Assignment 3

PLSC 21510/31510

2022

Assigned: April 21, 2022 Due: May 4, 2022

```
# quanteda stuff
require(quanteda)
require(quanteda.textstats)
require(quanteda.corpora)

# other stuff
require(tidyverse)
require(ggplot2)
require(lubridate)
```

In this assignment, we'll be working with the `data_corpus_sotu` data that come with `quanteda.corpora`.

```
# read about the corpus
?data_corpus_sotu

# save as `corp`
corp <- data_corpus_sotu
corp <- corpus(data_corpus_sotu, text_field = "text")
```

```
## Warning: text_field argument is not used.
```

```
head(docvars(corp))
```

```
##   FirstName  President       Date delivery type       party
## 1    George Washington 1790-01-08   spoken SOTU Independent
## 2    George Washington 1790-12-08   spoken SOTU Independent
## 3    George Washington 1791-10-25   spoken SOTU Independent
## 4    George Washington 1792-11-06   spoken SOTU Independent
## 5    George Washington 1793-12-03   spoken SOTU Independent
## 6    George Washington 1794-11-19   spoken SOTU Independent
```

1. Working with corpora

1.1

Which SOTU speech was the longest, in terms of number of tokens? A: the longest speech is by President Carter given in 1981 with 36905 tokens

```
num_tokens <- ntoken(corp)
df <- data.frame("num_tokens" = num_tokens,

                                "presidnet" = corp$President,
                                "date" = corp$Date) %>%
  arrange(-num_tokens)

head(df)
```

```
##                num_tokens presidnet       date
## Carter-1981         36905    Carter 1981-01-16
## Carter-1980b        36543    Carter 1980-01-21
## Truman-1946         30916    Truman 1946-01-21
## Taft-1910           30003      Taft 1910-12-06
## Roosevelt-1907      29735 Roosevelt 1907-12-03
## Roosevelt-1905      27110 Roosevelt 1905-12-05
```

1.2

Add a document-level variable (i.e. `docvars`) called `Year` that contains the year the address was delivered.

Hint: `Lubridate` is your friend.

```
corp$Year <- year(corp$Date)
head(docvars(corp))
```

```
##   FirstName  President       Date delivery type       party Year
## 1    George Washington 1790-01-08   spoken SOTU Independent 1790
## 2    George Washington 1790-12-08   spoken SOTU Independent 1790
## 3    George Washington 1791-10-25   spoken SOTU Independent 1791
## 4    George Washington 1792-11-06   spoken SOTU Independent 1792
## 5    George Washington 1793-12-03   spoken SOTU Independent 1793
## 6    George Washington 1794-11-19   spoken SOTU Independent 1794
```

1.3

Reshape the corpus such that each document is a paragraph. How many documents do we have?

```
corp_para <- corpus_reshape(corp, to = "paragraphs")
ndoc(corp_para)
```

```
## [1] 23884
```

1.4

Subset the corpus to include only speeches delivered after 2000. What *proportion* of the total documents are we left with? proportion of total documents = ndoc(corp2000)/ndoc(corp) = 20/23884 = 0.00083738

```
corp2000 <- corpus_subset(corp, Year > 2000 )
ndoc(corp2000)
```

```
## [1] 20
```

1.5

Create a DFM of the original SOTU corpus using the following preprocessing steps: 1) remove punctuation, 2) remove numbers, and 3) remove capitalization. Call it `dfm_sotu`. How many features (aka types) and tokens do we have?

```
token_sotu <- tokens(corp, split_hyphens = TRUE,
                remove_punct = TRUE,
                remove_numbers = TRUE
                ) %>%
  tokens_tolower(keep_acronyms = FALSE)
dfm_sotu <- dfm(token_sotu)
head(dfm_sotu)
```

```
## Document-feature matrix of: 6 documents, 24,913 features (97.07% sparse) and 7 docvars.
##                    features
## docs                fellow citizens  of the senate and house representatives  i
##   Washington-1790       2        3  69  97      2  41     3                3 11
##   Washington-1790b      3        5  89 122      2  45     3                3  8
##   Washington-1791       1        3 159 242      3  73     3                3  6
##   Washington-1792       1        7 139 195      2  56     3                3 21
##   Washington-1793       2        4 132 180      2  49     3                3 12
##   Washington-1794       3        9 187 273      2  86     4                3 18
##                    features
## docs                embrace
##   Washington-1790         1
##   Washington-1790b        0
##   Washington-1791         0
##   Washington-1792         0
##   Washington-1793         0
##   Washington-1794         0
## [ reached max_nfeat ... 24,903 more features ]
```

```
nfeat(dfm_sotu)
```

```
## [1] 24913
```

```
length(dfm_sotu)
```

```
## [1] 6004033
```

Heap's Law

Heap's Law tells us the relationship between number of tokens and number of types. As a reminder, Heap's law is $P = kT^b$, logp=b(log k+log t)where

- `P` = vocab size (num of types)
- `T` = number of tokens
- `k`, `b` are constants. In English,
  - 30 <= k <= 100
  - 0.4 <= b <= 0.6

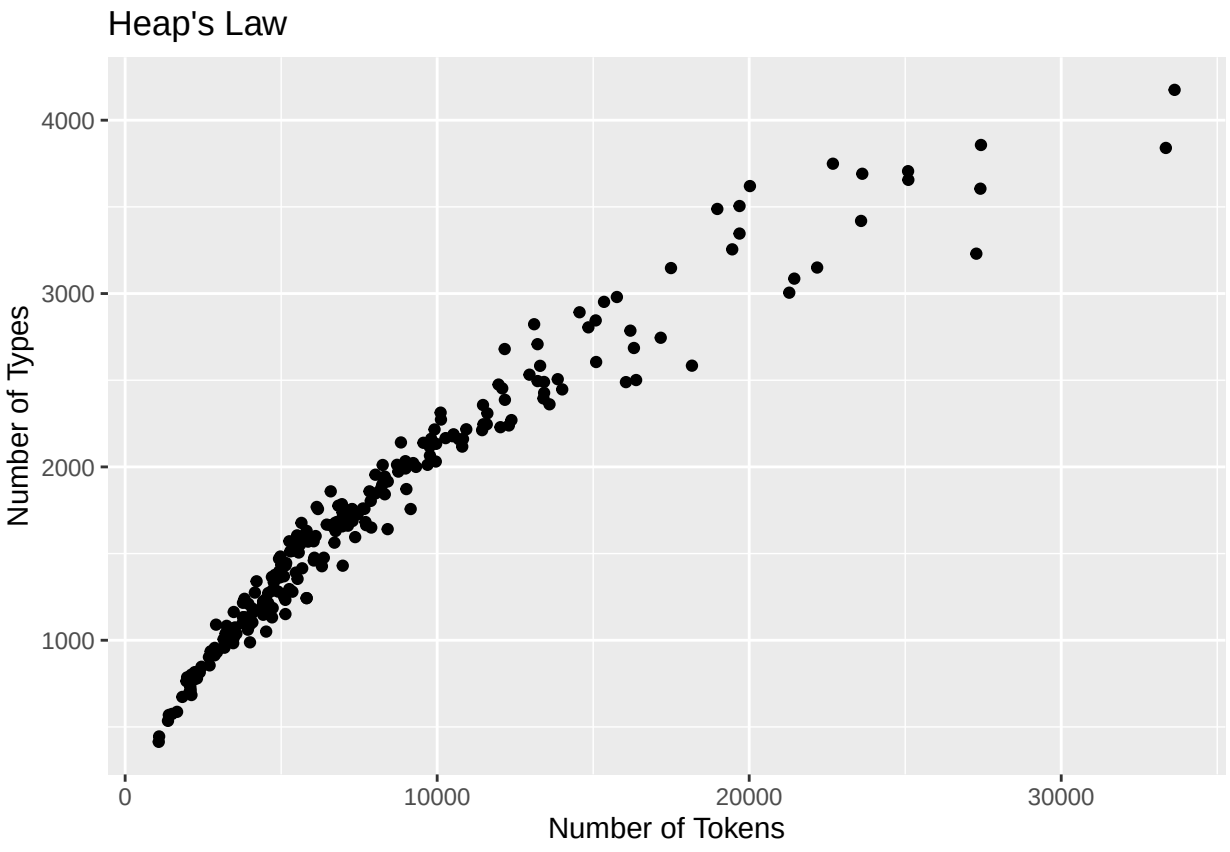
2.1

Using `dfm_sotu`, plot Heap's law. - X-axis: number of tokens in a document, - Y-axis: number of types in a document.

hint: check out `ntypes()` & `ntokens()`

tip: if you're not familiar with `ggplot`, read these materials first.

```
n_tokens <- ntoken(dfm_sotu)
n_types <- ntype(dfm_sotu)
heap_df <- convert(dfm_sotu, "data.frame")
first_plot <- heap_df %>%
  ggplot(aes(x = n_tokens, y = n_types)) +
  geom_point() +
xlab("Number of Tokens") +
  ylab("Number of Types") +
  ggtitle("Heap's Law")
first_plot
```



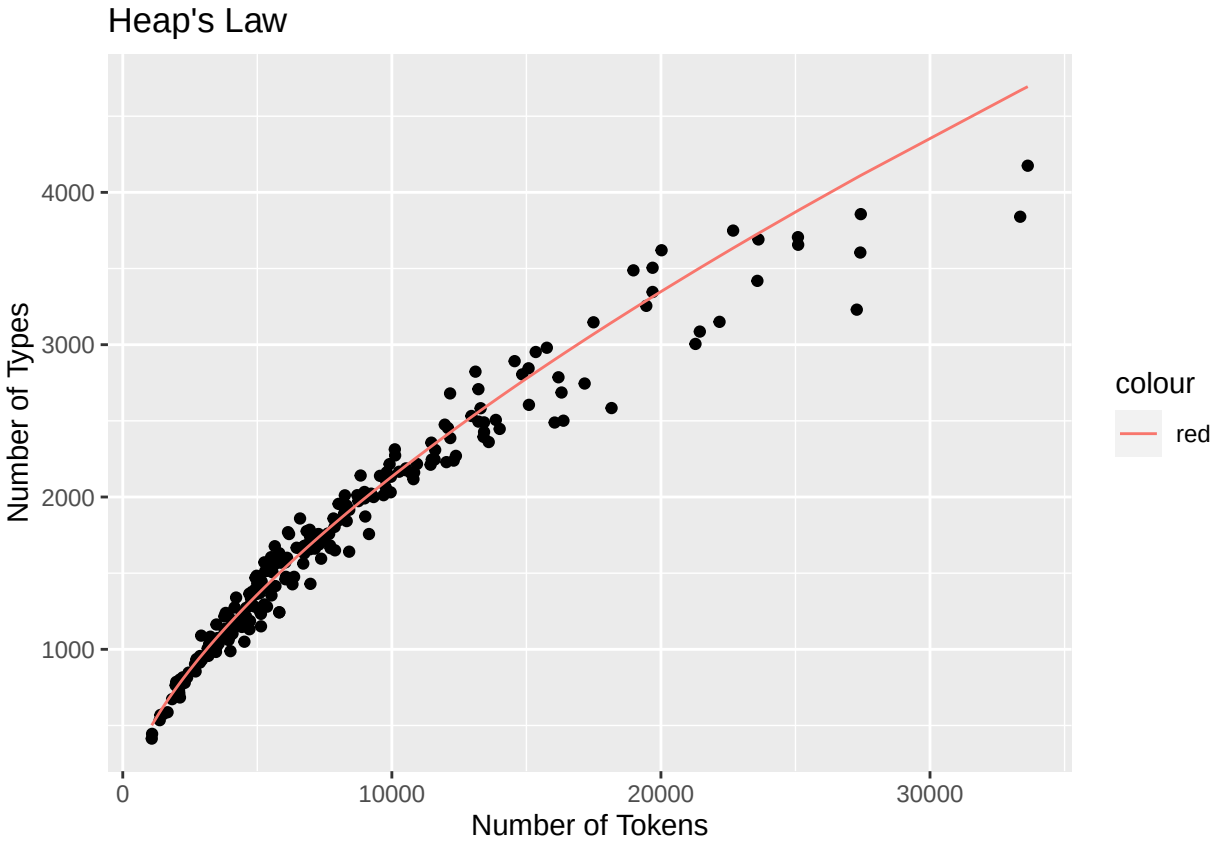
2.2

Add a layer to the plot above with red points showing the *expected* number of types as a function of the number of tokens. Use the following parameters:

k = 5.36 b = 0.65 - At first the expected line layer was identical to the token-type layer but they eventually drafted

```
n_tokens <- ntoken(dfm_sotu)
n_types <- ntype(dfm_sotu)
heap_df <- convert(dfm_sotu, "data.frame")
my_formula <- 5.36*(n_tokens^0.65)

ggplot(data = heap_df, aes(x = n_tokens, y = n_types)) +
  geom_point() +
  geom_line(data = heap_df, aes(x = n_tokens, y = my_formula, color = "red")) +
xlab("Number of Tokens") +
  ylab("Number of Types") +
  ggtitle("Heap's Law")
```



2.3

Replicate the plot you made above, but this time remove stop words. What do you notice?

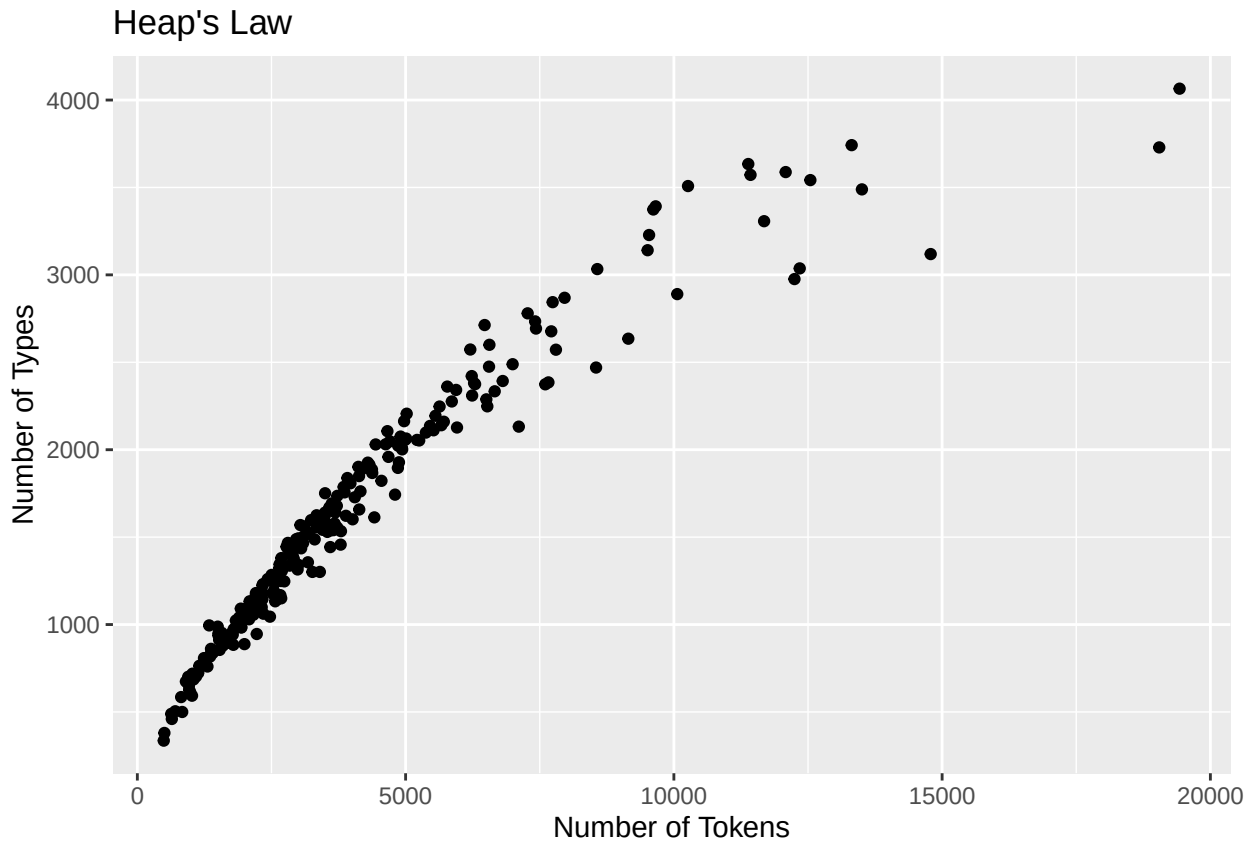
A: The relationship between the type-token did not change (it is still linear) however, some points have been removed which means that the gap between the number of types and tokens is smaller than before (mainly because tokens had a lot of stop words repetition unlike types) also the graph looks more stable

```
token_sotu_stop_words <- tokens(corp, split_hyphens = TRUE,
                remove_punct = TRUE,
                remove_numbers = TRUE

                ) %>%
```

```
  tokens_tolower(keep_acronyms = FALSE)%>%
  tokens_remove(pattern = stopwords("en"))
dfm_sotu_stop_words <- dfm(token_sotu_stop_words)

n_sw_tokens <- ntoken(dfm_sotu_stop_words)
n_sw_types <- ntype(dfm_sotu_stop_words)
heap_df <- convert(dfm_sotu_stop_words, "data.frame")
heap_df %>%
  ggplot(aes(x = n_sw_tokens, y = n_sw_types)) +
  geom_point() +
xlab("Number of Tokens") +
  ylab("Number of Types") +
  ggtitle("Heap's Law")
```



2.4

In Hipf's law, the number of types grows rapidly as the number of tokens increases, but eventually levels out. In your own words (formulas/equations not necessary) explain why this is so.

Hint: See slide 31 of the "Describing Texts" Slides on the Type-To-Token Ratio.

Answer Because the relationship between the type-token is subjected to the length of the document. Since the types only includes unique words which keeps growing until it covers all the unique words in a document while the types keep increasing because it covers all the words in the document not only the unique ones.

BONUS (Extra Credit)

We can estimate `k` and `b` by fitting a linear model, because the relationship between `log(P)` and `log(T)` is linear. Estimate `k` and `b` for the corpus that removes stop words.

```
# YOUR CODE HERE
```

Zipf's law

Zipf's Law tells us about the rank-frequency distribution.

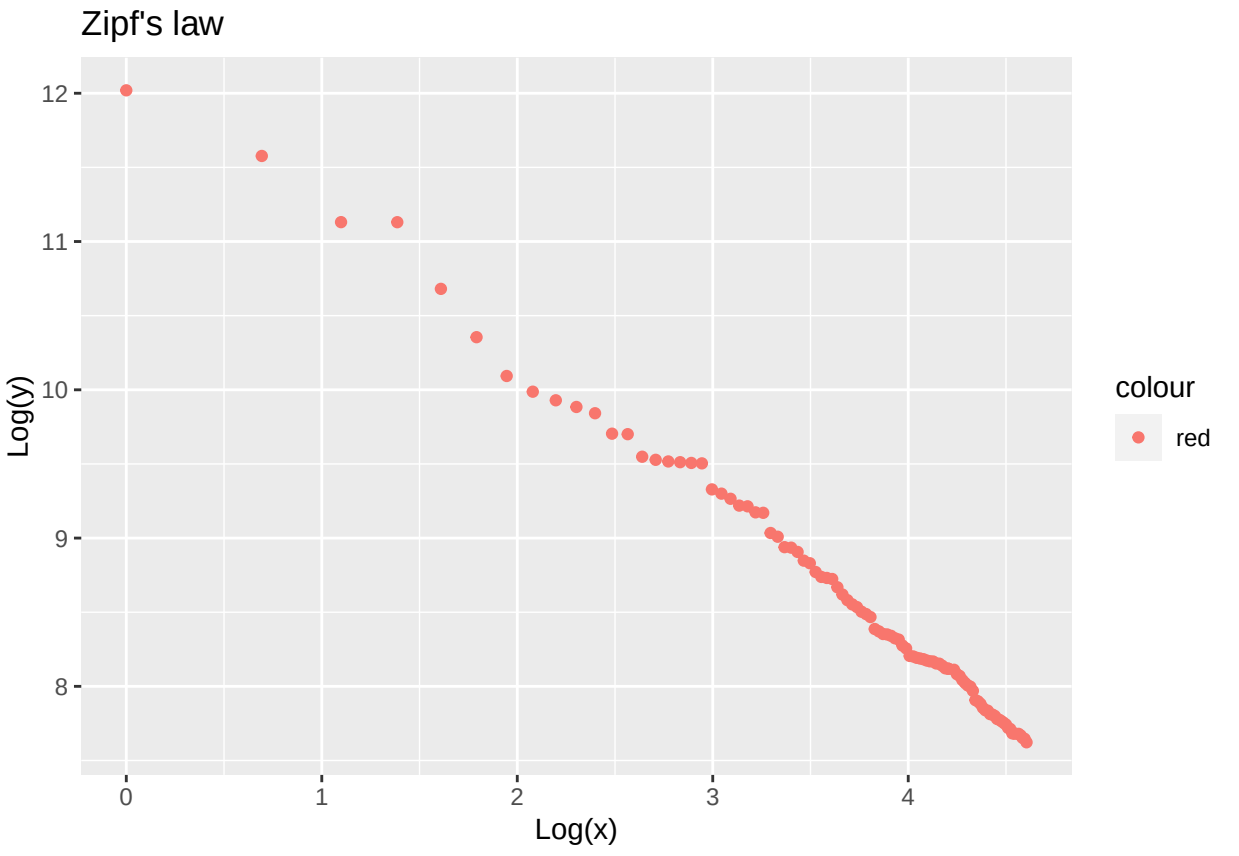
3.1

Using `dfm_sotu`, plot Zipf's law. - X-axis: log of ranks 1 through 100 - Y-axis log of frequency of top 100 terms from the DFM

Hint: `topfeaures()` A: I used textstat_frequency instead because it included the ranks

```
logy <- textstat_frequency(dfm_sotu, n = 100)
zip_df <- data.frame(logy)
plot_data <- data.frame( x=log(zip_df$rank), y=log(zip_df$frequency))

ggplot(plot_data, aes(x=x, y=y, color = "red")) +
  geom_point()+
  labs(title="Zipf's law", x='Log(x)', y='Log(y)')
```

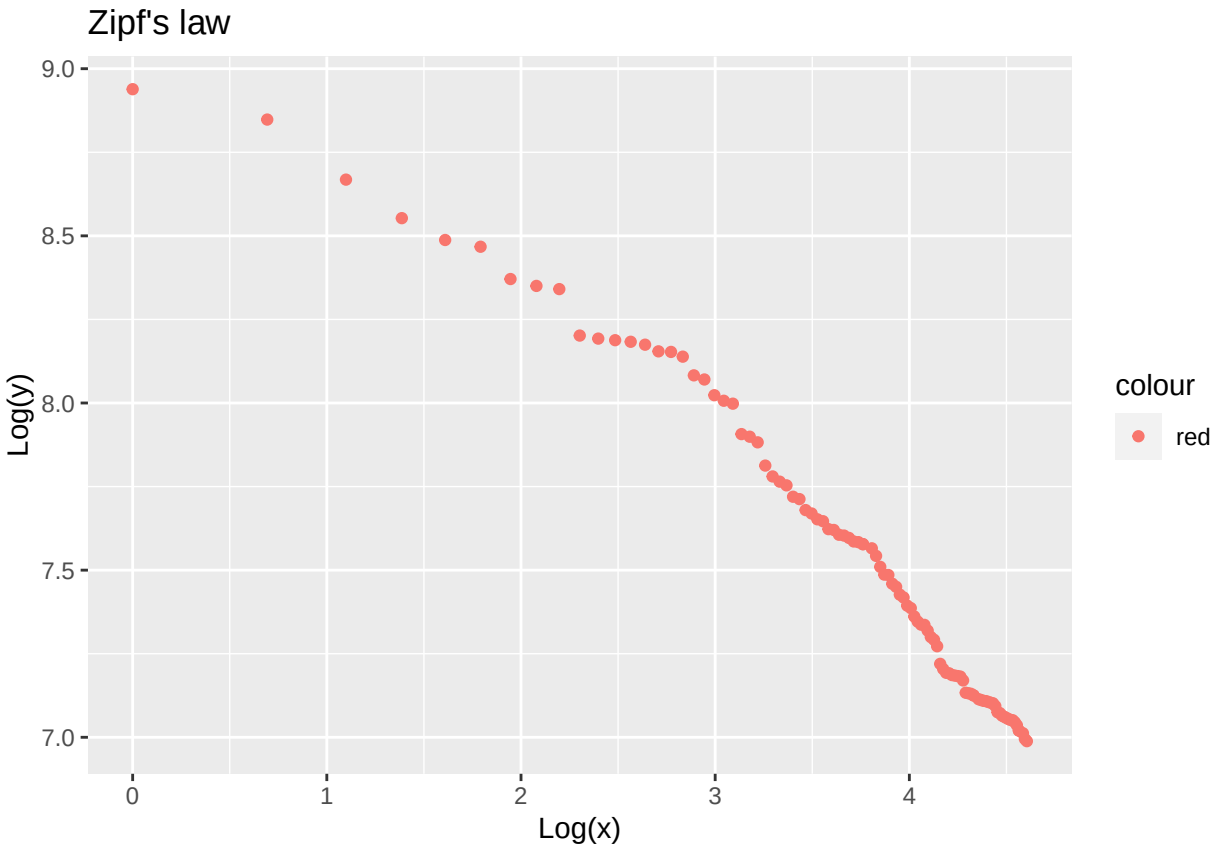


3.2

Replicate the plot above removing stop words. What do you notice?

A: There is a mild difference of the points between before and after removing the stop words(which makes sense because the ranking of the words changed), however the relationship between the frequency of the new words and its ranking is still inversely proportionate.

```
logy <- textstat_frequency(dfm_sotu_stop_words, n = 100)
zip_df <- data.frame(logy)
plot_data <- data.frame( x=log(zip_df$rank), y=log(zip_df$frequency))
ggplot(plot_data, aes(x=x, y=y, color = "red")) +
  geom_point()+
  labs(title="Zipf's law", x='Log(x)', y='Log(y)')
```



4 Collocations

4.1

Find bigrams collocations in Trump's State of the Union addresses. Display the top 5 (by z-score).

```
corp_gr <- corpus_group(corp, groups = President)
trump_gr <- corpus_subset(corp_gr, President == "Trump")
bigram_coll <- textstat_collocations(trump_gr, size = 2) %>%
```

```
  arrange(-z) %>%
  top_n(5)
```

```
## Selecting by z
```

```
head(bigram_coll)
```

```
##         collocation count count_nested length   lambda        z
## 1            we are    76            0      2 3.520544 22.52086
## 2         more than    37            0      2 6.500581 21.99630
## 3           we have    69            0      2 3.346940 21.07702
## 4 my administration    29            0      2 6.926015 19.05454
## 5       our country    50            0      2 4.563386 18.79205
```

4.2

What happens if you remove stop words? Based on your findings, do you think it's wise to remove stop words in this case?

A: I don't think it is necessary to remove stop words because the new collocation does not make sense and does not express any significant or meaningful information unlike the first collocation, when observed closely a pattern of unification and glorification can be deduced (i.e "we are", "we have", "our country")

```
tokens_trump <- tokens(trump_gr , remove_punct = TRUE, remove_symbols = TRUE) %>%
    tokens_remove(stopwords("en"))
bigram_coll_tokens <- textstat_collocations(tokens_trump, size = 2) %>%
  arrange(-z) %>%
  top_n(5)
```

```
## Selecting by z
```

```
head(bigram_coll_tokens)
```

```
##     collocation count count_nested length   lambda        z
## 1    thank much    30            0      2 6.131108 17.96812
## 2 united states    48            0      2 9.287578 17.18176
## 3     last year    19            0      2 5.798902 16.35820
## 4   health care     9            0      2 6.878049 12.82676
## 5     men women    11            0      2 7.833264 12.55518
```

4.3

Ignoring parts of speech information, almost all bigrams in a corpus occur more often than chance would lead us to expect. Why do you think that is?

Answer because any two words that come together in a corpus is a bigram. This means that one word can be in two bigrams at the same time which makes its frequency more likely to occur more than we would expect. This happens because almost all statistical analysis presupposes a specific pattern of bigrams to look for while other kinds of bigrams can occur outside this pattern. For example, looking for bigrams of nouns, adjectives and prepositions while ignoring subjects and verbs

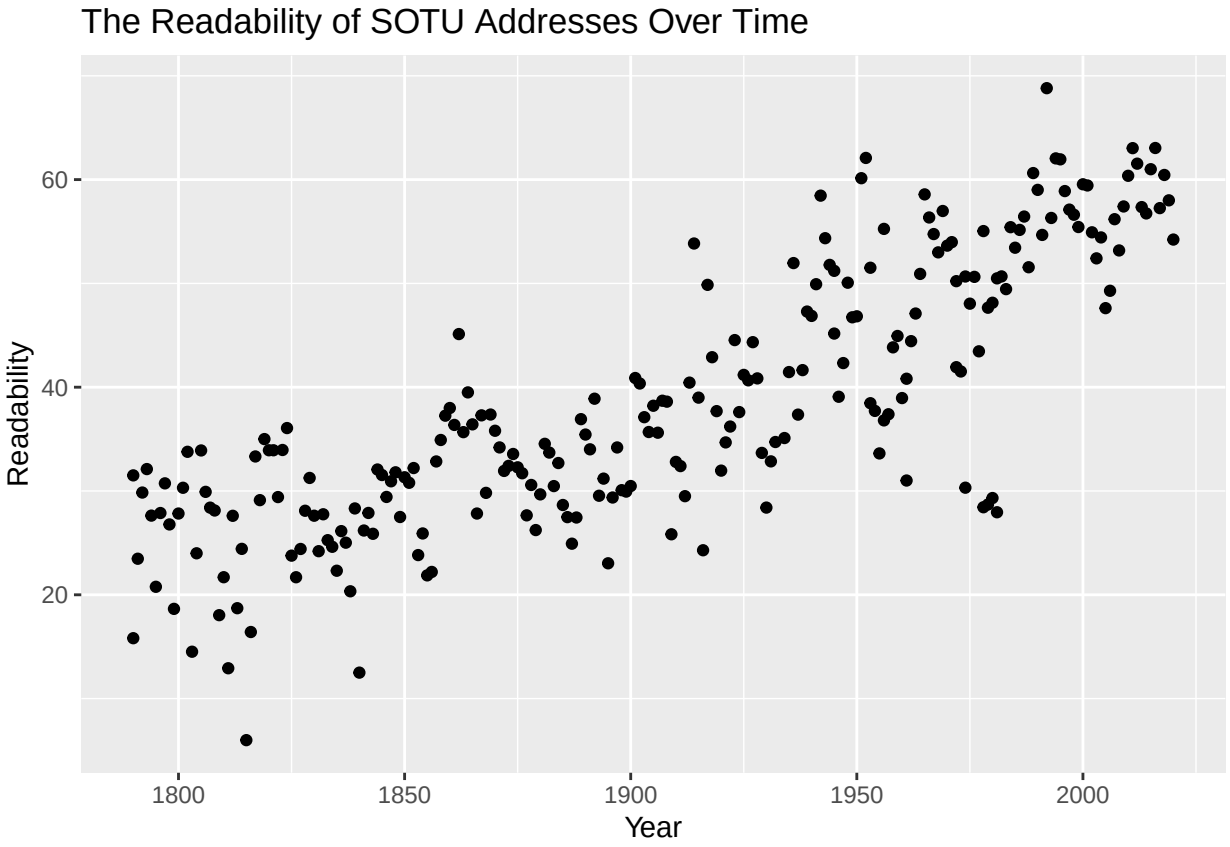
5. Readability

5.1

Plot the readability of SOTU addresses over time using FRE.

A: the readability increases over the years

```
read_corp <- textstat_readability(corp, "Flesch")
corp_df <- convert(corp, to = "data.frame")
corp_plot <- bind_cols(corp_df, read_corp)
  ggplot(data = corp_plot, aes(x = Year, y = Flesch)) +
  geom_point() +
    xlab("Year") +
  ylab("Readability") +
  ggtitle("The Readability of SOTU Addresses Over Time")
```

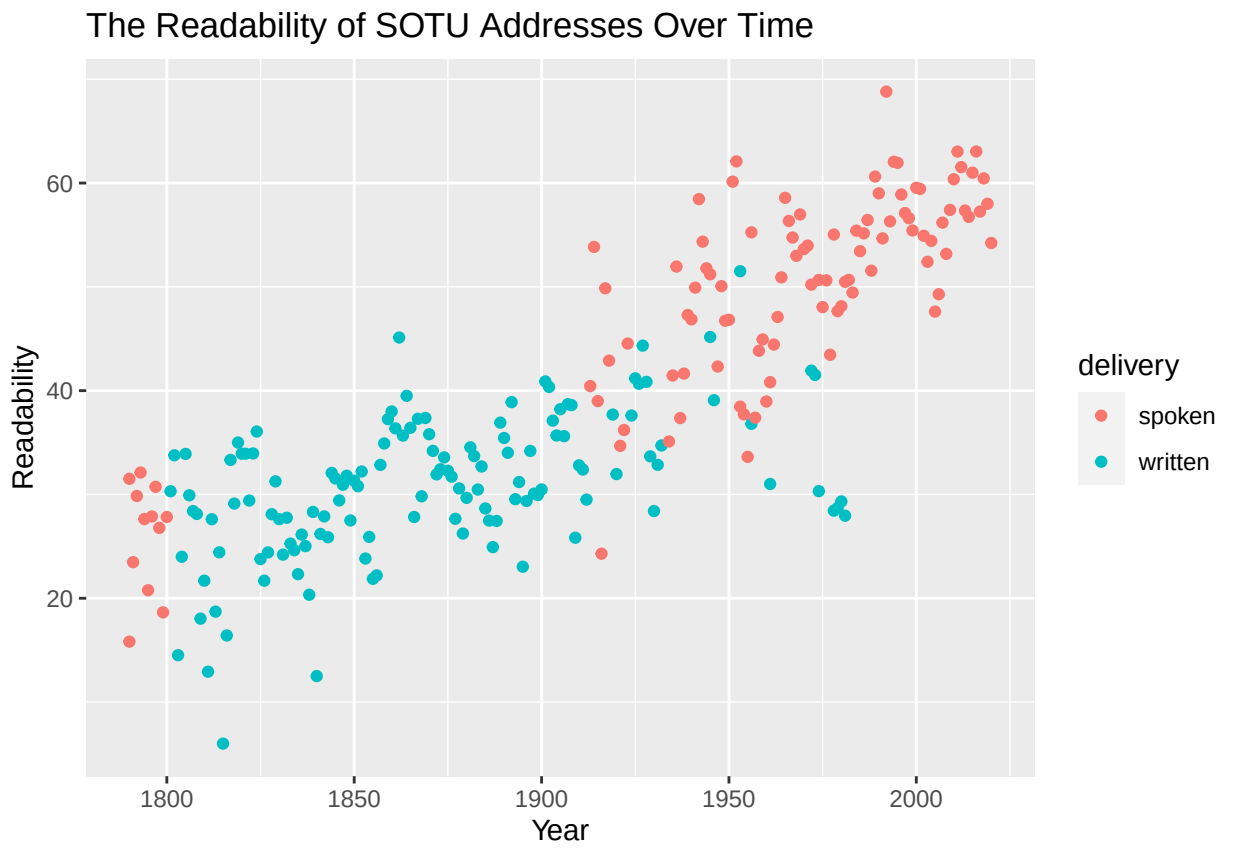


5.2

Amend the plot you made above such that each point is colored according to it's mode of delivery (i.e., spoken or written)

A: the delivery of speeches are now leaning more towards being spoken than written. Almost all recent spoken speeches have higher readability rates than the written ones.

```
ggplot(data = corp_plot, aes(x = Year, y = Flesch, color = delivery)) +
  geom_point() +
   xlab("Year") +
  ylab("Readability") +
  ggtitle("The Readability of SOTU Addresses Over Time")
```



5.3

Democrats are sometimes accused of being "out of touch" with the common man. Fit a simple linear model that tests this hypothesis. Do you find any evidence for it?

A: reject the hypothesis as there is a little to no correlation between the IV(the party) and the DV (which is readability)

```
corp_df$demo <- ifelse(corp_df$party == "Democratic", 1, 0)

corp_read_demo <- textstat_readability(corp , "Flesch")
corp_read_demo$party <- docvars(corp, "party")
corp_read_demo$demo <- ifelse(corp_read_demo$party == "Democratic", 1, 0)
view(corp_read_demo)
lm_demo <- lm(demo ~ Flesch, data = corp_read_demo)
lm_demo
```

```
##
```

```
## Call:
## lm(formula = demo ~ Flesch, data = corp_read_demo)
##
## Coefficients:
## (Intercept)       Flesch
##    0.084913     0.007909
```

```
summary(lm_demo)$r.squared
```

```
## [1] 0.04075386
```

Bonus 2

Matt Daniels tried to rank rappers by the size of their vocabulary. Read his analysis and evaluate one modeling choice he made (what he did, why it matters, if/how you'd do it differently.)

Answer: He ranked rappers by the number of unique words in their songs. The findings indicated that the more artists depart from rap, their lyrical diversity decreases. The author also found that the structure of rap is now closer to pop songs than before. I think what I would do differently is that I wouldn't exclude artists below 35000 words. His reason for only including 35000 artists was to be able to compare newer to older artists. However, we can solve this by depending on the rate/frequency of the unique words. By doing so, unique words will be divided by the total words of the artist and then we can compare the rates together.