

# Managing and Manipulating Data Using R

## Lecture 8, Acquiring data in R

Ozan Jaquette



## Introduction

Libraries we will use today [install if you don't have them]

```
library(dplyr)
library(readr)
library(haven)
library(readxl)
library(labelled)
```

# Data we will use today

IPEDS or HSLS  
Federal Student Aid  
Chetty

**Work on this**

## Common data formats

# Common data formats

- Comma-separated values (.csv)
- Excel (.xls or .xlsx)
- Text-formated data (.txt)
- Tab-separated values (.tsv)
- R (.Rdata or .rds)
- Stata (.dta)
- SPSS (.sav)
- SAS (.sas)

readr package



## readr

The `readr` package is part of `tidyverse`, which is designed to read in flat data files in R and transform them into data frames.

- We could load **`library(tidyverse)`** if we wanted to load all packages in `tidyverse` (e.g. `ggplot2`, `dplyr`, `tidyr`, `stringr`, `readr`, etc...)

```
library(tidyverse)
#> -- Attaching packages -----
#>  ggplot2 3.0.0      purrr   0.2.5
#>  tibble  1.3.4      dplyr   0.7.6
#>  tidyr   0.8.1      stringr 1.3.1
#>  readr   1.1.1      forcats 0.3.0
#> Warning: package 'tibble' was built under R version 3.3.2
#> Warning: package 'readr' was built under R version 3.3.2
#> -- Conflicts -----
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

- For the purpose of this lecture, we will just need to load **`library(readr)`**

## readr

No matter the flat file format you are working with, there are two important steps for reading in data with `readr`:

1. **a function to parse the file (`read_csv`)**
2. **column specification**

**Talk about `readr` (tidyverse), Notes from the reading, Notes from other sources**

## readr functions

**readr's** (tidyverse) functions

<b>Format</b>	<b>Function</b>
Comma-separated values (csv)	read_csv
Semicolon separated files	read_csv2
Tab-separated values (tsv)	read_tsv
Any delimiter	read_delim
Fixed width files	read_fwf
Text-formated data (txt)	read_table
Web log files	read_log

**What type of data could be read in**

## readr column specification

readr is pretty good at guessing each column's data type (e.g. character, double, etc.), however it is good practice to manually specify the data type for each column.

```
mtcars <- read_csv(readr_example("mtcars.csv"))  
#> Parsed with column specification:  
#> cols(  
#>   mpg = col_double(),  
#>   cyl = col_integer(),  
#>   disp = col_double(),  
#>   hp = col_integer(),  
#>   drat = col_double(),  
#>   wt = col_double(),  
#>   qsec = col_double(),  
#>   vs = col_integer(),  
#>   am = col_integer(),  
#>   gear = col_integer(),  
#>   carb = col_integer()  
#> )
```

## readr column specification

The output of the previous example shows us the column specification of the variables in the data. This is important because we could manually change column specification if we do not like readr's guess.

```
mtcars <- read_csv(readr_example("mtcars.csv"), col_types =  
  cols(  
    mpg = col_double(),  
    cyl = col_integer(),  
    disp = col_double(),  
    hp = col_integer(),  
    drat = col_double(),  
    vs = col_integer(),  
    wt = col_double(),  
    qsec = col_double(),  
    am = col_integer(),  
    gear = col_integer(),  
    carb = col_integer()  
  )  
)
```

## readr features

**skip:**  
**comment:**  
**col\_names:**

## readr demonstration csv

readr automatically treats the first line of data as column names.

```
read_csv("column 1, column 2, column 3
1,2,3
4,5,6"
)
#> # A tibble: 2 x 3
#>   `column 1` `column 2` `column 3`
#>       <int>      <int>      <int>
#> 1         1         2         3
#> 2         4         5         6
```

There are instances where you may want to tell R from what line to begin reading in data.

## readr demonstration csv

Notice the example below. The first two lines are comments about the data. We would need to use **skip = n** to skip n lines.

```
read_csv("This file contains data on student charges for the academic year.  
File name: IC2016_AY  
a, b, c  
1,2,3  
4,5,6", skip = 2  
)  
#> # A tibble: 2 x 3  
#>       a       b       c  
#>   <int> <int> <int>  
#> 1     1     2     3  
#> 2     4     5     6
```



## readr demonstration csv

We could also tell R to drop lines we specify as comments. With **comment = n**

```
read_csv("# This file contains data on student charges for the academic year.  
  a, b, c  
  1,2,3  
  4,5,6", comment = "#"  
)  
#> # A tibble: 2 x 3  
#>       a     b     c  
#>   <int> <int> <int>  
#> 1     1     2     3  
#> 2     4     5     6
```

```
read_csv("* This file contains data on student charges for the academic year.  
  a, b, c  
  1,2,3  
  4,5,6", comment = "*"  
)  
#> # A tibble: 2 x 3  
#>       a     b     c  
#>   <int> <int> <int>  
#> 1     1     2     3  
#> 2     4     5     6
```

## readr column names

We could tell R there are no column names with **col\_names = FALSE** or we could manually give R column names with **col\_names = c("", "", "")**

```
read_csv("1,2,3
         4,5,6", col_names = FALSE
         )
#> # A tibble: 2 x 3
#>       X1     X2     X3
#>   <int> <int> <int>
#> 1     1     2     3
#> 2     4     5     6
```

```
read_csv("1,2,3
         4,5,6", col_names = c("column 1", "column 2", "column 3")
         )
#> # A tibble: 2 x 3
#>   `column 1` `column 2` `column 3`
#>       <int>       <int>       <int>
#> 1         1         2         3
#> 2         4         5         6
```

## readr Student exercise

- Get in your homework groups
- Create a 3x3 tibble like the examples above (e.g. `read_csv("a,b,c...")`), treating the first line as column names
- Now on the first line add a sentence
- This time add a special character ( `*`, `#`, `!` ) at the beginning of the sentence and indicate it is a comment
- Delete the sentence and column names (should have a 2x2 tibble) and manually tell R column names

## readr demonstration csv

**NOT SURE IF TO MAKE THIS A DEMONSTRATION WHERE STUDENTS FOLLOW ALONG OR ANOTHER STUDENT EXERCISE**  
**Tying it all together**

Use `read_csv()` function from `readr` to import csv dataset into R without column specification. Follow along on your computers.

```
#ipeds_ay <- read_csv(file="../ic2017ay_small.csv")
```

Notice tuition and fee columns are read in as character type.

## readr demonstration csv

Use `read_csv()` function from `readr` to import csv dataset into R with column specification **[Would it be better to change to integer or double?]**

```
# {ipeds_ay <- read_csv("../ic2017ay_small.csv", col_types =  
#   cols(  
#     unitid = col_integer(),  
#     tuition1 = col_integer(),  
#     fee1 = col_integer(),  
#     hrchg1 = col_integer(),  
#     tuition2 = col_integer(),  
#     fee2 = col_integer(),  
#     hrchg2 = col_integer(),  
#     tuition3 = col_integer(),  
#     fee3 = col_integer(),  
#     hrchg3 = col_integer()  
#   )  
#) # }
```

## readr Running into errors

1. Make sure you have downloaded and saved flat file
2. Make sure to know the file path of where data is downloaded or saved (~/Desktop/educ263/data)
3. Make sure you set your working `setwd()` directory in R. To check your current working directory type `getwd()` in console.

haven package

## haven

Recap from lecture 5

`haven` is part of **tidyverse**, which enables users to import and export data from the following statistical packages:

- SAS
- SPSS
- Stata

Similar to `readr`, we could load the entire **library(tidyverse)** package to get `haven`. For the purpose of this lecture, we will just need to load **library(haven)**.



## haven functions

**haven's** (tidyverse) functions

Format	Function
SPSS	read_sav
SAS	read_sas
Stata	read_dta

## haven read and write Stata arguments

```
read_dta(file, encoding = NULL)  
write_data(data, path, version = 14)
```

### Arguments

- file: file path to data
- encoding: files prior to Stata 14 did not declare text encoding, files after Stata 14 do not need to declare encoding value
- data: data frame to save (write)
- path: file path to where data will be saved
- version: file version

### Link

## haven read and write Stata example

Use `read_dta()` function from `haven` to import Stata dataset into R

```
hsls <- read_dta("~/Desktop/lecture8/hsls_sch_small.dta", encoding=NULL)

# View data
head(hsls)
glimpse(hsls)
```

Use `write_dta` function from `haven` to write State data

```
write_dta(hsls, path = "~/Desktop/lecture8/hsls_sch_small.dta")
```

Student exercise Running into problems with variable/value labels?

readxl package

## readxl

The `readxl` package is part of tidyverse, which is designed to easily read data from Excel and into R.

- We could load **`library(tidyverse)`** if we wanted to load all packages in tidyverse. For the purpose of this lecture, we just need to load **`library(readxl)`**.

## readxl

readxl supports both .xls and .xlsx formats and is designed to work with tabular data. It does not require dependencies- making installing and operating fairly simple.

readxl has several example files where we could use as practice. The files include:

```
readxl_example()
#> [1] "clippy.xls"      "clippy.xlsx"    "datasets.xls"   "datasets.xlsx"
#> [5] "deaths.xls"      "deaths.xlsx"    "geometry.xls"   "geometry.xlsx"
#> [9] "type-me.xls"     "type-me.xlsx"
```

For now, lets use "datasets.xlsx"

```
excel_example <- readxl_example("datasets.xlsx")
```

## readxl features

- **sheet:** `read_excel(excel file, sheet = "sheet name")`
- **n\_max:** `read_excel(excel file, n_max = n)`
- **range:** `read_excel(excel file, range = "A:D")`
- **cell\_rows:** `read_excel(excel file, range = cell_rows(1:n))`
- **cell\_cols:** `read_excel(excel file, range = cell_cols("A:D"))`
- **na:** `read_excel(excel file, na = "n")`

## readxl sheet

```
#To view sheets in excel file
excel_sheets(excel_example)
#> [1] "iris"      "mtcars"    "chickwts"  "quakes"
```

```
xl_example <- read_excel(excel_example, sheet = "quakes")
head(xl_example)
#> # A tibble: 6 x 5
#>       lat    long depth  mag stations
#>   <dbl> <dbl> <dbl> <dbl>    <dbl>
#> 1 -20.42 181.62   562   4.8      41
#> 2 -20.62 181.03   650   4.2      15
#> 3 -26.00 184.10    42   5.4      43
#> 4 -17.97 181.66   626   4.1      19
#> 5 -20.42 181.96   649   4.0      11
#> 6 -19.68 184.31   195   4.0      12
```



```
readxl n_max
```

```
read_excel(excel_example, sheet = "quakes", n_max = 3)
#> # A tibble: 3 x 5
#>   lat    long depth  mag stations
#>   <dbl> <dbl> <dbl> <dbl>   <dbl>
#> 1 -20.42 181.62  562   4.8     41
#> 2 -20.62 181.03  650   4.2     15
#> 3 -26.00 184.10   42   5.4     43
```

## readxl range

```
read_excel(excel_example, sheet = "quakes", range = "C1:E4")
```

```
#> # A tibble: 3 x 3  
#>   depth mag stations  
#>   <dbl> <dbl>   <dbl>  
#> 1   562  4.8     41  
#> 2   650  4.2     15  
#> 3    42  5.4     43
```

```
read_excel(excel_example, sheet = "quakes", range = cell_rows(1:3))
```

```
#> # A tibble: 2 x 5  
#>   lat long depth mag stations  
#>   <dbl> <dbl> <dbl> <dbl>   <dbl>  
#> 1 -20.42 181.62  562  4.8     41  
#> 2 -20.62 181.03  650  4.2     15
```

```
head(read_excel(excel_example, sheet = "quakes", range = cell_cols("A:C")))
```

```
#> # A tibble: 6 x 3  
#>   lat long depth  
#>   <dbl> <dbl> <dbl>  
#> 1 -20.42 181.62  562  
#> 2 -20.62 181.03  650  
#> 3 -26.00 184.10   42  
#> 4 -17.97 181.66  626  
#> 5 -20.42 181.96  649  
#> 6 -19.68 184.31  195  
  
# using head() to only view first 6 rows
```

## readxl na

```
read_excel(excel_example, sheet = "quakes", na = "-20.42")
#> # A tibble: 1,000 x 5
#>       lat    long depth  mag stations
#>   <dbl> <dbl> <dbl> <dbl>    <dbl>
#> 1     NA 181.62   562  4.8      41
#> 2 -20.62 181.03   650  4.2      15
#> 3 -26.00 184.10    42  5.4      43
#> 4 -17.97 181.66   626  4.1      19
#> 5     NA 181.96   649  4.0      11
#> 6 -19.68 184.31   195  4.0      12
#> 7 -11.70 166.10    82  4.8      43
#> 8 -28.11 181.93   194  4.4      15
#> 9 -28.74 181.74   211  4.7      35
#> 10 -17.47 179.59   622  4.3      19
#> # ... with 990 more rows
```

## readxl Student exercise

- Save data** - Download and save Federal Student Financial Aid Data
- Read in data using `readxl` function
  - Read in first four rows (`n_max`) - Read in column Names to column State **hint**
  - `cell_cols` - Set value "A" to missing (`na`) **note** : you need to investigate in detail before setting anything to missing

## readxl Running into problems

1. Make sure you have downloaded and saved excel file
2. Make sure to know the file path of where data is downloaded or saved (~/Desktop/educ263/data)
3. Make sure you set your working `setwd()` directory in R. To check your current working directory type `getwd()` in console.
4. Make sure to choose the correct sheet (if applicable)
5. Pay attention to column names when setting range

Downloading data from web

## Downloading data from web

Could save time and reduce the steps of downloading, saving, and reading in data, by reading in data directly from the web.

- note that not all packages will work downloading data from web (`read_excel`)

For example, rather than downloading ipeds data and saving it in a folder, we could download the data directly from the web.

# Downloading data from web example using Raj Chetty data

1. Follow this link and under the “Mobility Report Cards...” tab select “click to view data”.
2. Choose “Online Data Table 1”
3. Right click and copy link address for “Excel” (Note: it is actually a csv file)

## Mobility Report Cards: The Role of Colleges in Intergenerational Mobility

Chetty, Friedman, Saez, Turner, and Yagan (2017)

Mobility Statistics and Student Outcomes by College and Birth Cohort

[Click to view data](#) ▼

Data Description	Download		
	Stata	Excel	Readme
Online Data Table 1 Preferred Estimates of Access and Mobility Rates by College	Stata	Excel	Readme
Online Data Table 2 Baseline Cross-Sectional Estimates by College	Stata	Excel	Readme
Online Data Table 3 Baseline Longitudinal Estimates by College and Child's Cohort	Stata	Excel	Readme



# Downloading data from web

[FIX OUTPUT (CUTTING OFF)]

```
#Paste url to excel "csv" file
data_url <- "http://www.equality-of-opportunity.org/data/college/mrc_table1.csv"

#Download data and read in using read_csv (readr)
mrc <- read_csv(data_url)

#View first 4 rows and 4 columns
mrc[1:4, 1:4]
#> # A tibble: 4 x 4
#>   super_opeid      name      czname state
#>   <int>          <chr>      <chr> <chr>
#> 1    2665 Vaughn College Of Aeronautics And Technology New York  NY
#> 2    7273          CUNY Bernard M. Baruch College New York  NY
#> 3    2688 City College Of New York - CUNY New York  NY
#> 4    7022          CUNY Lehman College New York  NY
```

# Downloading data from web

## Alternative approach

```
#Download data and read in link directly using read_csv (readr)
mrc <- read_csv("http://www.equality-of-opportunity.org/data/college/mrc_table1")
#> Parsed with column specification:
#> cols(
#>   super_opeid = col_integer(),
#>   name = col_character(),
#>   czname = col_character(),
#>   state = col_character(),
#>   par_median = col_integer(),
#>   k_median = col_integer(),
#>   par_q1 = col_double(),
#>   par_top1pc = col_double(),
#>   kq5_cond_parq1 = col_double(),
#>   ktop1pc_cond_parq1 = col_double(),
#>   mr_kq5_pq1 = col_double(),
#>   mr_ktop1_pq1 = col_double(),
#>   trend_parq1 = col_double(),
#>   trend_bottom40 = col_double(),
#>   count = col_double()
#> )

#View first 4 rows and 4 columns
mrc[1:4, 1:4]
#> # A tibble: 4 x 4
#>   super_opeid      name      czname state
```

# Problems downloading data (zip files) using IPEDS

1. Follow this link and under the “Survey Data” tab select “Complete data files”.
2. Choose “All years” and “All surveys” and click continue
3. Right click and copy link address for “IC2017\_AY”

**Years & Surveys**

All years

All surveys

Continue

Data files are available in ZIP format.

Year	Survey	Title	Data File	Stata Data File	Programs	Dictionary
2017	Institutional Characteristics	Directory information (Preliminary)	<a href="#">HD2017</a>	<a href="#">HD2017_STATA</a>	<a href="#">SPSS</a> , <a href="#">SAS</a> , <a href="#">STATA</a>	<a href="#">Dictionary</a>
2017	Institutional Characteristics	Educational offerings, organization, services and athletic associations (Preliminary)	<a href="#">IC2017</a>	<a href="#">IC2017_STATA</a>	<a href="#">SPSS</a> , <a href="#">SAS</a> , <a href="#">STATA</a>	<a href="#">Dictionary</a>
2017	Institutional Characteristics	Student charges for academic year programs (Preliminary)	<a href="#">IC2017_AY</a>	<a href="#">IC2017_AY_STATA</a>	<a href="#">SPSS</a> , <a href="#">SAS</a> , <a href="#">STATA</a>	<a href="#">Dictionary</a>

## Downloading data (zip files) using IPEDS

```
# Paste url and read in using read_csv  
# What happens when you try reading in this zip file?  
#ic2017_ay <- read_csv("https://nces.ed.gov/ipeds/datacenter/data/IC2017_AY.zip")  
  
# Need to download file first to unzip
```