Data Wrangling in R

C. Tobin Magle, PhD

11-30-2017

10:00-11:30 a.m.
Morgan Library

Computer Classroom 175

Based on

http://www.datacarpentry.org/R-ecology-lesson/03-dplyr.html

Outline

- Import data as a “tibble” using **read_csv**()

- 6 dplyr **verbs** for data manipulation
  - **select**, **filter**, **mutate**, **group_by**, **summarize**, **tally**

- Combining verbs with **pipes** **%>%**

- 2 tidyr verbs to reshape your data (**spread**, **gather**)

- **Cleaning** and **exporting** data (**is.na**, **write_csv**)

Setup

- Install R and R studio

http://www.datacarpentry.org/R-ecology-lesson/index.html#setup_instructions

- Download the quickstart files: http://bit.ly/2uemtAU

- See the Basic Analysis with R lesson if you’re unfamiliar with R or R studio

http://libguides.colostate.edu/coding-cookies/r-basic

What is the tidyverse?

- Packages for data manipulation
- Built for data tables
- Makes data manipulation easier than in base R
- Combine verbs using pipes (%>%)

Installing and loading packages

install.packages(“tidyverse”)

- Installs the package
- One time only (on each computer)

- Packages are installed on “D:/r-packages” on library PCs.

library(”tidyverse”)

- Loads the package
- Every time you start up R

- Tell R where your packages are using the lib.loc = “D:/r-packages” argument to the library function.

Data set: survey of small animals

- Stored in a data frame*
- **Rows**: observations of individual animals
- **Columns**: Variables that describe the animals
  - Species, sex, date, location, etc

*a tibble actually, but close enough

Import data in tidyverse

- **read_csv** – loads contents of a CSV file

- **Input**: a **file** path

- **Output** a “tibble” aka tbl_df
  - Prints data type under col name
  - Never converts characters to factors

Example: **read_csv**(**file** = “portal_data_joined.csv”)

select()

- Selects columns from a data frame
- **Input**: a tibble and a list of columns
- **Output**: a tibble with only columns specified above

Example: **select**(surveys, plot_id, species_id, weight)

filter()

- Choose rows based on a specific criterion

- **Input**: a tibble and relational expression (returns true/false)
    - **>**, **<**, **>=**, **<=**, **==**, **!=**
- **Output**: a tibble with rows that meet the relational expression

Example: **filter**(surveys, year **==** 1995)

Pipe operator **%>%**

- Allows you to combine multiple “verb” operations
- Syntax: **%>%** at the end of the line
- Output of the first line becomes input of next line, etc.
- Final output to the screen or a variable

Example:
```
surveys %>%
        filter(weight<5) %>%
        select(species_id, sex, weight)
```

Exercise #1

- Using pipes, subset the survey data to include
  - individuals collected before 1995 and
  - retain only the columns **year**, **sex**, and **weight**.

mutate()

- Creates a new column
- **Input:** a tibble, <new col name>= value
- **Output**: a tibble with a new column (defined above)

Example: mutate(surveys, weight_kg = weight/1000)

More mutate examples

```
surveys %>%
        mutate(weight_kg = weight / 1000)

surveys %>%
        mutate(weight_kg = weight / 1000,
               weight_kg2 = weight_kg *2)
surveys %>%
    mutate(weight_kg = weight / 1000) %>%
    head
```

Exercise #2

- Create a new data frame from the survey data that meets the following criteria:
    1. contains only the **species_id** column and a new column called **hindfoot_half**
    2. **hindfood_half** contains values that are half the **hindfoot_length** values.
    3. Only include records from 1990 and after

  - **Hint**: think about how the commands should be ordered to produce this data frame!

Summarizing data

group_by()

- Groups data in the table by an attribute
- Input: a tibble, a variable to group by
- Output: a tibble

summarize()

- Summarizes grouped data
- Input: a tibble. a summary statistic
- Output: a tibble

```
surveys %>%
        group_by(sex) %>%
      summarize(mean_weight = mean(weight, na.rm = TRUE))
```

Group by multiple categories

```
surveys %>%
        group_by(sex, species_id) %>%
        summarize(mean_weight = mean(weight,
                                         na.rm = TRUE))
```

Removing NA with filter

- **is.na**() – missing values TRUE, not missing = FALSE
  - **Input**: a column
  - **Output**: T/F vector

```
surveys %>%
          filter(!is.na(weight)) %>%
                    group_by(sex, species_id) %>%
                    summarize(mean_weight = mean(weight))
```

Limit what you print

```
surveys %>%
            filter(!is.na(weight)) %>%
            group_by(sex, species_id) %>%
            summarize(mean_weight = mean(weight)) %>%
            print(n = 15)
```

Multiple summary stats

```
surveys %>%
        filter(!is.na(weight)) %>%
        group_by(sex, species_id) %>%
        summarize(mean_weight = mean(weight),
                  min_weight = min(weight)
                 )
```

tally

- Count the number of observations in a group
- **Input**: a tibble
- **Output**: a tibble with a count of each group

Example:
```
surveys %>%
  group_by(sex) %>%
  tally
```

Exercise #3

- How many individuals were caught in each **plot_type** surveyed?

- Use **group_by**() and **summarize**() to find the **mean**, **min**, and **max** hindfoot length for each species (using species_id).

- **Bonus**: What was the heaviest animal measured in each year? Return the columns year, **genus**, **species_id**, and **weight**.

Reshaping data with tidyr

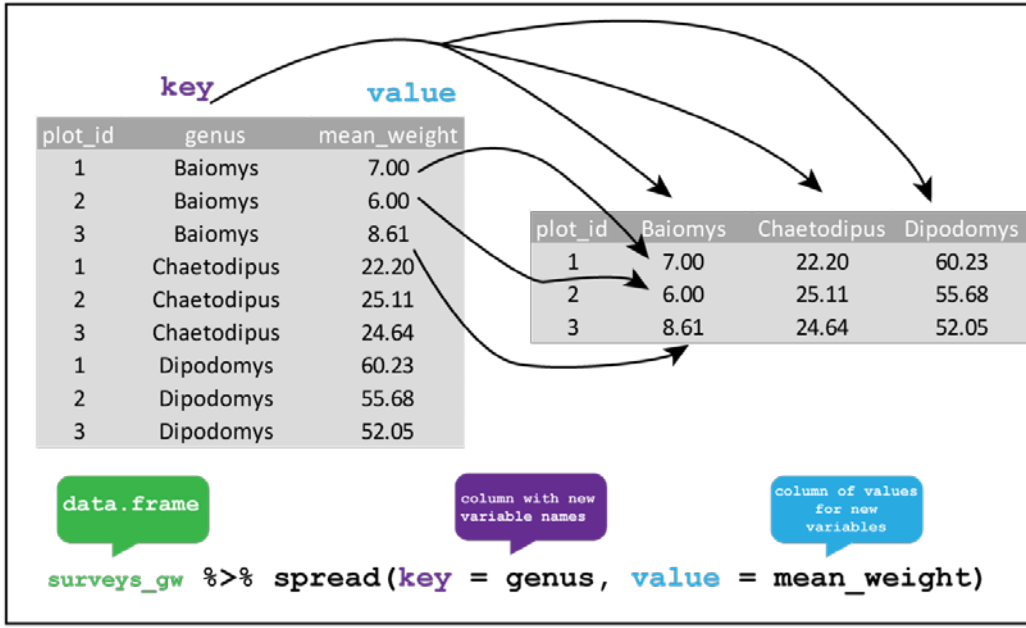
- Shape of your data depends on what you’re interested in doing
- Example: mean weight by genus
  - Need to reshape the data so rows are plot
- **Spreading**: makes a wide table
- **Gathering**: makes a long table

spread()

- Wide table: values in cells become column headers

- **Input**:
  - data: a tibble,
  - key column (values become new column names),
  - value column (to fill new column variables)

- **Output**: a tibble


key
value
data.frame
column with new variable names
column of values for new variables

plot_id	genus	mean_weight
1	Baiomys	7.00
2	Baiomys	6.00
3	Baiomys	8.61
1	Chaetodipus	22.20
2	Chaetodipus	25.11
3	Chaetodipus	24.64
1	Dipodomys	60.23
2	Dipodomys	55.68
3	Dipodomys	52.05

plot_id	Baiomys	Chaetodipus	Dipodomys
1	7.00	22.20	60.23
2	6.00	25.11	55.68
3	8.61	24.64	52.05

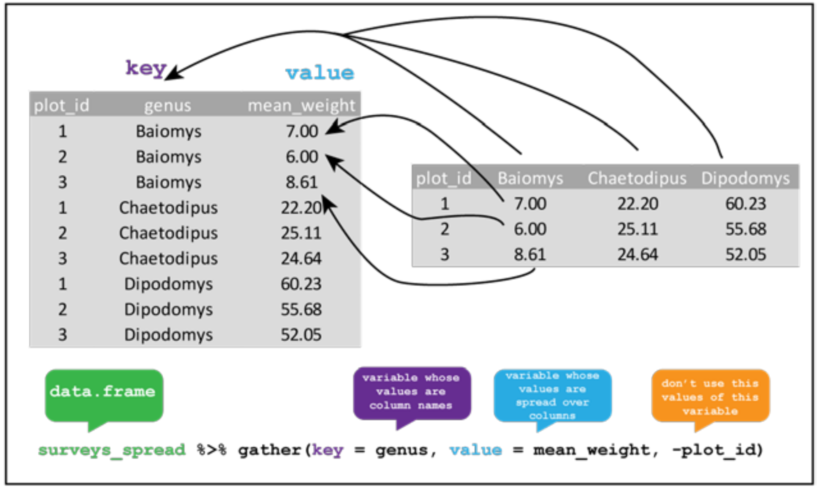
```
surveys_gw %>% spread(key = genus, value = mean_weight)
```

Example:

```
surveys_gw <- surveys %>%
                  filter(!is.na(weight)) %>%
                  group_by(genus, plot_id) %>%
                  summarize(mean_weight = mean(weight))
surveys_spread <- surveys_gw %>%
                 spread(key = genus, value = mean_weight)
surveys_gw %>%
          spread(genus, mean_weight, fill = 0) %>%
          head()
```

gather()

- Long table (column headers become values)
- **Input**:
  - **data**: tibble,
  - **key** column (create from col names)
  - **values** column (fill the key variable)
- **Output**: a tibble


key
value
data.frame
variable whose values are column names
variable whose values are spread over columns
don't use this values of this variable

plot_id	genus	mean_weight
1	Baiomys	7.00
2	Baiomys	6.00
3	Baiomys	8.61
1	Chaetodipus	22.20
2	Chaetodipus	25.11
3	Chaetodipus	24.64
1	Dipodomys	60.23
2	Dipodomys	55.68
3	Dipodomys	52.05

plot_id	Baiomys	Chaetodipus	Dipodomys
1	7.00	22.20	60.23
2	6.00	25.11	55.68
3	8.61	24.64	52.05

```
surveys_spread %>% gather(key = genus, value = mean_weight, -plot_id)
```

Example

```
surveys_spread %>%
            gather(key = genus,
                   value = mean_weight,
                   Baiomys:Spermophilus) %>%
             head()
```

```
surveys_spread %>%
             gather(key = genus,
                    value = mean_weight,
                    Baiomys:Spermophilus) %>%
              head()
```

Exercise #4:

- **Goal**: look at the relationship between mean values of weight and hindfoot length per year in different plot types.
- **Step 1**: Use gather() to create a dataset where we have a key column called measurement and a value column that takes on the value of either hindfoot_length or weight.
- **Step 2**: Calculate the average of each measurement in each year for each different plot_type.
- **Step 3**: spread() them into a data set with a column for hindfoot_length and weight.

Remove missing data

```
surveys_complete <- surveys %>%
 filter(species_id != "", # remove missing species_id
          !is.na(weight), # remove missing weight
          !is.na(hindfoot_length), # remove missing hindfoot_length
          sex != "") # remove missing sex
```

Data Cleaning: eliminate rare species

```
## Extract the most common species_id
species_counts <- surveys_complete %>%
            group_by(species_id) %>%
            tally %>%
            filter(n >= 50)
## Only keep the most common species
surveys_complete <- surveys_complete %>%
            filter(species_id %in% species_counts$species_id)
```

write_csv()

- Writes a data table to a file
- **Input**: a tibble, a file path
- **Output**: a file at the specified file path

Example: **write_csv**(surveys_complete,
path **=** “data/surveys_complete.csv")

Need help?

- Email: tobin.magle@colostate.edu
- Data Management Services website: http://lib.colostate.edu/services/data-management
- Data Carpentry: http://www.datacarpentry.org/
  - R Ecology Lesson: http://www.datacarpentry.org/R-ecology-lesson/03-dplyr.html
- Data wrangling cheat sheet: http://www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf