Data Wrangling & Species Richness

Jeff Oliver

30 October, 2019

[INTRODUCTORY SENTENCE]

Learning objectives

1. Demonstrate how to organize data in 'tidy' format
2. Develop quality control processes for data
3. Visualize measures of species richness and species diversity

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

Getting started

> Happy families are all alike; every unhappy family is unhappy in its own way - Leo Tolstoy

Tidy data

After Wickham 2014

1. Each variable forms a column.
2. Each observation forms a row.
3. Each type of observational unit forms a table.

Open the file in Excel. What can be done to make these data "tidy"?

First, the data are effectively broken in two, with the most recent observations separate from observations from preceding days. So in some cases, a row has more than one observation. To fix this, we move the cells from the preceding observations to rows after the most recent observations.

We also want to:

- Delete any columns without data
- Delete any rows without data
- Add a column title for date
- Make sure data for preceding observations lines up with column title (currently the Count column has species names and the Species column has count data)

Let's save this file as a CSV file so we can read it into R. We don't want to overwrite the original data file, so we'll call our file sweetwater-data-clean.csv.

Data wrangling

Now that the data are in CSV format, we can read it into R and take a look. We want to keep a record of all our work, so we are going to type commands into a script file and save the script. As we did before, we add a few lines of comments at the top of our script with some relevant information.

```
# Analyze sweetwater bird species richness
# Jeff Oliver
# jcoliver@email.arizona.edu
# 2019-11-01

bird_data <- read.csv(file = "data/sweetwater-data-clean.csv")
```

We can look at the first few rows with the `head` function:

```
head(bird_data)
```

```
##          Date Site Count              Species
## 1 October 4th    1     1         Mallard Duck
## 2               NA     1 Red-winged blackbird
## 3               NA     1      Gila woodpecker
## 4               NA     1                Finch
## 5               NA     1        Mourning Dove
## 6               NA     1          Phainopepla
```

And we see that in rows two through 6 there aren't any values for the Date and Site columns. This is a prime example of the difference between how a human interprets data and how a computer interprets the data. Let's open up our CSV file in Excel again. As humans, we can infer that the value for "Date" should be October 4th all the way to row 35. But the computer is stupid. The computer looks at row 3, sees no date, and infers that the date is *missing*. We need to explicitly tell the computer what the value for date is. Before we do that, we have the opportunity to go ahead and also change the date to the standard ISO date format of "YYYY-MM-DD". In this case, October 4th becomes 2019-10-04.

Go ahead and update all the dates and fill in values appropriately. You should also note that for observations prior to October 10th, there is an offset that (1) needs to be fixed and (2) prompts deletion of now empty rows.

Also, for the 2019-10-04 observations, do the same thing with data in the Site column - that is, fill in values where appropriate. Save the file as a CSV (you can use the same filename, sweetwater-data-clean.csv) and read it into R.

```
# Analyze sweetwater bird species richness
# Jeff Oliver
# jcoliver@email.arizona.edu
# 2019-11-01

bird_data <- read.csv(file = "data/sweetwater-data-clean.csv")
```

Looking at the updated data, we see those first six rows no longer have missing data.

```
head(bird_data)
```

```
##         Date Site Count              Species
## 1 2019-10-04    1     1         Mallard Duck
## 2 2019-10-04    1     1 Red-winged blackbird
## 3 2019-10-04    1     1      Gila woodpecker
## 4 2019-10-04    1     1                Finch
## 5 2019-10-04    1     1        Mourning Dove
## 6 2019-10-04    1     1          Phainopepla
```

We can also use the `summary` function to get an idea of the data set overall:

```
summary(bird_data)
```

```
##          Date         Site           Count                    Species
##  2019-09-28:20   Min.   :1.000   Min.   : 1.000   Gila Woodpecker :  3
##  2019-09-29:29   1st Qu.:2.000   1st Qu.: 1.000   Mallard Duck    :  3
##  2019-09-30:34   Median :3.000   Median : 1.000   mourning dove   :  3
##  2019-10-01: 5   Mean   :2.941   Mean   : 4.164   Blue Winged Teal:  2
##  2019-10-02:11   3rd Qu.:4.000   3rd Qu.: 3.000   Cooper's Hawk   :  2
##  2019-10-03: 1   Max.   :6.000   Max.   :80.000   Gambils quail   :  2
##  2019-10-04:34   NA's   :100                      (Other)         :119
```

There are still missing values in the Site column, but that's OK - those are observations from preceding days, which don't have any site data.

Species richness

OK, so now let's actually look at species richness. We are going to look at each date separately, and count the total number of species observed. To do this, we are going to use an additional package called dplyr; we will need to install the package and load it into memory before we try to use it. First, we check to see if the package is installed by looking at the Packages tab in the lower-right window. If dplyr *is* installed, we don't need to install it again. If dplyr is not listed, you can install it through the R console with the command:

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

Now that the package is installed, we also have to tell R to load the package into memory, so we add a call to `library` to our script. We often add all `library` commands to the top of our script.

```
# Analyze sweetwater bird species richness
# Jeff Oliver
# jcoliver@email.arizona.edu
# 2019-11-01

library(dplyr)
bird_data <- read.csv(file = "data/sweetwater-data-clean.csv")
```

You might see some read messages when you load dplyr. As long as you don't see the word "Error" everything should be fine.

Now we can use some dplyr functions to calculate the total number of species seen on each day.

```
richness <- bird_data %>%
  group_by(Date) %>%
  summarize(Total = n())
```

Let's look at the output by typing the name of our variable into the console:

```
richness
```

```
## # A tibble: 7 x 2
##   Date       Total
##   <fct>      <int>
## 1 2019-09-28    20
## 2 2019-09-29    29
## 3 2019-09-30    34
## 4 2019-10-01     5
```

```
## 5 2019-10-02    11
## 6 2019-10-03     1
## 7 2019-10-04    34
```

Pretty cool. We can even plot species richness for each day, too. We are going to use the ggplot2 package again, so first check to see if that package is installed. If not, install it by running `install.packages("ggplot2")` in the console, then update the script so that we have a call to `library(ggplot2)` at the top. Note we have to actually *run* this line in order to load ggplot2 into memory; just adding the code to script alone won't load it up.

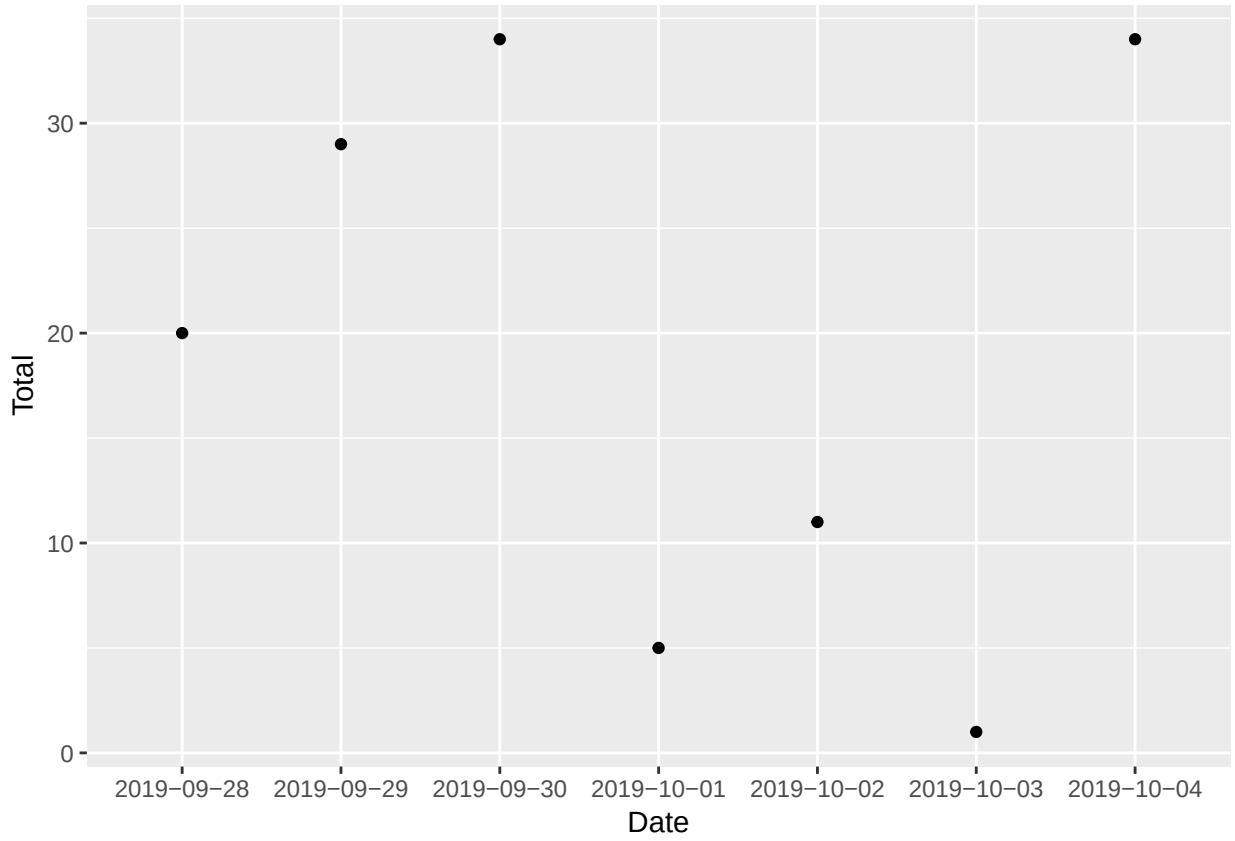
```
# Analyze sweetwater bird species richness
# Jeff Oliver
# jcoliver@email.arizona.edu
# 2019-11-01

library(dplyr)
library(ggplot2)
bird_data <- read.csv(file = "data/sweetwater-data-clean.csv")
```

Again, if you see red messages when loading ggplot2, that's OK, so long as the word "Error" doesn't show up.

To plot richness for each day,

```
richness_plot <- ggplot(data = richness,
                        mapping = aes(x = Date,
                                      y = Total)) +
  geom_point()
print(richness_plot)
```



Hey, a plot!

Let's take a moment though to think about the data that went into making these plots. In fact, let's look at the first six rows again:

```
head(bird_data)
```

```
##         Date Site Count              Species
## 1 2019-10-04    1     1         Mallard Duck
## 2 2019-10-04    1     1 Red-winged blackbird
## 3 2019-10-04    1     1      Gila woodpecker
## 4 2019-10-04    1     1               Finch
## 5 2019-10-04    1     1        Mourning Dove
## 6 2019-10-04    1     1          Phainopepla
```

Hmmm…row four has "Finch", which isn't a species. At Sweetwater Wetlands, House Finches and Lesser Goldfinches are common, and sometimes other rare finches show up. Another example is on row 17. We can take a look at this part of the data by specifying which rows to show. Here we ask for R to print out rows 15 through 20:

```
bird_data[15:20, ]
```

```
##          Date Site Count                              Species
## 15 2019-10-04    3     1                        Gambils quail
## 16 2019-10-04    3     1                          Phainopepla
## 17 2019-10-04    3     2                             Finch sp
## 18 2019-10-04    3     1                     Northern harrier
## 19 2019-10-04    3     1                   Unidentified raptor
## 20 2019-10-04    3     3 Humming Bird: Rubythroated, Black chin
```

In addition to "Finch sp" we all so see two more examples of observations not identified to species. Because we don't actually know what species these were, we will need to exclude them from our analyses.

Computers are stupid

Or, why quality control matters

We do not actually want to delete these records from our data, but rather we need a means of indicating whether or not they are identified to the species level. This should probably be done in the sweetwater-data-clean.csv file we've been working with. So open that file in Excel and add another column, immediately to the right of Species. Call this one "ID_to_species" and fill in values of 1 for those identified to species (e.g. Gila woodpecker would be scored as a 1) and values of 0 for those not identified to species (e.g. Finch sp and duck sp.). Save the file with the same name as before (sweetwater-data-clean.csv).

```
# Analyze sweetwater bird species richness
# Jeff Oliver
# jcoliver@email.arizona.edu
# 2019-11-01

library(dplyr)
library(ggplot2)
bird_data <- read.csv(file = "data/sweetwater-data-clean.csv")
```

Now we can update our richness calculation to only include observations that were identified to species. We do this by adding a filter to the process. Using the code we had before, before calling `group_by`, we use the `filter` function to only include those observations that have a 1 in the ID_to_species column.

```
richness <- bird_data %>%
  filter(ID_to_species == 1) %>%
  group_by(Date) %>%
  summarize(Total = n())
```

We can compare the old calculations for richness

```
## # A tibble: 7 x 2
##   Date       Total
##   <fct>      <int>
## 1 2019-09-28    20
## 2 2019-09-29    29
## 3 2019-09-30    34
## 4 2019-10-01     5
## 5 2019-10-02    11
## 6 2019-10-03     1
## 7 2019-10-04    34
```

to the new ones:

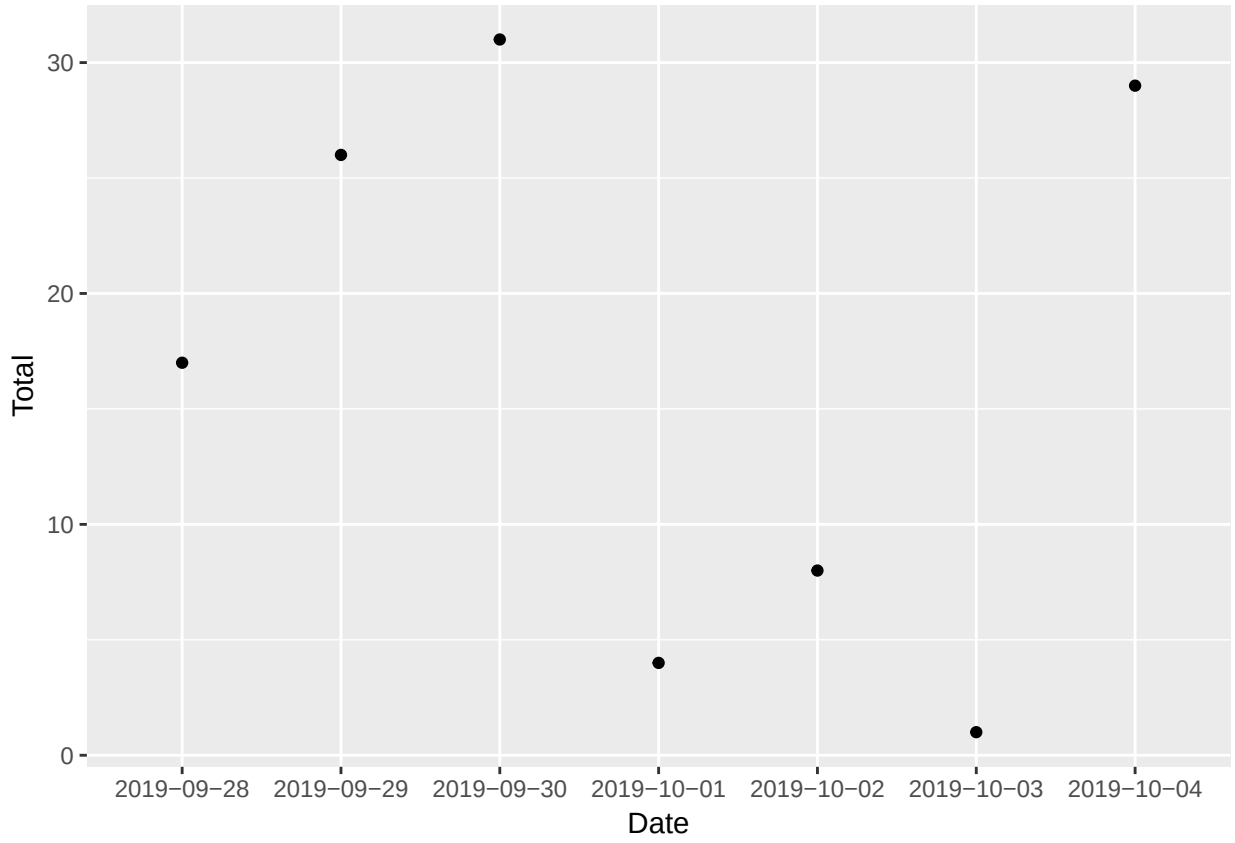
```
richness
```

```
## # A tibble: 7 x 2
##   Date       Total
##   <fct>      <int>
## 1 2019-09-28    17
## 2 2019-09-29    26
## 3 2019-09-30    31
```

```
## 4 2019-10-01     4
## 5 2019-10-02     8
## 6 2019-10-03     1
## 7 2019-10-04    29
```

And note the numbers have dropped a little when we exclude those observations. Now when we run the code for plotting, we see the results changed, too:

```
richness_plot <- ggplot(data = richness,
                        mapping = aes(x = Date,
                                      y = Total)) +
  geom_point()
print(richness_plot)
```



Fix: + Indicate species or not + Standardize spelling

Species richness

```
richness <- bird_data %>%
  group_by(Date) %>%
  summarize(Total = n())
```

Species diversity

There's considerable data wrangling necessary here. vegan::diversity needs a wide-formatted data set, but since there were multiple observations of a single species on October 4, this creates challenges for converting long data to wide-formatted data (because date is effectively a key, but there are multiple observations of some species for 2019-10-04).

```
bird_data_wide <- pivot_wider(data = bird_data[, -2],
                              names_from = Species,
                              values_from = Count,
                              values_fill = list(Count = 0))
```

```
species_counts <- as.matrix(as.data.frame(bird_data_wide[, 3:119]))
diversity <- vegan::diversity(x = bird_data_wide[, 3:119])
```

Could alternatively do a species accummulation curve, using `duplicated`

```
first <- bird_data[!duplicated(bird_data$Species), ]
first <- first[order(first$Date), ]
first_date <- first %>%
  group_by(Date) %>%
  summarize(new_species = n())
first_date$cumulative <- cumsum(x = first_date$new_species)
first_date$Date <- as.Date(x = first_date$Date)
plot(x = first_date$Date,
     y = first_date$cumulative,
     type = "l")
```

Diversity

Shannon's index

$$H = -\sum_{i=1}^{s} p_i \ln p_i$$

is the sum over s species, where p_i is the proportion individuals of the i^th species (n_i) out of the total number of inviduals (N).

use `vegan::diversity(x = bird_data$Count, index = "Shannon")`

Additional resources

- Wickham, H. 2014. Tidy data. *The Journal of Statistical Software* **59** http://www.jstatsoft.org/v59/i10/
- A nice introduction to calculating measures of species diversity
- A PDF version of this lesson

Back to learn-r main page

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