Documented code for task 1

Group I

15/5/2020

Contents

Library Imports

```r
library(readxl)
library(dplyr)
library(tidyr)
library(stringr)
```

Reading and processing raw data

Load data

```
FV19TOTA <- read_excel("data/FV19TOTA.xlsx") # No external preprocessing has been done
                                             # to the imported data
FV19TOTA <- FV19TOTA %>%
  rename("Party" = "Folketingsvalget 2019 efter valgresultat, område og tid",
         "Area" = "...2",
         "Votes" = "...3")
FV19TOTA$Votes <- as.integer(FV19TOTA$Votes)

characteristics_data <- read_excel("data/PollingDistrictCharacteristics.xlsx",skip=4)
```

Find unique parties and fill empty spaces between parties for joining later

```
FV19TOTA <- fill(FV19TOTA, "Party", .direction = "down")
Parties <- unique(FV19TOTA$Party)[2:15]
```

Find unique multi-member constituencies (storkredse) and format them to be used as keys for FV19TOTA dataset

```
MMCs <- unique(characteristics_data$Storkreds)[2:11]
for (i in 1:length(MMCs))
{
  MMCs[i] <- toupper(substr(MMCs[i], 4, nchar(MMCs[i])))
}
```

as well as provinces, which are manually defined

```
provinces = c("HOVEDSTADEN","SJÆLLAND-SYDDANMARK","MIDTJYLLAND-NORDJYLLAND")
```

Convenience functions allowing for subsetting data based on multi-member constituency (storkreds) and province (landsdel)

The functions add the desired new columns to the data frame passed to it. It does so by copying the Area column (which was named in the first code chunk) and removing all elements that are not in a given vector. It then fills the holes downwards.

```
addMMCCol <- function(df,vecOfVal)
{
  df$MMC <- df$Area

  for (row in 1:nrow(df))
  {
    area <- df[row, "MMC"]
    if (!(area %in% vecOfVal))
    {
      df$MMC[row] = NA
    }
  }

  df <- fill(df, "MMC", .direction = "down")
  return(df)
}
```

```r
addProvinceCol <- function(df,vecOfVal)
{
  df$Province <- df$Area

  for (row in 1:nrow(df))
  {
    area <- df[row, "Province"]
    if (!(area %in% vecOfVal))
    {
      df$Province[row] = NA
    }
  }

  df <- fill(df, "Province", .direction = "down")
  return(df)
}
```

Formatting

```r
vote_distribution <- data.frame()
for (i in 1:length(Parties))
{
  tmp <- subset(FV19TOTA, Party == Parties[i]) # Extract a temporary dataframe from
                                               # FV19TOTA for the corresponding party

  # Add and fill columns for multi-member constituencies and provinces.
  tmp <- addMMCCol(tmp,MMCs)
  tmp <- addProvinceCol(tmp,provinces)

  # Create a dateframe with the desired resulting columns
  party_df <- data.frame(Province = as.character(),
                         MMC = as.character(),
                         ND = as.character(),
                         Votes = as.integer(),
                         stringsAsFactors = FALSE)

  # Loop through multi-member constituencies and create sub-data-frames corresponding
  # to the data associated with each multi-member consituency
  for (j in 1:length(MMCs))
  {
    mmc_sliced = subset(tmp, MMC==MMCs[j])

    # Build data frame correspoding to MMC
    new_df <- data.frame(Province = mmc_sliced$Province,
                         MMC = mmc_sliced$MMC,
                         ND = mmc_sliced$Area,
                         Votes = mmc_sliced$Votes,
                         stringsAsFactors = FALSE)

    # Remove level of detail smaller than nomination district (based on lower case letters)
    new_df <- new_df[!str_detect(new_df$ND, "[a-z]"), ]
    new_df <- new_df[-1,]

    # Append newly created data frame for MMC to the data frame of entire party
    party_df <- bind_rows(party_df,new_df)
  }

  party_df$Party = Parties[i]
  # Move party column to first position for viewing convenience
  party_df <- party_df %>% select("Party", everything())
  # Concatenate data frame for party to data frame of all parties
  vote_distribution <- bind_rows(vote_distribution,party_df)
}

# An artifact of the above cleaning is that SJÆLLAND-SYDDANMARK and MIDTJYLLAND-NORDJYLLAND
# are still present in the ND column. They are manually removed with the following 4 lines.
# The same goes for a troublesome polling district DOKK1, which gets classified as an ND.
SS_idcs <- which(vote_distribution$ND == provinces[2])
MN_idcs <- which(vote_distribution$ND == provinces[3])
DOKK1_idcs <- which(vote_distribution$ND == "DOKK1")
vote_distribution <- vote_distribution[-c(SS_idcs,MN_idcs,DOKK1_idcs),] # Drops rows
```

Data format

The formatted data is now in the data frame vote_distribution. It follows a similar tree structure like the FV19TOTA dataset, however, it has been horizontalized, such that, for each A. Socialdemokratiet entry in the Party column, the columns to the right of it contain more specific information about the location, going from province (landsdel), to MMC = multi-member constituency (storkreds) to ND = nomination district (opstillingskreds). The last column, Votes, contains the votes for the corresponding nomination district.

```
head(vote_distribution)
```

```
##                  Party    Province                  MMC              ND Votes
## 1 A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS     1. ØSTERBRO  7660
## 2 A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS 2. SUNDBYVESTER  7215
## 3 A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS     3. INDRE BY  4243
## 4 A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS  4. SUNDBYØSTER  6608
## 5 A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS     5. NØRREBRO  5613
## 6 A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS   6. BISPEBJERG  4957
```

For example, if one wants to access the number of votes that Nye Borgerlige got in Roskilde, one could do the following

```
subset(vote_distribution,
       Party == "D. Nye Borgerlige" &
       ND == "8. ROSKILDE")
```

```
##                   Party             Province                 MMC          ND Votes
## 322 D. Nye Borgerlige SJÆLLAND-SYDDANMARK SJÆLLANDS STORKREDS 8. ROSKILDE  1116
```

and if one wanted to know the amount of votes that Stram Kurs got in the province Midtjylland-Nordjylland, one could type

```
sum(subset(vote_distribution,
           Party == "P. Stram Kurs" &
           Province == "MIDTJYLLAND-NORDJYLLAND")
           $Votes)
```

```
## [1] 18884
```

Seat allocation

Convenience functions

Convenience function for allocating seats in a multi-member constituency (storkreds) based on d'Hondt's method.

```
# a priority queue would be optimal for this, but it seems that an implementation using
# standard R results in O(nlogn) insertion time, which makes it no better than an ordered
# list. Data set size is small, so it does not matter much anyways.

# The below implementation is based on an unordered list philosophy that only updates the
# quotient for the party that gets a seat in an iteration.

d_Hondt <- function (p,v,s)
{
  allocated_seats <- c(1:length(p)) * 0 # Initialize a zero-vector of length number of
                                        # parties for storing allocated seats

  div_table <- data.frame("OV" = v, "V"  = v) # data frame for storing quotients
  div_table$div = 1

  for (i in 1:s)
  {
    # Find party which gets a seat allocated
    max_idx <- which(div_table$V == max(div_table$V))
    allocated_seats[max_idx] = allocated_seats[max_idx] + 1 # Allocate seat

    # Update divisor for party
    divisor <- div_table$div[max_idx] + 1
    div_table$div[max_idx] = divisor

    div_table$V[max_idx] = div_table$OV[max_idx] / divisor; # Assign new quotient to party
  }

  return(allocated_seats)
}
```

Convenience functions for checking if party qualifies for leveling seats (tillægsmandater)

```r
constituency_seat_qualification <- function(p,acs) # p = index of party of interest,
{                                                   # acs = allocated constituency seats

  return(acs[p] > 0) # Simply checks if party was allocated a constituency seat

}

two_thirds_over_mean_qualification <- function(P,pr,vd) # P = party of interest,
{                                                        # pr = provinces,
                                                         # vd = vote distribution data frame

  province_consituency_seats = c(39,50,46) # Constituency seats for the provinces
  count = 0
  for (i in 1:length(pr))
  {
    # Sum of votes for party in province
    party_votes = sum(subset(vd,
                             Party == P &
                             Province == pr[i])
                             $Votes)

    # sum of all votes in province
    province_votes = sum(subset(vd,
                                Province == pr[i])
                                $Votes)

    # Calculates mean votes per constituency seat in province
    mean_votes_per_seat = province_votes / province_consituency_seats[i]

    if (party_votes >= mean_votes_per_seat)
    {
      count = count + 1
    }
  }

  return(count >= 2) # Checks if party qualifies
}

percentage_qualification <- function(P,vd) # P = party of interest,
{                                           # vd = vote distribution data frame
  total_votes = sum(vd$Votes) # Sum of all votes
  party_votes = sum(subset(vd, Party == P)$Votes) # Sum of votes for the specific
                                                  # party of interest
  return( (party_votes / total_votes) > 0.02)
}
```

Allocation of seats for 2019 results

Allocating constituency seats (kredsmandater)

```r
# Creates vector of number of constituency seats for each MMC
constituency_seats <- c(16, 11, 10, 2, 20, 12, 18, 18, 13, 15)

# Vector containing the combined number of constituency seats for each party
allocated_constituency_seats <- c(1:length(Parties)) * 0

for (i in 1:length(MMCs))
{
  votes <- c()
  # Creates a vector of the number of votes each party got in the corresponding MMC
  for (j in 1:length(Parties))
  {
    votes <- c(votes, sum(subset(vote_distribution,
                                 Party == Parties[j] &
                                 MMC == MMCs[i])
                                 $Votes))
  }
  # Allocates seats based on d'Hondt's method
  MMC_seats <- d_Hondt(Parties, votes, constituency_seats[i])
  # Adds newly allocated seats to the total
  allocated_constituency_seats <-  allocated_constituency_seats + MMC_seats
}
```

Allocating leveling seats (tillægsmandater)

```r
# Find parties that qualify
qualified_parties <- c()

# Iterates through parties and adds the party to qualified_parties if one of the criteria
# is met
for (i in 1:length(Parties))
{
  # Checks if party got a constituency seat
  if (constituency_seat_qualification(i,allocated_constituency_seats))
  {
    qualified_parties <- c(qualified_parties, Parties[i])
    next
  }
  # Checks if party got at least as many votes as the mean number of votes given in a
  # province per constituency seat in 2 provinces.
  else if (two_thirds_over_mean_qualification(Parties[i],provinces,vote_distribution))
  {
    qualified_parties <- c(qualified_parties, Parties[i])
    next
  }
  # Checks if party got at least 2% of the votes nationwide
  else if (percentage_qualification(Parties[i],vote_distribution))
  {
    qualified_parties <- c(qualified_parties, Parties[i])
    next
  }
}

# Calculates "price per seat".
all_qualified_votes <- sum(subset(vote_distribution,
                                  Party %in% qualified_parties)
                                  $Votes)

seat_key <- all_qualified_votes / 175 # Since no nonpartisan politicians (løsgængere)
                                      # were elected, the divisor is 175.
allocated_leveling_seats <- c(1:length(Parties)) * 0
remainders <- c(1:length(Parties)) * 0

for (i in 1:length(Parties))
{
  if (Parties[i] %in% qualified_parties)
  {
    party_votes <- sum(subset(vote_distribution,
                              Party == Parties[i])
                              $Votes)

    # Finds propootional seat allocation and splits into integer part and remainder
    floored_q <- floor(party_votes / seat_key)
    remainders[i] <- party_votes / seat_key - floored_q

    # Allocates seats based on integer part of above and the already allocated seats
    party_leveling_seats <- floored_q - allocated_constituency_seats[i]
    allocated_leveling_seats[i] <- party_leveling_seats
  }
}

# Remainder of seats get allocated based on greatest remainder
leveling_seats <- c(11, 15, 14)
total_leveling_seats <- sum(leveling_seats)

# Finds how many seats are left to be allocated
seats_diff <- total_leveling_seats - sum(allocated_leveling_seats)

# Allocates seat to party with greatest remainder as long as there are seats left
while (seats_diff != 0)
{
  gr_idx <- which(remainders == max(remainders)) # Finds party with greatest remainder
  allocated_leveling_seats[gr_idx] = allocated_leveling_seats[gr_idx] + 1 # Allocates seat

  remainders[gr_idx] = 0
  seats_diff = seats_diff - 1
}
```

Final seat allocation (unaltered data)

The dataframe below summarizes the results of the seat allocation with unaltered data. It can be observed, the results match the current, official allocation of seats as expected.

```
allocated_seats <- data.frame("Party" = Parties,
                              "Constituency_seats" = allocated_constituency_seats,
                              "Leveling_seats" = allocated_leveling_seats,
                              "Total" = allocated_constituency_seats +
                                        allocated_leveling_seats)

allocated_seats
```

```
##                                  Party Constituency_seats Leveling_seats Total
## 1                  A. Socialdemokratiet                 44              4    48
## 2                   B. Radikale Venstre                 12              4    16
## 3         C. Det Konservative Folkeparti                  9              3    12
## 4                     D. Nye Borgerlige                  0              4     4
## 5               E. Klaus Riskær Pedersen                  0              0     0
## 6         F. SF - Socialistisk Folkeparti                 12              2    14
## 7                    I. Liberal Alliance                  0              4     4
## 8                 K. Kristendemokraterne                  0              0     0
## 9                   O. Dansk Folkeparti                 11              5    16
## 10                        P. Stram Kurs                  0              0     0
## 11 V. Venstre, Danmarks Liberale Parti                 39              4    43
## 12       Ø. Enhedslisten - De Rød-Grønne                  7              6    13
## 13                       Å. Alternativet                  1              4     5
## 14 Kandidater uden for partierne i alt                  0              0     0
```

Allocation of seats given Nørrebro, Østerbro, and Vesterbro did not vote

The process is identical to the above, except Nørrebro, Østerbro and Vesterbro are dropped from vote_distribution

```r
oesterbro_idcs <- which(vote_distribution$ND == "1. ØSTERBRO")
noerrebro_idcs <- which(vote_distribution$ND == "5. NØRREBRO")
vesterbro_idcs <- which(vote_distribution$ND == "9. VESTERBRO")

vote_distribution <- vote_distribution[-c(oesterbro_idcs,
                                          noerrebro_idcs,
                                          vesterbro_idcs),]
head(vote_distribution,n=7)
```

```
##                    Party    Province                 MMC              ND Votes
## 2  A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS 2. SUNDBYVESTER  7215
## 3  A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS     3. INDRE BY  4243
## 4  A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS  4. SUNDBYØSTER  6608
## 6  A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS   6. BISPEBJERG  4957
## 7  A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS     7. BRØNSHØJ  9577
## 8  A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS        8. VALBY  6491
## 10 A. Socialdemokratiet HOVEDSTADEN KØBENHAVNS STORKREDS    10. FALKONER  4824
```

Allocating constituency seats (kredsmandater)

```r
# Creates vector of number of constituency seats for each MMC
constituency_seats <- c(16, 11, 10, 2, 20, 12, 18, 18, 13, 15)

# Vector containing the combined number of constituency seats for each party
allocated_constituency_seats <- c(1:length(Parties)) * 0

for (i in 1:length(MMCs))
{
  votes <- c()
  # Creates a vector of the number of votes each party got in the corresponding MMC
  for (j in 1:length(Parties))
  {
    votes <- c(votes, sum(subset(vote_distribution,
                                 Party == Parties[j] &
                                 MMC == MMCs[i])
                                 $Votes))
  }
  # Allocates seats based on d'Hondt's method
  MMC_seats <- d_Hondt(Parties, votes, constituency_seats[i])
  # Adds newly allocated seats to the total
  allocated_constituency_seats <-  allocated_constituency_seats + MMC_seats
}
```

Allocating leveling seats (tillægsmandater)

```r
# Find parties that qualify
qualified_parties <- c()

# Iterates through parties and adds the party to qualified_parties if one of the criteria
# is met
for (i in 1:length(Parties))
{
  # Checks if party got a constituency seat
  if (constituency_seat_qualification(i,allocated_constituency_seats))
  {
    qualified_parties <- c(qualified_parties, Parties[i])
    next
  }
  # Checks if party got at least as many votes as the mean number of votes given in a
  # province per constituency seat in 2 provinces.
  else if (two_thirds_over_mean_qualification(Parties[i],provinces,vote_distribution))
  {
    qualified_parties <- c(qualified_parties, Parties[i])
    next
  }
  # Checks if party got at least 2% of the votes nationwide
  else if (percentage_qualification(Parties[i],vote_distribution))
  {
    qualified_parties <- c(qualified_parties, Parties[i])
    next
  }
}
# Calculates "price per seat".
all_qualified_votes <- sum(subset(vote_distribution,
                                  Party %in% qualified_parties)
                                  $Votes)

seat_key <- all_qualified_votes / 175 # Since no nonpartisan politicians (løsgængere)
                                      # were elected, the divisor is 175.
allocated_leveling_seats <- c(1:length(Parties)) * 0
remainders <- c(1:length(Parties)) * 0

for (i in 1:length(Parties))
{
  if (Parties[i] %in% qualified_parties)
  {
    party_votes <- sum(subset(vote_distribution,
                              Party == Parties[i])
                              $Votes)

    # Finds propootional seat allocation and splits into integer part and remainder
    floored_q <- floor(party_votes / seat_key)
    remainders[i] <- party_votes / seat_key - floored_q

    # Allocates seats based on integer part of above and the already allocated seats
    party_leveling_seats <- floored_q - allocated_constituency_seats[i]
    allocated_leveling_seats[i] <- party_leveling_seats
  }
}

# Remainder of seats get allocated based on greatest remainder
leveling_seats <- c(11, 15, 14)
total_leveling_seats <- sum(leveling_seats)

# Finds how many seats are left to be allocated
seats_diff <- total_leveling_seats - sum(allocated_leveling_seats)

# Allocates seat to party with greatest remainder as long as there are seats left
while (seats_diff != 0)
{
  gr_idx <- which(remainders == max(remainders)) # Finds party with greatest remainder
  allocated_leveling_seats[gr_idx] = allocated_leveling_seats[gr_idx] + 1 # Allocates seat

  remainders[gr_idx] = 0
  seats_diff = seats_diff - 1
}
```

Final seat allocation (without Nørrebro, Østerbro, and Vesterbro)

The dataframe below summarizes the results of the seat allocation after altering the data.

```r
allocated_seats_altered <- data.frame("Party" = Parties,
                                      "Constituency_seats" = allocated_constituency_seats,
                                      "Leveling_seats" = allocated_leveling_seats,
                                      "Total" = allocated_constituency_seats +
                                                allocated_leveling_seats)

allocated_seats_altered
```

```
##                                Party Constituency_seats Leveling_seats Total
## 1                A. Socialdemokratiet                 44              4    48
## 2                 B. Radikale Venstre                 12              3    15
## 3          C. Det Konservative Folkeparti              9              3    12
## 4                  D. Nye Borgerlige                  0              4     4
## 5               E. Klaus Riskær Pedersen              0              0     0
## 6        F. SF - Socialistisk Folkeparti             12              2    14
## 7                 I. Liberal Alliance                  0              4     4
## 8               K. Kristendemokraterne                 0              0     0
## 9                 O. Dansk Folkeparti                 12              5    17
## 10                      P. Stram Kurs                  0              0     0
## 11 V. Venstre, Danmarks Liberale Parti                39              5    44
## 12     Ø. Enhedslisten - De Rød-Grønne                 6              6    12
## 13                    Å. Alternativet                  1              4     5
## 14 Kandidater uden for partierne i alt                 0              0     0
```

The difference before and after alteration can be seen in the following dataframe. 4 Parties get a different number of seats. Radikale Venstre and Enhedslisten both lose a seat, while Dansk Folkeparti and Venstre each get an extra seat. This fits nicely with general understanding that left wing parties are more popular in the capital region, since removing a number of these votes swings the result slightly more in favour of the right wing.

```r
data.frame(Party = Parties,
           diff = allocated_seats_altered$Total - allocated_seats$Total)
```

```
##                                Party diff
## 1                A. Socialdemokratiet    0
## 2                 B. Radikale Venstre   -1
## 3          C. Det Konservative Folkeparti    0
## 4                  D. Nye Borgerlige    0
## 5               E. Klaus Riskær Pedersen    0
## 6        F. SF - Socialistisk Folkeparti    0
## 7                 I. Liberal Alliance    0
## 8               K. Kristendemokraterne    0
## 9                 O. Dansk Folkeparti    1
## 10                      P. Stram Kurs    0
## 11 V. Venstre, Danmarks Liberale Parti    1
## 12     Ø. Enhedslisten - De Rød-Grønne   -1
## 13                    Å. Alternativet    0
## 14 Kandidater uden for partierne i alt    0
```