Data basics

Richard Layton

2017-09-05

R packages used in data basics

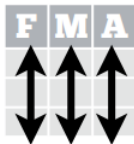| Package	For
tidyverse	ggplot2, dplyr, readr, tidyr
ggplot2	Graphing data
dplyr	Data manipulation
readr	Read rectangular data files (like csv or tsv)
tidyr	Set of functions that help you get to tidy data
knitr	Dynamic documents
readxl	Read .xls and .xslx files
VIM	Examine missing values in a data frame

At the conclusion of the tutorial, you will be able to

- ▶ Define a "tidy" dataset
- ▶ Read data from an Excel file
- ▶ Obtain datasets included with R and R packages
- ▶ Read and write CSV data files
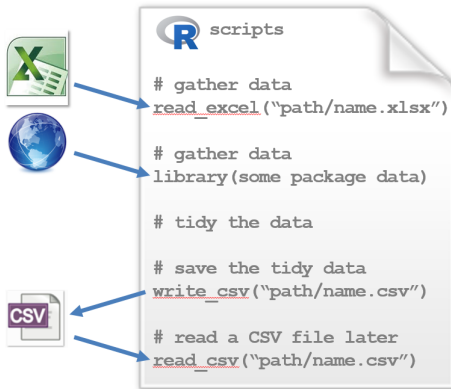
For data graphics success, data should be tidy

In a tidy data set:

Each **variable** is saved in its own **column**

&

Each **observation** is saved in its own **row**

Tidy data

table1

variables

observations

Source: Data Science with R by Garrett Grolemund,
http://garrettgman.github.io/tidying/

Sadly, untidy data is common

country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

table2

variables

observations

Sadly, untidy data is common

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

table4

variables

country	1999	2000
Afghanistan	19987071	20595360
Brazil	172006362	174504898
China	1272915272	1280428583

table5

observations

Source: Data Science with R by Garrett Grolemund,
http://garrettgman.github.io/tidying/

Some industry spreadsheets are really untidy
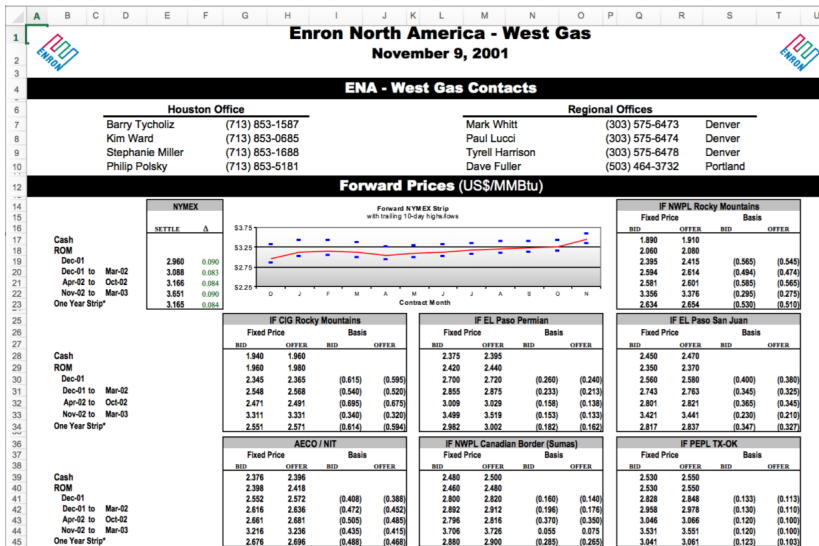
Source: Extract tables from messy spreadsheets with jailbreakr,
http://blog.revolutionanalytics.com/2016/08/jailbreakr.html

For now, assume your data in Excel are tidy

DSR-table1.xlsx

	A	B	C	D
1	country	year	cases	population
2	Afghanistan	1999	745	19987071
3	Afghanistan	2000	2666	20595360
4	Brazil	1999	37737	172006362
5	Brazil	2000	80488	174504898
6	China	1999	212258	1272915272
7	China	2000	213766	1280428583

Row 1 has the names of the variables

Row 2 starts the observations, one observation per row

- *readxl* is the package
- *read_excel()* is the function

The data frame that results from *read_excel()*

```
library(readxl)
tidy_data <- read_excel(
      path = "data/DSR-table1.xlsx"
    , sheet = "DSR-table1"
    )
glimpse(tidy_data)


## Observations: 6
## Variables: 4
## $ country    <chr> "Afghanistan", "Afghanistan"...
## $ year       <dbl> 1999, 2000, 1999, 2000, 1999...
## $ cases      <dbl> 745, 2666, 37737, 80488, 212...
## $ population <dbl> 19987071, 20595360, 17200636...
```

Or, if we print the data frame as a table

```r
library(knitr)
kable(tidy_data)
```

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

Note the relative path argument in *read_excel()*
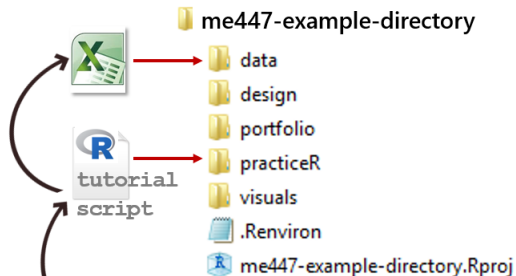

```
tidy_data <- read_excel(
    path = "data/DSR-table1.xlsx"
  , sheet = "DSR-table1"
  )
```

But, when your Excel data are not tidy

VADeaths.xlsx

	A	B	C	D	E
1		Rural		Urban	
2	Group	Men	Women	Men	Women
3	50-54	11.7	8.7	15.4	8.4
4	55-59	18.1	11.7	24.3	13.6
5	60-64	26.9	20.3	37	19.3
6	65-69	41	30.9	54.6	35.1
7	70-74	66	54.3	71.1	50

Row 1 has merged cells variable name information

Row 2 has more variable name information

Row 3 starts the data

The *read_excel()* results can be weird

```
untidy_data <- read_excel("data/VADeaths.xlsx")
glimpse(untidy_data)


## Observations: 6
## Variables: 5
## $ X__1  <chr> "Group", "50-54", "55-59", "60-64...
## $ Rural <chr> "Men", "11.7", "18.10000000000000...
## $ X__2  <chr> "Women", "8.6999999999999993", "1...
## $ Urban <chr> "Men", "15.4", "24.3", "37", "54....
## $ X__3  <chr> "Women", "8.4", "13.6", "19.3", "...
```

- All the data have been coerced into character data

Or, if we print the result as a table

```r
kable(untidy_data)
```

X__1	Rural	X__2	Urban
Group	Men	Women	Men
50-54	11.7	8.6999999999999993	15.4
55-59	18.100000000000001	11.7	24.3
60-64	26.9	20.3	37
65-69	41	30.9	54.6
70-74	66	54.3	71.099999999999

- The first data row is not actually an observation

However, *read_excel()* can skip lines

VADeaths.xlsx

	A	B	C	D	E
1		Rural		Urban	
2	Group	Men	Women	Men	Women
3	50-54	11.7	8.7	15.4	8.4
4	55-59	18.1	11.7	24.3	13.6
5	60-64	26.9	20.3	37	19.3
6	65-69	41	30.9	54.6	35.1
7	70-74	66	54.3	71.1	50

Row 1 has merged cells variable name information

Row 2 has more variable name information

Row 3 starts the data

```
untidy_data <- read_excel("data/VADeaths.xlsx"
    , skip = 1)
```

But we might lose information

```
kable(untidy_data)
```

Group	Men	Women	Men___1	Women___1
50-54	11.7	8.7	15.4	8.4
55-59	18.1	11.7	24.3	13.6
60-64	26.9	20.3	37.0	19.3
65-69	41.0	30.9	54.6	35.1
70-74	66.0	54.3	71.1	50.0

- ▶ Try to work with tidy data only for the time being
- ▶ Later we will work on how to make untidy data tidy

Read and write CSV files

If you have a data frame *tidy_data* in your workspace

```r
library(readr)
write_csv(tidy_data, "data/example_tidy_data.csv")
```

If you have file *example_tidy_data.csv* in your *data* folder

```r
new_copy <- read_csv("data/example_tidy_data.csv")
```

Examine a data frame

Graphing data and studying the data structure are complementary

- *class()*
- *glimpse()*
- *kable()*
- *head()*
- *tail()*

Identifying missing values (details in the tutorial)

- *summary()*
- *VIM* package *aggr()* function

Accessing data in R

```
data()
```

```
Data sets in package 'datasets':

AirPassengers   Monthly Airline Passenger Numbers 1949-1960
BJsales         Sales Data with Leading Indicator
BOD             Biochemical Oxygen Demand
CO2             Carbon Dioxide Uptake in Grass Plants
ChickWeight     Weight vs age of chicks on different diets
etc.
```

Many of these R data sets widely used in R examples

Commonly encountered

- ► `mtcars`
- ► `iris`
- ► `Titanic`
- ► `barley` in the *lattice* package

From the tidyverse

- ► `starwars` in the *dplyr* package
- ► `economics` in the *ggplot2* package
- ► `who` in the *tidyr* package

And other packages

- ► *fivethirtyeight* package
- ► so many more

Review

package::function()

- ▶ readxl::read_excel()
- ▶ readr::write_csv()
- ▶ readr::read_csv()
- ▶ tibble::glimpse()
- ▶ knitr::kable()
- ▶ VIM::aggr()

base R functions

- ▶ data()
- ▶ class()
- ▶ head()
- ▶ tail()
- ▶ summary()
- ▶ unloadNamespace()
- ▶ search()

ggplot2

- ▶ geom_smooth(method = lm, se = FALSE)