Inaugural Address Text Analysis

William Eerdmans

March 15, 2017

```
dirname <- file.path("C:/Users/wjeer/OneDrive/Side Projects/Presidential Inagural Addresses/Addresses")
docs <- Corpus(DirSource(dirname, encoding = "UTF-8"))
docs[[1]]$content[1]
```

```
## [1] "when it was first perceived, in early times, that no middle course for america remained between
```

```
# The following steps pre-process the raw text documents.
# Remove punctuations and numbers because they are generally uninformative.
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, removeNumbers)

# Convert all words to lowercase.
docs <- tm_map(docs, content_transformer(tolower))

# Remove stopwords such as "a", "the", etc.
docs <- tm_map(docs, removeWords, stopwords("english"))

# Use the SnowballC package to do stemming.
docs <- tm_map(docs, stemDocument)

# Remove excess white spaces between words.
docs <- tm_map(docs, stripWhitespace)

# You can inspect the first document to see what it looks like with
docs[[1]]$content[1]
```

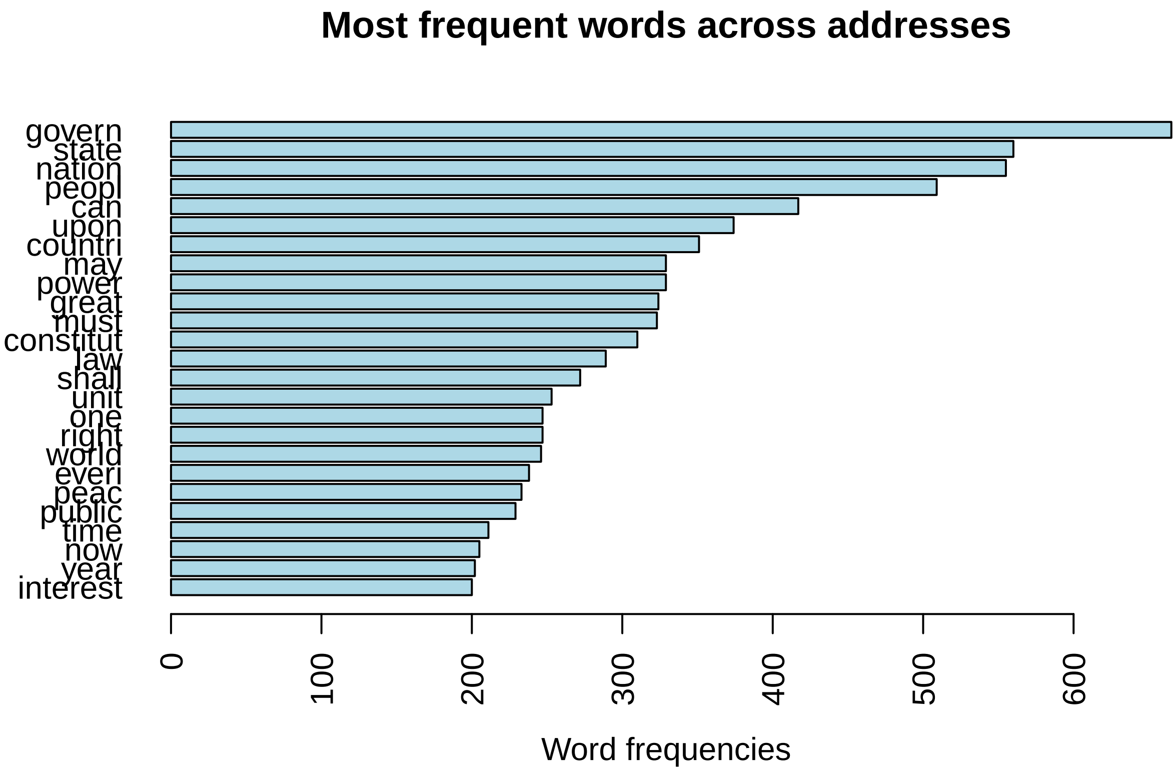
```
## [1] " first perceiv earli time middl cours america remain unlimit submiss foreign legislatur total i
```

```
# Convert all documents to a term frequency matrix.
tfm <- DocumentTermMatrix(docs)

# We can check the dimension of this matrix by calling dim()
print(dim(tfm))
```

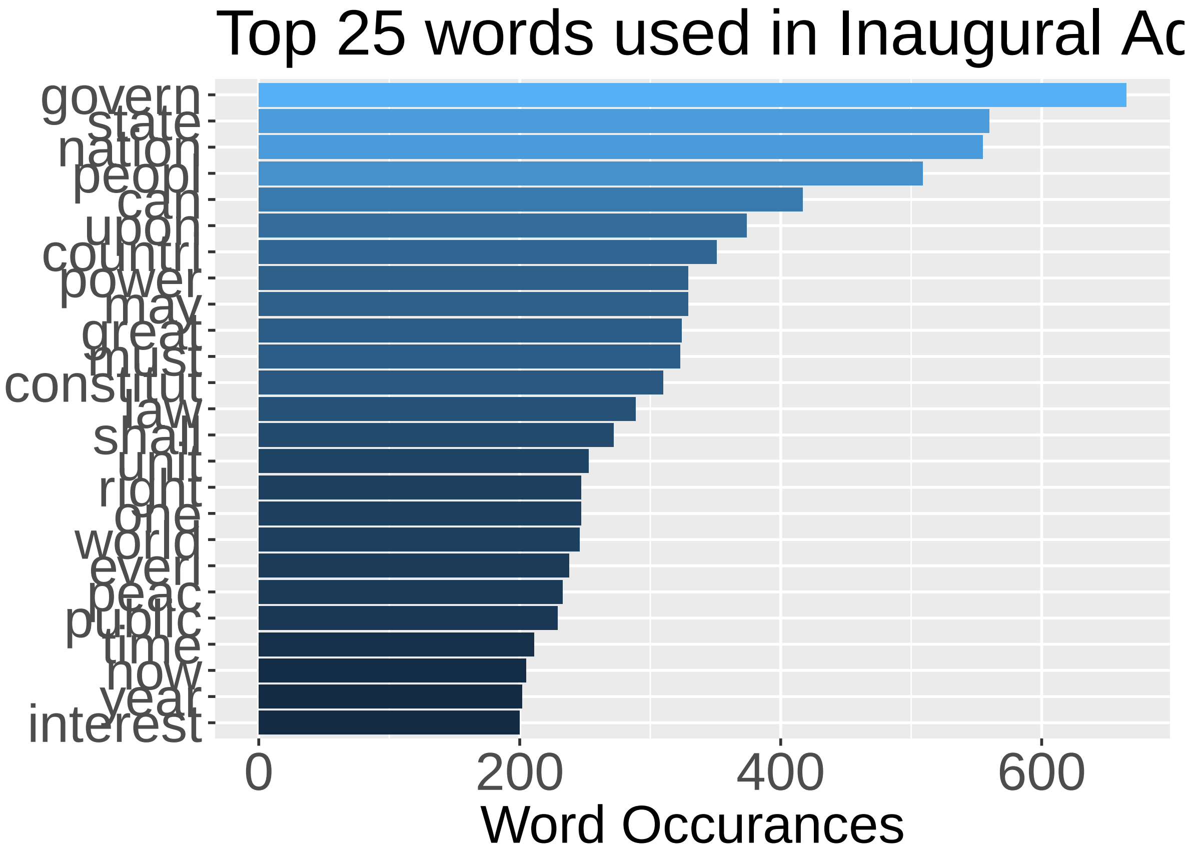
```
## [1]   42 5586
```

Find initial overall frequency of speechs, words used the most, etc.

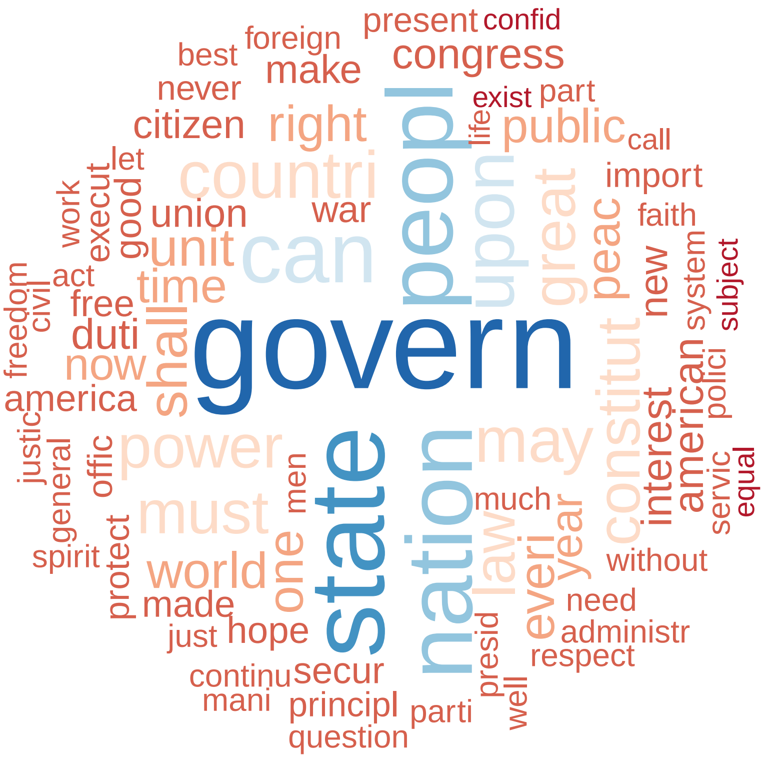
```
#Barplot of top 10 most frequent words
#initial strategy @ http://www.sthda.com/english/wiki/text-mining-and-word-cloud-fundamentals-in-r-5-si

m <- as.matrix(tfm)
v <- sort(colSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)

barplot(rev(d[2:26,]$freq), las = 2, names.arg = rev(d[2:26,]$word),
        horiz=TRUE,
        col ="lightblue", main ="Most frequent words across addresses",
        xlab = "Word frequencies",
        cex.names=1)
```



```
ggplot(d[2:26,], aes(x=reorder(word, freq), y=freq, fill = log(freq))) + geom_bar(stat="identity") + coo
```

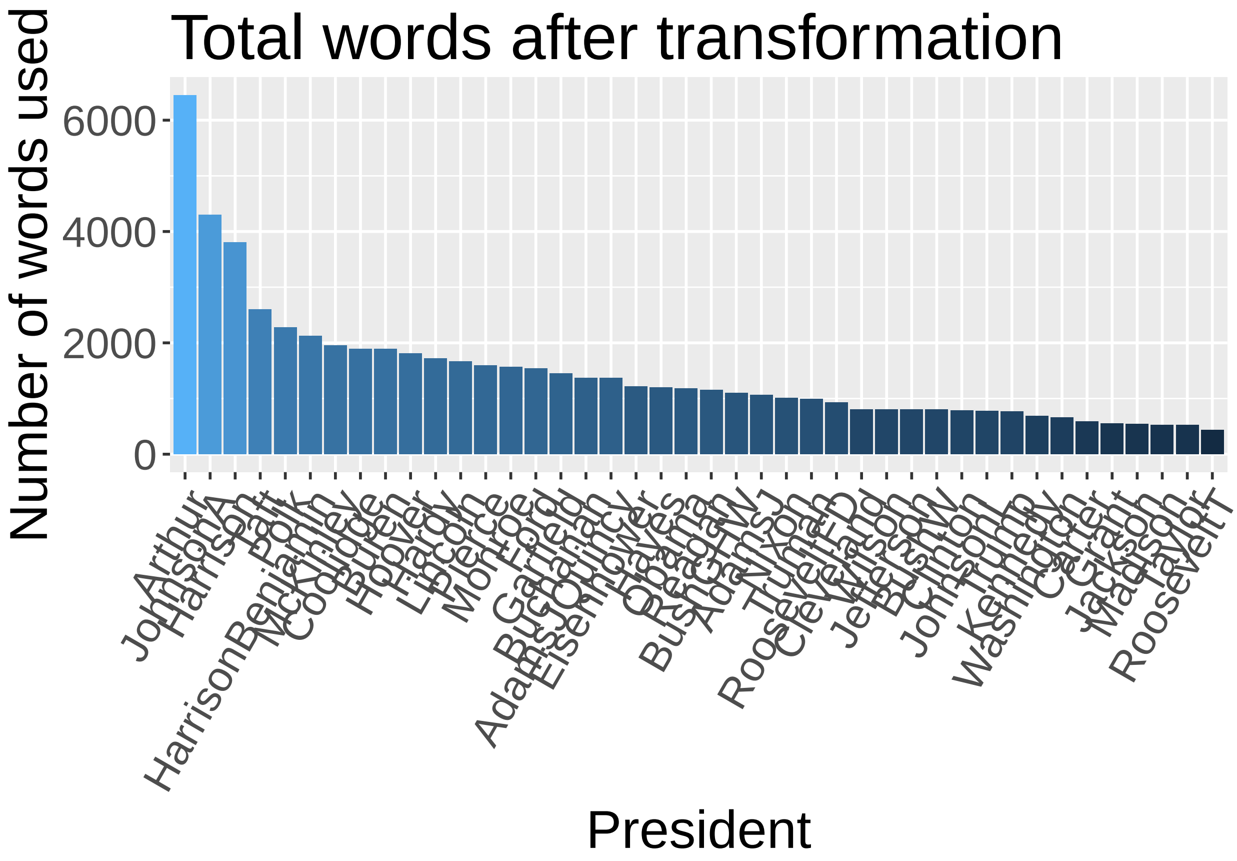


```
#Create word cloud or chart of most used words, removed will
wordcloud(words = d$word[-c(d$word == "will")], freq = d$freq, min.freq = 100,
          max.words=500, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "RdBu"))
```

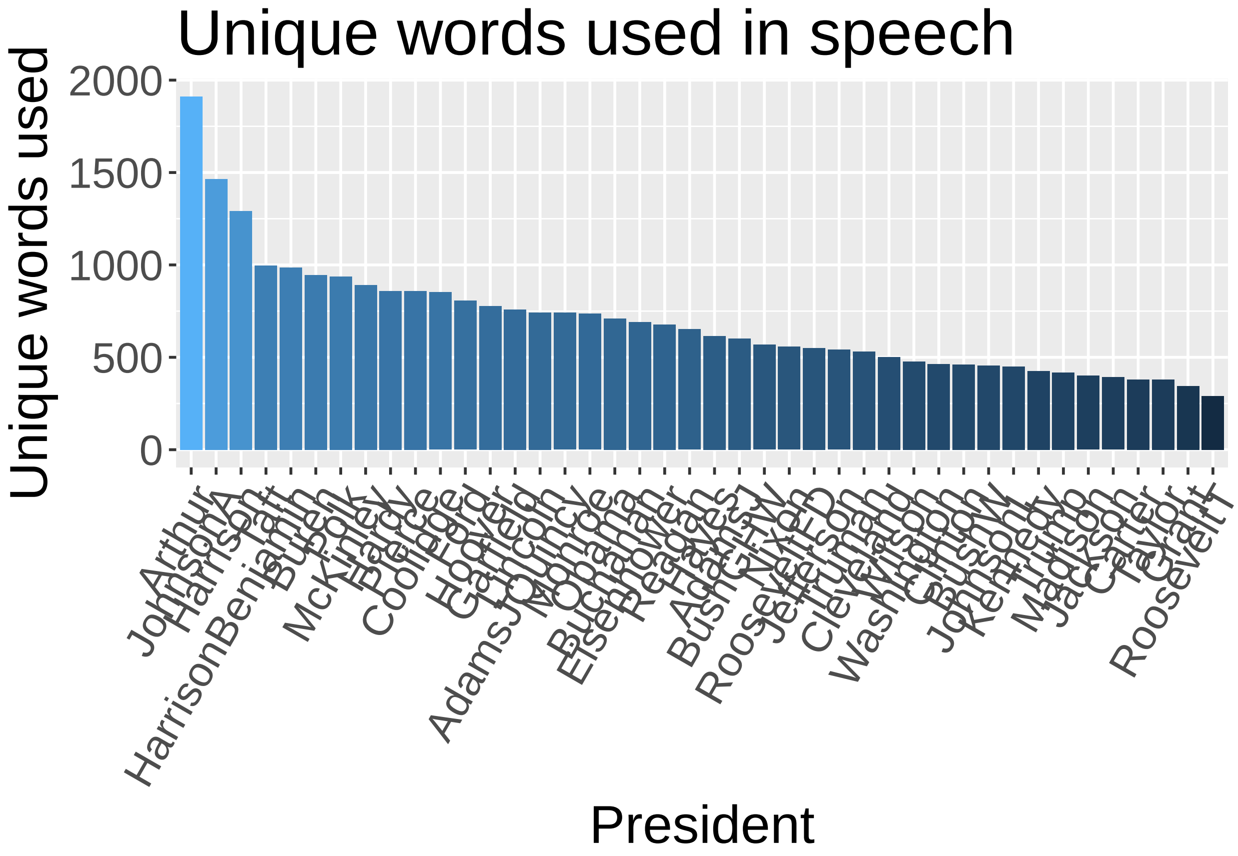


In this section find the most used words by President and unique words

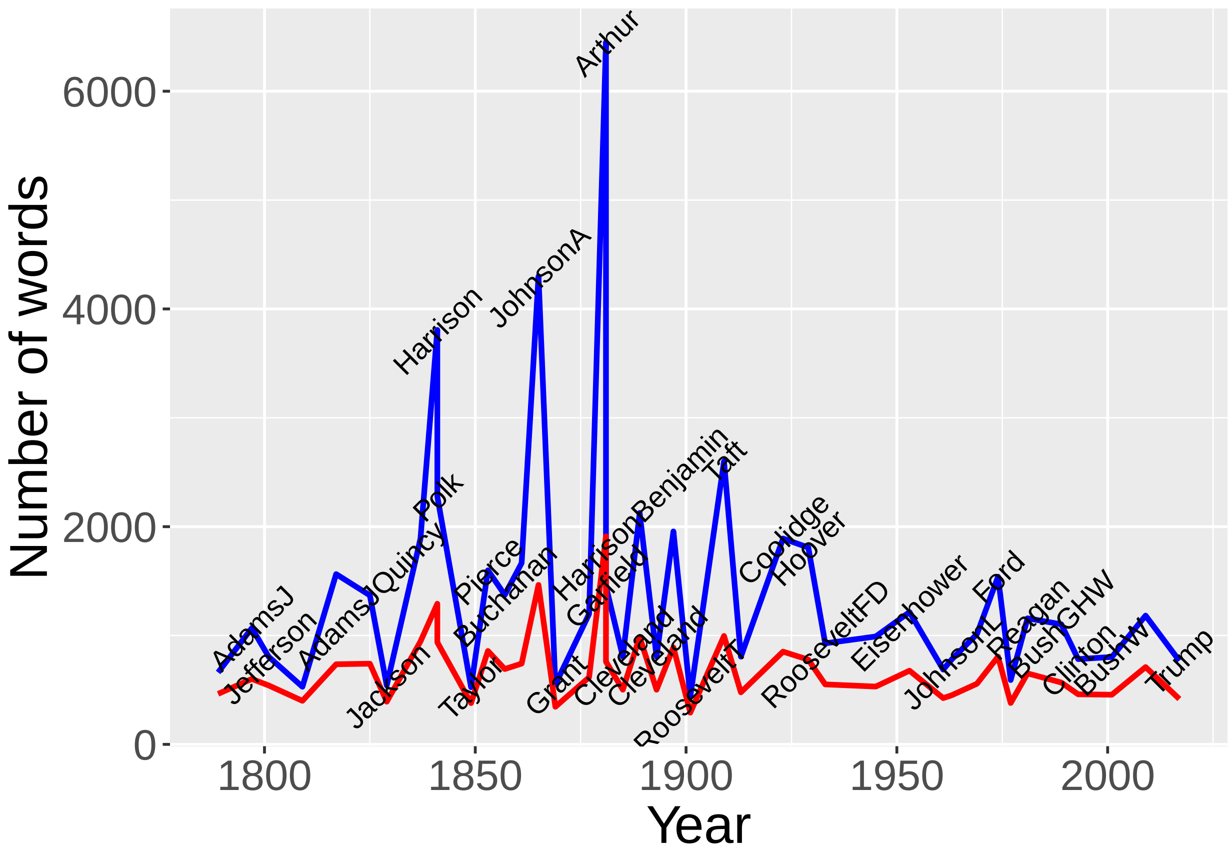
```
#Find the total words per speech after transformation
all_words <- rowSums(as.matrix(tfm))
tot_words <- data.frame(Last.Name = gsub('.{4}$', '', names(all_words)), Word_Frequency=all_words)

#Create bar chart of presidents and words
ggplot(tot_words, aes(x = reorder(Last.Name, -Word_Frequency), y = Word_Frequency, fill = log(Word_Freq
```



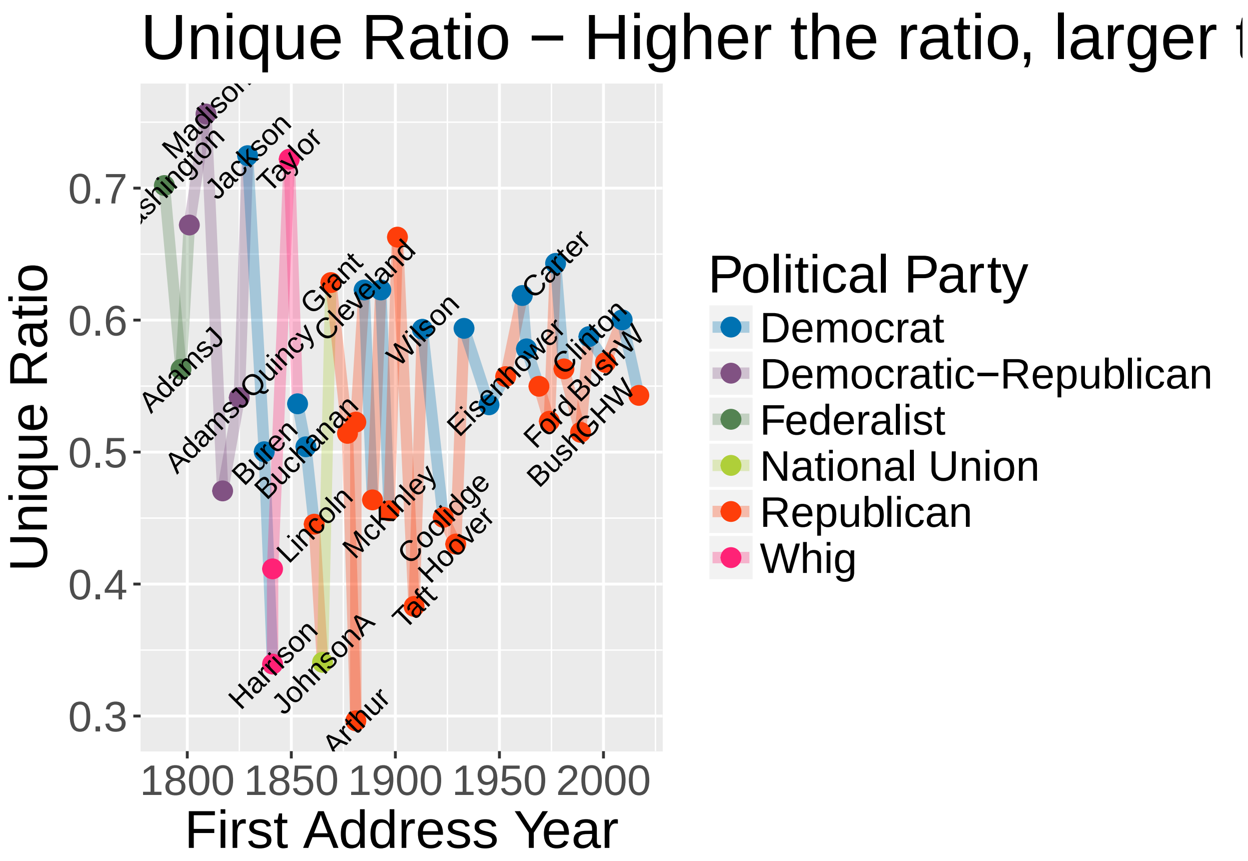
```r
#What about unique words?
a <- as.matrix(tfm)

#Make all duplicates just 1 (binary)
a[a>0] <- 1
unique_words <- rowSums(a)
uniq_words <- data.frame(Last.Name = gsub('.{4}$', '', names(unique_words)), Word_Frequency_Unique=uniq

#Create bar chart of presidents and words
ggplot(uniq_words, aes(x = reorder(Last.Name, -Word_Frequency_Unique), y = Word_Frequency_Unique, fill =
```



```r
#Bring in the Date of Inauguration and merge the two frequency dataframes to it
#Create time series graph showing number of words of speeches over time (just interesting, haha)
pres_dates <- read.csv("President_year.csv")
totWords_dates <- merge(tot_words, pres_dates, by = "Last.Name")
word_freq_dates <- merge(totWords_dates, uniq_words, by = "Last.Name" )

#create time series
ggplot(word_freq_dates) + geom_line(aes(x=start_year, y = Word_Frequency), color="blue", size = 1) + ge
```



```
#Unique word to total word ratio
word_freq_dates$uniq_ratio <- word_freq_dates$Word_Frequency_Unique / word_freq_dates$Word_Frequency

#create unique ratio graph
ggplot(word_freq_dates, aes(x = start_year, y = uniq_ratio,label = Last.Name)) + geom_line(aes(x = star
```

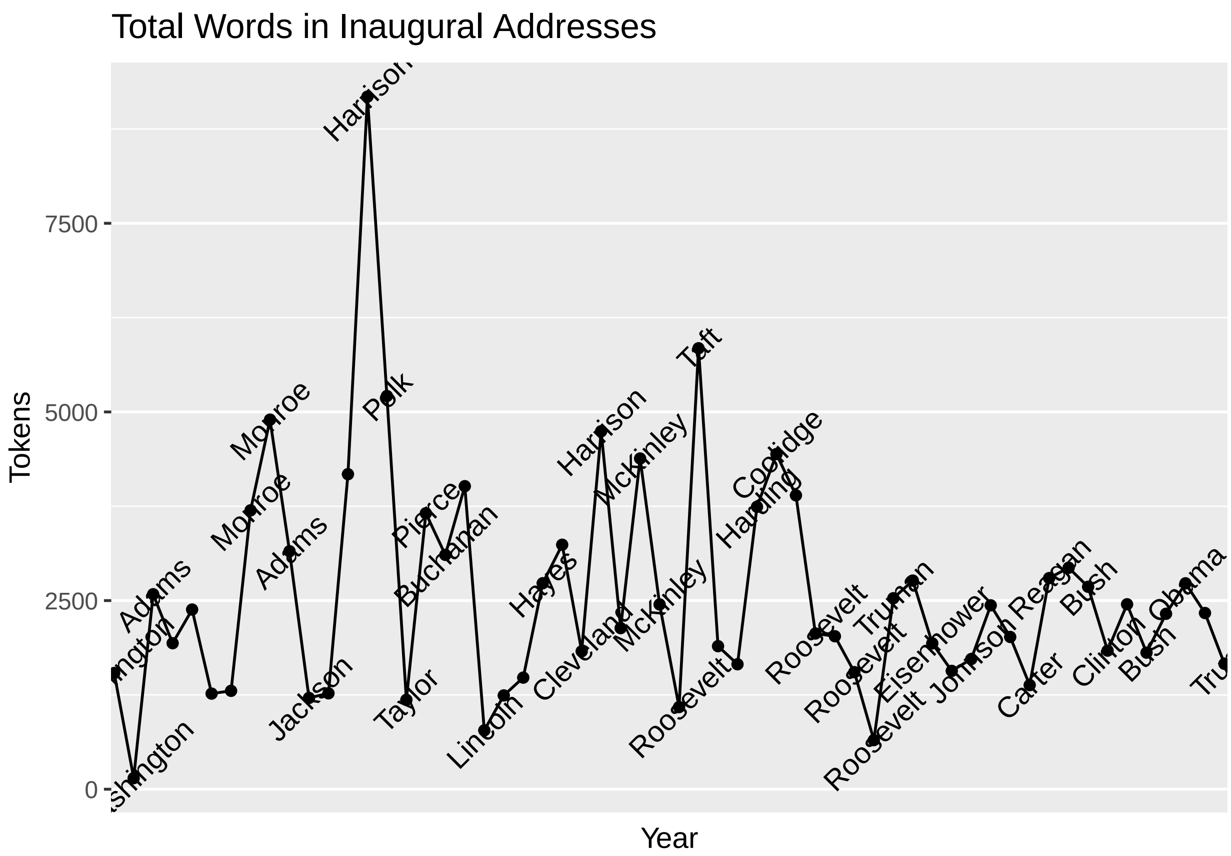


Find the bi, tri, and four-grams in the speeches overall. Having difficulties with RWeka and rJava, thus resorted to quanteda. This corpus includes all addresses, whereas mine only includes the first one and addresses to Congress after assinations.

```
#Looking at all inaugural addresses and lengths
tokenInfo <- summary(data_corpus_inaugural)
```

```
## Corpus consisting of 58 documents.
##
##             Text Types Tokens Sentences Year  President       FirstName
##  1789-Washington   626   1540        23 1789 Washington          George
##  1793-Washington    96    147         4 1793 Washington          George
##       1797-Adams   826   2584        37 1797      Adams            John
##   1801-Jefferson   716   1935        41 1801  Jefferson          Thomas
##   1805-Jefferson   804   2381        45 1805  Jefferson          Thomas
##     1809-Madison   536   1267        21 1809    Madison           James
##     1813-Madison   542   1304        33 1813    Madison           James
##      1817-Monroe  1040   3696       121 1817     Monroe           James
##      1821-Monroe  1262   4898       129 1821     Monroe           James
##       1825-Adams  1004   3154        74 1825      Adams    John Quincy
##     1829-Jackson   517   1210        25 1829    Jackson          Andrew
##     1833-Jackson   499   1271        29 1833    Jackson          Andrew
##    1837-VanBuren  1315   4175        95 1837  Van Buren          Martin
##    1841-Harrison  1893   9178       210 1841   Harrison  William Henry
##        1845-Polk  1330   5211       153 1845       Polk     James Knox
##      1849-Taylor   497   1185        22 1849     Taylor        Zachary
##      1853-Pierce  1166   3657       104 1853     Pierce       Franklin
##    1857-Buchanan   945   3106        89 1857   Buchanan           James
##     1861-Lincoln  1075   4016       135 1861    Lincoln         Abraham
##     1865-Lincoln   362    780        26 1865    Lincoln         Abraham
##       1869-Grant   486   1243        40 1869      Grant      Ulysses S.
##       1873-Grant   552   1479        43 1873      Grant      Ulysses S.
##       1877-Hayes   829   2730        59 1877      Hayes   Rutherford B.
##    1881-Garfield  1018   3240       111 1881   Garfield        James A.
##   1885-Cleveland   674   1828        44 1885  Cleveland          Grover
##    1889-Harrison  1355   4744       157 1889   Harrison        Benjamin
##   1893-Cleveland   823   2135        58 1893  Cleveland          Grover
##    1897-McKinley  1236   4383       130 1897   McKinley         William
##    1901-McKinley   857   2449       100 1901   McKinley         William
##   1905-Roosevelt   404   1089        33 1905  Roosevelt        Theodore
##        1909-Taft  1436   5844       159 1909       Taft  William Howard
##      1913-Wilson   661   1896        68 1913     Wilson         Woodrow
##      1917-Wilson   549   1656        59 1917     Wilson         Woodrow
##     1921-Harding  1172   3743       148 1921    Harding       Warren G.
##    1925-Coolidge  1221   4442       196 1925   Coolidge          Calvin
##      1929-Hoover  1086   3895       158 1929     Hoover         Herbert
##   1933-Roosevelt   744   2064        85 1933  Roosevelt     Franklin D.
##   1937-Roosevelt   729   2027        96 1937  Roosevelt     Franklin D.
##   1941-Roosevelt   527   1552        68 1941  Roosevelt     Franklin D.
##   1945-Roosevelt   276    651        26 1945  Roosevelt     Franklin D.
##      1949-Truman   781   2531       116 1949     Truman        Harry S.
##  1953-Eisenhower   903   2765       119 1953 Eisenhower       Dwight D.
##  1957-Eisenhower   621   1933        92 1957 Eisenhower       Dwight D.
##     1961-Kennedy   566   1568        52 1961    Kennedy         John F.
##     1965-Johnson   569   1725        93 1965    Johnson   Lyndon Baines
##       1969-Nixon   743   2437       103 1969      Nixon Richard Milhous
##       1973-Nixon   545   2018        68 1973      Nixon Richard Milhous
##      1977-Carter   528   1380        52 1977     Carter           Jimmy
##      1981-Reagan   904   2798       128 1981     Reagan          Ronald
##      1985-Reagan   925   2935       123 1985     Reagan          Ronald
##        1989-Bush   795   2683       141 1989       Bush          George
##     1993-Clinton   644   1837        81 1993    Clinton            Bill
##     1997-Clinton   773   2451       111 1997    Clinton            Bill
##        2001-Bush   622   1810        97 2001       Bush       George W.
##        2005-Bush   772   2325       100 2005       Bush       George W.
##       2009-Obama   939   2729       110 2009      Obama          Barack
##       2013-Obama   814   2335        88 2013      Obama          Barack
##       2017-Trump   582   1662        88 2017      Trump       Donald J.
##
## Source:  /home/paul/Dropbox/code/quanteda/* on x86_64 by paul
## Created: Fri Sep 12 12:41:17 2014
## Notes:
```

```
#Similar plot below, but this is total words without taking out unneccesary words like I, as, or, and,
ggplot(data=tokenInfo, aes(x=Year, y=Tokens, group=1, label = President)) + geom_line() + geom_point() +
```



```
#Find the bi-grams
bigrams <- dfm_trim(dfm(data_corpus_inaugural, tolower= TRUE, remove = stopwords("english"), removePunc
```

```
## Creating a dfm from a corpus ...

##    ... lowercasing

##    ... tokenizing

##    ... found 58 documents, 64,456 features

## ...

## removed 45,967 features, from 174 supplied (regex) feature types

## ... stemming features (English)

## , trimmed 592 feature variants

##    ... created a 58 x 17,897 sparse dfm
##    ... complete.
## Elapsed time: 1005 seconds.
```

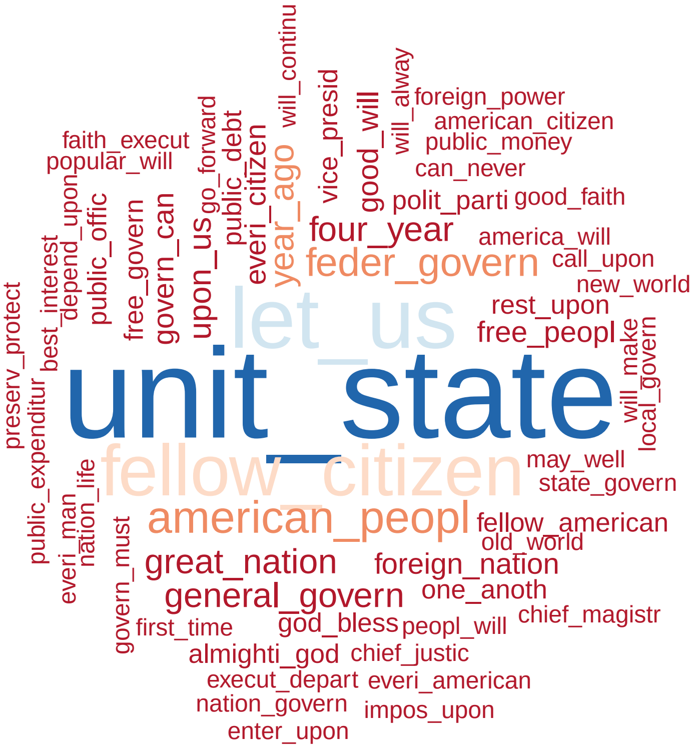
```
#Find the bigrams with more than 10 occurances
which(colSums(bigrams) >= 10)
```

```
##        unit_state           let_us    fellow_citizen    american_peopl
##                 1                2                 3                 4
##       feder_govern         year_ago         four_year      great_nation
##                 5                6                 7                 8
##     general_govern          upon_us     everi_citizen    foreign_nation
##                 9                10                11                12
##        govern_can         good_will        free_peopl         god_bless
##                13                14                15                16
##         rest_upon       polit_parti       free_govern       vice_presid
##                17                18                19                20
##      almighti_god         one_anoth   fellow_american      public_offic
##                21                22                23                24
##       public_debt       nation_life      chief_justic      america_will
##                25                26                27                28
##          may_well        peopl_will         can_never     execut_depart
##                29                30                31                32
##       govern_must      state_govern      public_money         will_make
##                33                34                35                36
##        go_forward        will_alway    everi_american       depend_upon
##                37                38                39                40
##     nation_govern        impos_upon        first_time         call_upon
##                41                42                43                44
##     best_interest public_expenditur         everi_man   preserv_protect
##                45                46                47                48
##         new_world     chief_magistr      local_govern         old_world
##                49                50                51                52
##        good_faith      will_continu  american_citizen     foreign_power
##                53                54                55                56
##      faith_execut        enter_upon      popular_will
##                57                58                59
```

```
#How many times did these words occur? #Trump uses United States and unite only 3 times, whereas Americ
bigrams_matrix <- data.frame(bigrams[,which(colSums(bigrams) >= 10)])

bigram_freq <- data.frame(bigram_freq = colSums(data.frame(bigrams[,which(colSums(bigrams) >= 10)])))

#Who used the most of these bigrams?
sort(rowSums(data.frame(bigrams[,which(colSums(bigrams) >= 10)])), decreasing = TRUE)
```

```
##   1841-Harrison       1845-Polk      1973-Nixon     1821-Monroe
##              52              47              39              38
##     1985-Reagan   1897-McKinley      1877-Hayes     1817-Monroe
##              38              36              35              33
##    1997-Clinton   1901-McKinley     1929-Hoover      1969-Nixon
##              33              30              30              29
##   1837-VanBuren       1909-Taft       1989-Bush       2005-Bush
##              27              26              26              26
##   1881-Garfield  1885-Cleveland   1889-Harrison     1981-Reagan
##              25              25              24              24
##      1825-Adams   1857-Buchanan   1925-Coolidge    1993-Clinton
##              23              22              21              21
##    1833-Jackson     1853-Pierce    1861-Lincoln  1893-Cleveland
##              20              20              20              20
##  1937-Roosevelt     1949-Truman      2013-Obama    1961-Kennedy
##              19              19              19              18
##  1801-Jefferson  1805-Jefferson    1921-Harding 1953-Eisenhower
##              17              15              15              15
##      2017-Trump 1789-Washington      1797-Adams      1869-Grant
##              15              14              14              14
##      2009-Obama    1829-Jackson     1849-Taylor     1917-Wilson
##              14              13              13              11
##  1933-Roosevelt     1977-Carter     1913-Wilson  1941-Roosevelt
##              11               9               8               8
## 1957-Eisenhower    1965-Johnson       2001-Bush    1809-Madison
##               8               8               8               7
##      1873-Grant    1865-Lincoln  1905-Roosevelt    1813-Madison
##               7               6               6               5
## 1793-Washington  1945-Roosevelt
##               3               3
```

```
# Favorite bigram
grams_pres <- data.frame(pres = rownames(bigrams_matrix))

grams_pres$fav_bigram <- c(colnames(bigrams_matrix[,2:ncol(bigrams_matrix)])[max.col(bigrams_matrix[,2:

# word cloud bigrams
wordcloud(words = rownames(bigram_freq), freq = bigram_freq$bigram_freq, min.freq = 10,
          max.words=500, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(6, "RdBu"))
```



```
#Find the tri-grams
trigrams <- dfm_trim(dfm(data_corpus_inaugural, tolower= TRUE, remove = stopwords("english"), removePun
```

```
## Creating a dfm from a corpus ...
```

```
##    ... lowercasing
```

```
##    ... tokenizing
##    ... found 58 documents, 113,794 features

## ...

## removed 107,422 features, from 174 supplied (regex) feature types

## ... stemming features (English)

## , trimmed 13 feature variants

##    ... created a 58 x 6,359 sparse dfm
##    ... complete.
## Elapsed time: 2081 seconds.
```

```
#Find the trigrams with more than 2 occurances
which(colSums(trigrams) > 2)
```

```
##    mr_chief_justic      four_year_ago      let_us_resolv
##                  1                  2                  3
##    chief_justic_mr     mr_vice_presid          let_us_go
##                  4                  5                  6
##  will_make_america     past_four_year       let_us_begin
##                  7                  8                  9
##  great_polit_parti  go_forward_togeth       world_let_us
##                 10                 11                 12
##      let_us_rememb    two_centuri_ago    will_bring_back
##                 13                 14                 15
##       let_us_never     vice_presid_mr     unit_state_can
##                 16                 17                 18
##     last_four_year      us_go_forward       let_us_build
##                 19                 20                 21
## chief_execut_offic     centuri_let_us  will_faith_execut
##                 22                 23                 24
##  god_bless_america peopl_still_believ
##                 25                 26
```

```
#How many times did these words occur?
colSums(data.frame(trigrams[,which(colSums(trigrams) > 2)]))
```

```
##    mr_chief_justic      four_year_ago      let_us_resolv
##                  9                  8                  7
##    chief_justic_mr     mr_vice_presid          let_us_go
##                  5                  5                  5
##  will_make_america     past_four_year       let_us_begin
##                  5                  4                  4
##  great_polit_parti  go_forward_togeth       world_let_us
##                  4                  4                  4
##      let_us_rememb    two_centuri_ago    will_bring_back
##                  4                  4                  4
##       let_us_never     vice_presid_mr     unit_state_can
##                  3                  3                  3
##     last_four_year      us_go_forward       let_us_build
##                  3                  3                  3
## chief_execut_offic     centuri_let_us  will_faith_execut
##                  3                  3                  3
##  god_bless_america peopl_still_believ
##                  3                  3
```

```
#Who used the most of these trigrams?
sort(rowSums(data.frame(trigrams[,which(colSums(trigrams) > 2)])), decreasing = TRUE)
```

```
##      1973-Nixon     1985-Reagan      2017-Trump      1969-Nixon 
##              15              10              10               9 
##    1997-Clinton    1961-Kennedy 1957-Eisenhower      2013-Obama 
##               7               6               5               5 
##  1937-Roosevelt    1993-Clinton      1877-Hayes     1949-Truman 
##               4               4               3               3 
##     1981-Reagan    1833-Jackson   1897-McKinley   1901-McKinley 
##               3               2               2               2 
##  1945-Roosevelt       1989-Bush       2005-Bush     1817-Monroe 
##               2               2               2               1 
##      1825-Adams   1837-VanBuren   1841-Harrison   1857-Buchanan 
##               1               1               1               1 
##    1865-Lincoln      1873-Grant   1889-Harrison       1909-Taft 
##               1               1               1               1 
##     1917-Wilson     1977-Carter       2001-Bush      2009-Obama 
##               1               1               1               1 
## 1789-Washington 1793-Washington      1797-Adams  1801-Jefferson 
##               0               0               0               0 
##  1805-Jefferson    1809-Madison    1813-Madison     1821-Monroe 
##               0               0               0               0 
##    1829-Jackson       1845-Polk     1849-Taylor     1853-Pierce 
##               0               0               0               0 
##    1861-Lincoln      1869-Grant   1881-Garfield  1885-Cleveland 
##               0               0               0               0 
##  1893-Cleveland  1905-Roosevelt     1913-Wilson    1921-Harding 
##               0               0               0               0 
##   1925-Coolidge     1929-Hoover  1933-Roosevelt  1941-Roosevelt 
##               0               0               0               0 
## 1953-Eisenhower    1965-Johnson 
##               0               0
```

```
#How many times did these words occur
trigrams_matrix <- data.frame(trigrams[,which(colSums(trigrams) >= 2)])

trigram_freq <- data.frame(bigram_freq = colSums(data.frame(trigrams[,which(colSums(trigrams) >= 2)])))

# Favorite Trigram
grams_pres$fav_trigram <- c(colnames(trigrams_matrix)[max.col(trigrams_matrix,ties.method="first")])

# Trigrams word cloud
wordcloud(words = rownames(trigram_freq), freq = trigram_freq$bigram_freq, min.freq = 2,
          max.words=500, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(7, "RdYlBu"))
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : mr_chief_justic could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : chief_justic_mr could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : mr_vice_presid could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : go_forward_togeth could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : world_let_us could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : let_us_rememb could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : will_bring_back could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : vice_presid_mr could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : last_four_year could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : us_go_forward could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : chief_execut_offic could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : will_faith_execut could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : chief_justic_presid could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : concern_thank_god could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : last_best_hope could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : us_begin_anew could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : presid_mr_chief could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : whole_constitut_vigor could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : four_major_cours could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : perfect_union_establish could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : union_establish_justic could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : establish_justic_insur could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : justic_insur_domest could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : insur_domest_tranquil could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : domest_tranquil_provid could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : common_defens_promot could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : brought_upon_us could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : interst_commerc_railroad could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : first_regular_session could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : earliest_practic_period could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : alon_can_suppli could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : everi_american_citizen could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : tradit_american_polici could not be fit on page. It will
## not be plotted.
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : origin_thirteen_state could not be fit on page. It will not
## be plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : mankind_let_us could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : let_us_take could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : new_world_togeth could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : make_everi_effort could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : peac_among_nation could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : pass_judgment_upon could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : confid_reli_upon could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : proclaim_liberti_throughout could not be fit on page. It
## will not be plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : shall_count_upon could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : nation_entangl_allianc could not be fit on page. It will
## not be plotted.
```

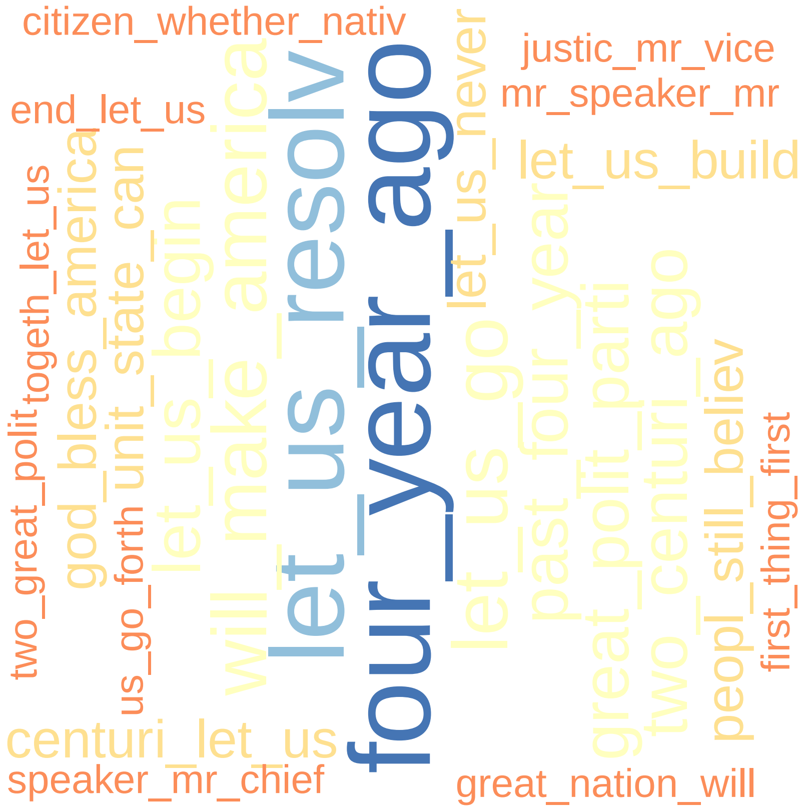
```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : justic_mr_presid could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : mr_presid_vice could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : presid_vice_presid could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : requir_turn_away could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : world_move_toward could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : let_us_invok could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : crucial_thing_uniti could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : next_four_year could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : nation_can_long could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : mr_major_leader could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : vice_presid_bush could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : will_go_away could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : fellow_citizen_look could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : special_interest_group could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : let_us_renew could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : creat_new_job could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : presid_carter_presid could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : presid_bush_presid could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : may_confid_expect could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : unit_state_congress could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : peopl_must_depend could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : peopl_let_us could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : america_must_continu could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : abil_preserv_protect could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : public_interest_depend could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : temporari_restrain_order could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : free_peopl_must could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : two_year_ago could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : citizen_let_us could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : may_god_bless could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : fifti_year_ago could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : everi_citizen_must could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : must_rest_upon could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : will_never_forget could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : make_war_upon could not be fit on page. It will not be
## plotted.
## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : forc_upon_us could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : indian_tribe_within could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : must_act_know could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : depend_upon_government could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : law_can_ever could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : go_far_toward could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : impos_upon_us could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : foreign_slave_trade could not be fit on page. It will not
## be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : american_merchant_marin could not be fit on page. It will
## not be plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : near_two_year could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : twenty.f_year_ago could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : twenty.f_year_henc could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : trust_impos_upon could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : unit_state_first could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = rownames(trigram_freq), freq = trigram_freq
## $bigram_freq, : exclus_metal_currenc could not be fit on page. It will not
## be plotted.
```



Perform topic modeling

```
# I run LDA and use Gibbs sampling as our method for identifying the optimal parameters
# Note: this make take some time to run (~10 mins)
set.seed(12345)
results <- LDA(tfm, k = 5, method = "Gibbs")

# Obtain the top w words (i.e., the w most probable words) for each topic, with the optional requiremen

#feel free to explore with different values of w and thresh
w=10
thresh = 0.005
set.seed(12345)
Terms <- terms(results, w,thresh)
Terms
```

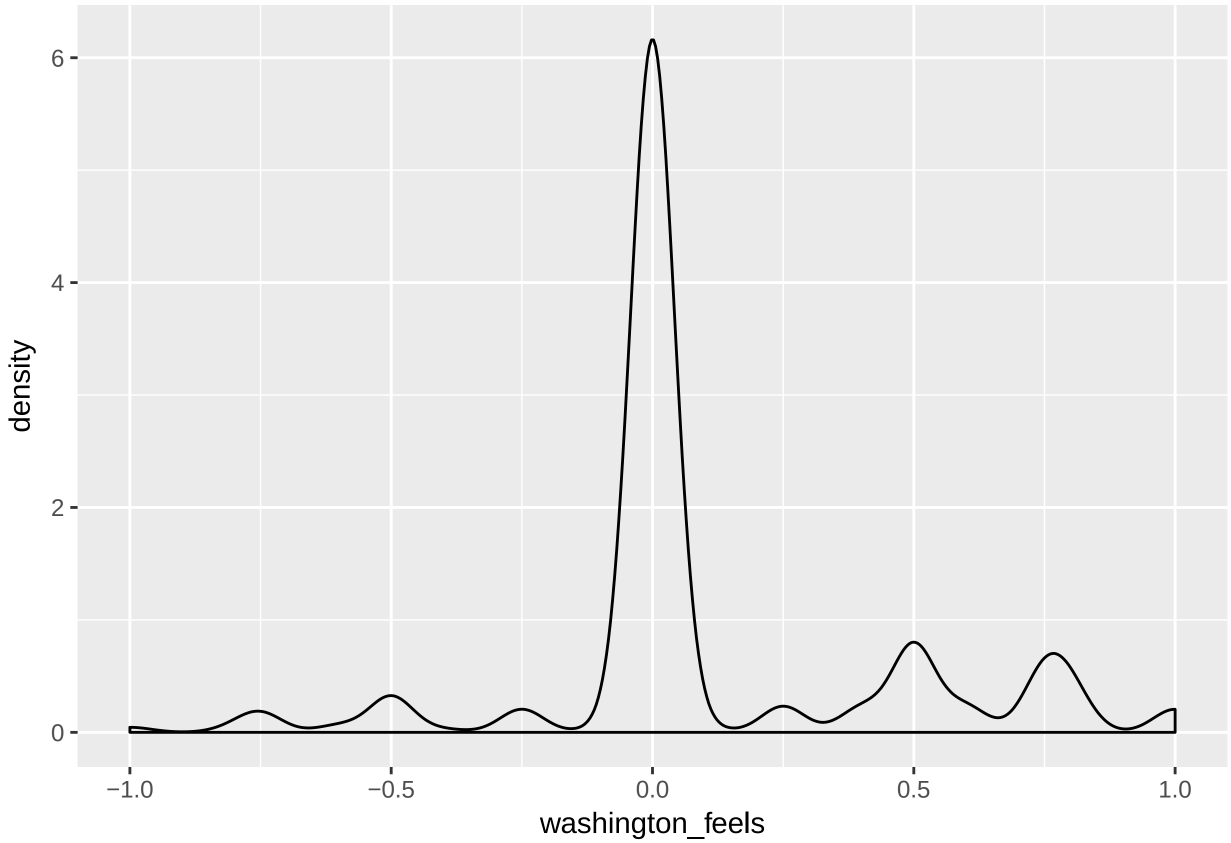
```
##       Topic 1    Topic 2  Topic 3    Topic 4  Topic 5
##  [1,] "america"  "civil"  "congress" "can"    "constitut"
##  [2,] "american" "faith"  "govern"   "govern" "countri"
##  [3,] "can"      "hope"   "import"   "great"  "everi"
##  [4,] "let"      "human"  "increas"  "law"    "govern"
##  [5,] "must"     "justic" "now"      "must"   "may"
##  [6,] "new"      "nation" "present"  "nation" "peopl"
##  [7,] "time"     "peac"   "state"    "polici" "power"
##  [8,] "will"     "peopl"  "territori" "secur"  "right"
##  [9,] "work"     "war"    "unit"     "upon"   "state"
## [10,] "world"    "world"  "year"     "will"   "will"
```

Perform how the sentiment changes over time

```
dirname <- file.path("C:/Users/wjeer/OneDrive/Side Projects/Presidential Inagural Addresses/Addresses")
docs <- Corpus(DirSource(dirname, encoding = "UTF-8"))

docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, removeNumbers)

# Convert all words to lowercase.
docs <- tm_map(docs, content_transformer(tolower))

# Remove excess white spaces between words.
docs <- tm_map(docs, stripWhitespace)

# Remove stopwords such as "a", "the", etc.
docs <- tm_map(docs, removeWords, stopwords("english"))

# Find the sentiment of the document
washington_feels <- get_sentiment(get_tokens(docs[[1]]$content, pattern = "\\W"), method="syuzhet")
get_sentiment(get_tokens(docs[[1]]$content, pattern = "\\W"), method="syuzhet")
```

```
##   [1]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.50  0.00 -0.25
##  [12]  0.00  0.00  0.80  0.00  0.00  0.00  0.00 -0.75 -0.75 -0.60  0.00
##  [23]  0.00  0.00  0.00  0.00 -0.60  0.00  0.00  0.00  0.00  0.00  0.00
##  [34] -0.50  0.00  0.00  0.00  0.50  0.00  0.00  0.00  0.80  0.25  1.00
##  [45]  0.00  0.50  0.50  0.00  0.00  0.60  0.00  0.80  0.00  0.00  0.00
##  [56]  0.00  0.00  0.00  0.00  0.25  0.00 -0.75  0.00  0.00  0.00  0.00
##  [67]  0.00  0.00  0.00 -0.25 -0.60  0.00 -0.25  0.25  0.00 -0.80  0.50
##  [78]  0.75  0.00  0.25 -0.50  0.00  0.00 -0.50  0.00  0.00  0.00  1.00
##  [89]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.50  0.00  0.00
## [100]  0.00  0.00  0.00  0.00  0.00  0.80  0.00  0.00  0.00  0.00  0.00
## [111]  0.00  0.00  0.00  0.50  0.00  0.00  0.00  0.00  0.00  0.00  0.00
## [122]  0.00 -0.50  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.75
## [133] -0.50  0.00 -0.80  0.80 -0.50  0.50  0.00  0.00  0.00  0.00 -0.75
## [144] -0.80  0.40 -0.10  0.00  0.00  0.00 -0.75  0.00  0.00 -0.50  0.00
## [155]  0.00  0.40 -0.25  0.00  0.00  0.00 -0.75  0.00  0.00  0.00 -0.75
## [166]  0.00  0.40 -0.25  0.00  0.00  0.00  0.00  0.00  0.00  0.00 -0.50
## [177] -1.00  0.50  0.00 -1.00 -0.50 -0.75  0.00  0.00 -0.50  0.40  0.75
## [188]  0.00  0.40  0.00  0.00  0.50  0.00 -0.25  0.00  0.00  0.00  0.75
## [199]  0.00  0.00  1.00  0.60 -0.25  0.00  0.80  0.00  0.25  0.80  0.40
## [210]  0.00  0.50  0.50  0.50  0.00  0.00  0.00  0.00  0.00  0.25  0.75
## [221]  0.00 -0.50  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
## [232]  0.00  0.50  0.00 -0.25  0.00 -0.50  0.00 -0.75  0.40  0.00  0.00
## [243]  0.00  0.00 -0.75  0.00  0.50  0.00  0.00  0.75  0.00  0.00  0.75
## [254]  0.00  0.00  0.80  0.00  0.50  0.00  0.00  0.00  0.00  0.00  0.00
## [265]  0.00  0.00  0.40  0.00  0.50  0.00  0.00  0.00 -0.50  0.00  0.80
## [276]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.80  0.00  0.00  0.00
## [287]  0.00 -0.75  0.00  0.00  0.00  0.80 -0.50  0.00  0.60  0.00  0.00
## [298]  0.00  0.00 -0.25  0.00  0.00  0.00 -0.25  0.00  0.50  0.00  0.60
## [309]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.25  0.00  0.00  0.00
## [320]  0.00  0.00  0.00  0.50  0.00  0.00  0.00  0.00 -0.50 -0.40  0.00
## [331]  0.00  1.00  0.00  0.00  0.80  0.00  0.00  0.00  0.00  0.00  0.00
## [342]  0.50  0.00  0.00  0.60  0.60  0.00  0.00  0.00  0.25  0.00  0.00
## [353]  1.00  0.00  0.00  0.75  0.00  0.50  0.75  0.00  0.80  0.00  0.00
## [364]  0.50  0.00 -0.50  0.00  0.00  0.80  0.25  0.60  0.75  0.00  0.00
##  [375]  0.00  0.25  0.00  0.00  0.00  0.00  0.00  1.00  0.00  0.00  0.75
##  [386]  0.00  0.80  0.50  0.00  0.00  0.00  0.40  0.00  0.00  1.00  0.75
##  [397]  0.50  0.00  0.25  0.50  0.00  0.00  0.00  0.00 -0.50  0.00  0.50
##  [408]  0.80  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
##  [419]  0.00  0.40  0.75  0.00  0.00  0.60  0.00  0.00  0.00  0.00  0.00
##  [430]  0.00  0.00  0.00  0.50  0.50  0.75  0.00 -0.60  0.00  0.40  0.00
##  [441]  0.00  0.00  0.50  0.00  0.00  0.75  0.50  0.00  0.00  0.25  0.00
##  [452]  0.50  0.00  0.75  0.00  0.80 -0.50  0.00  0.00  0.00  0.00  0.00
##  [463] -0.50  0.00  0.00  0.00  0.00  0.40  0.00  0.60  0.50  0.00  0.00
##  [474]  0.00  0.00  0.00  0.00  1.00  0.00  0.00  0.00  0.00  0.00  0.25
##  [485]  0.00  0.80  0.00  0.00  0.00  0.00  0.75  0.75 -0.50  0.00  0.80
##  [496]  0.40  0.50  0.00  1.00  0.00 -0.50  0.00 -0.25  0.00 -0.75  0.00
##  [507]  0.00  0.00 -0.25 -0.80  0.80  0.50  0.75  0.75  0.00  0.00  0.00
##  [518]  0.50  0.25  0.00  0.00  0.00  0.00  0.00  0.00 -0.75 -0.50  0.00
##  [529]  0.40  0.00  0.00  0.00  0.00  0.75  0.00  0.00  0.00  0.80 -0.25
##  [540]  0.00  0.00  0.00 -0.75 -0.50 -1.00 -0.10  0.00 -0.50  0.00  0.40
##  [551]  0.00  0.00 -0.25  0.00  0.00 -0.25  0.00  0.40  0.00  0.00  0.40
##  [562]  0.60  0.00  0.00  0.00  0.00  0.40  0.00  0.75  0.00  0.00  0.25
##  [573]  0.50  0.75  0.00 -0.50 -0.50  0.00 -0.50  0.00  0.00  0.00  0.75
##  [584] -1.00  0.50  0.75  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.50
##  [595]  0.00  0.00  0.50  1.00  0.00  0.50  0.00  0.00  0.50  0.00  0.40
##  [606]  0.75  0.25  0.75  0.50  0.80  0.75  0.00  0.50  0.00  0.50  0.00
##  [617]  0.75  0.00  0.00  0.00  1.00 -0.25  0.00  0.80  0.10  0.75  0.00
##  [628]  0.00  0.00  0.40  0.00  0.00  0.00  0.75  0.50  0.00  0.00  0.75
##  [639]  0.40  0.75  0.60  0.00  0.00  0.00  1.00  0.60  0.00  0.00  0.00
##  [650]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.60  0.00  0.00  0.00
##  [661]  0.00 -0.50  0.00  0.75  0.00  0.50 -0.25  0.00  0.00  0.00  0.00
##  [672]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.80  0.00  0.00  0.00
##  [683]  0.00  0.00  0.00  0.00  0.50  0.00  0.60  0.60  0.00  0.00  0.00
##  [694]  0.00  0.00  0.50  0.00 -0.50  0.00  0.00  0.00  0.00  0.00  0.50
##  [705]  0.50  0.00  1.00  0.00  0.00  0.50  0.00  0.50  0.80  0.50  0.00
##  [716]  0.00  0.00  0.80  0.00  0.00  0.00  0.00  0.75  0.25  0.00  0.00
##  [727]  0.00  0.00 -0.25  0.80  0.00  0.00  0.00  0.00  0.50  0.00  0.00
##  [738]  0.50  1.00  0.75  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
##  [749]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.75  0.75
##  [760]  0.00  0.00  0.00  0.75  0.00  0.00  0.80 -0.40  0.00  0.50  0.00
##  [771]  0.75  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.60  0.50  0.00
##  [782]  0.00  0.00  0.00  0.25  0.00  0.75  0.00  0.00  0.00  0.00  0.00
##  [793]  0.00  0.00  0.00  0.60 -0.50  0.50  0.00  0.50  0.00  0.50 -0.10
##  [804]  0.00 -0.75 -1.00 -0.25  0.00  0.50 -0.75  0.00  0.00  0.75  0.00
##  [815]  0.00  1.00  0.25  0.00  0.00  0.00  0.75  0.00  0.00 -0.25  0.00
##  [826]  0.50  0.25  0.50  1.00  0.25  0.00  0.00  0.00  0.00  0.00  0.00
##  [837]  0.00  0.00  0.75  0.00  0.00  0.75 -0.60  0.80  0.00  0.75  0.25
##  [848]  0.00  0.00  0.00  0.00  0.50  0.00 -0.50  0.00  0.00  0.00 -0.50
##  [859]  0.00  0.00  0.00  0.00  0.80  0.00  0.00  0.00  0.00  0.00  0.00
##  [870]  0.60  0.00  0.00  0.60  0.00  0.00  0.00  0.00  0.00  0.00  0.00
##  [881]  0.00  0.75  0.60  0.60  0.50  0.00  1.00  0.50  0.00  0.00  1.00
##  [892]  0.50  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.75  0.25
##  [903]  0.25  0.00  0.00  0.00 -0.25  0.00  0.00 -0.50 -0.75  0.00  0.00
##  [914]  0.50  0.00  0.10  0.00  0.50  0.00  0.00  0.00  0.00  0.75  0.00
##  [925]  0.80  0.00  0.40  0.00  0.00  0.00  0.00  1.00  0.50 -0.50  0.00
##  [936] -0.50  0.00  1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.75
##  [947]  0.50  0.50  0.00  0.00  0.75  1.00  0.50  0.25  0.00  0.00  0.00
##  [958]  0.00  0.00 -0.75  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
##  [969]  0.60  0.10  0.00  0.80  0.80  0.00  0.00  0.00  0.00  0.00  0.00
##  [980] -0.60  0.75  0.00  0.00  0.60  0.75  0.00  0.00  0.00  0.50  0.00
##  [991]  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.50  0.50  0.00  0.00
## [1002]  0.50  0.80  0.00  0.00  0.00  0.60  0.00  0.00  0.80  0.00 -0.60
## [1013]  0.25  0.00 -0.40  0.00  0.00  0.00  0.00  0.00  0.50  0.00  0.00
## [1024]  0.50  0.00  1.00  0.00  0.50  0.00  0.00  0.00  0.50  0.00  0.50
## [1035]  0.00  0.50 -0.75  0.00  0.00  0.00  0.50  0.00 -0.25  0.00 -0.25
## [1046]  0.00  0.50  0.40  0.00  0.00  0.50  0.10  0.00  0.00  1.00  0.80
## [1057]  0.00  0.00  0.75  0.50  0.40  0.75  0.00  0.00 -0.50  0.00  0.00
## [1068]  0.75  0.00  0.80  0.00  0.60
```

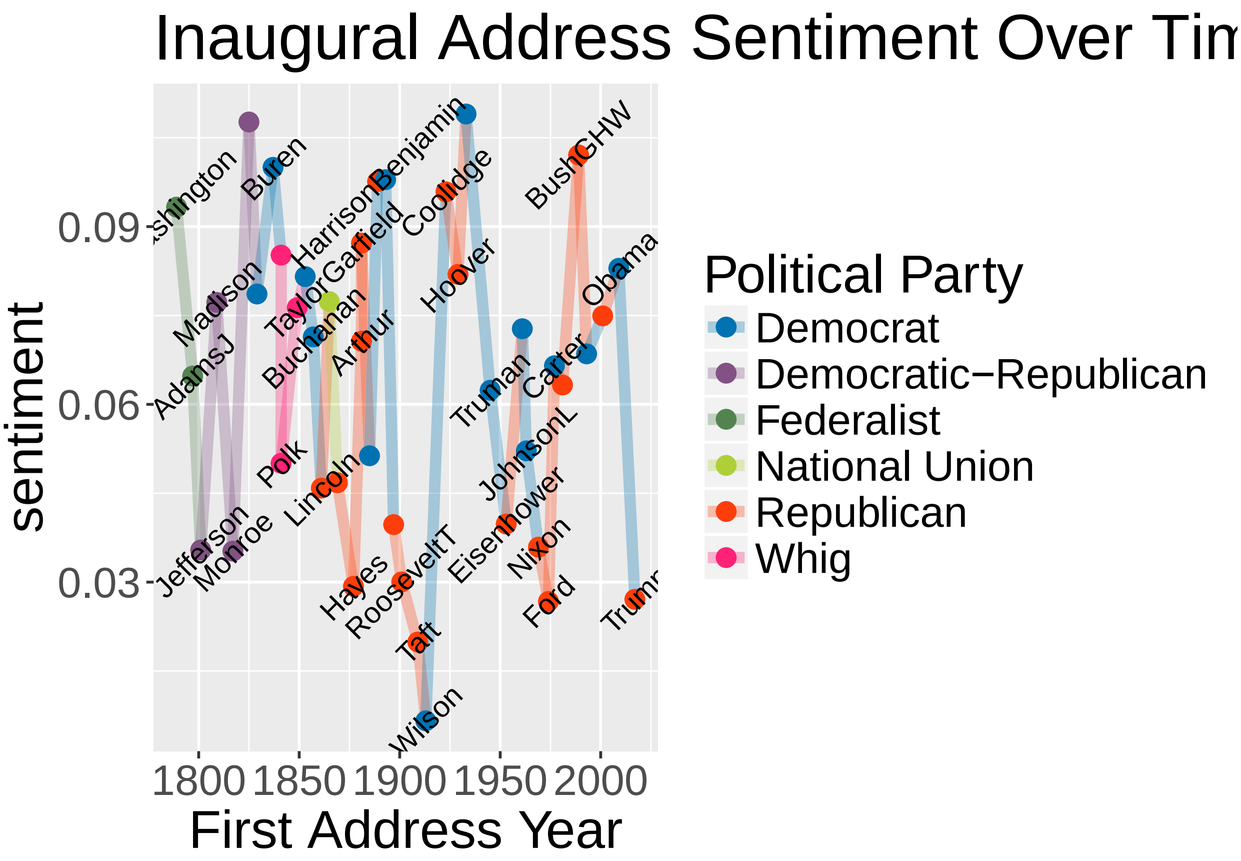
```
#Plot the distrubution of sentiment
ggplot() + geom_density(aes(x = washington_feels))
```



```
#Mean sentiment with stopwords is 0.0507763, sentiment increases to 0.1097948 for Washington using Syuz
mean(get_sentiment(get_tokens(docs[[1]]$content, pattern = "\\W"), method="syuzhet"))
```

```
## [1] 0.1097948
```

```
word_freq_dates <- word_freq_dates %>% arrange(start_year)
#Calculate sentiment for all presidents and add to the word_freq_dates dataframe
for (i in 1:42) {
  word_freq_dates$sentiment[i] <- mean(get_sentiment(get_tokens(docs[[i]]$content, pattern = "\\W"), me
}

#Plot sentiment over time
ggplot(word_freq_dates, aes(x = start_year, y = sentiment,label = Last.Name)) + geom_line(aes(x = start_
```



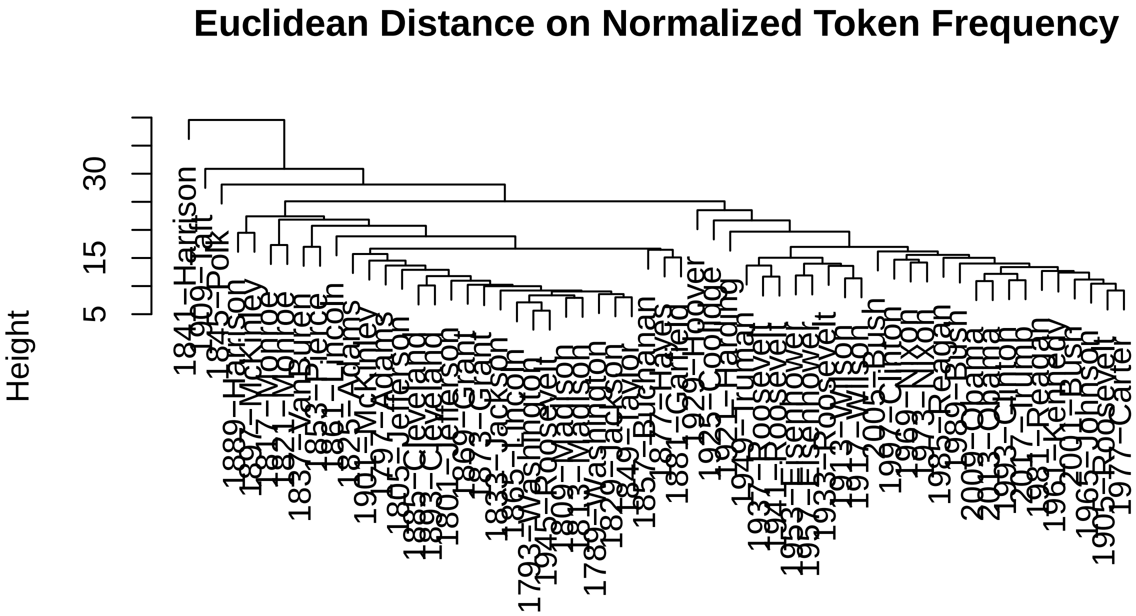
Lastly, perhaps look at how speeches appear similar

```
#https://cran.r-project.org/web/packages/quanteda/vignettes/quickstart.html

presDfm <- dfm(data_corpus_inaugural, tolower= TRUE, remove = stopwords("english"), removePunct = TRUE,
```

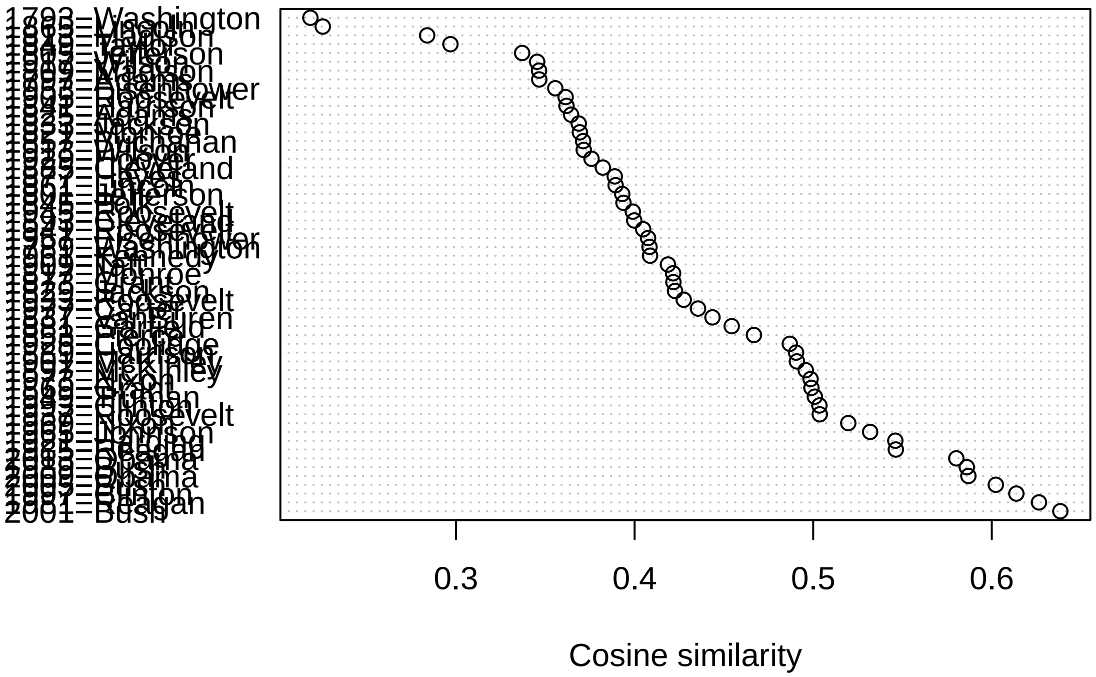
```
## Creating a dfm from a corpus ...

##    ... lowercasing

##    ... tokenizing

##    ... found 58 documents, 9,273 features

## ...

## removed 135 features, from 174 supplied (glob) feature types

## ... stemming features (English)

## , trimmed 3801 feature variants

##    ... created a 58 x 5,337 sparse dfm
##    ... complete.
## Elapsed time: 0.09 seconds.
```

```
presDfm <- dfm_trim(presDfm, min_count=5, min_docfreq=.25)
# hierarchical clustering - get distances on normalized dfm
presDistMat <- dist(as.matrix(dfm_weight(presDfm, "tfidf")))
# hierarchical clustering the distance object
presCluster <- hclust(presDistMat)
```

```
# label with document names
presCluster$labels <- docnames(presDfm)
# plot as a dendrogram
plot(presCluster, xlab = "", sub = "", main = "Euclidean Distance on Normalized Token Frequency")
```



```
#Find how similar Trumps Speech is to Clinton 1993
presDfm <- dfm(corpus_subset(data_corpus_inaugural),
               remove = stopwords("english"),
               stem = TRUE, removePunct = TRUE, removeSymbols = TRUE, removeNumbers = TRUE)
trumpSimil <- textstat_simil(presDfm, c("2017-Trump" , "1997-Clinton"), n = NULL,
                             margin = "documents", method = "cosine")
dotchart(as.list(trumpSimil)$"2017-Trump", xlab = "Cosine similarity")
```


1793–Washington
2001–Bush
0.3
0.4
0.5
0.6
Cosine similarity