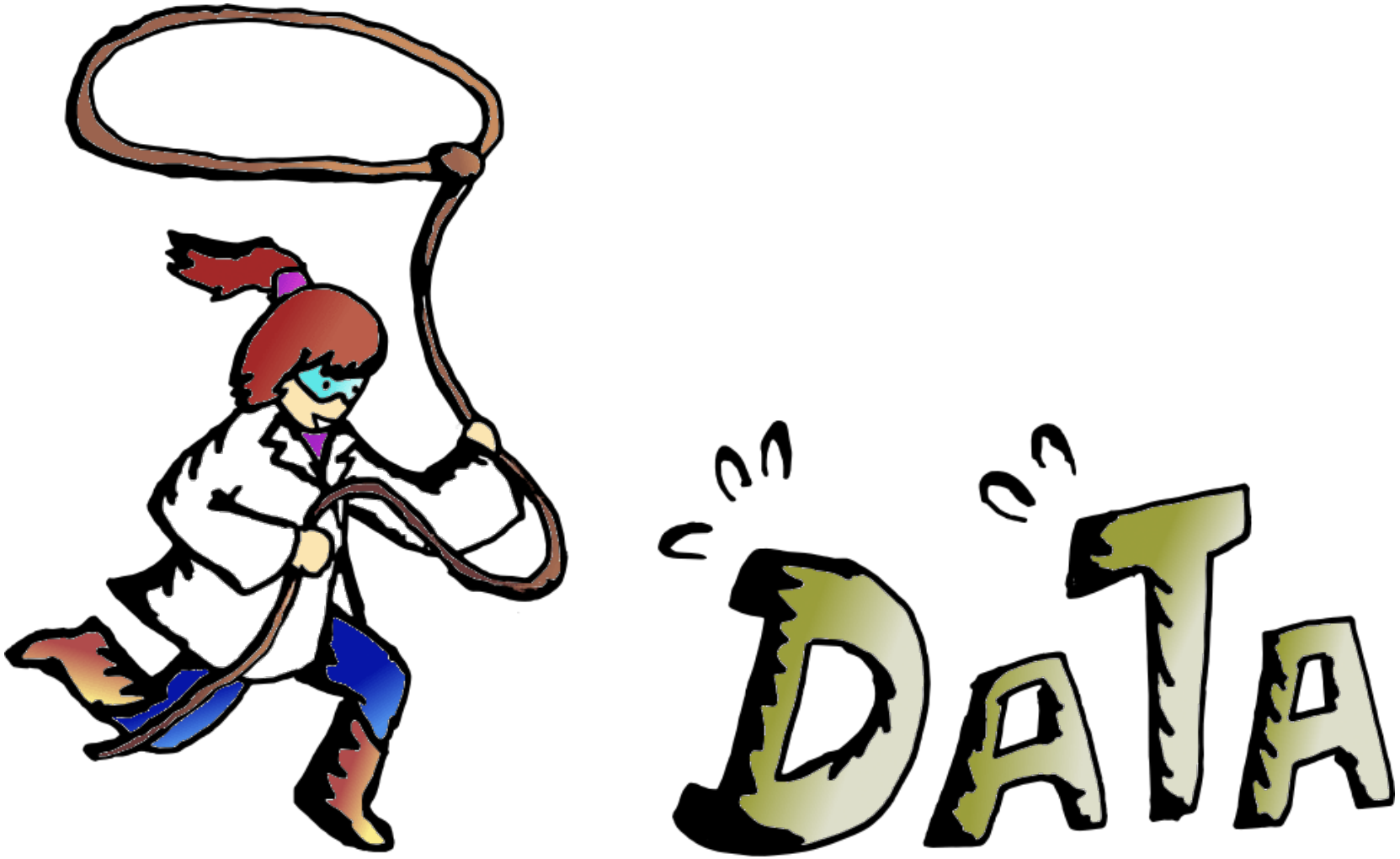
Data Wrangling

Andrew Redd, PhD.

R Bootcamp 2020


DATA

Manipulations

- Data integrity
- Reshaping
- Filtering
- Merging
- Summarizing

Packages that we will use

```
# Make tidyverse load quietly
options(tidyverse.quiet = TRUE)
library(tidyverse)   #< General use
library(tidyr)       #< Reshaping
library(wbstats)     #< World bank data.
library(countrycode) #< Country coding
library(assertthat)  #< Results checking
library(lubridate)   #< Date manipulations
requireNamespace('zoo')#< time series
```

Loading data

for `.RData` files use `load()`

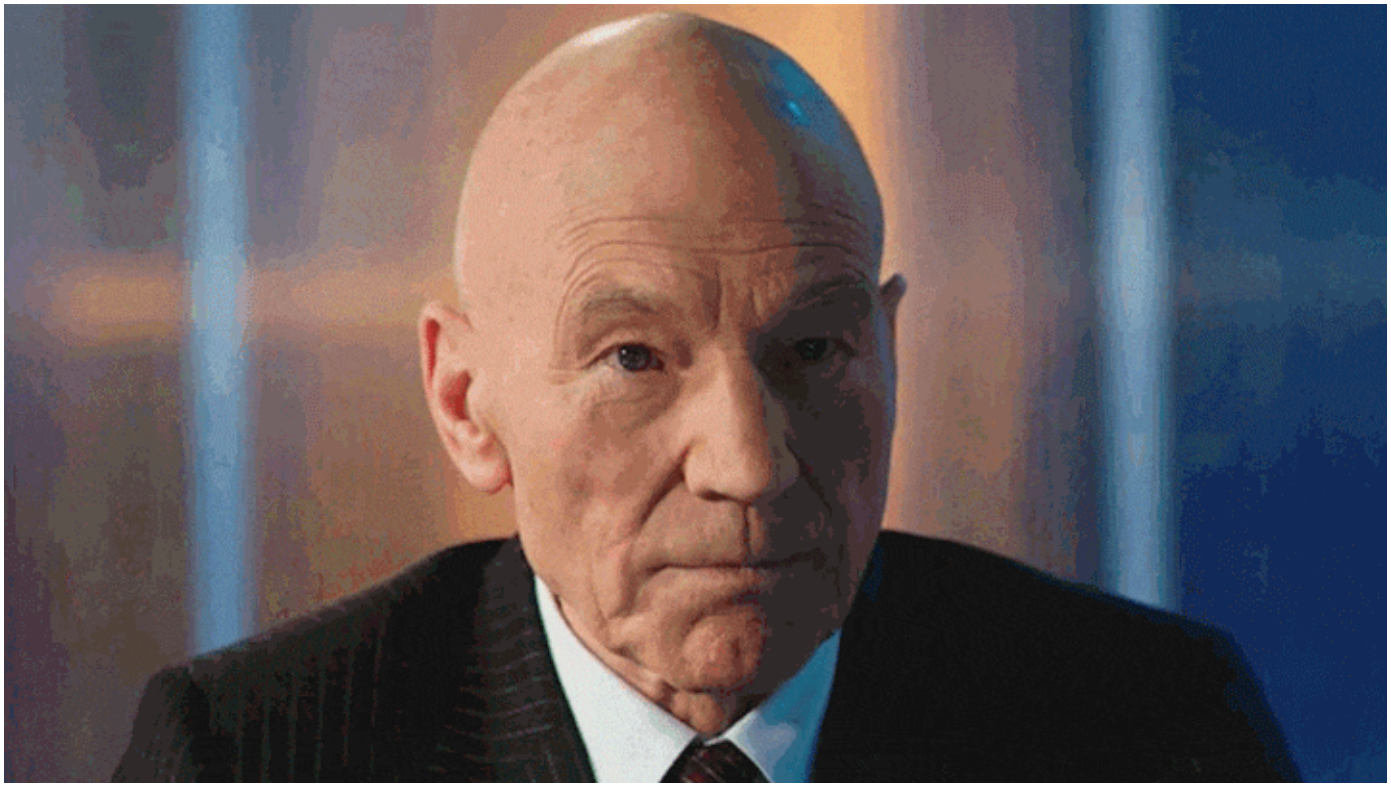
```
load("data/ebola.data.RData")
```

for `.rds` files use `readRDS()` and capture the results in a variable.

```
ebola <- readRDS("data/ebola.data.rds")
```

Always check your data

Any problems with the data?



Ebola Data Problem

The Most obvious is that `Country Report Date` should be repeated down the rows.

Fix with `dplyr::mutate()` + `zoo::na.locf()` (Missing last observation carried forward.)

`mutate()` variants

- `mutate()` - modify/add variables
- `mutate_at()` - modify a set of variables.
- `mutate_if()` - modify variables meeting a criteria
- `transmute()` - create a new set of variables based on previous.

Ebola Data Problem

```
ebola.data %<>%
    mutate_at('Country Report Date', zoo::na.locf)
```

Notes:

1. We used the assign-back pipe %<>% to modify in place. *Generally this is frowned upon*
2. Note the double colon for using na.locf from zoo without attaching the package.

`select()` - Choosing variables

Key Function
Use `select()` to choose the variables desired.

Basic Usage

```
select(data, ...)
```

Over the next few examples we will explore the forms `...` can take

`select()` - Variable Names

the easiest is with variable names:

```
ebola.data %>% select(Country, `Case def.`, `Total cases`) %>% head()
```

Country	Case def.	Total cases
Guinea	Confirmed	3351
Guinea	Probable	453
Guinea	Suspected	0
Guinea	All	3804
Liberia	Confirmed	3151
Liberia	Probable	1879

`select()` - Dropping by Variable Names

You can select everything **but** a variable with the minus operator

```
ebola.data %>% select(-`Total cases`) %>% head()
```

SheetName	Country	Case def.	Total deaths	Country Report Date
Jan 06, 2016	Guinea	Confirmed	2083	2015-12-27
Jan 06, 2016	Guinea	Probable	453	2015-12-27
Jan 06, 2016	Guinea	Suspected	0	2015-12-27
Jan 06, 2016	Guinea	All	2536	2015-12-27
Jan 06, 2016	Liberia	Confirmed	0	2015-05-09
Jan 06, 2016	Liberia	Probable	0	2015-05-09

`select()` - By the numbers

You can select by variable position as well.

```
ebola.data %>% select(1:4) %>% head()
```

SheetName	Country	Case def.	Total cases
Jan 06, 2016	Guinea	Confirmed	3351
Jan 06, 2016	Guinea	Probable	453
Jan 06, 2016	Guinea	Suspected	0
Jan 06, 2016	Guinea	All	3804
Jan 06, 2016	Liberia	Confirmed	3151
Jan 06, 2016	Liberia	Probable	1879

`select()` - by variable range

Use single colon `:` with variable names to select variables named and eveything in between:

```
ebola.data %>% select(Country:`Total cases`) %>% head()
```

Country	Case def.	Total cases
Guinea	Confirmed	3351
Guinea	Probable	453
Guinea	Suspected	0
Guinea	All	3804
Liberia	Confirmed	3151
Liberia	Probable	1879

`select()` - by helpers

selection helpers are also provided:

```
ebola.data %>% select(starts_with("Total")) %>% head()
```

Total cases	Total deaths
3351	2083
453	453
0	0
3804	2536
3151	0
1879	0
## `select()`	- The helpers

The available helpers are:

- `starts_with()`

`select()` - Multiple

You may use multiple forms together

```
ebola.data %>% select(last_col(), 2:3, `Total cases`) %>% head()
```

Country Report Date	Country	Case def.	Total cases
2015-12-27	Guinea	Confirmed	3351
2015-12-27	Guinea	Probable	453
2015-12-27	Guinea	Suspected	0
2015-12-27	Guinea	All	3804
2015-05-09	Liberia	Confirmed	3151
2015-05-09	Liberia	Probable	1879

Subsetting data

Key Function
Subset data with the `filter()` function.

The base R version is subset, but it is FAR less robust.

It takes the form of

```
filter(data, expr1, expr2, ...)
```

where `data` is the data set, and `expr1`, `expr2`, … are the criteria expressions evaluated *in the context of the data*. Data must meet *all* ctriteria to remain.

`filter()` Example

Subset data to only confirmed cases for Nigeria.

```
filter( ebola.data
      , Country == 'Nigeria'
      , `Case def.` == 'Confirmed'
      )
```

SheetName	Country	Case def.	Total cases	Total deaths	Country Report Date
Jan 06, 2016	Nigeria	Confirmed	19	7	2014-10-19
Dec 30, 2015	Nigeria	Confirmed	19	7	2014-10-19
Dec 23, 2015	Nigeria	Confirmed	19	7	2014-10-19
Dec 16	Nigeria	Confirmed	19	7	2014-10-19
Dec 9	Nigeria	Confirmed	19	7	2014-10-19
Dec 2	Nigeria	Confirmed	19	7	2014-10-19
Nov 25	Nigeria	Confirmed	19	7	2014-10-19

`filter()` Example 2

to perform an or use the single |

```
filter( ebola.data
      , (Country == 'Nigeria') | (Country == 'Sierra Leone')
      , `Case def.` == 'Confirmed'
      )
```

an alternate form would be to use `%in%`

```
filter( ebola.data
      , Country %in% c('Nigeria', 'Sierra Leone')
      , `Case def.` == 'Confirmed'
      )
```

`distinct()` - normalizing

From the previous filter example note that report date is repeated week after week.
:::{.keyfunction} To get only distinct observations, use `distinct()`. :::

distinct() - normalizing

```
`Confirmed Cases for Sierra Leone` <-
filter( ebola.data
      , Country == 'Sierra Leone'
      , `Case def.` == 'Confirmed'
      ) %>%
    select(last_col(), Country, `Case def.`, starts_with('Total')) %>%
    distinct()
```

Country Report Date	Country	Case def.	Total cases	Total deaths
2015-11-08	Sierra Leone	Confirmed	8704	3589
2015-11-01	Sierra Leone	Confirmed	8704	3589
2015-10-25	Sierra Leone	Confirmed	8704	3589
2015-10-18	Sierra Leone	Confirmed	8704	3589
2015-10-11	Sierra Leone	Confirmed	8704	3589
2015-10-04	Sierra Leone	Confirmed	8704	3589

Sorting Data

Key Function
To sort data use `arrange()`

`sort()` is the base version but again, less robust.

Arrange allows you to give sorting criteria.

arrange() Example

```
`Confirmed Cases for Sierra Leone` %>%
    arrange(`Country Report Date`, desc(`Case def.`)) %>%
    head()
```

Country Report Date	Country	Case def.	Total cases	Total deaths
2014-08-25	Sierra Leone	Confirmed	935	380
2014-09-05	Sierra Leone	Confirmed	1146	443
2014-09-06	Sierra Leone	Confirmed	1234	461
2014-09-07	Sierra Leone	Confirmed	1287	478
2014-09-13	Sierra Leone	Confirmed	1464	514
2014-09-14	Sierra Leone	Confirmed	1513	517

Reformatting data

- Wide Data
  - multiple observations for one unit are in columns
- Long Data
  - multiple observations for one unit are in rows.

Task: make ‘Case def.’ separate columns

We would like to make `Case def.` separate columns there are however 2 possible response variables:

1. Total Cases
2. Total Deaths

Options?

1. Subset to each value of `Case Def.` Then merge those together.
2. Choose our variable of interest and discard the rest, then pivot or `spread` the column.
3. Spread each column and then ‘bind’ the results together.

Option 1 - filter then merge

```
# Make subsets
confirmed <- ebola.data %>% filter(`Case def.` == 'Confirmed') %>%
    select(-`Case def.`) %>% distinct() %>%
    rename_at(vars(starts_with("total")), ~paste("Confirmed", .))
probable  <- ebola.data %>% filter(`Case def.` == 'Probable') %>%
    select(-`Case def.`) %>% distinct() %>%
    rename_at(vars(starts_with("total")), ~paste("Probable", .))
suspected <- ebola.data %>% filter(`Case def.` == 'Suspected') %>%
    select(-`Case def.`) %>%  distinct() %>%
    rename_at(vars(starts_with("total")), ~paste("Suspected", .))
all.cases <- ebola.data %>% filter(`Case def.` == 'All') %>%
    select(-`Case def.`) %>%  distinct() %>%
    rename_at(vars(starts_with("total")), ~paste("All", .))
```

Option 1 - filter then merge

```
# Join together
ebola.option1 <-
confirmed %>%
    full_join(probable) %>%
    full_join(suspected) %>%
    full_join(all.cases)
```

```
Message:## Joining, by = c("SheetName", "Country", "Country Report Date")
## Joining, by = c("SheetName", "Country", "Country Report Date")
## Joining, by = c("SheetName", "Country", "Country Report Date")
```

```
glimpse(ebola.option1)
```

Combining data

Key Function
Use the **`join`** family of functions to merge data together:

- `inner_join(a, b)` - keep only rows that match both `a` and `b`.
- `left_join(a, b)` - keep all rows of `a` and add columns in `b` to the rows that match. Unmatched rows will contain missing values.
- `right_join(a, b)` - same as left but swap `a` and `b`.
- `full_join(a, b)` - keep all rows of both `a` and `b`.
- `semi_join(a, b)` - keep all rows of `a` that match `b`, but don't add columns from `b`.
- `anti_join(a, b)` - keep only those rows of `a` that **don't** match `b`.

Operations have these parameters:

- `by` - variables to join on, defaults to common variables
- `suffix` - sufixes to add to distinguish common variables that are not part of `by`

Option 2 - pick 1 & spread

```
ebola.option2 <-
    ebola.data %>%
    select(SheetName, Country, `Case def.`, `Total cases`, `Country Report Date`) %>%
    tidyr::spread('Case def.', 'Total cases')
glimpse(ebola.option2)
```

```
## Observations: 838
## Variables: 7
## $ SheetName             <chr> "Apr 01", "Apr 01", "Apr ...
## $ Country               <chr> "Guinea", "Liberia", "Mal...
## $ `Country Report Date` <dttm> 2015-03-29, 2015-03-29, ...
## $ All                   <int> 3492, 9712, 8, 20, 1, 119...
## $ Confirmed             <int> 3068, 3151, 7, 19, 1, 854...
## $ Probable              <int> 414, 1879, 1, 1, 0, 287, ...
## $ Suspected             <int> 10, 4682, 0, 0, 0, 3142, ...
```

Option 3 - Spread each and merge

```
# spread each
cases <- ebola.data %>% select(-`Total deaths`) %>%
    tidyr::spread('Case def.', 'Total cases')
deaths <-  ebola.data %>% select(-`Total cases`) %>%
    tidyr::spread('Case def.', 'Total deaths')
ebola.option3 <-
    full_join( cases, deaths
              , c('SheetName', 'Country', 'Country Report Date')
              , suffix = c(".cases", ".deaths"))
glimpse(ebola.option3)
```

Summarization

Summarization

Key Function

`summarise(data, ...)`

Take the data and summarise it by performing the `...` operations to it.

```
summarize( ebola.option3
         , 'Observations' = n()
         , 'Number of countries' = n_distinct(Country)
         , "# of Reporting dates" = n_distinct(`Country Report Date`)
         , max.cases = max(All.cases, na.rm=TRUE)
         , max.deaths = max(All.deaths, na.rm=TRUE)
         )
```

Observations	Number of countries	# of Reporting dates	max.cases	max.deaths
838	11	120	14122	4806

Grouped Summarization

Key Function

`group_by(data, ...)`

Take the data and group it by variables specified in `...`,
all subsequent operations should be done by group.

```
ebola.option3 %>% group_by(Country) %>%
  summarise( "# of Reporting dates" = n_distinct(`Country Report Date`)
           , max.cases = max(All.cases, na.rm=TRUE)
           , max.deaths = max(All.deaths, na.rm=TRUE)
           )
```

Grouped Summarization

Country	# of Reporting dates	max.cases	max.deaths
Guinea	91	3810	2536
Italy	26	1	0
Liberia	53	10666	4806
Liberia2	27	9	3
Mali	68	8	6
Nigeria	16	22	8
Senegal	15	3	0
Sierra Leone	84	14122	3955
Spain	16	1	0
United Kingdom	47	1	0

Question

Remember the wide data problem?

Time to go back

Exercise

1. Decide on the variable of interest, our `value`

1.5 Summarise to reduce the data to one row per country x reporting date

2. `spread` out the number of cases by date.

5:00

Question

What should we do with our data?

This data set on it's own is not very interesting.

Let's build something interesting.

)) ## World Bank Data

The `wbstats` package provides access to the world bank database.

```
library(wbstats)
wbsearch('population', extra=TRUE)
```

```
## Warning: `wbsearch()` is deprecated as of wbstats 1.0.0.
```

Recoding Country

To merge the the world bank data to our ebola data we need a common country variable.

```
library(countrycode)
long.ebola <- mutate( ebola.option3
                    , iso3c = countrycode(Country, "country.name", "iso3c"))
```

Allways, allways, allways, check your results.

```
assert_that(!any(is.na(long.ebola$iso3c)))
```

```
## [1] TRUE
```

Get the desired population data

- `SP.URB.TOTL.ZS` - Percentage of Population in Urban Areas (in % of Total Population)
- `SP.POP.TOTL.MA.ZS` - Population, male (% of total)
- `SP.POP.TOTL` - Population, total
- `EN.POP.DNST` - Population density (people per sq km)
- `IN.POV.HCR.EST.TOTL` - Poverty HCR Estimates (%) - Total
- `NY.GDP.PCAP.CD` - GDP per capita (current US$)

```
pop.vars <- c( 'SP.URB.TOTL.ZS', 'SP.POP.TOTL.MA.ZS'
             , 'SP.POP.TOTL', 'EN.POP.DNST'
             , 'IN.POV.HCR.EST.TOTL', 'NY.GDP.PCAP.CD')
pop.data <- wb( country = unique(long.ebola$iso3c)
              , indicator = pop.vars
              , startdate = 2014, enddate=2014)
```

Look at the data

1. What format is it in?
2. Are there any problems?
3. Did we get get everything we expected?

Reshape and join together

```
meta.pop.data <- select(pop.data, variable=indicatorID, label=indicator) %>% distinct()
our.data <-
    pop.data %>%
    select(iso3c, value, indicatorID) %>%
    spread(indicatorID, value) %>%
    right_join(long.ebola)
```

```
Message:## Joining, by = "iso3c"
```

Exercise/break

Create a table 1

- Restrict the data to the most recent only.
- Columns should be Africa, Other, and Total.
- Rows should be summaries of the variables we have.
    - minimum, median, mean, maximum …

15:00