Tibbles and Tidy Data

Prof. Eric A. Suess

October 8, 2018

Today we will take a look at tibbles, tribbles, and importing data

```
library(tidyverse)
```

Traditional data.frame

```
my.data <- data.frame(
  x1 = c(12,34,45),
  x2 = c("M","F", "F")
)
my.data
```

```
##   x1 x2
## 1 12  M
## 2 34  F
## 3 45  F
```

```
class(my.data)
```

```
## [1] "data.frame"
```

The new data_frame tibble

```
my_data <- data_frame(
  x1 = c(12,34,45),
  x2 = c("M","F", "F")
)
my_data
```

```
## # A tibble: 3 x 2
##      x1 x2
##   <dbl> <chr>
## 1    12 M
## 2    34 F
## 3    45 F
```

```
class(my_data)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
mtcars
```

```
##                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4          21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag      21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710         22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive     21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
```

```
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```r
class(mtcars)
```

```
## [1] "data.frame"
```

```r
mtcars <- as.tibble(mtcars)
mtcars
```

```
## # A tibble: 32 x 11
##      mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##  * <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  21       6  160    110  3.9   2.62  16.5     0     1     4     4
##  2  21       6  160    110  3.9   2.88  17.0     0     1     4     4
##  3  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1
##  4  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1
##  5  18.7     8  360    175  3.15  3.44  17.0     0     0     3     2
##  6  18.1     6  225    105  2.76  3.46  20.2     1     0     3     1
##  7  14.3     8  360    245  3.21  3.57  15.8     0     0     3     4
##  8  24.4     4  147.    62  3.69  3.19  20       1     0     4     2
##  9  22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2
## 10  19.2     6  168.   123  3.92  3.44  18.3     1     0     4     4
## # ... with 22 more rows
```

tribble

```r
my_data <- tribble(
  ~x1,  ~x2,
  12,   "M",
  34,   "F",
  45,   "F"
)
my_data
```

```
## # A tibble: 3 x 2
##      x1 x2
##   <dbl> <chr>
## 1    12 M
## 2    34 F
## 3    45 F
```

```r
class(my_data)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

Access rows and columns

```r
my_data
```

```
## # A tibble: 3 x 2
##      x1 x2
##   <dbl> <chr>
## 1    12 M
## 2    34 F
## 3    45 F
```

```r
# columns

my_data[1]
```

```
## # A tibble: 3 x 1
##      x1
##   <dbl>
## 1    12
## 2    34
## 3    45
```

```r
my_data[[1]]
```

```
## [1] 12 34 45
```

```r
my_data$x1
```

```
## [1] 12 34 45
```

```r
my_data$x2
```

```
## [1] "M" "F" "F"
```

```r
# rows

my_data[1,]
```

```
## # A tibble: 1 x 2
##      x1 x2
##   <dbl> <chr>
## 1    12 M
```

```
my_data[2,]
```

```
## # A tibble: 1 x 2
##      x1 x2
##   <dbl> <chr>
## 1    34 F
```

```
my_data[3,]
```

```
## # A tibble: 1 x 2
##      x1 x2
##   <dbl> <chr>
## 1    45 F
```

Read comma separated value data

```
my_data2 <- read_csv(
  "x1, x2
   12,  M
   34,  F
   45,  F"
)
my_data2
```

```
## # A tibble: 3 x 2
##      x1 x2
##   <int> <chr>
## 1    12 M
## 2    34 F
## 3    45 F
```

Parsing

```
x <- c("TRUE", "FALSE", "FALSE")
str(x)
```

```
##  chr [1:3] "TRUE" "FALSE" "FALSE"
```

```
class(x)
```

```
## [1] "character"
```

```
x <- parse_logical(x)
str(x)
```

```
##  logi [1:3] TRUE FALSE FALSE
```

```
class(x)
```

```
## [1] "logical"
```

```r
x <- c(23,34,45)
str(x)
```

```
##  num [1:3] 23 34 45
```

```r
class(x)
```

```
## [1] "numeric"
```

```r
x <- parse_integer(x)
str(x)
```

```
##  int [1:3] 23 34 45
```

```r
class(x)
```

```
## [1] "integer"
```

```r
x <- c("2018-10-06", "2018-10-07", "2018-10-08")
str(x)
```

```
##  chr [1:3] "2018-10-06" "2018-10-07" "2018-10-08"
```

```r
class(x)
```

```
## [1] "character"
```

```r
x <- parse_date(x)
str(x)
```

```
##  Date[1:3], format: "2018-10-06" "2018-10-07" "2018-10-08"
```

```r
class(x)
```

```
## [1] "Date"
```

```r
problems(x)
```

```
## # tibble [0 x 4]
## # ... with 4 variables: row <int>, col <int>, expected <chr>, actual <chr>
```

```r
x <- parse_number(x)
str(x)
```

```
##  num [1:3] 2018 2018 2018
```

```r
class(x)
```

```
## [1] "numeric"
```

```r
x <- parse_double(x)
str(x)
```

```
##  num [1:3] 2018 2018 2018
```

```r
class(x)
```

```
## [1] "numeric"
```

Strings, ASCII code

```r
charToRaw("Prof. Suess")
```

```
##  [1] 50 72 6f 66 2e 20 53 75 65 73 73
```

Dates

```r
parse_date("2018-10-08")
```

```
## [1] "2018-10-08"
```

```r
library(hms)

parse_time("06:30 pm")
```

```
## 18:30:00
```

```r
parse_date("10/08/18", "%m/%d/%y")
```

```
## [1] "2018-10-08"
```

Parsing a file

This is a useful section to read over. It is about read an external .csv file into R.

```r
challenge <- read_csv(readr_example("challenge.csv"))
```

```
## Parsed with column specification:
## cols(
##   x = col_integer(),
##   y = col_character()
## )
```

```
## Warning in rbind(names(probs), probs_f): number of columns of result is not
## a multiple of vector length (arg 1)
```

```
## Warning: 1000 parsing failures.
## row # A tibble: 5 x 5 col     row col   expected       actual      file
## ... ................. ... ..........................................................................
## See problems(...) for more details.
```

```r
problems(challenge)
```

```
## # A tibble: 1,000 x 5
##      row col   expected       actual      file
##    <int> <chr> <chr>          <chr>       <chr>
##  1  1001 x     no trailing c~ .238379750~ '/home/esuess/R/x86_64-pc-linux~
##  2  1002 x     no trailing c~ .411679971~ '/home/esuess/R/x86_64-pc-linux~
##  3  1003 x     no trailing c~ .746071676~ '/home/esuess/R/x86_64-pc-linux~
##  4  1004 x     no trailing c~ .723450553~ '/home/esuess/R/x86_64-pc-linux~
##  5  1005 x     no trailing c~ .614524137~ '/home/esuess/R/x86_64-pc-linux~
##  6  1006 x     no trailing c~ .473980569~ '/home/esuess/R/x86_64-pc-linux~
##  7  1007 x     no trailing c~ .578461039~ '/home/esuess/R/x86_64-pc-linux~
##  8  1008 x     no trailing c~ .241593722~ '/home/esuess/R/x86_64-pc-linux~
##  9  1009 x     no trailing c~ .114378662~ '/home/esuess/R/x86_64-pc-linux~
## 10  1010 x     no trailing c~ .298344632~ '/home/esuess/R/x86_64-pc-linux~
## # ... with 990 more rows
```

```r
challenge <- read_csv(
  readr_example("challenge.csv"),
  col_types = cols(
    x = col_integer(),
    y = col_character()
  )
)
```

```
## Warning in rbind(names(probs), probs_f): number of columns of result is not
## a multiple of vector length (arg 1)
```

```
## Warning: 1000 parsing failures.
## row # A tibble: 5 x 5 col     row col   expected       actual      file
## ... ................. ... ..........................................................................
## See problems(...) for more details.
```

```r
challenge <- read_csv(
  readr_example("challenge.csv"),
  col_types = cols(
    x = col_double(),
    y = col_character()
  )
)
head(challenge)
```

```
## # A tibble: 6 x 2
##       x y
##   <dbl> <chr>
## 1   404 <NA>
## 2  4172 <NA>
## 3  3004 <NA>
## 4   787 <NA>
## 5    37 <NA>
## 6  2332 <NA>
```

```r
tail(challenge)
```

```
## # A tibble: 6 x 2
##       x y
##   <dbl> <chr>
## 1 0.805 2019-11-21
## 2 0.164 2018-03-29
## 3 0.472 2014-08-04
## 4 0.718 2015-08-16
## 5 0.270 2020-02-04
## 6 0.608 2019-01-06
```

```r
challenge <- read_csv(
  readr_example("challenge.csv"),
  col_types = cols(
    x = col_double(),
    y = col_date()
  )
)
head(challenge)
```

```
## # A tibble: 6 x 2
##       x y
##   <dbl> <date>
## 1   404 NA
## 2  4172 NA
## 3  3004 NA
## 4   787 NA
## 5    37 NA
## 6  2332 NA
```

```r
tail(challenge)
```

```
## # A tibble: 6 x 2
##       x y
##   <dbl> <date>
## 1 0.805 2019-11-21
## 2 0.164 2018-03-29
## 3 0.472 2014-08-04
## 4 0.718 2015-08-16
## 5 0.270 2020-02-04
## 6 0.608 2019-01-06
```

Writing a csv

Write it, go find it, open it in MS Excel.

```r
write_csv(challenge, "challenge.csv")
```

Write an .rds file. This is an R dataset file.

```r
write_rds(challenge, "challenge.rds")
read_rds("challenge.rds")
```

```
## # A tibble: 2,000 x 2
##        x y
##    <dbl> <date>
##  1   404 NA
##  2  4172 NA
##  3  3004 NA
##  4   787 NA
##  5    37 NA
##  6  2332 NA
##  7  2489 NA
##  8  1449 NA
##  9  3665 NA
## 10  3863 NA
## # ... with 1,990 more rows
```