Introduction to R for Data Management and Analysis

Marcel Ramos, MPH

Announcements

- Downloading files from GitHub
    - Use the ‘raw’ button when at the page
    - Use Git GUI software
        - https://git-scm.com/download/gui/windows
    - Use functions that work with URLs (`read.csv`)
- Additional topics to cover
    - Formulas
    - Aggregating
    - Reshaping your data

Formula notation in R

- Uses the ~ for denoting a formula
  - `y ~ m*x + b`
- Good for specifying linear models
- Mainly used in base R code
- Useful for creating crosstabs!
  - `xtabs(A ~ B, data = blue)`
- Look out for `formula` class inputs
  - see `?xtabs`
  - see `?t.test`
- Usually requires a `data` input / argument in a supported function

Sorting and aggregating data

- order function - returns an index of ordered positions
- tidyverse: arrange - returns the arranged data
- aggregate - summarize data by a categorical variable
  - aggregate(mtcars$mpg, by = list(mtcars$cyl), FUN = "mean")
- tapply
  - tapply(mtcars$mpg, mtcars$cyl, mean)
- tidyverse: group_by and summarize

Transformations / Manipulations

- long to wide format
- `dplyr` and `tidyr` packages

Outline for today

- Review exercises
- Combining data manipulations
- Reshaping data
- Plotting in base R
- Exploratory Data Analysis
- Intro to ggplot2
- Saving graphics

But first, a quote...

The data may not contain the answer. The combination of some data and an aching desire for an answer does not ensure that a reasonable answer can be extracted from a given body of data.
-John Tukey

Review

- Exercises
- Creating a `data.frame`

Using the ***nycflights13*** *dataset*

```
library(nycflights13); library(dplyr)
flights %>% group_by(carrier) %>%
  summarise(avg_depdelay = mean(dep_delay, na.rm = TRUE), coun
    left_join(airlines) %>%
      arrange(avg_depdelay) %>%
        head
```

```
## # A tibble: 6 x 4
##   carrier avg_depdelay count name
##   <chr>          <dbl> <int> <chr>
## 1 US              3.78 20536 US Airways Inc.
## 2 HA              4.90   342 Hawaiian Airlines Inc.
## 3 AS              5.80   714 Alaska Airlines Inc.
## 4 AA              8.59 32729 American Airlines Inc.
## 5 DL              9.26 48110 Delta Air Lines Inc.
## 6 MQ             10.6  26397 Envoy Air
```

Reshaping data

- Useful to prepare data for visualizations
- long vs wide
- long format - multiple observations per row (survival data)
- wide format - a single observation per row

Reshaping data using ***gather***

```r
data(iris); library(tidyr)
longdata <- gather(tbl_df(iris), key = measure, n,
  Sepal.Length:Petal.Width) %>% separate(measure, c("type",
    "dimension"))
longdata %>% group_by(Species, type, dimension) %>%
  summarise(avg_dim = mean(n, na.rm = TRUE))
```

```
## # A tibble: 12 x 4
## # Groups:   Species, type [6]
##    Species    type  dimension avg_dim
##    <fct>      <chr> <chr>       <dbl>
##  1 setosa     Petal Length      1.46
##  2 setosa     Petal Width       0.246
##  3 setosa     Sepal Length      5.01
##  4 setosa     Sepal Width       3.43
##  5 versicolor Petal Length      4.26
##  6 versicolor Petal Width       1.33
```

Pew example

```r
library(readr)
(pew <- read_csv("../Data/pew.csv"))
```

```
## Parsed with column specification:
## cols(
##   religion = col_character(),
##   `<$10k` = col_double(),
##   `$10-20k` = col_double(),
##   `$20-30k` = col_double(),
##   `$30-40k` = col_double(),
##   `$40-50k` = col_double(),
##   `$50-75k` = col_double(),
##   `$75-100k` = col_double(),
##   `$100-150k` = col_double(),
##   `>150k` = col_double(),
##   `Don't know/refused` = col_double()
## )
```

Long dataset

```
pew %>% pivot_longer(-religion, names_to = "income", values_to
```

```
## # A tibble: 6 x 3
##   religion income  count
##   <chr>    <chr>   <dbl>
## 1 Agnostic <$10k      27
## 2 Agnostic $10-20k    34
## 3 Agnostic $20-30k    60
## 4 Agnostic $30-40k    81
## 5 Agnostic $40-50k    76
## 6 Agnostic $50-75k   137
```

- `-religion` - don't include religion when reshaping
- `names_to` - create an income variable out of the columns
- `values_to` - cell values are counts

group_by operations

- Allow users to group different levels of categories of 1 or more variables
- Efficient summirization

Using group_by (1)

```
pew %>% pivot_longer(-religion, names_to = "income", values_to
  group_by(income) %>% summarise(totals = sum(count))
```

```
## # A tibble: 10 x 2
##    income                totals
##    <chr>                  <dbl>
##  1 $10-20k                 2781
##  2 $100-150k               3197
##  3 $20-30k                 3357
##  4 $30-40k                 3302
##  5 $40-50k                 3085
##  6 $50-75k                 5185
##  7 $75-100k                3990
##  8 <$10k                   1930
##  9 >150k                   2608
## 10 Don't know/refused      6121
```

Using group_by (2)

```
pew %>% pivot_longer(-religion, names_to = "income", values_to
  group_by(religion) %>% summarise(totals = sum(count))
```

```
## # A tibble: 18 x 2
##    religion                totals
##    <chr>                    <dbl>
##  1 Agnostic                   826
##  2 Atheist                    515
##  3 Buddhist                   411
##  4 Catholic                  8054
##  5 Don’t know/refused         272
##  6 Evangelical Prot          9472
##  7 Hindu                      257
##  8 Historically Black Prot   1995
##  9 Jehovah's Witness          215
## 10 Jewish                     682
## 11 Mainline Prot             7470
```

Plotting and Graphing

- Exploratory Data Analysis
- Base R graphics
- Intro `ggplot2`
- Saving graphics

Plotting systems in R

- 'Base' graphics
- lattice
- ggplot2

Exploratory Data Analysis

- Informal representation data
- Looking for patterns, outliers, etc.
- Get familiar with your data!

Types of graphs

- Historgram
- Scatterplot
    - Scatterplot matrix
- Boxplots / dotplots (`ggplot2`)
- Violin plots (`ggplot2`)
- Q-Q plots
- Mosaic plots
- and many more!

par function

- Check parameters for graphing
- Allows you to control the finer details of plotting

ggplot2 - Grammar of Graphics

- Different syntax
  - Slight learning curve
- Plots are built in layers
- Operations add layers to the plot

Saving outputs

- Common formats for saving plots:
  - PDF
  - SVG
  - PNG/TIFF
- but there are more

Output sandwhich

- Start with a function `pdf`, `png`, `jpeg`, etc.

- End in `dev.off()` for closing the graphics window

Saving plots in `ggplot2`

- `ggplot2` graphics require a `print` (or a call) before it gets rendered in the file.
- `ggsave` - added to make it easier to save plotting objects

Recommended resources

- Fundamentals of Data Visualization
  - Claus O. Wilke
- R Graphics Cookbook
  - Winston Chang