HW1 Answer Key

Christian Baehr

2/10/2025

Load in required packages.

```
rm(list=ls())
pacman::p_load(quanteda, quanteda.corpora, quanteda.textstats, dplyr, ggplot2,
               readtext, stringr, gutenbergr, stylest2, text.alignment)
```

Question 1

```
load("hw1/unga_speech_corpus.RData")

doc1 <- which(txtstatesyears$year==2017 & txtstatesyears$country=="United States of America")[1]
doc2 <- which(txtstatesyears$year==2017 & txtstatesyears$country=="United Kingdom")[1]
doc3 <- which(txtstatesyears$year==2017 & txtstatesyears$country=="Australia")[1]
txtstatesyears <- txtstatesyears[c(doc1, doc2, doc3) , ]

txtstatesyears$country[which(txtstatesyears$country=="United States of America")] <- "US"
txtstatesyears$country[which(txtstatesyears$country=="United Kingdom")] <- "UK"

## convert to corpus, with "text" as the variable with text data
unga.corpus <- corpus(txtstatesyears, text_field = "text", docid_field = "country")

## tokenize the speeches (no pre processing yet)
unga.tokens <- tokens(unga.corpus)
```

1a)

```
## function to compute TTR
##
## @param x tokenized quanteda corpus
calculate_TTR <- function(x){
  ntype(x)/lengths(x)
}
calculate_TTR(unga.tokens)
```

```
##        US        UK Australia
## 0.5105263 0.5073529 0.3769883
```

```
## function to compute Guiraud's index of lexical richness
##
## @param x tokenized quanteda corpus
calculate_G <- function(x){
  ntype(x)/sqrt(lengths(x))
```

```r
}
calculate_G(unga.tokens)
```

```
##        US        UK Australia
##  7.037120  8.367479 16.371888
```

1b)

```r
## create dfm of the UNGA speeches
unga.dfm <- tokens(unga.corpus, remove_punct = T) |>
  dfm(tolower=F)
textstat_simil(unga.dfm, margin = "documents", method = "cosine")
```

```
## textstat_simil object; method = "cosine"
##              US    UK Australia
## US        1.000 0.766     0.714
## UK        0.766 1.000     0.804
## Australia 0.714 0.804     1.000
```

Question 2

2a) Calculating TTR & Similarity w/ Stemming

```r
## Processing
unga.tokens <- tokens(unga.corpus, remove_punct = T) |>
  tokens_wordstem()

## TTR
ttr <- calculate_TTR(unga.tokens) %>% setNames(c("USA", "UK", "France"))
r <- calculate_G(unga.tokens) %>% setNames(c("USA", "UK", "France"))

## Similarity
unga.dfm <- dfm(unga.tokens, tolower = F)
sim <- textstat_simil(unga.dfm, margin = "documents", method = "cosine")

## print results
cat("TTR scores w. stemming: \n", ttr, "\n\n")
```

```
## TTR scores w. stemming:
##  0.5146199 0.526971 0.3638498
```

```r
cat("G scores w. stemming: \n", r, "\n\n")
```

```
## G scores w. stemming:
##  6.729528 8.180789 15.01955
```

```r
cat("Cosine similarity w. stemming: \n"); prmatrix(as.matrix(sim))
```

```
## Cosine similarity w. stemming:
```

```
##                  US        UK Australia
## US        1.0000000 0.7607298 0.7041919
## UK        0.7607298 1.0000000 0.8010190
## Australia 0.7041919 0.8010190 1.0000000
```

2b) Calculating TTR & Similarity w/o Stopwords

```r
## Processing
unga.tokens <- tokens(unga.corpus, remove_punct = T) |>
  tokens_remove(stopwords("english"))

## TTR
ttr <- calculate_TTR(unga.tokens)
r <- calculate_G(unga.tokens) |>
  setNames(c("US", "UK", "France"))

## Similarity
unga.dfm <- dfm(unga.tokens, tolower = F)
sim <- textstat_simil(unga.dfm, margin = "documents", method = "cosine")

## print results
cat("TTR scores w/o stopwords: \n", ttr, "\n\n")
```

```
## TTR scores w/o stopwords:
##  0.7078652 0.7355372 0.6393782
```

```r
cat("G scores w/o stopwords: \n", r, "\n\n")
```

```
## G scores w/o stopwords:
##  6.677987 8.090909 19.86193
```

```r
cat("Cosine similarity w/o stopwords: \n"); prmatrix(as.matrix(sim))
```

```
## Cosine similarity w/o stopwords:
```

```
##                  US        UK Australia
## US        1.0000000 0.4275372 0.1195156
## UK        0.4275372 1.0000000 0.1524853
## Australia 0.1195156 0.1524853 1.0000000
```

2c) Calculating TTR & Similarity w/ all lowercase

```r
## Processing
unga.tokens <- tokens(unga.corpus, remove_punct = T) |>
  tokens_tolower()

## TTR
ttr <- calculate_TTR(unga.tokens)
r <- calculate_G(unga.tokens)

## Similarity
unga.dfm <- dfm(unga.tokens)
sim <- textstat_simil(unga.dfm, margin = "documents", method = "cosine")

# print results
cat("TTR scores w. lowercase: \n", ttr, "\n\n")
```

```
## TTR scores w. lowercase:
##  0.502924 0.526971 0.3890845
```

```r
cat("G scores w. lowercase: \n", r, "\n\n")
```

```
## G scores w. lowercase:
##  6.576584 8.180789 16.06123
```

```r
cat("Cosine similarity w. lowercases: \n"); prmatrix(as.matrix(sim))
```

```
## Cosine similarity w. lowercases:

##                  US        UK Australia
## US        1.0000000 0.7863459 0.7416247
## UK        0.7863459 1.0000000 0.8214111
## Australia 0.7416247 0.8214111 1.0000000
```

2d) TF-IDF

```r
unga.dfm.tfidf <- tokens(unga.corpus, remove_punct = T) |>
  dfm() |>
  dfm_tfidf()

textstat_simil(unga.dfm.tfidf, margin = "documents", method = "cosine")
```

```
## textstat_simil object; method = "cosine"
##               US     UK Australia
## US        1.0000 0.0862    0.0123
## UK        0.0862 1.0000    0.0474
## Australia 0.0123 0.0474    1.0000
```

Question 3

3a)

```r
## file names
files <- c("hw1/wealth_of_nations.txt", "hw1/theory_moral_sentiments.txt")

## read each text as a corpus object
smith <- readtext(files) |>
  corpus()

## docvar with titles
smith$title <- c("theory", "wealth")
```

3b)

```r
## remove symbols/punctuation/numbers/stopwords, lowercase and remove hyphens
smith.tok <- smith |>
  tokens(remove_symbols = T, remove_punct = T, remove_numbers = T, split_hyphens = T) |>
  tokens_tolower() |>
  tokens_remove(stopwords())
```

3c)

```r
## use tfidf weighting with numerator the proportion of
## document tokens of that type
smith.dfm <- dfm(smith.tok) |>
  dfm_tfidf(scheme_tf = "prop", base = exp(1))
```

```r
topfeatures(dfm_subset(smith.dfm, title=="theory"), 15) # pretty close!
```

```
##     sympathy          joy   sympathize       sorrow     sufferer        grief
## 0.0021620311 0.0008386060 0.0007730899 0.0007599867 0.0005503352 0.0005503352
##       breast     emotions     applause  substantive      demerit    judgments
## 0.0005372320 0.0005241288 0.0005241288 0.0004848191 0.0004586127 0.0003930966
## prepositions        verbs     external
## 0.0003930966 0.0003799934 0.0003668901
```

Question 4

```r
sentence1 <- "Trump's immigration crackdown sparks humanitarian crisis at the US border"
sentence2 <- "Trump's immigration reforms strengthen US national security and US economy"

## remove punctuation and tokenize
sentences.tokens <- corpus(c(sentence1, sentence2)) |>
  tokens(remove_punct = T)

sentences.dfm <- dfm(sentences.tokens, tolower = T)

s1 <- as.matrix(sentences.dfm)[1,] # feature vector for sentence1
s2 <- as.matrix(sentences.dfm)[2,] # feature vector for sentence2

## Euclidean distance
euclidean <- sqrt( sum( ( s1-s2 )^2 ) )

## Manhattan distance
manhattan <- sum( abs( s1-s2 ) )

## Jaccard distance
num <- length( intersect(sentences.tokens[[1]], sentences.tokens[[2]]) )
denom <- length( union(sentences.tokens[[1]],sentences.tokens[[2]]) )
jaccard <- num / denom

## Cosine similarity
cosine <- sum(s1 * s2) /( sqrt(sum(s1^2)) * sqrt(sum(s2^2)) )

## Levenshtein distance for surveyance and surveillance
text1 <- "At the theatre, my neighbour wore her favourite jewellery"
text2 <- "At the theater, my neighbor wore her favorite jewelry"
levenshtein <- adist(text1, text2)

dl <- stringdist::stringdist(text1, text2, method="dl")

## print
cat("Euclidean distance:", euclidean, "\n\n",
    "Manhattan distance:", manhattan, "\n\n",
    "Jaccard similarity:", jaccard, "\n\n",
    "Cosine similarity:", cosine, "\n\n",
    "Levenshtein distance:", levenshtein, "\n\n",
    "Damerau-Levenshtein distance:", dl)
```

```
## Euclidean distance: 3.741657
```

```
##
##  Manhattan distance: 14
##
##  Jaccard similarity: 0.1875
##
##  Cosine similarity: 0.3651484
##
##  Levenshtein distance: 6
##
##  Damerau-Levenshtein distance: 5
```

Question 5

5a) Contingency table for UK Manifestos

```r
## get text from UK political manifestos speeches
corpus <- corpus_subset(data_corpus_ukmanifestos)
text <- tokens(corpus, remove_punct = T) |>
  tokens_tolower() |>
  paste(collapse = " ")

## get entry of contingency table for the collocation
o11 <- str_count(text, "northern(?= ireland)")
o12 <- str_count(text, "northern(?! ireland)")
o21 <- str_count(text, "(?<!northern )ireland")
N <- tokens(text) |>
  tokens_ngrams(n = 2) |>
  ntoken() |>
  unname()
o22 <- N - o21 - o11 - o12

## contingency table
out <- matrix(c(o11, o12, o21, o22),
                 ncol = 2,
                 byrow = T)
rownames(out) <- c("Northern", "Not Northern")
colnames(out) <- c("Ireland", "Not Ireland")
print(out)
```

```
##              Ireland Not Ireland
## Northern         594          34
## Not Northern     431     1083371
```

```r
## expected frequency
E11 <- (o11+o12)/N * (o11 + o21)/N * N
# N12 <- N - (o11 + o21)
# E21 <- (o11+o21)/N * N21/N * N
# E12 <- (o11+o12)/N * N12/N * N
# E22 <- N12/N * N21/N * N

## get Chi-square value
## (o11-E11)^2/E11 + (o21-E21)^2/E21 + (o12-E12)^2/E12 + (o22-E22)^2/E22

## print
cat("Observed frequency:", o11, "\n\n",
```

```
    "Expected frequency:", E11)
```

```
## Observed frequency: 594
##
##  Expected frequency: 0.5935837
```

5b) Collocation for "Northern Ireland" using quanteda

```
textstat_collocations(corpus, min_count = 5) |>
  data.frame() |>
  select(c("collocation", "lambda", "z")) |>
  filter(collocation == "northern ireland")
```

```
##         collocation   lambda        z
## 36 northern ireland 10.56993 69.40865
```

5c) Collocations using quanteda

```
(collout1 <- textstat_collocations(corpus, min_count = 5) |>
   arrange(-lambda) |>
   slice(1:10) |>
   data.frame() |>
   select(c("collocation", "count", "lambda", "z")))
```

```
##           collocation count   lambda         z
## 12246    gymru newydd    19 18.15165  9.018201
## 12470     PLAID CYMRU    15 17.92208  8.889637
## 9576   05-apr-2001 00    33 17.59416 10.714374
## 13232          rt hon     7 17.19616  8.458261
## 13233    sierra leone     7 17.19616  8.458261
## 13421     veal crates     6 17.05306  8.367138
## 10108 adeiladwn gymru    19 17.05304 10.343824
## 13627          ad hoc     5 16.88600  8.257417
## 13628       bona fide     5 16.88600  8.257417
## 13629     magna carta     5 16.88600  8.257417
```

```
(collout2 <- textstat_collocations(corpus, min_count = 5) |>
   arrange(-count) |>
   slice(1:10) |>
   data.frame() |>
   select(c("collocation", "count", "lambda", "z")))
```

```
##      collocation count    lambda         z
## 5         of the  7368 1.6319809 117.11299
## 1        we will  6042 4.3460648 217.11342
## 10        in the  4876 1.7995549 105.12821
## 1099      to the  2985 0.5036708  25.59629
## 4173     and the  2647 0.3311830  15.99929
## 56       for the  2593 1.3769813  62.10876
## 2        will be  2436 3.5747854 138.16377
## 46        on the  1806 1.8120498  65.53095
## 8        we have  1677 3.2692852 108.77172
## 4          it is  1448 4.1497773 121.39666
```

Question 6

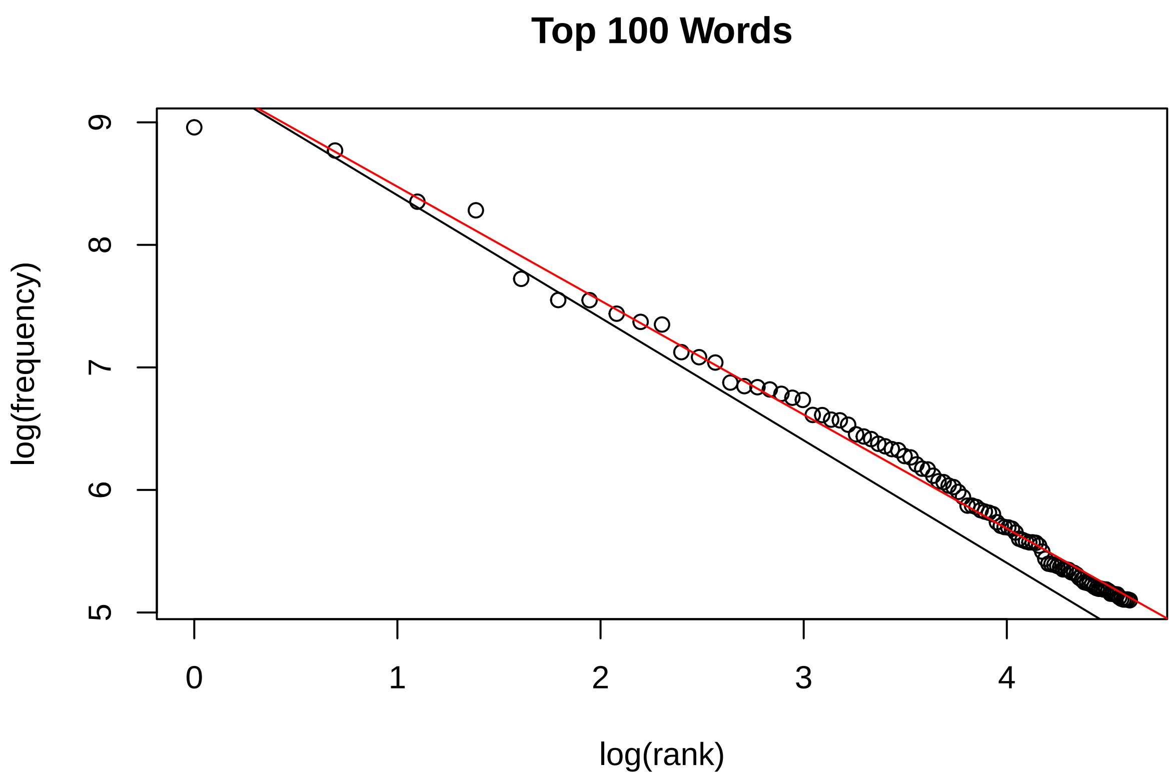
```r
dfm <- smith |>
  corpus_subset(title=="theory") |>
  tokens(remove_punct = T) |>
  dfm()

## regression to check if slope is approx -1.0
regression <- lm(log(topfeatures(dfm, 100)) ~ log(1:100))
summary(regression)
```

```
##
## Call:
## lm(formula = log(topfeatures(dfm, 100)) ~ log(1:100))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44581 -0.05555 -0.00767  0.07384  0.16639
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.404740   0.035457  265.24   <2e-16 ***
## log(1:100)  -0.929926   0.009448  -98.42   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08724 on 98 degrees of freedom
## Multiple R-squared:   0.99,  Adjusted R-squared:  0.9899
## F-statistic:  9687 on 1 and 98 DF,  p-value: < 2.2e-16
```

```r
confint(regression)
```

```
##                  2.5 %     97.5 %
## (Intercept)  9.3343774  9.4751034
## log(1:100)  -0.9486754 -0.9111762
```

```r
# create plot to illustrate zipf's law
plot(log(1:100), log(topfeatures(dfm, 100)),
     xlab="log(rank)", ylab="log(frequency)", main="Top 100 Words")
abline(regression, col="red")
abline(a = regression$coefficients["(Intercept)"], b = -1, col = "black")
```



Question 7

```r
## Heap's Law
## M = kT^b
## where:
## M = vocab size
## T = number of tokens
## k, b are constants

dfm <- smith |>
  corpus_subset(title=="theory") |>
  tokens(remove_punct = T) |>
  dfm()

num_tokens <- sum(rowSums(dfm))
M <- nfeat(dfm)
k <- 44

## solve for b
b <- log(M/k)/log(num_tokens)
print(b)
```

```
## [1] 0.4330382
```

```r
## Now without lowercase

dfm <- smith |>
  corpus_subset(title=="theory") |>
  tokens(remove_punct = T) |>
```

```r
  dfm(tolower=F)

num_tokens <- sum(rowSums(dfm))
M <- nfeat(dfm)
k <- 44

## solve for b
b <- log(M/k)/log(num_tokens)
print(b)
```

```
## [1] 0.4373448
```

Question 8

```r
corpus <- txtstatesyears |>
  filter(country %in% c("United States of America", "China")) |>
  corpus(text_field="text")

## key words in context
corpus_subset(corpus, country == "United States of America") |>
  tokens(remove_punct = T) |>
  kwic("nation", window = 5)
corpus_subset(corpus, country == "United States of America") |>
  tokens(remove_punct = T) |>
  kwic("industry", window = 5)

corpus_subset(corpus, country == "China") |>
  tokens(remove_punct = T) |>
  kwic("nation", window = 5)
corpus_subset(corpus, country == "China") |>
  tokens(remove_punct = T) |>
  kwic("industry", window = 5)
```

Question 9

9a)

```r
load("hw1/unga_speech_corpus.RData")

unga.df.sub <- txtstatesyears |>
  filter(country=="United States of America" & year %in% c(2005:2015))
unga.sub <- corpus(unga.df.sub, text_field="text")
docvars(unga.sub, "year") <- unga.df.sub$year

unga.sub <- unga.sub |>
  corpus_reshape("sentence")

unga.df <- cbind(as.character(unga.sub), docvars(unga.sub)["year"]) |>
  setNames(c("text", "year"))

unga.split <- split(unga.df, as.factor(unga.df$year))
```

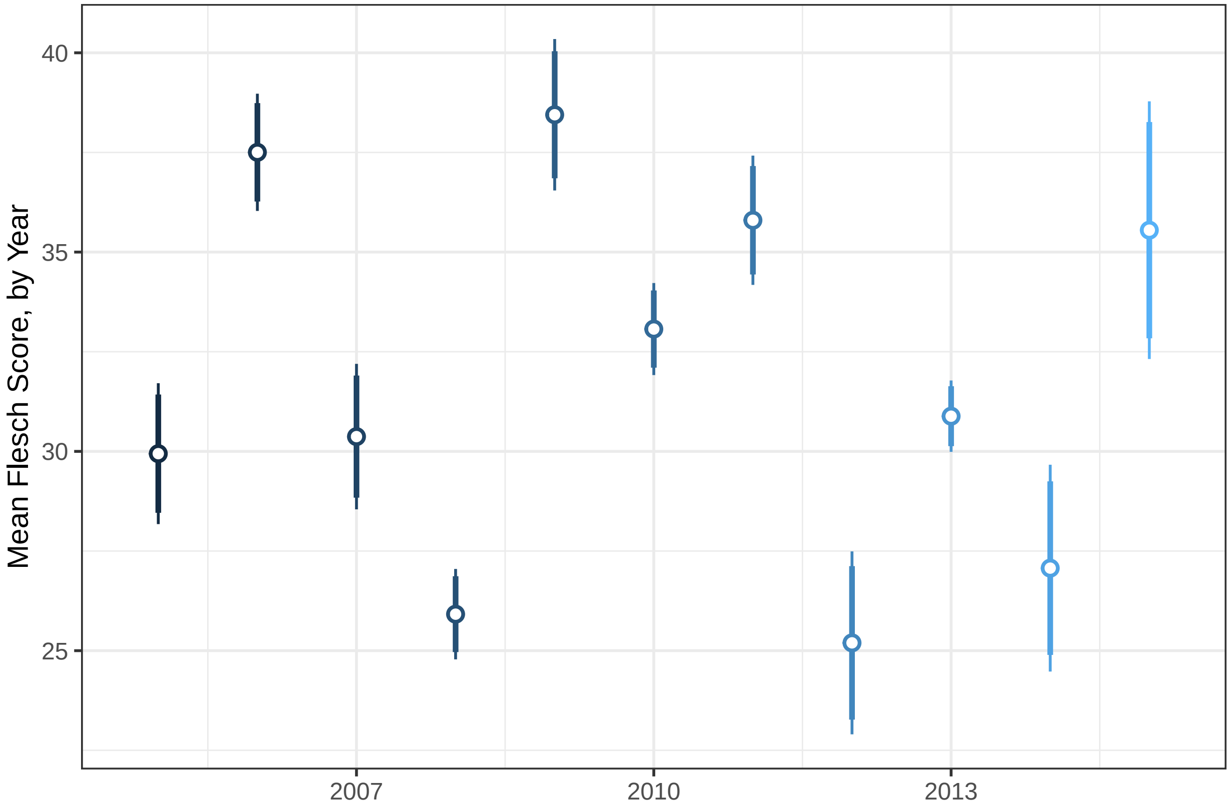
```r
boot.fre <- function(year) { # accepts df of texts (year-specific)
  n <- nrow(year) # number of texts
  docnums <- sample(1:n, size=n, replace=T) # sample texts WITH replacement
  docs.boot <- corpus(year[docnums, "text"])
  docnames(docs.boot) <- 1:length(docs.boot) # something you have to do
  fre <- textstat_readability(docs.boot, measure = "Flesch") # compute FRE for each
  return(mean(fre[,"Flesch"])) # return flesch scores only
}

lapply(unga.split, boot.fre) # apply to each df of party texts
```

```
## $`2005`
## [1] 29.37264
##
## $`2006`
## [1] 36.90584
##
## $`2007`
## [1] 30.96104
##
## $`2008`
## [1] 25.43996
##
## $`2009`
## [1] 38.10053
##
## $`2010`
## [1] 33.31522
##
## $`2011`
## [1] 35.43833
##
## $`2012`
## [1] 25.81929
##
## $`2013`
## [1] 30.61103
##
## $`2014`
## [1] 27.61242
##
## $`2015`
## [1] 32.60372
```

```r
iter <- 10 # NUMBER OF BOOTSTRAP SAMPLES (usually would want more, >=100)

## for loop to compute as many samples as specified
for(i in 1:iter) {
  if(i==1) {boot.means <- list()} # generate new list

  # store the results in new element i
  boot.means[[i]] <- lapply(unga.split, boot.fre)
  print(paste("Iteration", i))
}
```

```
## [1] "Iteration 1"
## [1] "Iteration 2"
## [1] "Iteration 3"
## [1] "Iteration 4"
## [1] "Iteration 5"
## [1] "Iteration 6"
## [1] "Iteration 7"
## [1] "Iteration 8"
## [1] "Iteration 9"
## [1] "Iteration 10"
```

```
## combine the point estimates to a data frame and compute statistics by party
boot.means.df <- do.call(rbind.data.frame, boot.means)
mean.boot <- apply(boot.means.df, 2, mean)
sd.boot <- apply(boot.means.df, 2, sd)

## create data frame for plot
plot_df <- data.frame(sort(unique(unga.df$year)), mean.boot, sd.boot) |>
  setNames(c("year", "mean", "se"))

## confidence intervals
ci90 <- qnorm(0.95)
ci95 <- qnorm(0.975)

## ggplot point estimate + variance
ggplot(plot_df, aes(colour = year)) + # general setup for plot
  geom_linerange(aes(x = year,
                     ymin = mean - se*ci90,
                     ymax = mean + se*ci90),
                 lwd = 1, position = position_dodge(width = 1/2)) + # plot 90% interval
  geom_pointrange(aes(x = year,
                      y = mean,
                      ymin = mean - se*ci95,
                      ymax = mean + se*ci95),
                  lwd = 1/2, position = position_dodge(width = 1/2),
                  shape = 21, fill = "WHITE") + # plot point estimates and 95% interval
  #coord_flip() + # fancy stuff
  theme_bw() + # fancy stuff
  xlab("") + ylab("Mean Flesch Score, by Year") + # fancy stuff
  theme(legend.position = "none") # fancy stuff
```



9b)

```r
## mean Flesch statistic by year
flesch_point <- unga.df$text |>
  textstat_readability(measure = "Flesch") |>
  group_by(unga.df$year) |>
  summarise(mean_flesch = mean(Flesch)) |>
  setNames(c("year", "mean")) |>
  arrange(as.numeric(year))

cbind(flesch_point, "bs_mean" = plot_df$mean)
```

```
##    year     mean  bs_mean
## 1  2005 30.11911 29.94279
## 2  2006 37.35916 37.50301
## 3  2007 29.98124 30.37059
## 4  2008 26.34213 25.91582
## 5  2009 38.01141 38.44575
## 6  2010 32.97639 33.06885
## 7  2011 35.99020 35.79871
## 8  2012 25.05206 25.19546
## 9  2013 31.05528 30.88282
## 10 2014 26.93190 27.07043
## 11 2015 34.88778 35.55001
```

9c)

```r
## calculate the FRE score and the Dale-Chall score.
fre_and_dc_measures <- textstat_readability(unga.sub, c("Flesch", "FOG"))

## compute correlations
readability_cor <- cor(cbind(fre_and_dc_measures$Flesch, fre_and_dc_measures$FOG))

## print
print(readability_cor[1,2])
```

```
## [1] -0.8870674
```

Question 10

```r
docs <- corpus( readtext(c("hw1/data/jefferson_draft.txt", "hw1/data/final_version.txt")))

# set gap to default (-1)
sw2 <- smith_waterman(as.character(docs)[1], as.character(docs)[2],
                      type="words", gap=-1)

# increase match, decrease mismatch --> increases extent of plagiarism. Why?
sw3 <- smith_waterman(as.character(docs)[1], as.character(docs)[2],
                      type="words",
                      match= 3,
                      mismatch = 0,
                      gap=-1)

print(sw2$sw)
```

```
## [1] 1246
```

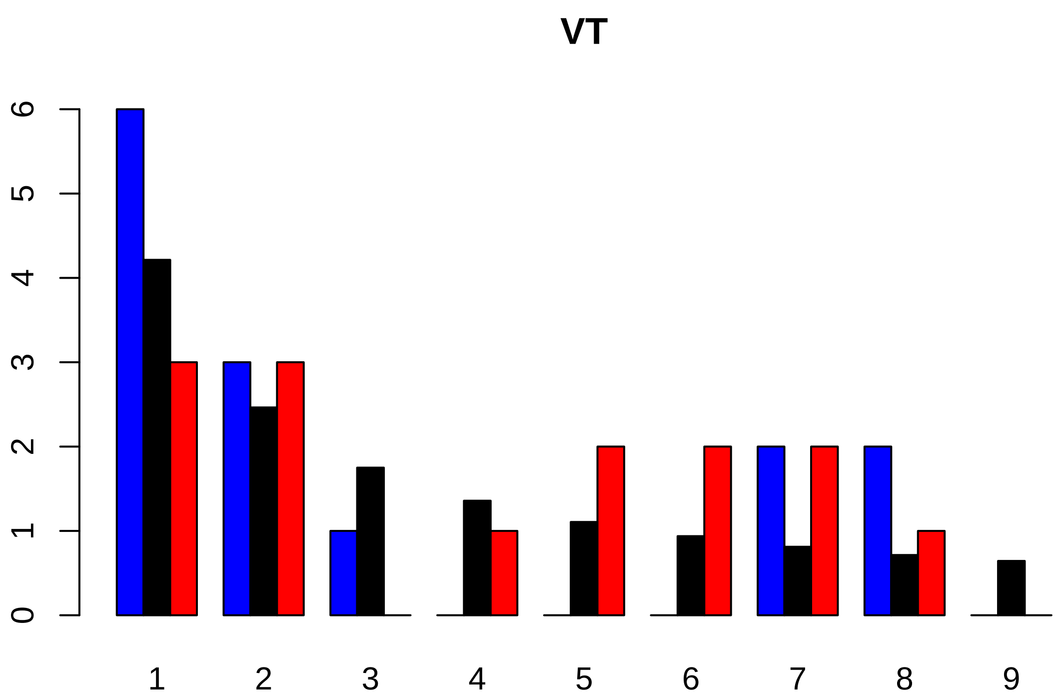
```r
print(sw3$sw)
```

```
## [1] 2310
```

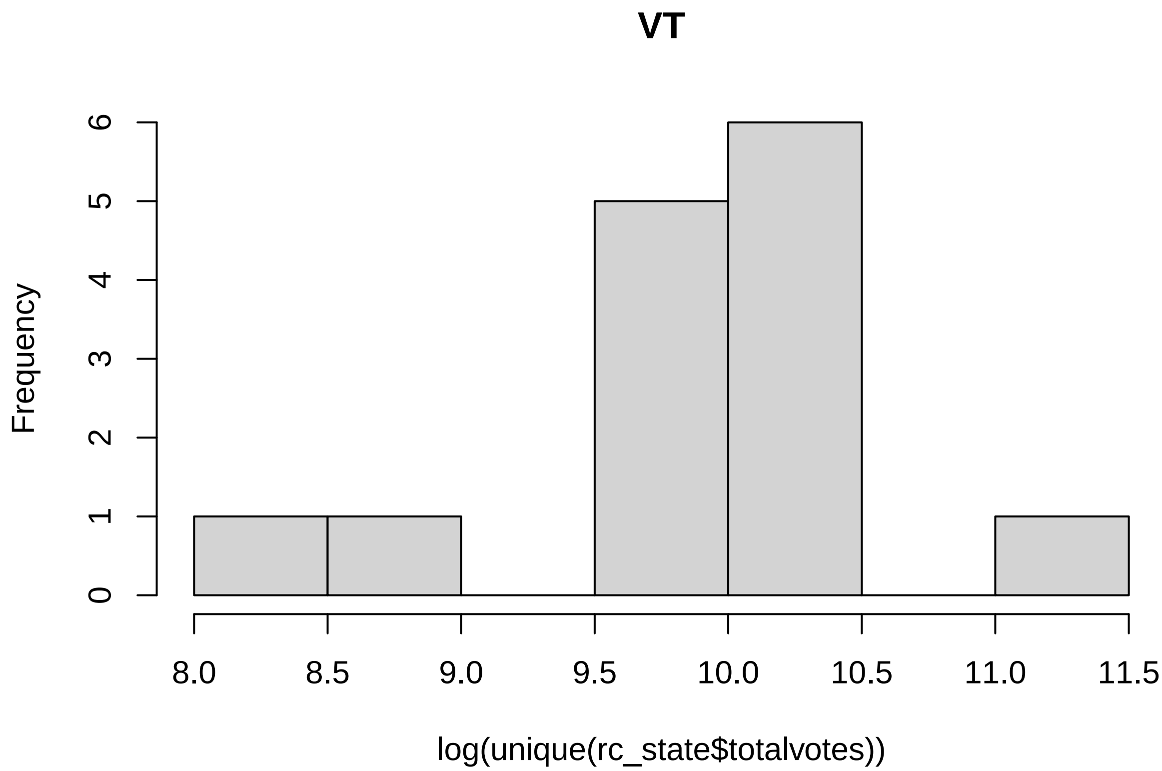
Question 11

```r
rc <- read.csv("hw1/countypres_2000-2020.csv")
rc20 <- rc[rc$year=="2020",]
rc20DR <- rc20[rc20$party%in%c("REPUBLICAN", "DEMOCRAT"),]

# look at specific state
# TX works

state <- "VT"
rc_state <- rc20DR[rc20DR$state_po==state,]

#look at leading digit v frequency
digits <- rc_state$candidatevotes
first_digits <-  as.numeric(substr(digits, 1, 1))

rc_state$first_digs <- first_digits
```

```r
dems <- rc_state[rc_state$party=="DEMOCRAT",]
total_dems <- table(factor(dems$first_digs, levels=1:9) )

rep <-  rc_state[rc_state$party=="REPUBLICAN",]
total_reps <- table(factor(rep$first_digs, levels=1:9) )

#benford expectations
ben_props  <- c(0.301, .176, .125, .097, 0.079, 0.067, 0.058, 0.051, 0.046)
names(ben_props)<- c("1" ,"2", "3" ,"4" ,"5", "6", "7", "8", "9")

the_benprops <- ben_props*length(unique(rc_state$county_name))

# now do barplot
dat <- rbind(total_dems, the_benprops, total_reps)
rownames(dat) <- c("dem", "benford", "rep")
colnames(dat) <- c("1" ,"2", "3" ,"4" ,"5", "6", "7", "8", "9")

#x11()
barplot(dat, col = c("blue","black","red"), beside = T, main=state)
```




```r
# article:
# https://www.reuters.com/article/world/fact-check-deviation-from-benfords-law-does-not-prove-election-
# Election Integrity Partnership

# look at magnitude of districts
#x11()
hist(log(unique(rc_state$totalvotes)), main=state) # TX has broad range of magnitudes, VT does not
```


VT
Frequency
0
1
2
3
4
5
6
8.0
8.5
9.0
9.5
10.0
10.5
11.0
11.5
log(unique(rc_state$totalvotes))