# Introduction to Data Reading and Tidying

*Abhinav Anand*

## Setup

The following discussion assumes we have donwloaded R and RStudio. The packages `readr` and `tidyr` need to be installed prior to running the commands below. They are included in the `tidyverse`

1. For downloading R, visit https://cran.r-project.org/
2. For downloading RStudio visit https://www.rstudio.com/

## Reading and Parsing Data Files

The following discussion assumes that all data files referenced are in the same folder as the R codes.

### Reading Plain-Text Files (`.csv`, `.tsv` etc.)

We will be working with the following set of files to illustrate the ideas regarding reading real-life, empirical data files.

```
file_fin_risk <- "FMC_T4_read_file_fin_risk.csv"
file_gdppc <- "FMC_T4_read_file_gdppc.csv"
file_US_corp_spread <- "FMC_T4_read_file_US_corp_spread.csv"
```

While we may read files in formats such as `.xls`, `.xlsx` etc. ("excel files") in R by using the package `readxl`, it is advised by many writers to convert such files into plain-text `.csv` format (comma separated format) and then open them by the `readr` package functions.

### `read_csv()`

`read_csv()` reads `.csv` files. For semicolon separated files, `read_csv2()` function is used.

```r
(fin_risk <- readr::read_csv(file_fin_risk)) #file path
```

```
## Parsed with column specification:
## cols(
##   Country = col_character(),
##   Year = col_integer(),
##   `Risk Points for Foreign Debt as a % of GDP` = col_double(),
##   `Risk Points for Exchange Rate Stability` = col_double(),
##   `Risk Points for Debt Service as a % of XGS` = col_double(),
##   `Risk Points for Current Account as % of XGS` = col_double(),
##   `Risk Points for International Liquidity` = col_double(),
##   `Aggregate Financial Risk` = col_double()
## )
```

```
## # A tibble: 4,380 x 8
##    Country  Year `Risk Points for F~ `Risk Points for ~ `Risk Points for ~
##    <chr>   <int>               <dbl>              <dbl>              <dbl>
##   1 Albania  1984                5.33               9                    NA
##   2 Albania  1985                6                  9                    NA
##   3 Albania  1986                6                  9                    NA
##   4 Albania  1987                6                  8.42                 NA
##   5 Albania  1988                6.25               8                    NA
##   6 Albania  1989                6.54               8                    NA
##   7 Albania  1990                6.96               8                    NA
##   8 Albania  1991                6.75               6.5                  NA
##   9 Albania  1992                4.58               5                    NA
## 10 Albania  1993                 4                   5                    NA
## # ... with 4,370 more rows, and 3 more variables: `Risk Points for Current
## #   Account as % of XGS` <dbl>, `Risk Points for International
## #   Liquidity` <dbl>, `Aggregate Financial Risk` <dbl>
```

read_csv uses the first row as the column names of data. If however, we know this to not be true (sometimes there are a few lines of metadata at the top of the file) we can instruct read_csv to refrain from such behavior.

```
(US_corp_spread <- readr::read_csv(file_US_corp_spread))
```

```
## Warning: Missing column names filled in: 'X2' [2], 'X3' [3], 'X4' [4],
## 'X6' [6], 'X7' [7], 'X8' [8], 'X9' [9]
```

```
## Warning: Duplicated column names deduplicated: 'FRED Graph Observations' =>
## 'FRED Graph Observations_1' [5]
```

```
## # A tibble: 41 x 9
##    `FRED Graph Obse~ X2     X3    X4    `FRED Graph Obs~ X6    X7    X8
##    <chr>             <chr>  <chr> <chr> <chr>            <chr> <chr> <chr>
##  1 Federal Reserve ~ <NA>   <NA>  <NA>  Federal Reserve~ <NA>  <NA>  <NA>
##  2 Link: https://fr~ <NA>   <NA>  <NA>  Link: https://f~ <NA>  <NA>  <NA>
##  3 Help: https://fr~ <NA>   <NA>  <NA>  Help: https://f~ <NA>  <NA>  <NA>
##  4 Economic Researc~ <NA>   <NA>  <NA>  Economic Resear~ <NA>  <NA>  <NA>
##  5 Federal Reserve ~ <NA>   <NA>  <NA>  Federal Reserve~ <NA>  <NA>  <NA>
##  6 <NA>              <NA>   <NA>  <NA>  <NA>             <NA>  <NA>  <NA>
##  7 AAA               Moody~ <NA>  <NA>  BAA              Mood~ <NA>  <NA>
##  8 <NA>              <NA>   <NA>  <NA>  <NA>             <NA>  <NA>  <NA>
##  9 Frequency: Annua~ <NA>   <NA>  <NA>  Frequency: Annu~ <NA>  <NA>  <NA>
## 10 observation_date  AAA    <NA>  <NA>  observation_date BAA   <NA>  <NA>
## # ... with 31 more rows, and 1 more variable: X9 <chr>
```

```
(US_corp_spread_skip <- readr::read_csv(file_US_corp_spread,
                                        skip = 10)
  )
```

```
## Warning: Missing column names filled in: 'X3' [3], 'X4' [4], 'X7' [7],
## 'X8' [8]

## Warning: Duplicated column names deduplicated: 'observation_date' =>
## 'observation_date_1' [5]

## # A tibble: 31 x 9
##    observation_date   AAA X3    X4    observation_date_1   BAA X7    X8
##    <date>           <dbl> <chr> <chr> <date>             <dbl> <chr> <chr>
##  1 1985-01-01       12.1  <NA>  <NA>  1985-01-01         13.3  <NA>  <NA>
##  2 1986-01-01       10.0  <NA>  <NA>  1986-01-01         11.4  <NA>  <NA>
##  3 1987-01-01        8.36 <NA>  <NA>  1987-01-01          9.72 <NA>  <NA>
##  4 1988-01-01        9.88 <NA>  <NA>  1988-01-01         11.1  <NA>  <NA>
##  5 1989-01-01        9.62 <NA>  <NA>  1989-01-01         10.6  <NA>  <NA>
##  6 1990-01-01        8.99 <NA>  <NA>  1990-01-01          9.94 <NA>  <NA>
##  7 1991-01-01        9.04 <NA>  <NA>  1991-01-01         10.4  <NA>  <NA>
##  8 1992-01-01        8.2  <NA>  <NA>  1992-01-01          9.13 <NA>  <NA>
##  9 1993-01-01        7.91 <NA>  <NA>  1993-01-01          8.67 <NA>  <NA>
## 10 1994-01-01        6.92 <NA>  <NA>  1994-01-01          7.65 <NA>  <NA>
## # ... with 21 more rows, and 1 more variable: `BAA-AAA` <dbl>
```

### Notes

1. When the data file does not have column names we can use `col_names = FALSE` to tell `read_csv()` not to treat the first row as headings, and instead label them sequentially from $X1$ to $Xn$.
2. While base R has the classic `read.csv()` function to read `.csv` files, usage of `read_csv()` is encouraged since the latter is said to be around 10 times faster than the former. This is critical when file sizes become large. Additionally, the files are read as tibbles and hence retain their readability, flexibility and reproduceability.
3. Excel files can be read with `readxl()`. Files in formats foreign to R, such as Stata files (`.dta`) can be read using the tidyverse package `haven`.
4. R can also write dataframes into a `.csv` file by use of the command

write_csv().

# Tidying Data

```
(gdppc <- readr::read_csv(file_gdppc)) #which format?
```

```
## Parsed with column specification:
## cols(
##   .default = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
## # A tibble: 264 x 61
##    `Series Name`           `Series Code`  `Country Name`    `Country Code`
##    <chr>                   <chr>          <chr>             <chr>
##  1 GDP per capita (curren~ NY.GDP.PCAP.CD Afghanistan       AFG
##  2 GDP per capita (curren~ NY.GDP.PCAP.CD Albania           ALB
##  3 GDP per capita (curren~ NY.GDP.PCAP.CD Algeria           DZA
##  4 GDP per capita (curren~ NY.GDP.PCAP.CD American Samoa    ASM
##  5 GDP per capita (curren~ NY.GDP.PCAP.CD Andorra           AND
##  6 GDP per capita (curren~ NY.GDP.PCAP.CD Angola            AGO
##  7 GDP per capita (curren~ NY.GDP.PCAP.CD Antigua and Barb~ ATG
##  8 GDP per capita (curren~ NY.GDP.PCAP.CD Arab World        ARB
##  9 GDP per capita (curren~ NY.GDP.PCAP.CD Argentina         ARG
## 10 GDP per capita (curren~ NY.GDP.PCAP.CD Armenia           ARM
## # ... with 254 more rows, and 57 more variables: `1960 [YR1960]` <chr>,
## #   `1961 [YR1961]` <chr>, `1962 [YR1962]` <chr>, `1963 [YR1963]` <chr>,
## #   `1964 [YR1964]` <chr>, `1965 [YR1965]` <chr>, `1966 [YR1966]` <chr>,
## #   `1967 [YR1967]` <chr>, `1968 [YR1968]` <chr>, `1969 [YR1969]` <chr>,
## #   `1970 [YR1970]` <chr>, `1971 [YR1971]` <chr>, `1972 [YR1972]` <chr>,
## #   `1973 [YR1973]` <chr>, `1974 [YR1974]` <chr>, `1975 [YR1975]` <chr>,
## #   `1976 [YR1976]` <chr>, `1977 [YR1977]` <chr>, `1978 [YR1978]` <chr>,
## #   `1979 [YR1979]` <chr>, `1980 [YR1980]` <chr>, `1981 [YR1981]` <chr>,
## #   `1982 [YR1982]` <chr>, `1983 [YR1983]` <chr>, `1984 [YR1984]` <chr>,
## #   `1985 [YR1985]` <chr>, `1986 [YR1986]` <chr>, `1987 [YR1987]` <chr>,
## #   `1988 [YR1988]` <chr>, `1989 [YR1989]` <chr>, `1990 [YR1990]` <chr>,
## #   `1991 [YR1991]` <chr>, `1992 [YR1992]` <chr>, `1993 [YR1993]` <chr>,
## #   `1994 [YR1994]` <chr>, `1995 [YR1995]` <chr>, `1996 [YR1996]` <chr>,
## #   `1997 [YR1997]` <chr>, `1998 [YR1998]` <chr>, `1999 [YR1999]` <chr>,
## #   `2000 [YR2000]` <chr>, `2001 [YR2001]` <chr>, `2002 [YR2002]` <chr>,
## #   `2003 [YR2003]` <chr>, `2004 [YR2004]` <chr>, `2005 [YR2005]` <chr>,
## #   `2006 [YR2006]` <chr>, `2007 [YR2007]` <chr>, `2008 [YR2008]` <chr>,
## #   `2009 [YR2009]` <chr>, `2010 [YR2010]` <chr>, `2011 [YR2011]` <chr>,
## #   `2012 [YR2012]` <chr>, `2013 [YR2013]` <chr>, `2014 [YR2014]` <chr>,
## #   `2015 [YR2015]` <chr>, `2016 [YR2016]` <chr>
```

```
head(fin_risk) #which format?
```

```
## # A tibble: 6 x 8
##   Country  Year `Risk Points for F~ `Risk Points for E~ `Risk Points for ~
##   <chr>   <int>               <dbl>               <dbl>              <dbl>
## 1 Albania  1984                5.33                9                    NA
## 2 Albania  1985                6                   9                    NA
## 3 Albania  1986                6                   9                    NA
## 4 Albania  1987                6                   8.42                 NA
## 5 Albania  1988                6.25                8                    NA
## 6 Albania  1989                6.54                8                    NA
## # ... with 3 more variables: `Risk Points for Current Account as % of
## #   XGS` <dbl>, `Risk Points for International Liquidity` <dbl>,
## #   `Aggregate Financial Risk` <dbl>
```

## The Tidy Format

The tidy format has three characteristics:

1. Each variable is a column

2. Each observation is a row
3. Each value is a cell

fin_risk is a tidy dataset, gdppc is not.

Not all formats of data are equally good for analysis. For the tidyverse, the best format to work with is the "tidy" format. dplyr, ggplot2 and all the other packages in the tidyverse are designed to work best with tidy data.

```
(fin_risk_tidy <- fin_risk %>%
  dplyr::rename(risk_foreign =
                  `Risk Points for Foreign Debt as a % of GDP`) %>%
  dplyr::rename(risk_exchange =
                  `Risk Points for Exchange Rate Stability`) %>%
  dplyr::rename(risk_debt =
                  `Risk Points for Debt Service as a % of XGS`) %>%
  dplyr::rename(risk_CA =
                  `Risk Points for Current Account as % of XGS`) %>%
  dplyr::rename(risk_liq =
                  `Risk Points for International Liquidity`) %>%
  dplyr::rename(risk_agg_fin = `Aggregate Financial Risk`)
 )
```

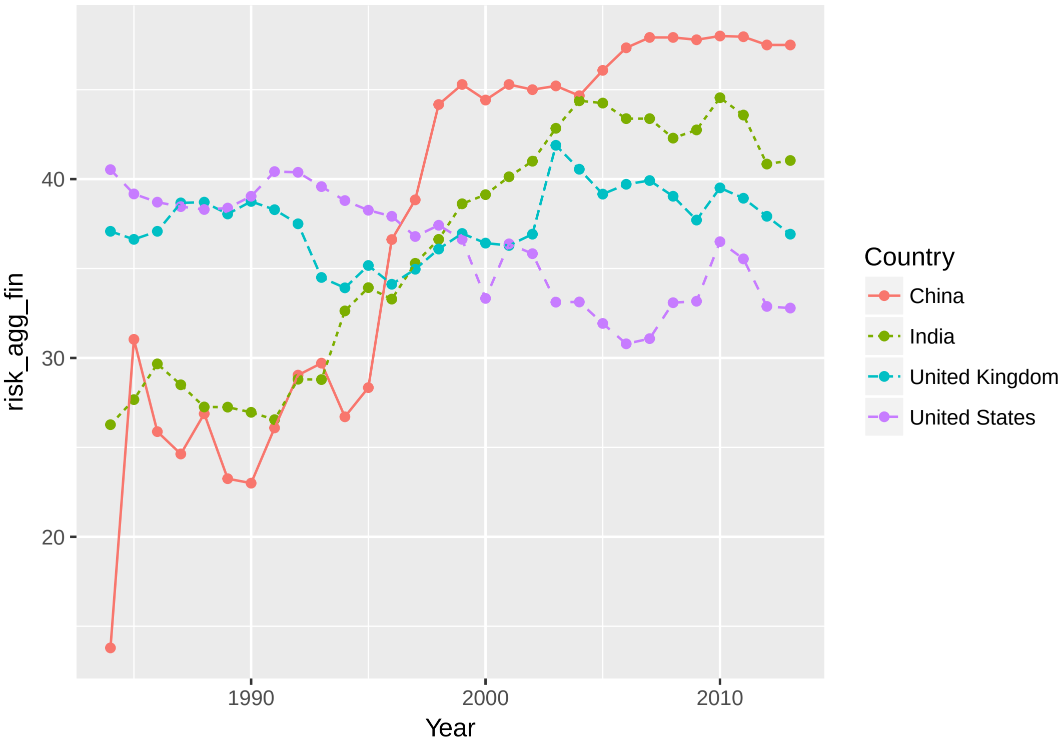
```
## # A tibble: 4,380 x 8
##    Country  Year risk_foreign risk_exchange risk_debt risk_CA risk_liq
##    <chr>   <int>        <dbl>         <dbl>     <dbl>   <dbl>    <dbl>
##  1 Albania  1984         5.33          9           NA      NA       NA
##  2 Albania  1985         6             9           NA      NA       NA
##  3 Albania  1986         6             9           NA      NA       NA
##  4 Albania  1987         6             8.42        NA      NA       NA
##  5 Albania  1988         6.25          8           NA      NA       NA
##  6 Albania  1989         6.54          8           NA      NA       NA
##  7 Albania  1990         6.96          8           NA      NA       NA
##  8 Albania  1991         6.75          6.5         NA      NA       NA
##  9 Albania  1992         4.58          5           NA      NA       NA
## 10 Albania  1993          4              5           NA       NA        NA
## # ... with 4,370 more rows, and 1 more variable: risk_agg_fin <dbl>
```

```
(fin_risk_tidy_summ <- fin_risk_tidy %>%
    dplyr::group_by(Year) %>%
    dplyr::summarise(risk_agg_min =
                       min(risk_agg_fin, na.rm = T),
                     risk_agg_max =
                       max(risk_agg_fin, na.rm = T),
                     risk_agg_med =
                       median(risk_agg_fin, na.rm = T),
                     risk_agg_mean =
                       mean(risk_agg_fin, na.rm = T),
                     risk_agg_std =
                       sd(risk_agg_fin, na.rm = T),
                     risk_agg_iqr =
                       IQR(risk_agg_fin, na.rm = T)
                     )

)
```

```
## # A tibble: 30 x 7
##     Year risk_agg_min risk_agg_max risk_agg_med risk_agg_mean risk_agg_std
##    <int>        <dbl>        <dbl>        <dbl>         <dbl>        <dbl>
##  1  1984            0         45.1         22.5          19.3         14.3
##  2  1985            0        116.          26.1          23.8         14.9
##  3  1986            0         44.8         25.9          23.7         12.2
##  4  1987            0         45.5         25.9          23.6         12.3
##  5  1988            0         46.6         26.7          24.1         12.5
##  6  1989            0         46           26.4          24.2         12.5
##  7  1990            0         45.8         26.6          24.8         12.3
##  8  1991            0         45.8         27.7          25.2         12.8
##  9  1992            0         46.0         29.7          25.9         13.2
## 10  1993            0         44.4         29.9          26.0         12.9
## # ... with 20 more rows, and 1 more variable: risk_agg_iqr <dbl>
```

```
ggplot(data = filter(fin_risk_tidy,
                     Country %in% c("United Kingdom",
                                    "United States",
                                    "China",
                                    "India")
                     ),
       mapping = aes(x = Year,
                     y = risk_agg_fin,
                     color = Country)) +
  geom_point() +
  geom_line(mapping = aes(linetype = Country))
```



## Gathering

This is used to “gather” data from the wide format, to the long format.

```r
gdppc %>% head(.)
```

```
## # A tibble: 6 x 61
##   `Series Name`                `Series Code` `Country Name` `Country Code`
##   <chr>                        <chr>         <chr>          <chr>
## 1 GDP per capita (current US$) NY.GDP.PCAP.~ Afghanistan    AFG
## 2 GDP per capita (current US$) NY.GDP.PCAP.~ Albania        ALB
## 3 GDP per capita (current US$) NY.GDP.PCAP.~ Algeria        DZA
## 4 GDP per capita (current US$) NY.GDP.PCAP.~ American Samoa ASM
## 5 GDP per capita (current US$) NY.GDP.PCAP.~ Andorra        AND
## 6 GDP per capita (current US$) NY.GDP.PCAP.~ Angola         AGO
## # ... with 57 more variables: `1960 [YR1960]` <chr>, `1961
## #   [YR1961]` <chr>, `1962 [YR1962]` <chr>, `1963 [YR1963]` <chr>, `1964
## #   [YR1964]` <chr>, `1965 [YR1965]` <chr>, `1966 [YR1966]` <chr>, `1967
## #   [YR1967]` <chr>, `1968 [YR1968]` <chr>, `1969 [YR1969]` <chr>, `1970
## #   [YR1970]` <chr>, `1971 [YR1971]` <chr>, `1972 [YR1972]` <chr>, `1973
## #   [YR1973]` <chr>, `1974 [YR1974]` <chr>, `1975 [YR1975]` <chr>, `1976
## #   [YR1976]` <chr>, `1977 [YR1977]` <chr>, `1978 [YR1978]` <chr>, `1979
## #   [YR1979]` <chr>, `1980 [YR1980]` <chr>, `1981 [YR1981]` <chr>, `1982
## #   [YR1982]` <chr>, `1983 [YR1983]` <chr>, `1984 [YR1984]` <chr>, `1985
## #   [YR1985]` <chr>, `1986 [YR1986]` <chr>, `1987 [YR1987]` <chr>, `1988
## #   [YR1988]` <chr>, `1989 [YR1989]` <chr>, `1990 [YR1990]` <chr>, `1991
## #   [YR1991]` <chr>, `1992 [YR1992]` <chr>, `1993 [YR1993]` <chr>, `1994
## #   [YR1994]` <chr>, `1995 [YR1995]` <chr>, `1996 [YR1996]` <chr>, `1997
## #   [YR1997]` <chr>, `1998 [YR1998]` <chr>, `1999 [YR1999]` <chr>, `2000
## #   [YR2000]` <chr>, `2001 [YR2001]` <chr>, `2002 [YR2002]` <chr>, `2003
## #   [YR2003]` <chr>, `2004 [YR2004]` <chr>, `2005 [YR2005]` <chr>, `2006
## #   [YR2006]` <chr>, `2007 [YR2007]` <chr>, `2008 [YR2008]` <chr>, `2009
## #   [YR2009]` <chr>, `2010 [YR2010]` <chr>, `2011 [YR2011]` <chr>, `2012
## #   [YR2012]` <chr>, `2013 [YR2013]` <chr>, `2014 [YR2014]` <chr>, `2015
## #   [YR2015]` <chr>, `2016 [YR2016]` <chr>
```

```
col_yr_1 <- "1960 [YR1960]"
col_yr_end <- "2016 [YR2016]"

(gdppc_tidy <- gdppc %>%
    dplyr::select(-c(`Series Name`,
                     `Series Code`,
                     `Country Code`
                     )
                  ) %>%
    dplyr::rename(Country = `Country Name`) %>%
    tidyr::gather(col_yr_1:col_yr_end,
                  key = "Year",
                  value = "GDP_per_capita"
                  ) %>%
    dplyr::arrange(Country)

  )
```

```
## # A tibble: 15,048 x 3
##    Country     Year          GDP_per_capita
##    <chr>       <chr>         <chr>
##  1 Afghanistan 1960 [YR1960] 59.7773265084
##  2 Afghanistan 1961 [YR1961] 59.8781528089
##  3 Afghanistan 1962 [YR1962] 58.4928738323
##  4 Afghanistan 1963 [YR1963] 78.7827580363
##  5 Afghanistan 1964 [YR1964] 82.2084438594
##  6 Afghanistan 1965 [YR1965] 101.2904712742
##  7 Afghanistan 1966 [YR1966] 137.899361897
##  8 Afghanistan 1967 [YR1967] 161.3220000885
##  9 Afghanistan 1968 [YR1968] 129.5066538443
## 10 Afghanistan 1969 [YR1969] 129.7985414084
## # ... with 15,038 more rows
```

### Spreading

### Joining