guidance.

Do smaller dogs live longer?

The study of dogs is a surprisingly fruitful area of research! In this tutorial, we will make use of some data that was made available by the authors of a 2008 *Genetics* article, *Single-nucleotide-polymorphism-based association mapping of dog stereotypes.* These data are stored in a CSV file.

Our main analysis aim is to investigate the anecdotal claim that smaller breeds (such as Chihuahuas) live longer than larger breeds (such as Saint Bernards).

Our first ggplot plot

It will take us some time to understand *how* ggplot works, but let's start by quickly creating our first ggplot.

```
library(ggplot2)
dogs <- read.csv("dogs.csv",stringsAsFactors = FALSE)
ggplot(dogs,aes(x = height,y = weight)) +
  geom_point()
```

A first look at the data

When you load data into R for the first time, it is important to get a basic understanding of the data frame and its contents.

```
# Add your code here.
```

What different types of data are in this table?

The often overlooked scatterplot

In this tutorial, we will learn about ggplot2 through one of the most basic data visualizations: *the scatterplot.*

The scatterplot is easily overlooked because it is so simple. But it can be one of the most effective ways to visualize relationships. And it has many uses.

With embellishments (adding labels, varying color, shape, size, *etc*), scatterplots can produce stunning visualizations.

Our first ggplot2 (with "ugly" code)

Recall, our objective is to investigate the relationship between size and longevity in dogs. We'll use weight as a proxy for size.

```
# Add your code here.
```

Now for some more elegant code that can accomplish the same thing. (This more elegant code comes with a "for experts only" warning.)

```
# Add your code here.
```

For the moment I want to focus on the "uglier" code because it highlights better the key elements of a ggplot2 plot:

1. The first input is the data (stored in a data frame).
2. The second input is an "aesthetic mapping", created using `aes` that defines how columns are mapped to features of the plot (axes, shapes, colors, *etc*).
3. A "geom", short for "geometric object", specifies the type of plot. ggplot2 has an excellent on-line reference at ggplot2.tidyverse.org explaining all the "geoms", from bar charts to contour plots, with code examples for each.
4. ggplot2 outputs a *ggplot object* `p`, which can be drawn to the screen with `print(p)` or just `p` (then hit "enter" or "return").

The distinguishing feature of ggplot2 is that plots are created by *adding layers*. This layering allows for infinite variety of plots to be created. The layering approach means that ggplot2 is easily extendible, and many R packages have been developed to enhance ggplot2. (We will use two of these packages, ggrepel and cowplot.)

A few improvements

No plot is ever right the first time. *What are ways we can improve our plot?* Add code for your improved plot here:

```
# Add your code here.
```

In ggplot, plots can be improved either by modifying the inputs to the functions you are already using (for example, `geom_point` has many settings that can be fiddled with), or by *adding layers*. The ggplot online reference has many helpful examples that illustrate the variety of ways plots can be improved.

Save your work

We've worked hard, and make considerable progress. This is a good point to save our work in an image file that can be shared with others. *What type of image file should we use?*

```
# Add your code here.
```

D. melanogaster

Plot the best-fit line

It has been estimated that an increase of 28 lbs in a dog's body weight corresponds to about a 1-year drop in expected lifespan (with a maximum lifespan of about 13 years). How well does this estimate agree with our data? Let's investigate this question by plotting the line that *best fits* these data (specifically, this is the "least-squares" fit).

```
# Add your code here.
```

Now let's add this "best fit" line to the plot. In case we make a mistake, let's call our new plot "p2" to avoid writing over our previous plot:

```
# Add your code here.
```

Now let's add another another "abline" layer to compare our estimate against the previous estimate (and name the new plot object "p3"):

```
# Add your code here.
```

Notice how easy it was to add layers to an existing plot!

Which breeds fit the trend, and which don't?

It would be helpful if we could tell which breeds are being plotted. Adding text labels to a ggplot is also done by adding a layer.

There's one catch here—there simply isn't enough real estate on the plot to accommodate all the breed names. This is a great opportunity to play around with a clever package, **ggrepel**, that adds the labels in a way that makes them more readable, and only adds them to the plot when possible. Let's appreciate how simply this sophisticated plot is created.

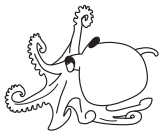
```
# Add your code here.
```

Notice that the labels are automatically redrawn as the plot is resized. Give it a try!

A QTL for weight

In the *Genetics* paper, the strongest *quantitative trait locus* (QTL) for weight was a QTL on chromosome 7. Using your ggplot coding skills, create a scatterplot to visualize the relationship between weight and the allele frequency at this QTL. *Is your code reproducible? i.e., could I run your code to get the same plot?*

```
# Add your code here.
```



O. bimaculoides

Exploration: a surprising subtlety with color

So far, we have focussed on using the co-ordinate plane to visualize relationships. But other "aesthetics"—color, size, shape, *etc*—can also be used to tell an evocative story. In the last chapter of this tutorial, we will explore the use of color to visualize relationships. In so doing, we will discover a complication.

We learned that the "shortcoat" column had values of 0 or 1 (with a few NAs).

Let's try varying the color of the points using the shortcoat column:

```
p <- ggplot(dogs,aes(x = weight,y = aod,color = shortcoat)) +
  geom_point() +
  theme_cowplot()
p
```

Are we happy with this plot? How could it be improved? Write your code for an improved plot here:

```
# Add your code here.
```

Optional: Try varying the shape of the points instead of color. If you have have written your ggplot code well, this should only involve a slight change to your code. You can use `scale_shape_manual` to select the shapes. Running `plot(0:23,pch = 0:23)` will give you the full list of shapes to choose from.

```
# Add your code here.
```

More on color

Carefully chosen colors can often be the difference between an effective visualization and an ineffective one. One the few complaints I have with ggplot2 is that the default colors choices are quite poor. So when you use ggplot2, you will often need to make adjustments to the colors. One rule-of-thumb is that "warmer" colors such as red and orange tend to draw the reader's attention.

There are several good resources on use of color in data visualization, and I will mention a couple here: Color Brewer (https://colorbrewer2.org); and a short article, "Color blindness", written by Bang Wong (*Nature Methods*, 2011). For more discussion, see the "Fundamentals of data visualization" book by Claus Wilke.

In our scatterplot, the best color choice is less clear, so I will let you experiment with different choices. To override the color defaults, add a `scale_color_manual` layer:

```
# Add your code here.
```

There are several different ways to specify colors. I prefer specifying colors by name; to get the full list of color names, run `colors()`.

The function that controls the color of a discrete variable has an odd name: `scale_color_manual`. This is because, in ggplot2, all methods that control the mapping of variables to colors, shapes, sizes, axes, *etc*, start with `scale_`.

Obviously, there is much more to ggplot2. But once you are comfortable with these basic elements, you will find that almost everything else in ggplot2 is a variation of what we covered in this lesson.

T. thermophila

Programming challenge: Mapping the genetic basis of physiological and behavioral traits in outbred mice

In this programming challenge, you will use simple visualizations to gain insight into biological data.

You are working in a lab studying the genetics of physiological and behavioral traits in mice. The lab has just completed a large study of mice from an outbred mouse population, "CFW" ("Carworth Farms White"). The aim of the study is to identify genetic contributors to variation in behaviour and musculoskeletal traits.

Note hese challenges are roughly ordered in increasing level of complexity. *Do not be discouraged if you have difficulty completing every one.*

Collaboration strategy

Before diving into the problems, first agree on a collaboration strategy with your teammates. Important aspects include communication and co-ordination practices, and setting goals and deadlines. How will your team collaborate on code, and share solutions? (Consider online resources such as Etherpad or the UofC-hosted Google Drive.) The aim is not just to tackle the challenges, but also to do so collaboratively.

Instructions

- Locate the files for this exercise on your computer (see "Materials" below).
- Make sure your R working directory is set to the same directory containing the tutorial materials; use `getwd()` to check this. (If you need to change your R working directory, you can either use the `setwd()` function or, in RStudio, you can select **Session > Set Working Directory > Choose Directory…**.)
- Some of the programming challenges require uploading an image file containing a plot. Use `ggsave` to save your plot as a file. Any standard image format (e.g., PDF, PNG) is fine.
- No additional R packages are needed beyond what we used in the examples above.

Materials

- **pheno.csv:** CSV file containing physiological and behavioral phenotype data on 1,219 male mice from the CFW outbred mouse stock. Data are from Parker *et al*, 2016. Use **readpheno.R**

to read the phenotype data from the CSV file into a data frame. After filtering out some of the samples, this script should create a new data frame, `pheno`, containing phenotype data on 1,092 samples (these are the rows of the data frame).

- **hmdp.csv:** CSV file containing bone-mineral density measurements taken in 878 male mice from the Hybrid Mouse Diversity Panel (HMDP). Data are from Farber *et al*, 2011. To load the data into your R environment, run this code:

```
hmdp <- read.csv("hmdp.csv",stringsAsFactors = FALSE)
```

  This will create a data frame, `hmdp`, containing BMD data on 878 mice (these are the rows of the data frame).

- **gwscan.csv:** CSV file containing results of a "genome-wide scan" for abnormal BMD. (The association *p*-values were computed using GEMMA 0.96.) To read the results of the genome-wide scan, run the following code:

```
gwscan <- read.csv("gwscan.csv",stringsAsFactors = FALSE)
gwscan <- transform(gwscan,chr = factor(chr,1:19))
```

  This will create a data frame, `gwscan`. Each row of the data frame is a genetic variant (a single nucleotide polymorphism, or "SNP"). The columns are chromosome ("chr"), base-pair position on the chromosome ("pos"), and the *p*-value for a test of association between variant genotype and trait value ("abnormalBMD"). The value stored in the "abnormalBMD" column is $-\log_{10}(P)$, where P is the *p*-value.

- **geno_rs29477109.csv:** CSV file containing estimated genotypes at one SNP (rs29477109) for 1,038 CFW mice. Use the following code to read the genotype data into your R environment:

```
geno <- read.csv("geno_rs29477109.csv",stringsAsFactors = FALSE)
geno <- transform(geno,id = as.character(id))
```

  This will create a new data frame, `geno`, with 1,038 rows (samples). The genotypes are encoded as "dosages"—that is, the expected number of times the alternative allele is observed in the genotype. This will be an integer (0, 1, 2), or a real number between 0 and 2 when there is some uncertainty in the estimate of the genotype. For this SNP, the reference allele is T and the alternative allele is C. Therefore, dosages 0, 1 and 2 correspond to genotypes TT, CT and CC, respectively (genotypes CT and TC are equivalent).

- **wtccc.png:** Example genome-wide scan ("Manhattan plot") from Fig. 4 of the WTCCC paper. The *p*-values highlighted in green show the regions of the human genome most strongly associated with Crohn's disease risk.

A couple tips

- Some "geoms" you may find useful: `geom_point`, `geom_histogram`, `geom_boxplot`.
- In some cases it may be useful to convert to a *factor*.

Part A: Exploratory analysis of muscle development and conditioned fear data

Your first task is to create plots to explore the data.

1. A basic initial step in an exploratory analysis is to visualize the distribution of the data. It is

often convenient if the distribution is normal, or "bell shaped".

- Visualize the empirical distribution of tibialis anterior (TA) muscle weight (column "TA") with a histogram. Units are mg. *Hint:* Try using function `geom_histogram`.
- Is the distribution of TA weight roughly normal? Are there mice with unusually large or unusually small values ("outliers")? If so, how many "outliers" are there?

2. It is also important to understand relationships among measured quantities. For example, the development of the tibia bone (column "tibia") could influence TA muscle weight. Create a scatterplot (`geom_point`) to visualize the relationship between TA weight and tibia length. (Tibia length units are mm.) Based on this plot, what can you say about the relationship between TA weight and tibia length? Quantify this relationship by fitting a linear model, before and after removing the outlying TA values. (*Hint*: Use the `lm` and `summary` functions. See also the "r.squared" return value in `help(summary.lm)`.)
3. The "AvToneD3" column contains data collected from a behavioral test called the "Conditioned Fear" test.
   - Visualize the empirical distribution of AvToneD3 ("freezing to cue") with a histogram. Is the distribution of AvToneD3 approximately normal?
   - Freezing to cue is a proportion (a number between 0 and 1). A common way to obtain a more "normal" quantity is to transform it using the "logit" function.[1] Visualize the empirical distribution of the logit-transformed phenotype. Is the transformed phenotype more "bell shaped"? After the transformation, do you observe unusually small or unusually large values?
   - A common concern with behavioral tests is that the testing devices can lead to measurement error. It is especially a concern when multiple devices are used, as the devices can give slightly different measurements, even after careful calibration. Create a plot to visualize the relationship between (transformed) freezing to cue and the device used ("FCbox" column). *Hint:* Try a boxplot (`geom_boxplot`). Based on this plot, does the apparatus used affect these behavioral test measurements?

Part B: Exploratory analysis of bone-mineral density data

Now you will examine data on bone-mineral density (BMD) in mice. This is a trait that is important for studying human diseases such as osteoporosis (units are mg/cm^2).

- Plot the distribution of BMD in CFW mice (see column "BMD"). What is most notable about the distribution?
- Compare these data against BMD measurements taken in a "reference" mouse population, the Hybrid Mouse Diversity Panel. To compare, create two histograms, and draw one on top of the other. What difference do you observe in the BMD distributions? For a correct comparison, consider that: (1) BMD in CFW mice was measured in the femurs of male mice only; (2) BMD in HMDP mice was recorded in g/cm^2. *Hints:* Potentially useful functions include `xlim` and `labs` from the ggplot2 package, and `plot_grid` from the cowplot package. The `binwidth` argument in `geom_histogram` may also be useful.

[1] R code: `logit <- function(x) log((x + 0.001) / (1 - x + 0.001))`

Part C: Mapping the genetic basis of osteopetrotic bones

A binary trait, "abnormal BMD", was defined that signals whether an individual mouse had "abnormal", or osteopetrotic, bones. It takes a value of 1 when BMD falls on the "long tail" of the distribution (BMD greater than 90 mg/cm^2), otherwise zero.

GEMMA was used to carry out a "genome-wide association study" (GWAS) for this trait; that is, support for association with abnormal BMD was evaluated at 79,824 genetic variants (single nucleotide polymorphisms, or "SNPs") on chromosomes 1–19. At each SNP, a p-value quantifies the support for an association with abnormal BMD.

1. Your first task is to get an overview of the association results by creating a "Manhattan plot". Follow as closely as possible the provided prototype, **wtccc.png**, which shows a genome-wide scan for Crohn's disease. (Don't worry about highlighting the strongest p-values in green.) *Hints:* Replicating some elements of this plot may be more challenging than others, so start with a simple plot, and try to improve on it. Recall the adage that creating plots requires relatively little effort *provided the data are in the right form*—consider adding appropriate columns to the `gwscan` data frame before writing your ggplot code. Functions from the ggplot2 package that you may find useful include `geom_point`, `scale_color_manual` and `scale_x_continuous`.
    - In your plot, you should observe that the most strongly associated SNPs cluster closely together in small regions of the genome. This is common—it is due to a genetic phenomenon known as linkage disequilibrium (LD). It is a consequence of low recombination rates between markers in small populations. How many SNPs have "strong" statistical support for association with abnormal BMD, specifically with a $-\log_{10} p$-value > 6? How many "distinct" regions of the genome are strongly associated with abnormal BMD at this p-value threshold?
    - What p-value does a $-\log_{10} p$-value of 6 correspond to?
    - Using your plot, identify the "distinct region" (this is called a "quantitative trait locus", or QTL) with the strongest association signal. What is, roughly, the size of the QTL in Megabases (Mb) if we define the QTL by base-pair positions of the SNPs with $-\log_{10} p$-value > 6? Using the UCSC Genome Browser, get a rough count of the number of genes that are *transcribed* in this region. (Parker *et al*, 2016 identified *Col1a1* as a candidate BMD gene.) *Hint:* All SNP positions are based on *NCBI Mouse Genome Assembly 38* (mm10, December 2011).
2. Your next task is to visualize the relationship between genotype and phenotype. From the genome-wide scan of abnormal BMD, you should find that rs29477109 is the SNP most strongly associated with abnormal BMD. Here you will look closely at the relationship between BMD and the genotype at this SNP. In developing your visualization, consider that:
    - The samples listed in the phenotype and genotype tables are not the same. So you will need to align the two tables to properly show analyze the relationship. *Hint:* Function `match` could be useful for this.
    - The genotypes, stored in file **geno_rs29477109.csv**, are encoded as "dosages" (numbers between 0 and 2). You could start with a scatterplot of BMD vs. dosage. But ultimately it is more effective if the genotypes (CC, CT and TT) are plotted instead. *Hints:* In effect, what you need to do is convert from a continuous variable (dosage) to a discrete variable (genotype). One approach is to create a `factor` column from the "dosage"

column. (For dosages that are not exactly 0, 1 or 2, you could simply round to the nearest whole number.) A boxplot is recommended; see function `geom_boxplot`.

Based on your plot, how would describe (in plain language) the relationship between the genotype and BMD?

 *D. rerio*