Introduction to data summarizing and visualization

Jeff Oliver

26 August, 2019

The R programming language provides many tools for data analysis and visualization, but all the options can be daunting. This lesson provides an introduction to wrangling data in R and using graphics to tell a story with data.

Learning objectives

1. Understand the difference between files and R objects
2. Modify data for proper data hygiene
3. Summarize information from raw data
4. Visualize data to convey information

[DESCRIPTION OR MOTIVATION; 2-4 sentences that would be used for an announcement]

Getting started

The tools: R and RStudio

For this lesson, we will use the R programming language in the RStudio environment. RStudio provides a convenient interface for working with files and packages in R. If you have not done so already, install R and RStudio; details can be found on the installation page.

Preparing our workplace

Key to successful programming is organization. In RStudio, we use Projects to organize our work (a "Project" is really just a fancy name for a folder that contains all our files). For this lesson, we'll create a new project through the File menu (File > New Project). In the first dialog, select "New Directory", and select "New Project" in the second dialog. Next you'll be prompted to provide a directory name. This will be the name of our project, so we should give it an informative name. For this lesson, we will be using data from vegetation surveys of Tumacacori National Historical Park, for the directory name, enter "vegetation". We need also to tell RStudio where to put the lesson on our computer; for this lesson, we will place the folder on our Desktop, so it is easy to find. In your own work, you may find it better to place project folders in your Documents folder.

The last thing we need to do to set up our workspace is to use file organization that reinforces best practices. In general, there should be a one-way flow of information: we take information from *data* and write code to produce *output*. We want to avoid any output from messing up our data, so we create separate folders for each. We want to create two folders, one for our data and one for any output, which may include results of statistical analyses or data visualization. In the R console,

```
dir.create("data")
dir.create("output")
```

[Get data]

Data are csv at https://irma.nps.gov/DataStore/DownloadFile/569336 Webpage with files and other information at https://irma.nps.gov/DataStore/Reference/Profile/2233448

Data in R

Data *outside* R

[Open file in Excel]

[Load in data]

```
plot_data <- read.csv(file = "data/tumacacori-vegetation.csv")
```

[QA/QC]

`head` `tail` Point out `<NA>` in tail output

Cleaning up

Missing data

```
plot_data <- na.omit(plot_data)
```

Subsetting data

```
families_keep <- c("Amaranthaceae", "Brassicaceae", "Poaceae", "Fabaceae")
subset_data <- plot_data[plot_data$Family %in% families_keep, ]
subset_data$Family <- factor(subset_data$Family)
```

Summarizing data

Using `summary`

Summary statistics for groups

use `mean` and `sd` for subset(s) of data

Get % cover for each family/plot

```
install.packages("dplyr")
```

```
library("dplyr")
```

```
family_data <- subset_data %>%
  group_by(Plot_Code, Family, Community) %>%
  summarize(Family_Percent_Cover = sum(Percent_Cover))
```

Visualizing data

The ggplot2 package

In order to visualize the data, we are going to use the ggplot2 package. Like we did for the dplyr package, we will first need to install the package:

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

Now that the package in installed, we can load the functions into memory with the `library` function:

```
library("ggplot2")
```

First rule of plots

Draw it by hand first

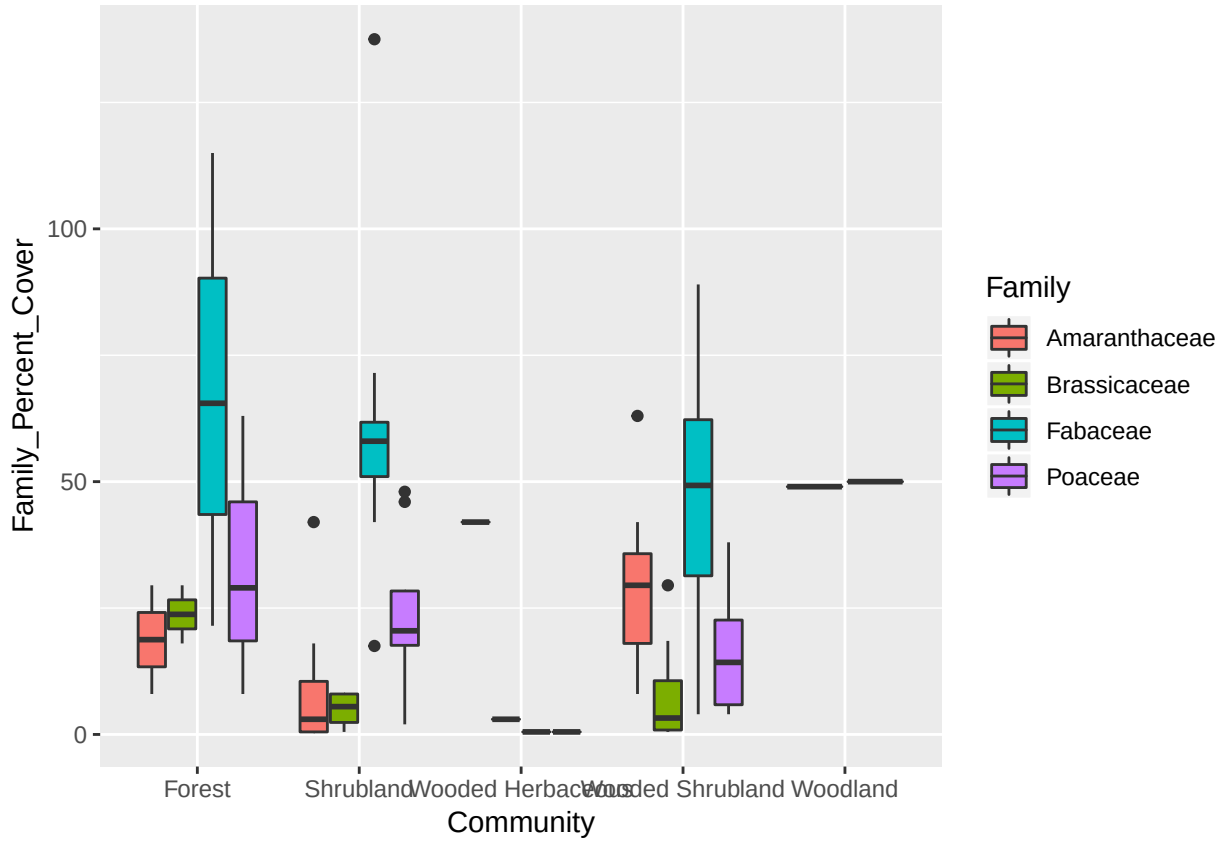
Code it second

Simple boxplot, no colors x is Community y is Percent_Cover group is family

Telling the story

Add group colors for story color is family

```
cover_plot <- ggplot(data = family_data,
                     mapping = aes(x = Community, y = Family_Percent_Cover,
                     fill = Family)) +
  geom_boxplot()
print(cover_plot)
```



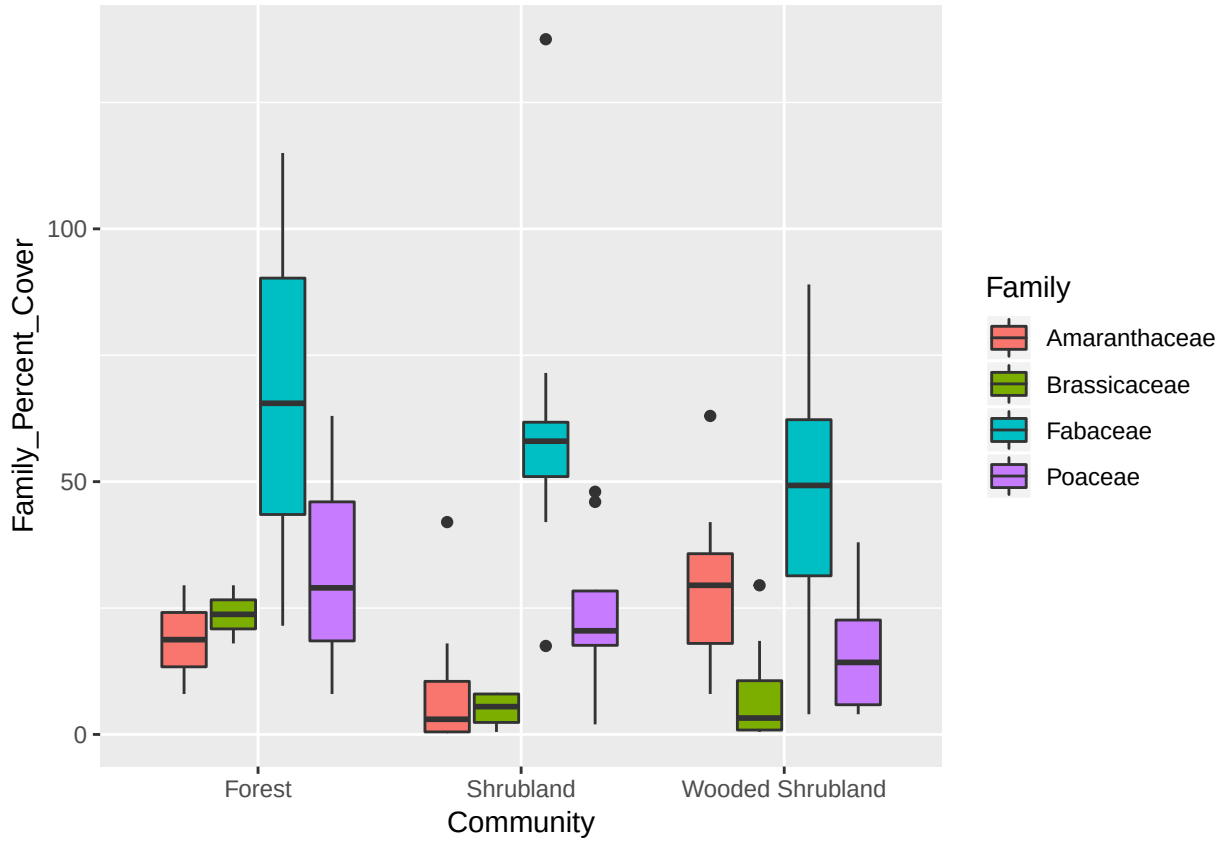
Look at the Wooded Herbaceous and Woodland - they're just lines. Why?

```
table(family_data$Family, family_data$Community)
```

```
##
##                 Forest Shrubland Wooded Herbaceous Wooded Shrubland
##   Amaranthaceae      2         7                 1                 7
##   Brassicaceae       2         4                 1                 8
##   Fabaceae           3         8                 1                 8
##   Poaceae            3         8                 1                 8
##
##                 Woodland
##   Amaranthaceae        0
##   Brassicaceae         0
##   Fabaceae             1
##   Poaceae              1
```

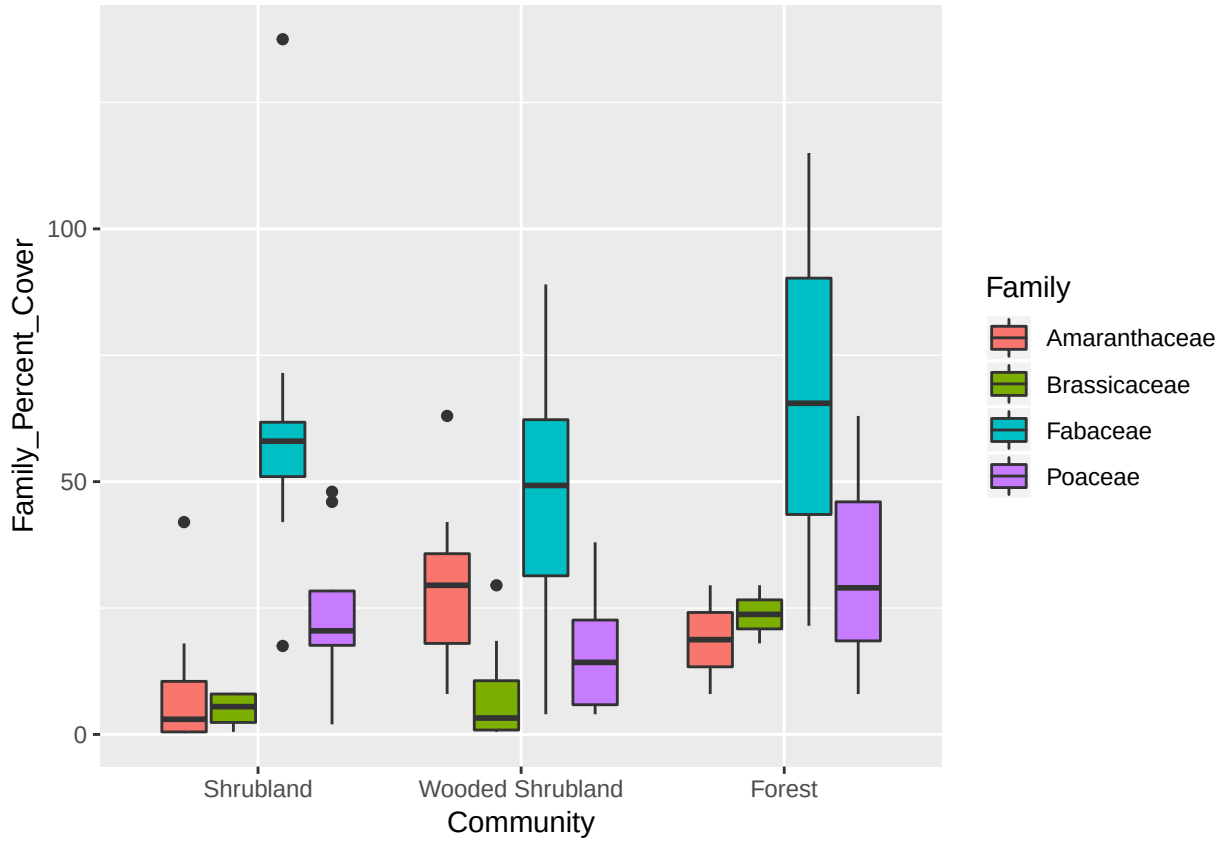
Since we only have those single plots for Wooded Herbaceous and Woodland, we should probably exclude them from our plot.

```
family_data <- family_data[!(family_data$Community %in% c("Wooded Herbaceous", "Woodland")), ]

cover_plot <- ggplot(data = family_data,
                     mapping = aes(x = Community, y = Family_Percent_Cover,
                     fill = Family)) +
  geom_boxplot()
print(cover_plot)
```



Re-level physio class so X-axis is shown in increasing tree cover

```
family_data$Community <- factor(family_data$Community,
                                levels = c("Shrubland", "Wooded Shrubland", "Forest"))

cover_plot <- ggplot(data = family_data,
                     mapping = aes(x = Community, y = Family_Percent_Cover,
                     fill = Family)) +
  geom_boxplot()
print(cover_plot)
```



Additional resources

- Official ggplot documentation
- A handy cheatsheet for ggplot
- A PDF version of this lesson

Back to learn-r main page

Questions? e-mail me at jcoliver@email.arizona.edu.