NLP Project Report

Najada Feimi, Elnaz Khaveh, Hamed Vaheb

12/29/2022

Contents

Import Libraries **2**

Introduction **2**

Describe Dataset . . . 2

Goal and Procedure . . . 5

Preprocessing **6**

Remove punctuation . . . 6

Remove special characters . . . 6

Remove numbers . . . 6

Converting to lowercase . . . 7

Remove "stopwords" . . . 7

Remove particular stopwords . . . 8

Retain compouned words . . . 8

Strip unnecesary whitespace . . . 8

Type Check . . . 9

Create Doc Term Matrix . . . 9

Organize by frequency . . . 9

Remove sparse words . . . 10

Word Frequency **10**

Plot . . . 11

Relationships Between Terms **12**

Clustering by Term Similarity **14**

Hierarchal Clustering . . . 14

K-means Clustering . . . 16

Import Libraries

```
library(knitr)
library(dplyr)
library(ggplot2)
library(broom)
library(janitor)
library(renv)
library(purrr)
library(tm)
library(SnowballC)
library(RColorBrewer)
library(ggplot2)
library(wordcloud)
library(biclust)
library(cluster)
library(igraph)
library(fpc)
library(magrittr)

library(textreuse)
library(slam)

#packages <- c("tm", "SnowballCC", "RColorBrewer", "ggplot2", "wordcloud", "biclust", "cluster", "igrap
#install.packages(packages, dependencies = TRUE)
```

Introduction

Describe Dataset

The dataset used for this project is president speeches ob.tained from this link.

Using the following script in Python, we first created a dataframe of the website's speeches:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Scrapes transcripts for inaugural addresses


def get_urls(url):
    '''Returns list of transcript urls'''

    page = requests.get(url).text
    soup=BeautifulSoup(page, 'lxml')
    url_table = soup.find("table", class_='table').find_all("a")
    return [u["href"] for u in url_table]
```

```
urls = get_urls("https://www.presidency.ucsb.edu/documents/presidential-documents-archive-guidebook/inau

transcripts = pd.DataFrame()

def get_transcripts(urls, transcripts):
    for u in urls:
        page = requests.get(u).text
        soup = BeautifulSoup(page, 'lxml')
        t_president = soup.find("h3", class_="diet-title").text
        t_year = soup.find("span", class_="date-display-single").text.split(',')[1].strip()
        t_content = soup.find("div", class_="field-docs-content").text
        record = {
            'president' : t_president,
            'year' : t_year,
            'content' : t_content
        }
        transcripts = transcripts.append(record, ignore_index=True)
    return transcripts

data = get_transcripts(urls,transcripts)
data.to_csv("us_presidents_transcripts.csv", sep="|")
```

In what follows, we load the dataframe:

```
df <- read.csv("https://raw.githubusercontent.com/berserkhmdvhb/MADS-NLP/main/data/presidents-speech.csv
```

```
df |> dplyr::glimpse()
```

```
## Rows: 59
## Columns: 4
## $ X         <int> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17~
## $ president <chr> "George Washington", "George Washington", "John Adams", "Tho~
## $ year      <int> 1789, 1793, 1797, 1801, 1805, 1809, 1813, 1817, 1821, 1825, ~
## $ content   <chr> "\nFellow-Citizens of the Senate and of the House of Represe~
```

```
df |> summary()
```

```
##        X         president              year        content
##  Min.   : 0.0   Length:59          Min.   :1789   Length:59
##  1st Qu.:14.5   Class :character   1st Qu.:1847   Class :character
##  Median :29.0   Mode  :character   Median :1905   Mode  :character
##  Mean   :29.0                      Mean   :1905
##  3rd Qu.:43.5                      3rd Qu.:1963
##  Max.   :58.0                      Max.   :2021
```

In what follows, text files are generated from each row of dataframe and are stored in "texts" folder:

```
#presidents <- df[["president"]]|> unique() |>as.list()
```

```
for(i in 1:nrow(df)) {       # for-loop over rows
  df_i <- df[i, ]
  name <- df_i$president
  year <- df_i$year
  text <- df_i$content
  file_name <- paste(as.character(year),
                     as.character(name),
                     sep="-")
  file_name <- paste(file_name, ".txt",
                     sep="")
  loc <- paste("/home/hamed/Documents/R/MADS-NLP/data/texts/", file_name, sep="")
  #writeLines(text, loc)
}
```

```
loc <- "/home/hamed/Documents/R/MADS-NLP/data/texts"
docs <- tm::VCorpus(DirSource(loc))
summary(docs)
```

```
##                                Length Class             Mode
## 1789-George Washington.txt     2      PlainTextDocument list
## 1793-George Washington.txt     2      PlainTextDocument list
## 1797-John Adams.txt            2      PlainTextDocument list
## 1801-Thomas Jefferson.txt      2      PlainTextDocument list
## 1805-Thomas Jefferson.txt      2      PlainTextDocument list
## 1809-James Madison.txt         2      PlainTextDocument list
## 1813-James Madison.txt         2      PlainTextDocument list
## 1817-James Monroe.txt          2      PlainTextDocument list
## 1821-James Monroe.txt          2      PlainTextDocument list
## 1825-John Quincy Adams.txt     2      PlainTextDocument list
## 1829-Andrew Jackson.txt        2      PlainTextDocument list
## 1833-Andrew Jackson.txt        2      PlainTextDocument list
## 1837-Martin van Buren.txt      2      PlainTextDocument list
## 1841-William Henry Harrison.txt 2     PlainTextDocument list
## 1845-James K. Polk.txt         2      PlainTextDocument list
## 1849-Zachary Taylor.txt        2      PlainTextDocument list
## 1853-Franklin Pierce.txt       2      PlainTextDocument list
## 1857-James Buchanan.txt        2      PlainTextDocument list
## 1861-Abraham Lincoln.txt       2      PlainTextDocument list
## 1865-Abraham Lincoln.txt       2      PlainTextDocument list
## 1869-Ulysses S. Grant.txt      2      PlainTextDocument list
## 1873-Ulysses S. Grant.txt      2      PlainTextDocument list
## 1877-Rutherford B. Hayes.txt   2      PlainTextDocument list
## 1881-James A. Garfield.txt     2      PlainTextDocument list
## 1885-Grover Cleveland.txt      2      PlainTextDocument list
## 1889-Benjamin Harrison.txt     2      PlainTextDocument list
## 1893-Grover Cleveland.txt      2      PlainTextDocument list
## 1897-William McKinley.txt      2      PlainTextDocument list
## 1901-William McKinley.txt      2      PlainTextDocument list
## 1905-Theodore Roosevelt.txt    2      PlainTextDocument list
## 1909-William Howard Taft.txt   2      PlainTextDocument list
## 1913-Woodrow Wilson.txt        2      PlainTextDocument list
## 1917-Woodrow Wilson.txt        2      PlainTextDocument list
## 1921-Warren G. Harding.txt     2      PlainTextDocument list
```

```
## 1925-Calvin Coolidge.txt        2      PlainTextDocument list
## 1929-Herbert Hoover.txt         2      PlainTextDocument list
## 1933-Franklin D. Roosevelt.txt  2      PlainTextDocument list
## 1937-Franklin D. Roosevelt.txt  2      PlainTextDocument list
## 1941-Franklin D. Roosevelt.txt  2      PlainTextDocument list
## 1945-Franklin D. Roosevelt.txt  2      PlainTextDocument list
## 1949-Harry S. Truman.txt        2      PlainTextDocument list
## 1953-Dwight D. Eisenhower.txt   2      PlainTextDocument list
## 1957-Dwight D. Eisenhower.txt   2      PlainTextDocument list
## 1961-John F. Kennedy.txt        2      PlainTextDocument list
## 1965-Lyndon B. Johnson.txt      2      PlainTextDocument list
## 1969-Richard Nixon.txt          2      PlainTextDocument list
## 1973-Richard Nixon.txt          2      PlainTextDocument list
## 1977-Jimmy Carter.txt           2      PlainTextDocument list
## 1981-Ronald Reagan.txt          2      PlainTextDocument list
## 1985-Ronald Reagan.txt          2      PlainTextDocument list
## 1989-George Bush.txt            2      PlainTextDocument list
## 1993-William J. Clinton.txt     2      PlainTextDocument list
## 1997-William J. Clinton.txt     2      PlainTextDocument list
## 2001-George W. Bush.txt         2      PlainTextDocument list
## 2005-George W. Bush.txt         2      PlainTextDocument list
## 2009-Barack Obama.txt           2      PlainTextDocument list
## 2013-Barack Obama.txt           2      PlainTextDocument list
## 2017-Donald J. Trump.txt        2      PlainTextDocument list
## 2021-Joseph R. Biden.txt        2      PlainTextDocument list
```

```
inspect(docs[1])
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 1
##
## [[1]]
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 8617
```

```
writeLines(as.character(docs[1]))
```

```
## list(list(content = c("", "Fellow-Citizens of the Senate and of the House of Representatives:", "Amo
## "Such being the impressions under which I have, in obedience to the public summons, repaired to the
## "By the article establishing the executive department it is made the duty of the President \"to reco
## "Besides the ordinary objects submitted to your care, it will remain with your judgment to decide ho
## "To the foregoing observations I have one to add, which will be most properly addressed to the House
## "Having thus imparted to you my sentiments as they have been awakened by the occasion which brings u
## ""), meta = list(author = character(0), datetimestamp = list(sec = 48.8706912994385, min = 46, hour =
## list()
## list()
```

Goal and Procedure

This project is dedicated to investigating text similarity between speeches from different presidents of US during various years, starting from 1789 and ending with 2021.

In Preprocessing section, numerous text mining tasks are implemented on all the docs.

In Word Frequency section, frequency of different terms in documents are analyzed and visualized.

Preprocessing

Remove punctuation

```
docs <- tm::tm_map(docs,removePunctuation)
writeLines(as.character(docs[1]))
```

```
## list(list(content = c("", "FellowCitizens of the Senate and of the House of Representatives", "Among
## "Such being the impressions under which I have in obedience to the public summons repaired to the pr
## "By the article establishing the executive department it is made the duty of the President to recomm
## "Besides the ordinary objects submitted to your care it will remain with your judgment to decide how
## "To the foregoing observations I have one to add which will be most properly addressed to the House
## "Having thus imparted to you my sentiments as they have been awakened by the occasion which brings us
## ""), meta = list(author = character(0), datetimestamp = list(sec = 48.8706912994385, min = 46, hour =
## list()
## list()
```

Remove special characters

```
for (j in seq(docs)) {
    docs[[j]] <- gsub("/", " ", docs[[j]])
    docs[[j]] <- gsub("@", " ", docs[[j]])
    docs[[j]] <- gsub("\\|", " ", docs[[j]])
    docs[[j]] <- gsub("\u2028", " ", docs[[j]])  # This is an ascii character that did not translate, s
}
writeLines(as.character(docs[1]))
```

```
## list(c("", "FellowCitizens of the Senate and of the House of Representatives", "Among the vicissitud
## "Such being the impressions under which I have in obedience to the public summons repaired to the pr
## "By the article establishing the executive department it is made the duty of the President to recomm
## "Besides the ordinary objects submitted to your care it will remain with your judgment to decide how
## "To the foregoing observations I have one to add which will be most properly addressed to the House
## "Having thus imparted to you my sentiments as they have been awakened by the occasion which brings us
## ""))
## list()
## list()
```

Remove numbers

```
docs <- tm::tm_map(docs, removeNumbers)
writeLines(as.character(docs[1]))
```

```
## list(c("", "FellowCitizens of the Senate and of the House of Representatives", "Among the vicissitud
## "Such being the impressions under which I have in obedience to the public summons repaired to the pr
## "By the article establishing the executive department it is made the duty of the President to recomm
## "Besides the ordinary objects submitted to your care it will remain with your judgment to decide how
## "To the foregoing observations I have one to add which will be most properly addressed to the House
## "Having thus imparted to you my sentiments as they have been awakened by the occasion which brings us
## ""))
## list()
## list()
```

Converting to lowercase

```
docs <- tm::tm_map(docs, tolower)
docs <- tm::tm_map(docs, PlainTextDocument)
DocsCopy <- docs
writeLines(as.character(docs[1]))
```

```
## list(list(content = c("", "fellowcitizens of the senate and of the house of representatives", "among
## "such being the impressions under which i have in obedience to the public summons repaired to the pr
## "by the article establishing the executive department it is made the duty of the president to recomm
## "besides the ordinary objects submitted to your care it will remain with your judgment to decide how
## "to the foregoing observations i have one to add which will be most properly addressed to the house
## "having thus imparted to you my sentiments as they have been awakened by the occasion which brings us
## ""), meta = list(author = character(0), datetimestamp = list(sec = 49.0042490959167, min = 46, hour =
## list()
## list()
```

Remove "stopwords"

```
# For a list of the stopwords, see:
length(stopwords("english"))
```

```
## [1] 174
```

```
docs <- tm::tm_map(docs, removeWords, stopwords("english"))
docs <- tm::tm_map(docs, PlainTextDocument)
writeLines(as.character(docs[1]))
```

```
## list(list(content = c("", "fellowcitizens  senate  house  representatives", "among  vicissitudes
## "  impressions  obedience  public summons repaired  present station  peculiarly improper  o
## "  article establishing  executive department  made  duty  president  recommend  consideration  m
## "besides  ordinary objects submitted  care  will remain  judgment  decide  far  exercise  occasio
## "  foregoing observations  one  add  will  properly addressed  house representatives  concerns
## " thus imparted  sentiments  awakened  occasion  brings us together  shall take  present leave
## ""), meta = list(author = character(0), datetimestamp = list(sec = 49.2290160655975, min = 46, hour =
## list()
## list()
```

Remove particular stopwords

```
#docs <- tm::tm_map(docs, removeWords, c("syllogism", "tautology"))
# Just remove the words "syllogism" and "tautology".
# These words don't actually exist in these texts. But this is how you would remove them if they had.
```

Retain compouned words

If you wish to preserve a concept is only apparent as a collection of two or more words, then you can combine them or reduce them to a meaningful acronym before you begin the analysis. Here, I am using examples that are particular to qualitative data analysis.

```
for (j in seq(docs))
{
  docs[[j]] <- gsub("fake news", "fake_news", docs[[j]])
  docs[[j]] <- gsub("inner city", "inner-city", docs[[j]])
  docs[[j]] <- gsub("politically correct", "politically_correct", docs[[j]])
}
docs <- tm_map(docs, PlainTextDocument)
```

Remove common word endings Common words ending e.g. “ing”, “es”, “s”

```
## Note: I did not run this section of code for this particular example.
docs_st <- tm_map(docs, stemDocument)
docs_st <- tm_map(docs_st, PlainTextDocument)
writeLines(as.character(docs_st[1])) # Check to see if it worked.
```

```
## list(list(content = c("", "fellowcitizen senat hous repres", "among vicissitud incid life event fill
## "impress obedi public summon repair present station peculiar improp omit first offici act fervent su
## "articl establish execut depart made duti presid recommend consider measur shall judg necessari expe
## "besid ordinari object submit care will remain judgment decid far exercis occasion power deleg fifth
## "forego observ one add will proper address hous repres concern will therefor brief possibl first hon
## ""), meta = list(author = character(0), datetimestamp = list(sec = 49.3727238178253, min = 46, hour =
## list()
## list()
```

```
# docs <- docs_st
```

Strip unnecesary whitespace

```
docs <- tm_map(docs, stripWhitespace)
writeLines(as.character(docs[1]))
```

```
## list(list(content = c("", "fellowcitizens senate house representatives", "among vicissitudes incident
## " impressions obedience public summons repaired present station peculiarly improper omit first offic
## " article establishing executive department made duty president recommend consideration measures shal
## "besides ordinary objects submitted care will remain judgment decide far exercise occasional power d
## " foregoing observations one add will properly addressed house representatives concerns will therefo
```

```
## " thus imparted sentiments awakened occasion brings us together shall take present leave without reso
##     sec = 49.2749984264374, min = 46, hour = 18, mday = 29, mon = 11, year = 122, wday = 4, yday = 3
## list()
## list()
```

Type Check

Be sure to use the following script once you have completed preprocessing. This tells R to treat the preprocessed documents as text documents.

```
docs <- tm::tm_map(docs, stripWhitespace)
writeLines(as.character(docs[1]))
```

```
## list(list(content = c("", "fellowcitizens senate house representatives", "among vicissitudes incident
## " impressions obedience public summons repaired present station peculiarly improper omit first offici
## " article establishing executive department made duty president recommend consideration measures shal
## "besides ordinary objects submitted care will remain judgment decide far exercise occasional power do
## " foregoing observations one add will properly addressed house representatives concerns will therefor
## " thus imparted sentiments awakened occasion brings us together shall take present leave without reso
##     sec = 49.2749984264374, min = 46, hour = 18, mday = 29, mon = 11, year = 122, wday = 4, yday = 3
## list()
## list()
```

Create Doc Term Matrix

```
dtm <- tm::DocumentTermMatrix(docs)
dtm
```

```
## <<DocumentTermMatrix (documents: 59, terms: 9495)>>
## Non-/sparse entries: 40113/520092
## Sparsity           : 93%
## Maximal term length: 23
## Weighting          : term frequency (tf)
```

Storing transpose of matrix

```
tdm <- tm::TermDocumentMatrix(docs)
tdm
```

```
## <<TermDocumentMatrix (terms: 9495, documents: 59)>>
## Non-/sparse entries: 40113/520092
## Sparsity           : 93%
## Maximal term length: 23
## Weighting          : term frequency (tf)
```

Organize by frequency

```
freq <- colSums(as.matrix(dtm))
length(freq)
```

```
## [1] 9495
```

```
ord <- order(freq)
m <- as.matrix(dtm)
dim(m)
```

```
## [1]   59 9495
```

Store the matrix to memory

```
#write.csv(m, file="DocumentTermMatrix.csv")
```

Remove sparse words

```
#  Start by removing sparse terms:
dtms <- removeSparseTerms(dtm, 0.2) # This makes a matrix that is 20% empty space, maximum.
dtms
```

```
## <<DocumentTermMatrix (documents: 59, terms: 25)>>
## Non-/sparse entries: 1298/177
## Sparsity           : 12%
## Maximal term length: 10
## Weighting          : term frequency (tf)
```

Word Frequency

```
freq <- colSums(as.matrix(dtm))
```

Least frequent

```
head(table(freq), 20)
```

```
## freq
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 4154 1404  788  535  359  291  226  177  162  146  112   85   82   90   43   59
##   17   18   19   20
##   49   35   47   38
```

The top number is the frequency with which words appear and the bottom number reflects how many words appear that frequently.

Most frequent:

```
tail(table(freq), 40)
```

```
## freq
## 134 137 138 139 142 143 147 150 155 157 159 171 179 184 185 198 207 210 221 222
##   1   1   2   1   1   1   1   1   1   2   1   1   1   2   1   1   1   1   1   1
## 227 232 240 250 256 267 302 303 304 314 318 337 341 346 373 374 488 567 576 942
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

View a table of the terms we selected when we removed sparse terms in subsection Remove sparse words

```
freq <- sort(colSums(as.matrix(dtm)), decreasing=TRUE)
freq |> head(20)
```

```
##       will     people government        can       upon       must      great
##        942        576        567        488        374        373        346
##        may     states      world      shall    country     nation      every
##        341        337        318        314        304        303        302
##        one      peace        new      power        now     public
##        267        256        250        240        232        227
```

Identify all terms that appear frequently

```
findFreqTerms(dtm, lowfreq=50) |> head(20)
```

```
##  [1] "act"            "action"         "administration" "also"
##  [5] "always"         "america"        "american"       "americans"
##  [9] "among"          "another"        "authority"      "become"
## [13] "believe"        "best"           "better"         "beyond"
## [17] "business"       "called"         "can"            "cause"
```

Another approach to perform the same task:

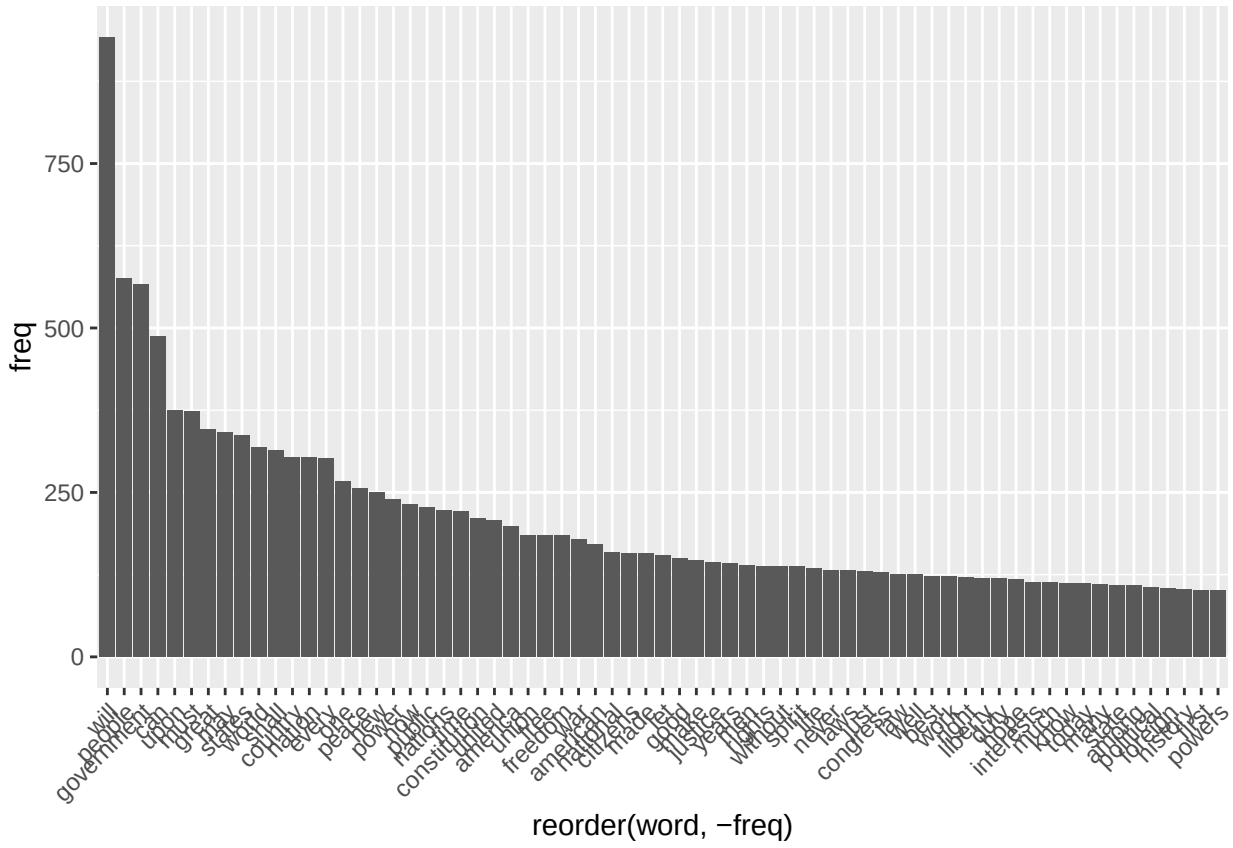
```
wf <- data.frame(word=names(freq), freq=freq)
head(wf)
```

```
##                  word freq
## will             will  942
## people         people  576
## government government  567
## can               can  488
## upon             upon  374
## must             must  373
```

Plot

```
p <- ggplot(subset(wf, freq>100), aes(x = reorder(word, -freq), y = freq)) +
  geom_bar(stat = "identity") +
  theme(axis.text.x=element_text(angle=45, hjust=1))

p
```



Relationships Between Terms

```
tm::findAssocs(dtm, c("government" , "states"), corlimit=0.75)
```

```
## $government
## system
##   0.79
##
## $states
##   powers sections   united
##     0.77     0.76     0.76
```

```
findAssocs(dtms, "government", corlimit=0.70) # specifying a correlation limit of 0.95
```

```
## $government
## states
##   0.75
```

Word Clouds

Plot words that occur at least 25 times.

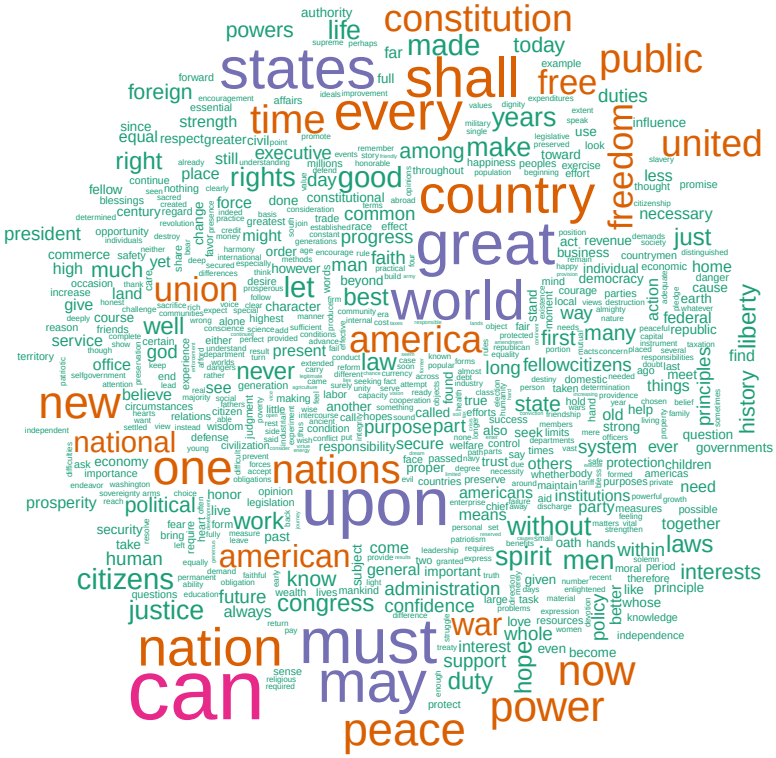
Colorized version:

```
set.seed(142)
wordcloud::wordcloud(names(freq), freq, min.freq=20, scale=c(5, .1), colors=brewer.pal(6, "Dark2"))
```

```
## Warning in wordcloud::wordcloud(names(freq), freq, min.freq = 20, scale = c(5, :
## government could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud::wordcloud(names(freq), freq, min.freq = 20, scale = c(5, :
## people could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud::wordcloud(names(freq), freq, min.freq = 20, scale = c(5, :
## will could not be fit on page. It will not be plotted.
```



Plot words that occur at least 100 times.

```
set.seed(142)
dark2 <- brewer.pal(6, "Dark2")
wordcloud::wordcloud(names(freq), freq, max.words=100, rot.per=0.2, colors=dark2)
```

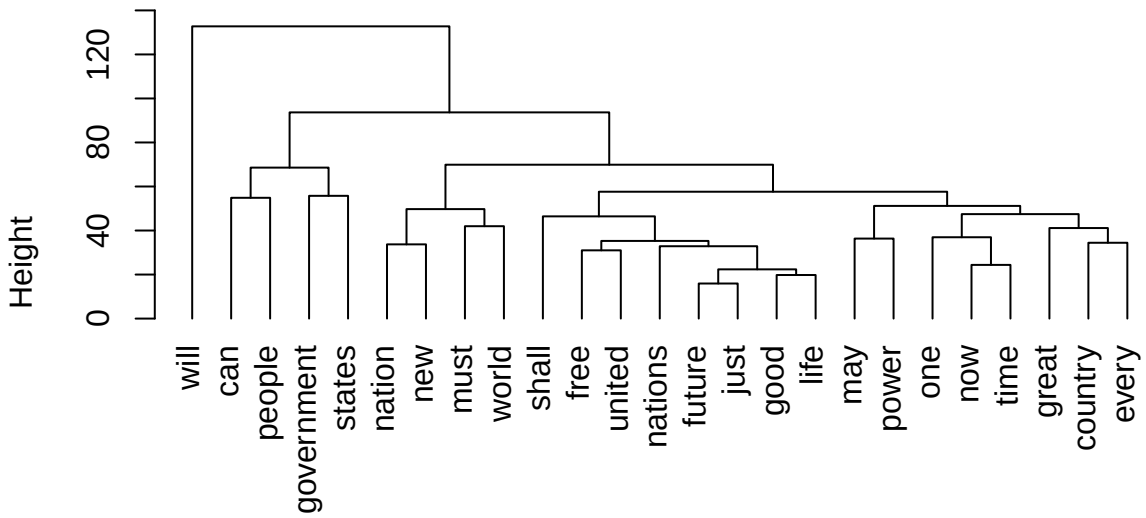
Clustering by Term Similarity

Hierarchal Clustering

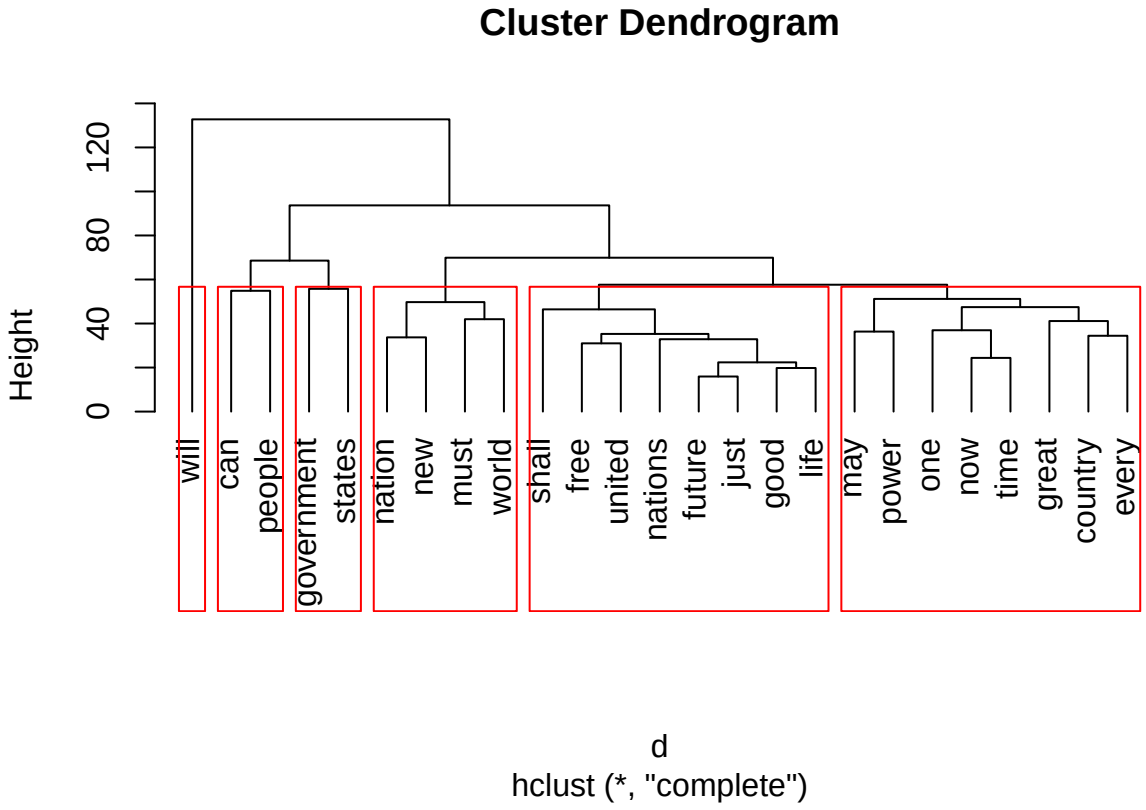
```
d <- dist(t(dtms), method="euclidian")
fit <- hclust(d=d, method="complete")   # for a different look try substituting: method="ward.D"
fit
```

```
##
## Call:
## hclust(d = d, method = "complete")
##
## Cluster method   : complete
## Distance         : euclidean
## Number of objects: 25
```

```
plot(fit, hang=-1)
```

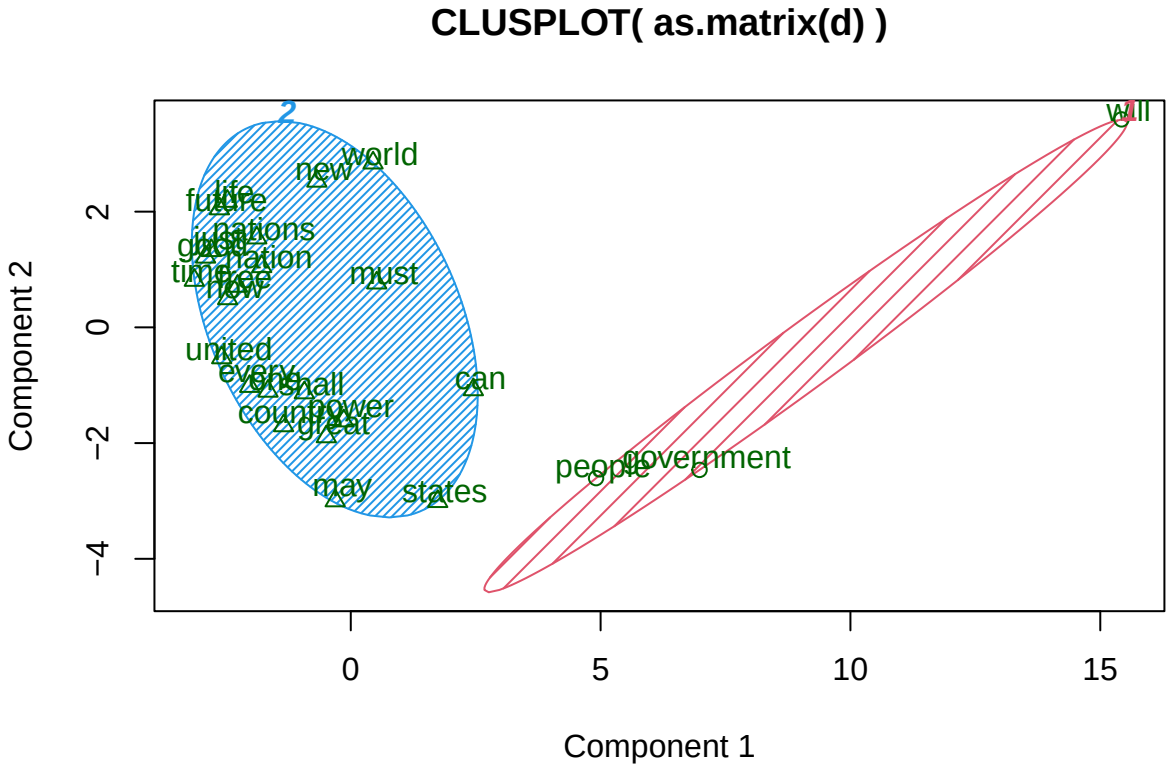



```
plot.new()
plot(fit, hang=-1)
groups <- cutree(fit, k=6)   # "k=" defines the number of clusters you are using
rect.hclust(fit, k=6, border="red") # draw dendogram with red borders around the 6 clusters
```



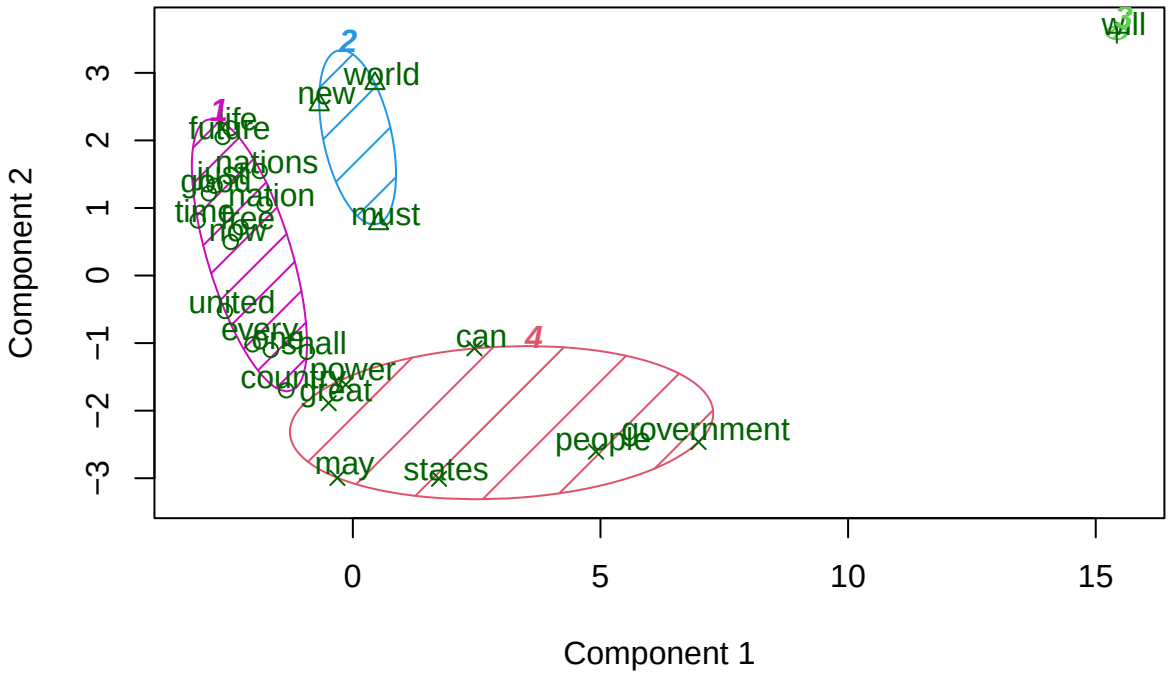
K-means Clustering

```
d <- dist(t(dtms), method="euclidian")
kfit <- kmeans(d, 2)
clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0)
```



These two components explain 77.48 % of the point variability.

```
d <- dist(t(dtms), method="euclidian")
kfit <- kmeans(d, 4)
clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0)
```



Doc Simlarity

//TODO: Perform doc similarity using the textreuse library.

```
#loc <- "/home/hamed/Documents/R/MADS-NLP/data/texts"
#docs <- tm::VCorpus(DirSource(loc))

loc <- "/home/hamed/Documents/R/MADS-NLP/data/texts"
corpus <- TextReuseCorpus(dir=loc)


comparisons <- pairwise_compare(corpus, jaccard_similarity)
compare_df <- pairwise_candidates(comparisons)
compare_df <- as.data.frame(compare_df,
                            col.names = names(compare_df))
#compare_df <- compare_df[order(compare_df$score,decreasing=TRUE)]
compare_df <- compare_df[order(compare_df$score,decreasing=TRUE),]
compare_df |> head(20)
```

```
##                                a                         b      score
## 386            1817-James Monroe        1821-James Monroe 0.02384178
## 491       1825-John Quincy Adams       1845-James K. Polk 0.02251564
## 677  1841-William Henry Harrison       1845-James K. Polk 0.02228458
## 724           1845-James K. Polk      1857-James Buchanan 0.01910381
## 1046    1877-Rutherford B. Hayes   1881-James A. Garfield 0.01898857
## 540           1829-Andrew Jackson      1849-Zachary Taylor 0.01824135
```

```
## 1118       1885-Grover Cleveland    1893-Grover Cleveland 0.01803833
## 860          1857-James Buchanan    1897-William McKinley 0.01773767
## 734           1845-James K. Polk    1897-William McKinley 0.01736912
## 492       1825-John Quincy Adams      1849-Zachary Taylor 0.01730467
## 392            1817-James Monroe       1845-James K. Polk 0.01709736
## 1621          1969-Richard Nixon       1973-Richard Nixon 0.01708706
## 632         1837-Martin van Buren       1845-James K. Polk 0.01702180
## 729           1845-James K. Polk 1877-Rutherford B. Hayes 0.01701129
## 586          1833-Andrew Jackson       1845-James K. Polk 0.01680973
## 857          1857-James Buchanan    1885-Grover Cleveland 0.01643192
## 855          1857-James Buchanan 1877-Rutherford B. Hayes 0.01634117
## 499       1825-John Quincy Adams 1877-Rutherford B. Hayes 0.01601562
## 730           1845-James K. Polk   1881-James A. Garfield 0.01575558
## 442            1821-James Monroe       1845-James K. Polk 0.01548165
```

```
max(dtms)
```

```
## [1] 47
```

```
dim(dtms)
```

```
## [1] 59 25
```

1

Cosine Similarity

```
cosine_dist_mat <- 1 - crossprod_simple_triplet_matrix(dtms)/(sqrt(col_sums(dtms^2) %*% t(col_sums(dtms

cosine_dist_mat
```

```
##             Terms
## Terms              can   country     every      free    future      good
##   can        0.0000000 0.3013170 0.3196871 0.3970037 0.3110346 0.2664495
##   country    0.3013170 0.0000000 0.2134492 0.3652111 0.4106707 0.3386620
##   every      0.3196871 0.2134492 0.0000000 0.4134426 0.2906490 0.3322699
##   free       0.3970037 0.3652111 0.4134426 0.0000000 0.4161580 0.3816995
##   future     0.3110346 0.4106707 0.2906490 0.4161580 0.0000000 0.2954737
##   good       0.2664495 0.3386620 0.3322699 0.3816995 0.2954737 0.0000000
##   government 0.2579230 0.2047350 0.2285687 0.4167750 0.4221402 0.3390539
##   great      0.3042754 0.2390763 0.2096882 0.4302002 0.3422671 0.3008674
##   just       0.3563657 0.3436575 0.2022730 0.4530560 0.2901312 0.4011886
##   life       0.3891276 0.5365314 0.4278774 0.3598682 0.3989592 0.3532778
##   may        0.2454736 0.2091903 0.2500473 0.3684674 0.4220194 0.2937487
##   must       0.2257792 0.3539029 0.3447757 0.3947449 0.2996883 0.3999765
##   nation     0.2810751 0.4035222 0.3253524 0.5014245 0.2368747 0.2633727
##   nations    0.3268656 0.3906216 0.4437372 0.3606915 0.3747685 0.4074177
##   new        0.3658506 0.6269561 0.4465935 0.5652935 0.3712807 0.4494017
##   now        0.2702416 0.3961642 0.3232435 0.4636810 0.3613852 0.3781249
##   one        0.1981736 0.2578042 0.2801983 0.3996113 0.3642686 0.3632012
##   people     0.1876246 0.2312582 0.2164757 0.3129386 0.3493964 0.2671543
##   power      0.3523297 0.3045336 0.3592473 0.4283691 0.4823880 0.3825696
```

```
##   shall      0.3109780 0.3616201 0.3589731 0.3534992 0.4197767 0.3228838
##   states     0.3775741 0.2668935 0.2919410 0.3686965 0.5228231 0.4072595
##   time       0.2443284 0.3741071 0.2302306 0.3895374 0.1702277 0.3522756
##   united     0.3648980 0.2623717 0.2403582 0.3262696 0.4483419 0.3755336
##   will       0.1884575 0.2630200 0.2057079 0.3901750 0.2137291 0.2273583
##   world      0.3037327 0.5509632 0.5317638 0.4174611 0.3732763 0.4578542
##              Terms
## Terms         government     great      just      life       may      must
##   can          0.2579230 0.3042754 0.3563657 0.3891276 0.2454736 0.2257792
##   country      0.2047350 0.2390763 0.3436575 0.5365314 0.2091903 0.3539029
##   every        0.2285687 0.2096882 0.2022730 0.4278774 0.2500473 0.3447757
##   free         0.4167750 0.4302002 0.4530560 0.3598682 0.3684674 0.3947449
##   future       0.4221402 0.3422671 0.2901312 0.3989592 0.4220194 0.2996883
##   good         0.3390539 0.3008674 0.4011886 0.3532778 0.2937487 0.3999765
##   government   0.0000000 0.2538097 0.3284080 0.5131929 0.1897588 0.3346813
##   great        0.2538097 0.0000000 0.2831543 0.4549428 0.2135467 0.3918152
##   just         0.3284080 0.2831543 0.0000000 0.4844806 0.3634034 0.4016780
##   life         0.5131929 0.4549428 0.4844806 0.0000000 0.5740470 0.3630353
##   may          0.1897588 0.2135467 0.3634034 0.5740470 0.0000000 0.3689675
##   must         0.3346813 0.3918152 0.4016780 0.3630353 0.3689675 0.0000000
##   nation       0.4031149 0.3087375 0.3363669 0.2829575 0.4428263 0.3346222
##   nations      0.4886689 0.4644433 0.4336468 0.3544381 0.4583313 0.4073543
##   new          0.5282904 0.4976656 0.4564718 0.3729945 0.6337560 0.3379381
##   now          0.3539362 0.3478060 0.4525142 0.4819565 0.3803697 0.3446933
##   one          0.2253451 0.3185289 0.4083071 0.5308138 0.2428221 0.3131816
##   people       0.1561980 0.2122676 0.3205800 0.4053061 0.1893500 0.2844634
##   power        0.3144943 0.3182054 0.5045919 0.6500729 0.1761881 0.5603424
##   shall        0.2626669 0.3262349 0.3740483 0.4798238 0.2220753 0.4286209
##   states       0.1503715 0.2484336 0.3144041 0.6534657 0.1793855 0.5053126
##   time         0.3544366 0.2798467 0.2648991 0.3886612 0.3363113 0.2596588
##   united       0.2204035 0.1972449 0.3001157 0.5073253 0.2731160 0.4412357
##   will         0.2554974 0.2392831 0.2468342 0.3862783 0.2956313 0.2099962
##   world        0.5681668 0.5193071 0.5327552 0.3016199 0.5944010 0.2184216
##              Terms
## Terms           nation   nations       new       now       one    people
##   can        0.2810751 0.3268656 0.3658506 0.2702416 0.1981736 0.1876246
##   country    0.4035222 0.3906216 0.6269561 0.3961642 0.2578042 0.2312582
##   every      0.3253524 0.4437372 0.4465935 0.3232435 0.2801983 0.2164757
##   free       0.5014245 0.3606915 0.5652935 0.4636810 0.3996113 0.3129386
##   future     0.2368747 0.3747685 0.3712807 0.3613852 0.3642686 0.3493964
##   good       0.2633727 0.4074177 0.4494017 0.3781249 0.3632012 0.2671543
##   government 0.4031149 0.4886689 0.5282904 0.3539362 0.2253451 0.1561980
##   great      0.3087375 0.4644433 0.4976656 0.3478060 0.3185289 0.2122676
##   just       0.3363669 0.4336468 0.4564718 0.4525142 0.4083071 0.3205800
##   life       0.2829575 0.3544381 0.3729945 0.4819565 0.5308138 0.4053061
##   may        0.4428263 0.4583313 0.6337560 0.3803697 0.2428221 0.1893500
##   must       0.3346222 0.4073543 0.3379381 0.3446933 0.3131816 0.2844634
##   nation     0.0000000 0.3911856 0.2389233 0.3628019 0.3765086 0.2904789
##   nations    0.3911856 0.0000000 0.4748058 0.4749950 0.5669782 0.3968177
##   new        0.2389233 0.4748058 0.0000000 0.4120310 0.4726255 0.4367703
##   now        0.3628019 0.4749950 0.4120310 0.0000000 0.3042476 0.2278992
##   one        0.3765086 0.5669782 0.4726255 0.3042476 0.0000000 0.2424710
##   people     0.2904789 0.3968177 0.4367703 0.2278992 0.2424710 0.0000000
##   power      0.4872611 0.6527536 0.6956918 0.5153625 0.2507013 0.2587369
```

```
##   shall      0.4901205 0.4562546 0.5808853 0.3135053 0.4146651 0.2428110
##   states     0.5151173 0.5438493 0.6967915 0.3601278 0.3113200 0.2187172
##   time       0.2190216 0.4210667 0.2539761 0.2155948 0.2571156 0.2440165
##   united     0.4477340 0.4010402 0.6261884 0.3173412 0.3682137 0.2665430
##   will       0.2334926 0.3713594 0.3063816 0.1803612 0.2196700 0.1838601
##   world      0.3381502 0.2622823 0.2473251 0.4499794 0.4556484 0.4291283
##              Terms
## Terms             power     shall    states      time    united      will
##   can        0.3523297 0.3109780 0.3775741 0.2443284 0.3648980 0.1884575
##   country    0.3045336 0.3616201 0.2668935 0.3741071 0.2623717 0.2630200
##   every      0.3592473 0.3589731 0.2919410 0.2302306 0.2403582 0.2057079
##   free       0.4283691 0.3534992 0.3686965 0.3895374 0.3262696 0.3901750
##   future     0.4823880 0.4197767 0.5228231 0.1702277 0.4483419 0.2137291
##   good       0.3825696 0.3228838 0.4072595 0.3522756 0.3755336 0.2273583
##   government 0.3144943 0.2626669 0.1503715 0.3544366 0.2204035 0.2554974
##   great      0.3182054 0.3262349 0.2484336 0.2798467 0.1972449 0.2392831
##   just       0.5045919 0.3740483 0.3144041 0.2648991 0.3001157 0.2468342
##   life       0.6500729 0.4798238 0.6534657 0.3886612 0.5073253 0.3862783
##   may        0.1761881 0.2220753 0.1793855 0.3363113 0.2731160 0.2956313
##   must       0.5603424 0.4286209 0.5053126 0.2596588 0.4412357 0.2099962
##   nation     0.4872611 0.4901205 0.5151173 0.2190216 0.4477340 0.2334926
##   nations    0.6527536 0.4562546 0.5438493 0.4210667 0.4010402 0.3713594
##   new        0.6956918 0.5808853 0.6967915 0.2539761 0.6261884 0.3063816
##   now        0.5153625 0.3135053 0.3601278 0.2155948 0.3173412 0.1803612
##   one        0.2507013 0.4146651 0.3113200 0.2571156 0.3682137 0.2196700
##   people     0.2587369 0.2428110 0.2187172 0.2440165 0.2665430 0.1838601
##   power      0.0000000 0.4360201 0.3234218 0.4166865 0.3998290 0.4231578
##   shall      0.4360201 0.0000000 0.2449347 0.3897523 0.3418936 0.2887096
##   states     0.3234218 0.2449347 0.0000000 0.4248667 0.1442522 0.3265694
##   time       0.4166865 0.3897523 0.4248667 0.0000000 0.3697408 0.1750546
##   united     0.3998290 0.3418936 0.1442522 0.3697408 0.0000000 0.3062891
##   will       0.4231578 0.2887096 0.3265694 0.1750546 0.3062891 0.0000000
##   world      0.7102814 0.5616665 0.7295113 0.3166116 0.5751415 0.3533194
##              Terms
## Terms             world
##   can        0.3037327
##   country    0.5509632
##   every      0.5317638
##   free       0.4174611
##   future     0.3732763
##   good       0.4578542
##   government 0.5681668
##   great      0.5193071
##   just       0.5327552
##   life       0.3016199
##   may        0.5944010
##   must       0.2184216
##   nation     0.3381502
##   nations    0.2622823
##   new        0.2473251
##   now        0.4499794
##   one        0.4556484
##   people     0.4291283
##   power      0.7102814
```

```
##   shall      0.5616665
##   states     0.7295113
##   time       0.3166116
##   united     0.5751415
##   will       0.3533194
##   world      0.0000000
```

```
dtms[,1]
```

```
## <<DocumentTermMatrix (documents: 59, terms: 1)>>
## Non-/sparse entries: 56/3
## Sparsity           : 5%
## Maximal term length: 3
## Weighting          : term frequency (tf)
```

```
cosine_sim <- tcrossprod_simple_triplet_matrix(dtms[,1], dtms[,2])/sqrt(row_sums(dtms[,2]^2) %*% t(row_s

#cosine_sim
```