# DATA 607 Project 1

Daniel Moscoe

2/24/2021

**Introduction** How can we transform a text file with partially structured data into a more useful form? In this project I examine one such file that reports the proceedings of a chess tournament. By exploiting the structure of the text file and employing tools in R, I show how to create a tibble containing all the information in the text file in a form more amenable to further analysis. Finally, I generate a CSV containing a subset of this tibble.

**Procedure Overview** There are 6 steps to the transformation procedure.

1. Create a tibble containing the raw data from the text file, delimited by line breaks.
2. Reorganize the rows of the table so that all the information pertaining to each player resides in a single row.
3. Reorganize the columns of the table so that each field resides in a single column.
4. Resolve anomalies in the data.
5. Compute the average pre-tournament opponent rating for each player.
6. Create a CSV containing the requested information on each player.

**Procedure Detail**

1. Create `chess_line_delim`, a tibble containing information from the raw text file. Although the text file contains headers, I ignore them for now. Column names are added in step 3.

```
data_on_github = "https://raw.githubusercontent.com/dmoscoe/SPS/main/DATA607/proj1/tournamentinfo.txt"

chess_line_delim <- read.delim(url(data_on_github), header = FALSE, sep = "\n")
head(chess_line_delim)
```

```
##                                                                                           V1
## 1  -----------------------------------------------------------------------------------------
## 2  Pair | Player Name                     |Total|Round|Round|Round|Round|Round|Round|Round|
## 3  Num  | USCF ID / Rtg (Pre->Post)      | Pts |  1  |  2  |  3  |  4  |  5  |  6  |  7  |
## 4  -----------------------------------------------------------------------------------------
## 5     1 | GARY HUA                       |6.0  |W  39|W  21|W  18|W  14|W   7|D  12|D   4|
## 6    ON | 15445895 / R: 1794   ->1817    |N:2  |W    |B    |W    |B    |W    |B    |W    |
```

2. The raw text file contains three rows per player. Two of these lines contain data, and the third consists of a bar of hyphens that acts as a row border. Here I collapse the data from the two informative rows into a single row for each player, and I discard the border row. The result is a vector of strings, `chess_line_collapse`, that will be easy to divide into columns.

```
chess_line_collapse <- c()
j <- 1
for (i in seq(1: length(chess_line_delim[,1]))) {
  if (i %% 3 == 2) {
    next_line <- str_c(chess_line_delim[i, 1], chess_line_delim[i + 1, 1])
    chess_line_collapse <- c(chess_line_collapse, next_line)
    j <- j + 1
  }
}
head(chess_line_collapse)
```

```
## [1] " Pair | Player Name                     |Total|Round|Round|Round|Round|Round|Round|  Num
## [2] "    1 | GARY HUA                       |6.0  |W  39|W  21|W  18|W  14|W   7|D  12|D   4|   ON
## [3] "    2 | DAKSHESH DARURI                 |6.0  |W  63|W  58|L   4|W  17|W  16|W  20|W   7|   MI
## [4] "    3 | ADITYA BAJAJ                    |6.0  |L   8|W  61|W  25|W  21|W  11|W  13|W  12|   MI
## [5] "    4 | PATRICK H SCHILLING             |5.5  |W  23|D  28|W   2|W  26|D   5|W  19|D   1|   MI
## [6] "    5 | HANSHI ZUO                      |5.5  |W  45|W  37|D  12|D  13|D   4|W  14|W  17|   MI
```

3. The raw text file uses pipes and hyphens to mimic the structure of a table. As a result of this pseudo-tabular structure, each field of the dataset resides at a fixed position in each row. To examine the position of each field, I export `chess_line_collapse` as a txt file.

```
for (i in seq(1:length(chess_line_collapse))) {
  write_lines(chess_line_collapse[i], str_c(path,"chess_line_collapse.txt"), append = TRUE)
}
```

Viewing the txt file with a fixed-width font, and placing a "ruler" across the top of the file, allow me to easily read off the starting and ending position of each field.

I use these positions to construct `tidy_chess`, a tibble with each player's information in one row, and each field in one column. This tibble contains more information than is necessary to produce the CSV file that is the final product for this project. However, since the information is easy to capture, I preserve it.

```
tidy_chess <- read_fwf(chess_line_collapse,
                       trim_ws = TRUE, skip = 1, fwf_cols(
                         PairNum = c(1, 6),
                         USCFID = c(98, 105),
                         State = c(93, 94),
                         PlayerName = c(8, 40),
                         PreRating = c(112, 118),
                         RtgPost = c(121, 129),
                         TotalPtsN = c(133, 133),
                         TotalPoints = c(42, 46),
                         Rd1Opp = c(51, 52),
                         Rd1Color = c(137, 137),
                         Rd1Result = c(48, 48),
                         Rd2Opp = c(57, 58),
                         Rd2Color = c(143, 143),
                         Rd2Result = c(54, 54),
                         Rd3Opp = c(63, 64),
                         Rd3Color = c(149, 149),
                         Rd3Result = c(60, 60),
                         Rd4Opp = c(69, 70),
```

```
                          Rd4Color = c(155, 155),
                          Rd4Result = c(66, 66),
                          Rd5Opp = c(75, 76),
                          Rd5Color = c(161, 161),
                          Rd5Result = c(72, 72),
                          Rd6Opp = c(81, 82),
                          Rd6Color = c(167, 167),
                          Rd6Result = c(78, 78),
                          Rd7Opp = c(87, 88),
                          Rd7Color = c(173, 173),
                          Rd7Result = c(84, 84)

))
head(tidy_chess)
```

```
## # A tibble: 6 x 29
##   PairNum USCFID State PlayerName PreRating RtgPost TotalPtsN TotalPoints Rd1Opp
##     <dbl>  <dbl> <chr> <chr>      <chr>     <chr>       <dbl>       <dbl>  <dbl>
## 1       1 1.54e7 ON    GARY HUA   1794      1817            2         6       39
## 2       2 1.46e7 MI    DAKSHESH ~ 1553      1663            2         6       63
## 3       3 1.50e7 MI    ADITYA BA~ 1384      1640            2         6        8
## 4       4 1.26e7 MI    PATRICK H~ 1716      1744            2         5.5     23
## 5       5 1.46e7 MI    HANSHI ZUO 1655      1690            2         5.5     45
## 6       6 1.51e7 OH    HANSEN SO~ 1686      1687            3         5       34
## # ... with 20 more variables: Rd1Color <chr>, Rd1Result <chr>, Rd2Opp <dbl>,
## #   Rd2Color <chr>, Rd2Result <chr>, Rd3Opp <dbl>, Rd3Color <chr>,
## #   Rd3Result <chr>, Rd4Opp <dbl>, Rd4Color <chr>, Rd4Result <chr>,
## #   Rd5Opp <dbl>, Rd5Color <chr>, Rd5Result <chr>, Rd6Opp <dbl>,
## #   Rd6Color <chr>, Rd6Result <chr>, Rd7Opp <dbl>, Rd7Color <chr>,
## #   Rd7Result <chr>
```

4. Some entries in `PreRating` and `RtgPost` end with a suffix beginning "P...". This suffix indicates the number of games contributing to the rating. In order to compute average opponent pre-chess ratings for each player, I remove the suffix from each entry in `PreRating` where it appears. This is easily accomplished with `parse_number()`.

```
tidy_chess <- tidy_chess %>%
  mutate(PreRating = parse_number(PreRating),
         RtgPost = parse_number(RtgPost))
```

5. To compute the average pre-chess opponent rating for each player, I add columns to `tidy_chess`. Each added column contains opponent ratings for one round.

```
tidy_chess$Rd1OppRtg <- tidy_chess$PreRating[match(tidy_chess$Rd1Opp, tidy_chess$PairNum)]
tidy_chess$Rd2OppRtg <- tidy_chess$PreRating[match(tidy_chess$Rd2Opp, tidy_chess$PairNum)]
tidy_chess$Rd3OppRtg <- tidy_chess$PreRating[match(tidy_chess$Rd3Opp, tidy_chess$PairNum)]
tidy_chess$Rd4OppRtg <- tidy_chess$PreRating[match(tidy_chess$Rd4Opp, tidy_chess$PairNum)]
tidy_chess$Rd5OppRtg <- tidy_chess$PreRating[match(tidy_chess$Rd5Opp, tidy_chess$PairNum)]
tidy_chess$Rd6OppRtg <- tidy_chess$PreRating[match(tidy_chess$Rd6Opp, tidy_chess$PairNum)]
tidy_chess$Rd7OppRtg <- tidy_chess$PreRating[match(tidy_chess$Rd7Opp, tidy_chess$PairNum)]
```

I then calculate the average across the new columns for each player to determine the player's average pre-chess opponent rating.

```
tidy_chess <- tidy_chess %>%
  rowwise() %>%
  mutate(AvgPreChessRtgOpp = round(
    mean(
      c(Rd1OppRtg, Rd2OppRtg, Rd3OppRtg, Rd4OppRtg, Rd5OppRtg, Rd6OppRtg, Rd7OppRtg),
      na.rm = TRUE),
    0))
head(tidy_chess)
```

```
## # A tibble: 6 x 37
## # Rowwise:
##   PairNum USCFID State PlayerName PreRating RtgPost TotalPtsN TotalPoints Rd1Opp
##     <dbl>  <dbl> <chr> <chr>          <dbl>   <dbl>     <dbl>       <dbl>  <dbl>
## 1       1 1.54e7 ON    GARY HUA        1794    1817         2         6       39
## 2       2 1.46e7 MI    DAKSHESH ~      1553    1663         2         6       63
## 3       3 1.50e7 MI    ADITYA BA~      1384    1640         2         6        8
## 4       4 1.26e7 MI    PATRICK H~      1716    1744         2         5.5     23
## 5       5 1.46e7 MI    HANSHI ZUO      1655    1690         2         5.5     45
## 6       6 1.51e7 OH    HANSEN SO~      1686    1687         3         5       34
## # ... with 28 more variables: Rd1Color <chr>, Rd1Result <chr>, Rd2Opp <dbl>,
## #   Rd2Color <chr>, Rd2Result <chr>, Rd3Opp <dbl>, Rd3Color <chr>,
## #   Rd3Result <chr>, Rd4Opp <dbl>, Rd4Color <chr>, Rd4Result <chr>,
## #   Rd5Opp <dbl>, Rd5Color <chr>, Rd5Result <chr>, Rd6Opp <dbl>,
## #   Rd6Color <chr>, Rd6Result <chr>, Rd7Opp <dbl>, Rd7Color <chr>,
## #   Rd7Result <chr>, Rd1OppRtg <dbl>, Rd2OppRtg <dbl>, Rd3OppRtg <dbl>,
## #   Rd4OppRtg <dbl>, Rd5OppRtg <dbl>, Rd6OppRtg <dbl>, Rd7OppRtg <dbl>,
## #   AvgPreChessRtgOpp <dbl>
```

6. Finally, I create a CSV with the required information by selecting from `tidy_chess`.

```
final_product <- tidy_chess %>%
  select(PlayerName,State,TotalPoints,PreRating, AvgPreChessRtgOpp)
write_csv(final_product, str_c(path,"final_product.csv"))
head(final_product)
```

```
## # A tibble: 6 x 5
## # Rowwise:
##   PlayerName          State TotalPoints PreRating AvgPreChessRtgOpp
##   <chr>               <chr>       <dbl>     <dbl>             <dbl>
## 1 GARY HUA            ON            6        1794              1605
## 2 DAKSHESH DARURI     MI            6        1553              1469
## 3 ADITYA BAJAJ        MI            6        1384              1564
## 4 PATRICK H SCHILLING MI            5.5      1716              1574
## 5 HANSHI ZUO          MI            5.5      1655              1501
## 6 HANSEN SONG         OH            5        1686              1519
```

**Discussion** Transforming data into a useful form almost always relies on identifying and exploiting structure in the raw data. Sometimes this structure is found in reliable patterns within the data itself. In this case, extracting sequences based on a match with a regular expression can be useful. Sometimes there is a delimiter, like a comma or a tab, that distinguishes each entry from its neighbors. The data examined in this project was not consistently delimited. However, it was presented in a pseudo-tabular structure, presumably

so that it could be easily read by a person. In preparing this data for analysis, I relied on this pseudo-tabular structure, along with a "ruler" that helped me identify the starting and ending positions of each field. While this strategy is simple, there is also something satisfying about relying only on the structure intentionally supplied by the "authors" of this file. It enabled me to easily capture all the data in the file, rather than narrowing my efforts to only the information required for producing the final product. The consequence is that options for future analysis of this data are broader than they otherwise might have been.

I believe there is room for improvement in my approach to computing the average pre-chess opponent rating, since this is the most repetitive part of my code. Is there a way I could have used a loop to perform this same column-creating operation seven times, without writing seven separate lines? Is there some other way to compute the required result without generating additional columns in the first place?

In future assignments I will look for ways to create more visually appealing R Markdown documents. I would like to learn more about dividing my work into visually recognizable sections, or including a sidebar for navigation.