Problem 3

The American Presidency Project at UCSB has the text from all of the State of the Union speeches by US presidents, in which the president speaks to Congress to report on the situation in the country. We will use web scraping, text formatting and pattern matching to grab the data; and then do some statistical analysis on them.

(a)

From the website, I download the *index.html* file and use pattern matching to pull out the individual URLs for each speech in order to download individual HTML files. Files are converted to UNIX line-ending using *fromdos*.

```
#### Download all html files
system("wget -q -O 'index_pres.html'
        'http://www.presidency.ucsb.edu/sou.php#axzz265cEKp1a'")
system("fromdos index_pres.html")
indexPres <- readLines('index_pres.html',warn= FALSE)
# Get speech text source
patUrl1 <-
'\\s{16}<td width=\\"\\d{2}\\" align=\\"center\\" class=\\"doclist\\"><a href=\\"'
indexPres <- indexPres[grep(patUrl1,indexPres,perl= TRUE)]
indexPres <- sapply(indexPres,
                    function(x){gsub(patUrl1,"",x)},USE.NAMES= FALSE)
patUrl2 <- '\\">\\d{4}<\\/a>(\\*|)<\\/td>'
indexPres <- sapply(indexPres,
                    function(x){gsub(patUrl2,"",x)},USE.NAMES= FALSE)
# Get file id from the source url
patUrl3 <-
'http:\\/\\/www\\.presidency\\.ucsb\\.edu\\/ws\\/index\\.php\\?pid='
fileid <- sapply(indexPres,
                 function(x){gsub(patUrl3,"",x)},USE.NAMES= FALSE)
# Download all files and convert to unix
sapply(1:length(fileid),
       function(i){
       system(paste("wget -q -O '",fileid[i],".html' '",indexPres[i],"'",sep=""));
       system(paste("fromdos ",fileid[i],".html",sep=""))})
```

(b)

For each speech, I use pattern matching to extract the body of the speech while retaining the name of the president and the year of the speech. The function is applied with vector operations using *sapply()*.

For the *speechVec*, text pre-processing is done by

1. Replacing HTML line-end with UNIX ones
2. Removing all the HTML format operators
3. Modifying all the HTML special characters to similar UTF-8 ones

```
# Import all *.html lines
ff <- sapply(fileid, function(x) {
    readLines(paste(x, ".html", sep = ""), warn = FALSE)
})
# Get the president name
patName <- "^<title>(.*?)<\\/title>"
namePres <- ff[grep(patName, ff, perl = TRUE)]
namePres <- sapply(namePres, function(x) {
    gsub(patName, "\\1", x)
}, USE.NAMES = FALSE)
namePres <- sapply(namePres, function(x) {
    return(unlist(strsplit(x, ":"))[1])
}, USE.NAMES = FALSE)
# Get the talk date
patDate <- "^.*<span class=\\\"docdate\\\">(.*?)<\\/span>.*$"
dateTalk <- ff[grep(patDate, ff, perl = TRUE)]
dateTalk <- sapply(dateTalk, function(x) {
    gsub(patDate, "\\1", x)
}, USE.NAMES = FALSE)
dateTalk <- sapply(dateTalk, function(x) {
    gsub("^.*\\s", "", x)
}, USE.NAMES = FALSE)
# Get the speech content text and prune for nice-format print
patText <- "^.*<span class=\\\"displaytext\\\">(.*?)<\\/span>.*$"
speechVec <- ff[grep(patText, ff, perl = TRUE)]
speechVec <- sapply(speechVec, function(x) {
    gsub(patText, "\\1<p>", x)
}, USE.NAMES = FALSE)  # grab speech text
speechVec <- sapply(speechVec, function(x) {
    gsub("(<p.*?>|<\\/p>|<br>)", "\n", x)
}, USE.NAMES = FALSE)  # for line ending
speechVec <- sapply(speechVec, function(x) {
    gsub("<.*?>", "", x)
}, USE.NAMES = FALSE)  # remove all html format
speechVec <- sapply(speechVec, function(x) {
    x <- gsub("&mdash;", " -- ", x)
    x <- gsub(" ", "  ", x)
    x <- gsub("&lsquo;", " ' ", x)
    x <- gsub("&#8226;", " \\. ", x)
    x <- gsub("&lt;", " < ", x)
    x <- gsub("&deg;", " degree ", x)
    x <- gsub("&pound;", " pound ", x)
    x <- gsub("&fra.*?;", " 1/2 ", x)
    x <- gsub("&O.*?;", "O", x)
    x <- gsub("&e.*?;", "e", x)
}, USE.NAMES = FALSE)  # html special char
```

(c)

Each speech is stored as a single character vector with all non-text stripped out. The encoding is converted from WINDOWS-1251 to UTF-8. Meanwhile, the information about the tags of "Laughter" and "Applause" and the number of times it was used are kept as a record for each speech. The *speechVec[i]* will be printed out in a nicely-formatted manner.

```
# Remove audience response tags (laughter & applause)
patLau <- "\\[.*?(Laughter|laughter).*?\\]"
patApp <- "\\[.*?(Applause|applause).*?\\]"
getlauNum <- function(x) {
    if (length(gregexpr(patLau, x, perl = TRUE)[[1]]) == 1 && gregexpr(patLau,
        x, perl = TRUE)[[1]] == -1) {
        return(0)
    } else {
        return(length(gregexpr(patLau, x, perl = TRUE)[[1]]))
    }
}
getappNum <- function(x) {
    if (length(gregexpr(patApp, x, perl = TRUE)[[1]]) == 1 && gregexpr(patApp,
        x, perl = TRUE)[[1]] == -1) {
        return(0)
    } else {
        return(length(gregexpr(patApp, x, perl = TRUE)[[1]]))
    }
}
lauNum <- sapply(speechVec, getlauNum, USE.NAMES = FALSE)
appNum <- sapply(speechVec, getappNum, USE.NAMES = FALSE)
speechVec <- sapply(speechVec, function(x) {
    iconv(x, from = "WINDOWS-1251", to = "UTF-8", sub = " ")
})
speechVec <- sapply(speechVec, function(x) {
    x <- gsub(patLau, "", x, perl = TRUE)
    x <- gsub(patApp, "", x, perl = TRUE)
})
names(speechVec) <- NULL
```

(d)

The collection of speeches is stored in a clean fashion of list elements. This is easy later for plotting variables changes over time.

```
listSpeech <- list()
listSpeech$id <- fileid
listSpeech$name <- namePres
listSpeech$date <- as.integer(dateTalk)
listSpeech$numLaughter <- lauNum
listSpeech$numApplause <- appNum
listSpeech$speech <- speechVec
```

(e) (f)

Words and sentences are extracted from each speech, and are stored as individual elements of a (rather long) character vector. Counts are also done on both words and sentences.

```
# Speech analysis
getWords <- function(x) {
    x <- gsub("'", "", x, perl = TRUE)
    x <- gsub("\\W+", " ", x, perl = TRUE)
    xs <- unlist(strsplit(x, "[ ]+", perl = TRUE))
    return(xs[xs != ""])
}
getSents <- function(x) {
    x <- gsub(" (Mr|Ms|Mrs|Dr|St|Sr|Jr)\\.", "\\1", x, perl = TRUE)
    x <- gsub("[\\.!\\?][ \t]+", "\n", x, perl = TRUE)
    xs <- unlist(strsplit(x, "\n", perl = TRUE))
    return(xs[xs != ""])
}
listSpeech$wc <- sapply(speechVec, function(x) {
    return(length(getWords(x)))
}, USE.NAMES = FALSE)
listSpeech$sc <- sapply(speechVec, function(x) {
    return(length(getSents(x)))
}, USE.NAMES = FALSE)
listSpeech$wMean <- sapply(speechVec, function(x) {
    return(mean(nchar(getWords(x))))
}, USE.NAMES = FALSE)
listSpeech$wSD <- sapply(speechVec, function(x) {
    return(sd(nchar(getWords(x))))
}, USE.NAMES = FALSE)
listSpeech$sMean <- sapply(speechVec, function(x) {
    return(mean(nchar(getSents(x))))
}, USE.NAMES = FALSE)
listSpeech$sSD <- sapply(speechVec, function(x) {
    return(sd(nchar(getSents(x))))
}, USE.NAMES = FALSE)
```

(g) (h)

We now start to extract some features of interest from the speeches to analyze how the speeches have changed over time. The result of all this is a list with each element containing the information about a speech: the speech as a single string, the vector of sentences, the vector of words, the word counts, and the additional quantification of variables about the speech from (g) as well as the non-verbal variables from (c).

1. Length in words and sentences *wc,sc*
2. Average and SD of word and sentence lengths *wMean,wSD,sMean,sSD*
3. Number of quotations in each speech, mean length (in words), and SD of length (in words) of the quotations in each speech *quoNum,quoMean,quoSD*
4. The most common meaningful words, where non-meaningful words are pre-defined *cmw*
5. Counts of the following words or word stems:

   I, we

   America,n

   democracy,tic

   republic

Democrat,ic

Republican

free,dom

war

God – not including God bless

God Bless

Jesus, Christ, Christian

Woman – I think would be interesting

```
tmpList <- matrix(rep(0, 226 * 15), nrow = 226, ncol = 15)
## Speech list with element-wise analysis
speechList <- list()  #empty list
system("wget -O 'common_words.txt' 'http://www.textfixer.com/resources/common-english-words.txt'")
commonWords <- readLines("common_words.txt", warn = FALSE)
commonWords <- unlist(strsplit(commonWords, ",", perl = TRUE))
for (i in 1:length(fileid)) {
    ss <- list()
    # Global attr
    ss$id <- fileid[i]
    ss$name <- namePres[i]
    ss$date <- as.integer(dateTalk[i])
    ss$numLaughter <- lauNum[i]
    ss$numApplause <- appNum[i]
    ss$speech <- speechVec[i]
    # Indiv attr
    talkWords <- getWords(speechVec[i])
    talkSents <- getSents(speechVec[i])
    ss$words <- talkWords  # words vector
    ss$sents <- talkSents  # sentence vector
    ss$wc <- length(talkWords)  # word count
    ss$sc <- length(talkSents)  # sentence count
    ss$wMean <- mean(nchar(talkWords))  # avg word length
    ss$wSD <- sd(nchar(talkWords))  # word length sd
    ss$sMean <- mean(nchar(talkSents))  # avg sentence length
    ss$sSD <- sd(nchar(talkSents))  # sentence length sd; ss[14]
    patQuo <- "\"(.*?)\""  # quotation pattern
    quo <- talkSents[grep(patQuo, talkSents, perl = TRUE)]
    if (length(quo) != 0) {
        # get quotation attr
        quo <- sapply(quo, function(x) {
            gsub(patQuo, "\\1", x)
        }, USE.NAMES = FALSE)
        ss$quoNum <- length(quo)
        ss$quoMean <- mean(nchar(quo))
        ss$quoSD <- sd(nchar(quo))
    } else {
        ss$quoNum <- 0
        ss$quoMean <- 0
        ss$quoSD <- 0
    }  #ss[17]
```

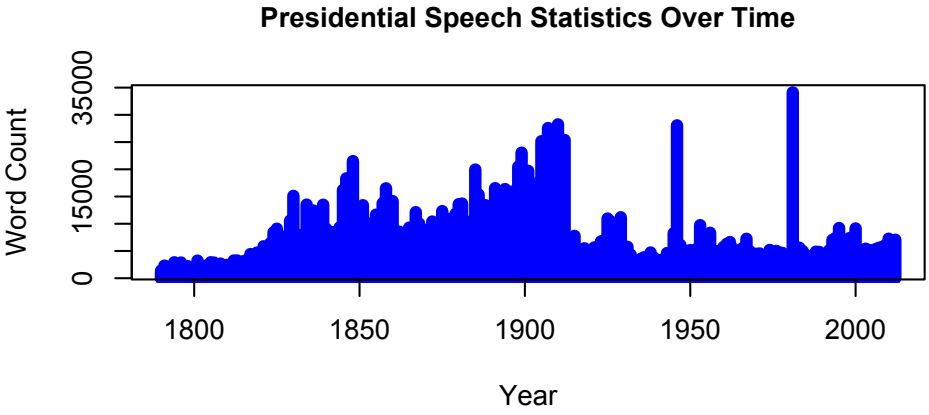
```
    cmw <- sort(table(talkWords), decreasing = TRUE)
    cmw <- cmw[which(!(names(cmw) %in% commonWords))]  # get meaningful words
    ss$cmw <- cmw[cmw >= 10]  #arbitrary cut-off for display
    ss$strIwe <- cmw[grep("^(I|[Ww]e)$", names(cmw))]  #string 'I|We'; ss[19]
    ss$strAme <- cmw[grep("[Aa]merica(|n)", names(cmw))]
    ss$strDem <- cmw[grep("[Dd]emocra(cy|tic)", names(cmw))]
    ss$strRep <- cmw[grep("[Rr]epublic(|n)", names(cmw))]
    ss$strFree <- cmw[grep("^[Ff]ree(|dom)$", names(cmw))]
    ss$strWar <- cmw[grep("^[Ww]ar(|s)$", names(cmw))]
    ss$strGod <- cmw[grep("^[Gg]od(|s)$", names(cmw))]
    ss$strChr <- cmw[grep("(Jesus|Christ|Christian)", names(cmw))]
    ss$strWoman <- cmw[grep("^[Ww]om[ae]n$", names(cmw))]  #Mystring 'Woman'; ss[27]
    ssGodBless <- talkSents[grep("[Gg]od [Bb]less", talkSents, perl = TRUE)]
    if (length(ssGodBless) != 0) {
        #string 'God Bless' from sentences
        ss$strGodBless <- sapply(ssGodBless, function(x) {
            return(length(gregexpr("[Gg]od [Bb]less", x, perl = TRUE)[[1]]))
        }, USE.NAMES = FALSE)
    } else {
        ss$strGodBless <- 0
    }
    # add to speechList and listSpeech
    speechList[[i]] <- ss
    tmpList[i, 1:3] <- unlist(ss[15:17])  #quo
    tmpList[i, 4:13] <- sapply(ss[19:28], sum)  #cmw
}
# prepare for plotting
listSpeech$quoNum <- tmpList[, 1]
listSpeech$quoMean <- tmpList[, 2]
listSpeech$quoSD <- tmpList[, 3]
listSpeech$strIwe <- tmpList[, 4]
listSpeech$strAme <- tmpList[, 5]
listSpeech$strDem <- tmpList[, 6]
listSpeech$strRep <- tmpList[, 7]
listSpeech$strFree <- tmpList[, 8]
listSpeech$strWar <- tmpList[, 9]
listSpeech$strGod <- tmpList[, 10]
listSpeech$strChr <- tmpList[, 11]
listSpeech$strWoman <- tmpList[, 12]
listSpeech$strGodBless <- tmpList[, 13]
```

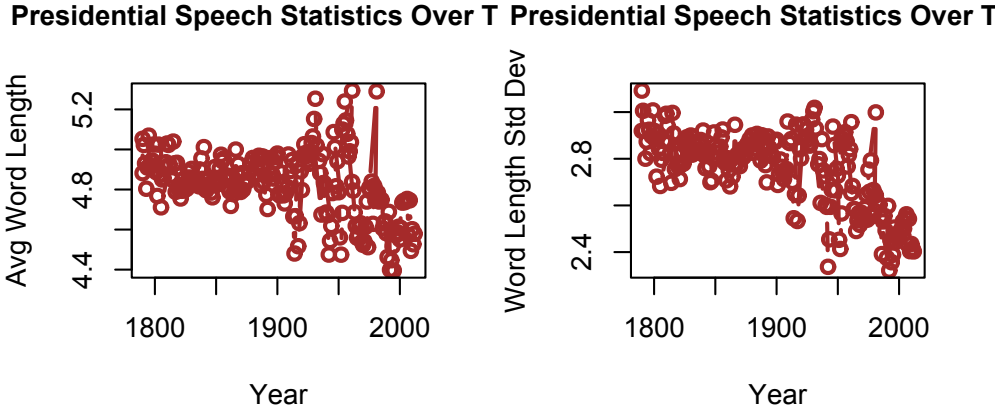
(i) (j)

Some basic plots that show how the variables have changed over time are given below.

```
# Create plots
attach(listSpeech)
plotTitle1 = "Presidential Speech Statistics Over Time"
xlab = "Year"
# over time
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
par(cex.main = 1)
typ = "h"
```

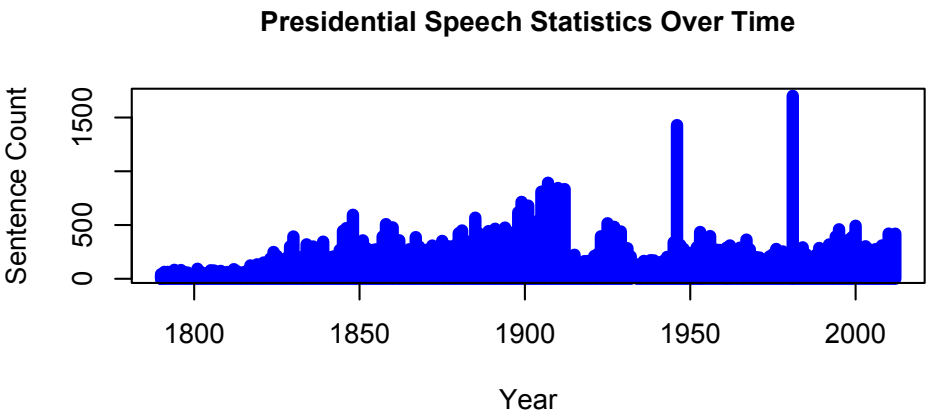
```
col = "blue"
lwd = 6
plot(date, wc, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Word Count")
typ = "b"
lwd = 2
col = "brown"
plot(date, wMean, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Avg Word Length")
plot(date, wSD, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Word Length Std Dev")
```

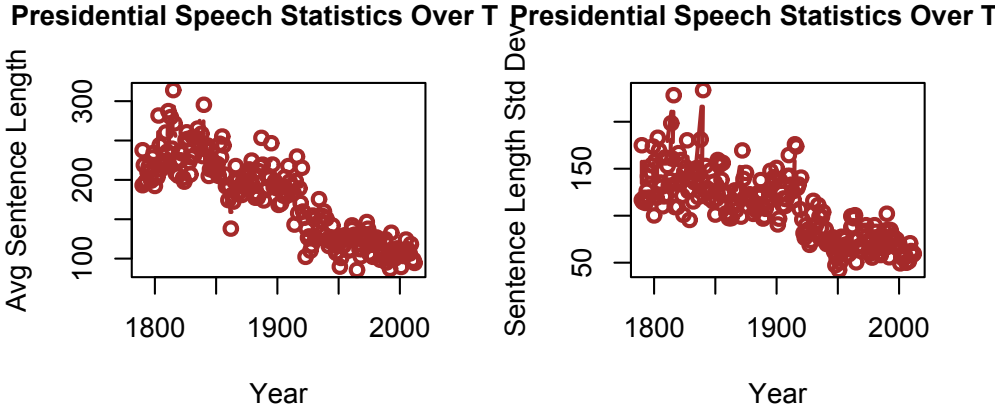




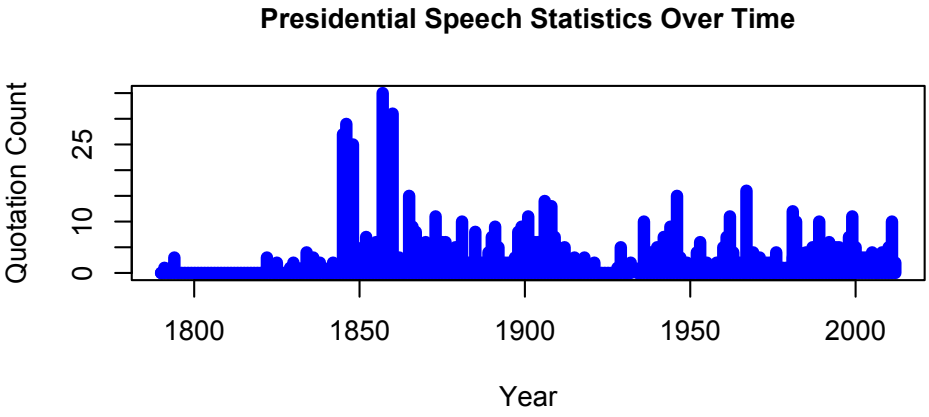

```
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
par(cex.main = 1)
typ = "h"
col = "blue"
lwd = 6
plot(date, sc, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Sentence Count")
typ = "b"
lwd = 2
```

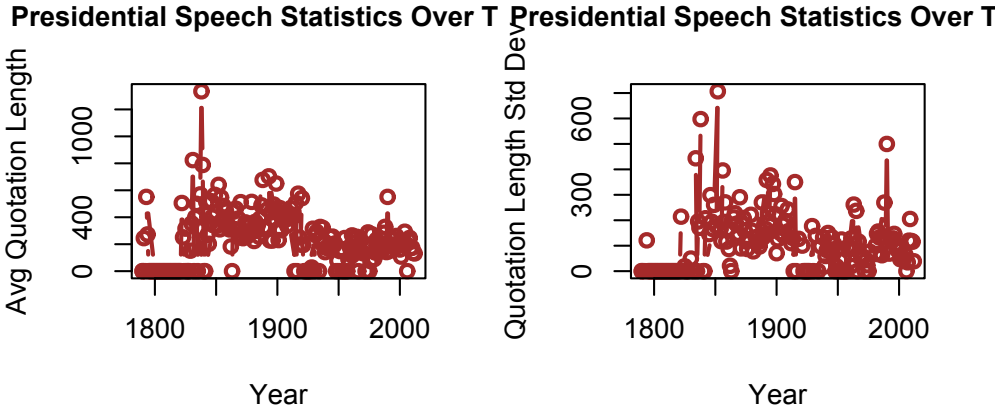
```
col = "brown"
plot(date, sMean, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Avg Sentence Length")
plot(date, sSD, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Sentence Length Std Dev")
```



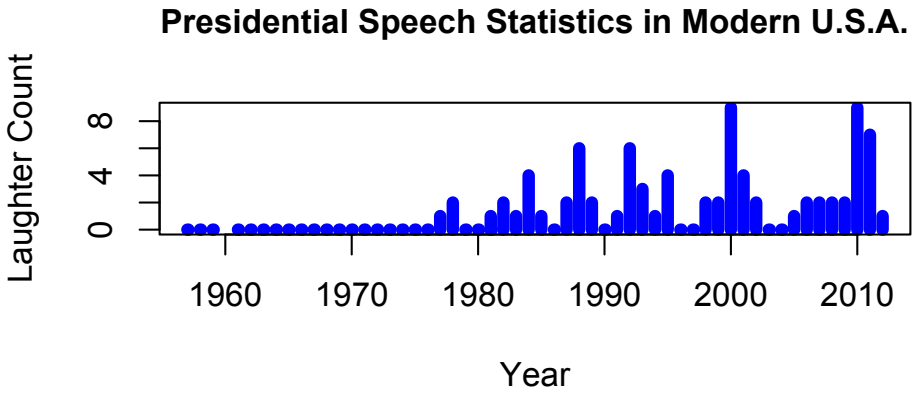


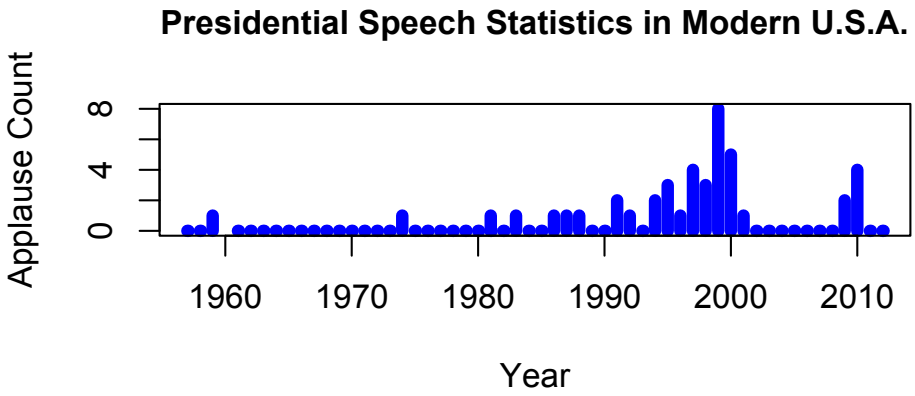

```
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
par(cex.main = 1)
typ = "h"
lwd = 6
col = "blue"
plot(date, quoNum, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Quotation Count")
typ = "b"
lwd = 2
col = "brown"
plot(date, quoMean, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Avg Quotation Length")
plot(date, quoSD, type = typ, col = col, lwd = lwd, main = plotTitle1, xlab = xlab,
    ylab = "Quotation Length Std Dev")
```





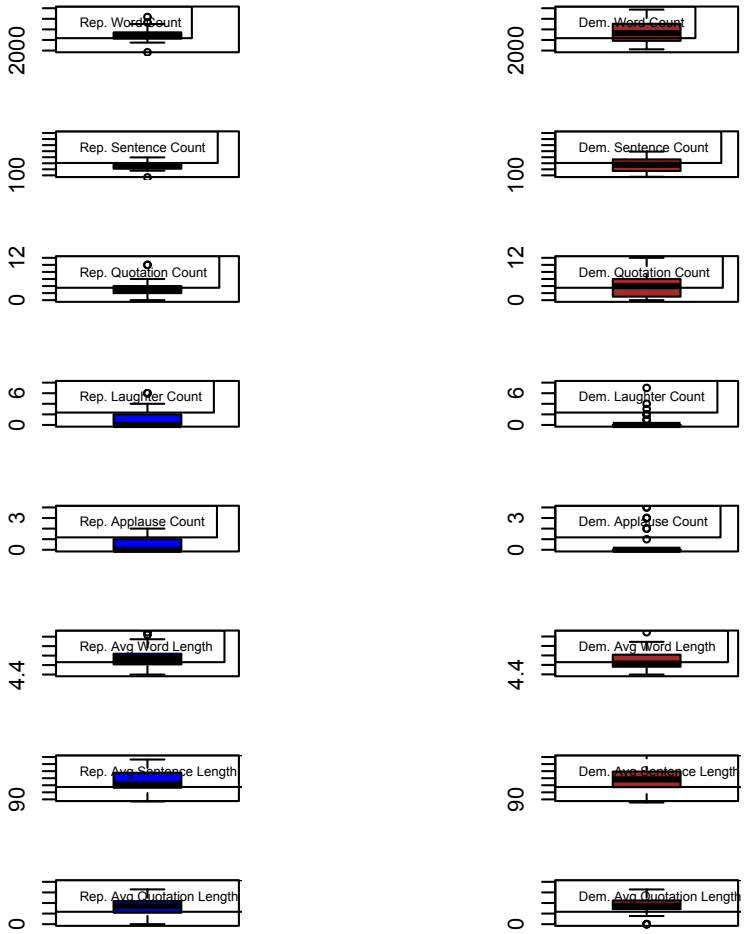

```
# recent years
plotTitle2 = "Presidential Speech Statistics in Modern U.S.A."

recY <- 1:max(c(which(numLaughter > 0), which(numApplause > 0)))
typ = "h"
lwd = 6
col = "blue"
par(mfrow = c(2, 1), cex.main = 1)
plot(date[recY], numLaughter[recY], type = typ, col = col, lwd = lwd, main = plotTitle2,
    xlab = xlab, ylab = "Laughter Count")
plot(date[recY], numApplause[recY], type = typ, col = col, lwd = lwd, main = plotTitle2,
    xlab = xlab, ylab = "Applause Count")
```



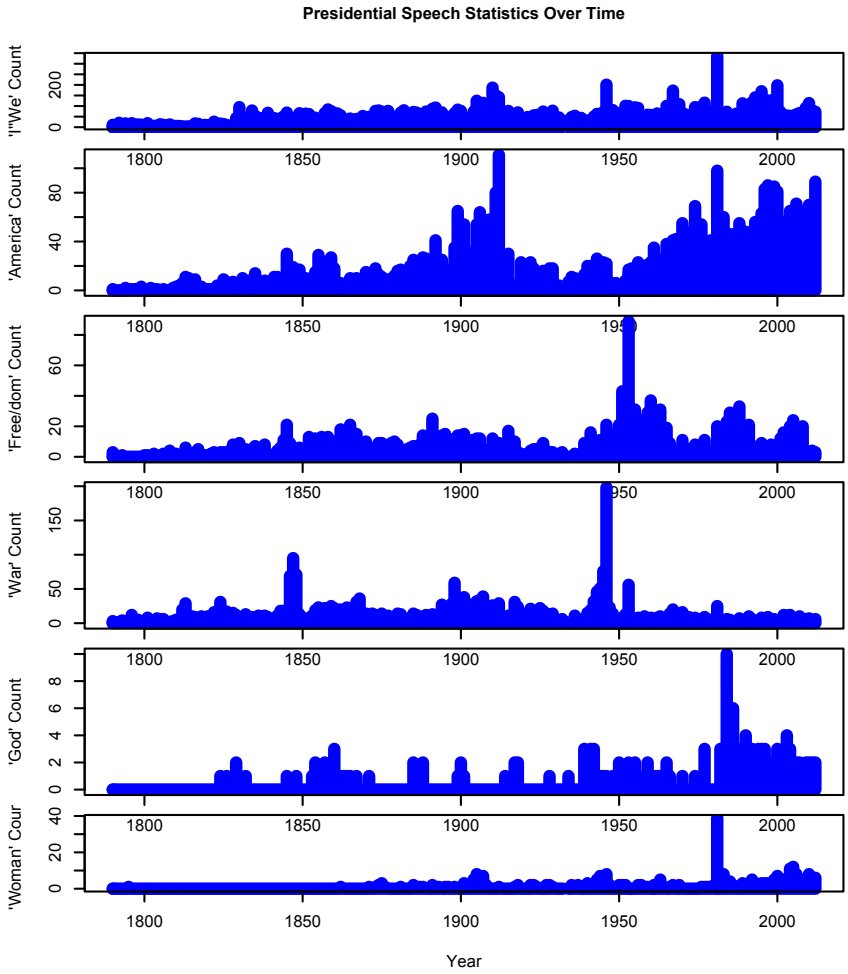



```
# Rep vs Dem
plotTitle3 = "Republican vs. Democratic Presidential Speech Statistics (Since 1932)"

repPres <- c("Dwight D. Eisenhower", "Richard Nixon", "Gerald R. Ford", "Ronald Reagan",
    "George Bush", "George W. Bush")
demPres <- c("Franklin D. Roosevelt", "Harry S. Truman", "John F. Kennedy",
    "Lyndon B. Johnson", "Jimmy Carter", "William J. Clinton", "Barack Obama")
repY <- which(name %in% repPres)
demY <- which(name %in% demPres)
layout(matrix(1:16, 8, 2))
par(mar = c(1.5, 6, 1.5, 6))
col = "blue"
cex = 0.6
boxplot(wc[repY], col = col, ylim = c(2000, 10000))
legend("topleft", "Rep. Word Count", cex = cex)
boxplot(sc[repY], col = col, ylim = c(100, 800))
legend("topleft", "Rep. Sentence Count", cex = cex)
boxplot(quoNum[repY], col = col, ylim = c(0, 12))
legend("topleft", "Rep. Quotation Count", cex = cex)
boxplot(numLaughter[repY], col = col, ylim = c(0, 8))
legend("topleft", "Rep. Laughter Count", cex = cex)
```

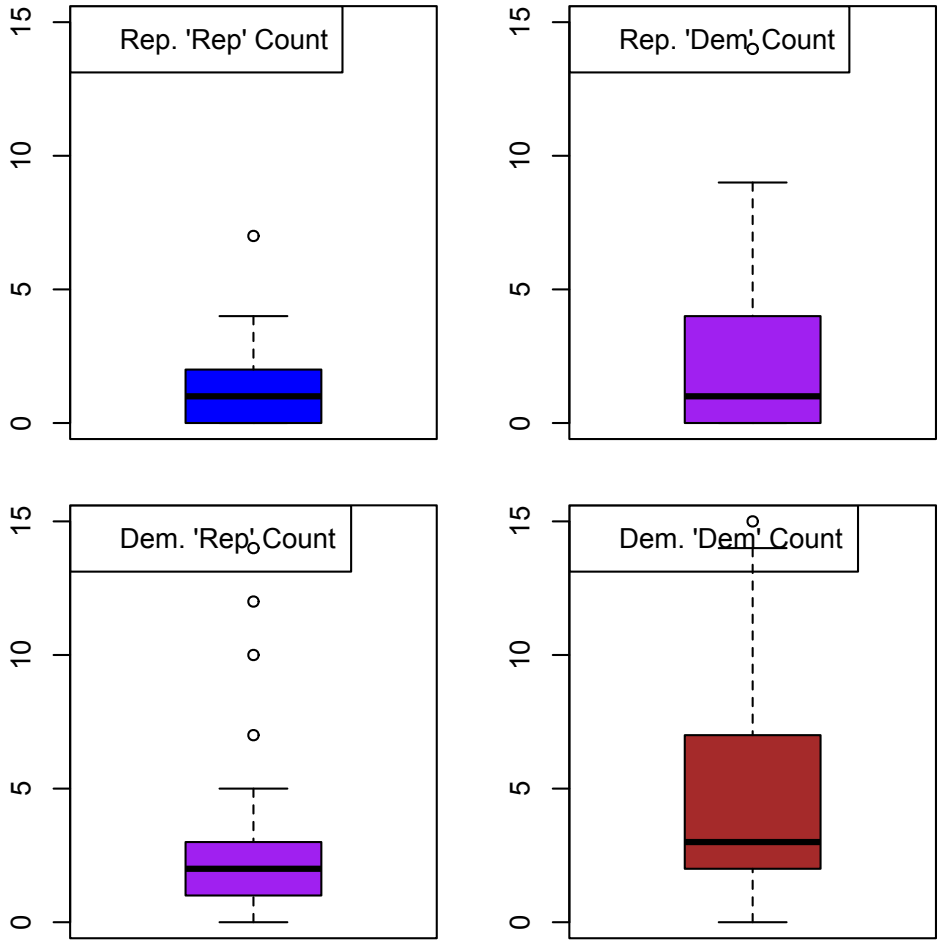
```
boxplot(numApplause[repY], col = col, ylim = c(0, 4))
legend("topleft", "Rep. Applause Count", cex = cex)
boxplot(wMean[repY], col = col)
legend("topleft", "Rep. Avg Word Length", cex = cex)
boxplot(sMean[repY], col = col, ylim = c(90, 150))
legend("topleft", "Rep. Avg Sentence Length", cex = cex)
boxplot(quoMean[repY], col = col, ylim = c(0, 400))
legend("topleft", "Rep. Avg Quotation Length", cex = cex)
col = "brown"
cex = 0.6
boxplot(wc[demY], col = col, ylim = c(2000, 10000))
legend("topleft", "Dem. Word Count", cex = cex)
boxplot(sc[demY], col = col, ylim = c(100, 800))
legend("topleft", "Dem. Sentence Count", cex = cex)
boxplot(quoNum[demY], col = col, ylim = c(0, 12))
legend("topleft", "Dem. Quotation Count", cex = cex)
boxplot(numLaughter[demY], col = col, ylim = c(0, 8))
legend("topleft", "Dem. Laughter Count", cex = cex)
boxplot(numApplause[demY], col = col, ylim = c(0, 4))
legend("topleft", "Dem. Applause Count", cex = cex)
boxplot(wMean[demY], col = col)
legend("topleft", "Dem. Avg Word Length", cex = cex)
boxplot(sMean[demY], col = col, ylim = c(90, 150))
legend("topleft", "Dem. Avg Sentence Length", cex = cex)
boxplot(quoMean[demY], col = col, ylim = c(0, 400))
legend("topleft", "Dem. Avg Quotation Length", cex = cex)
```



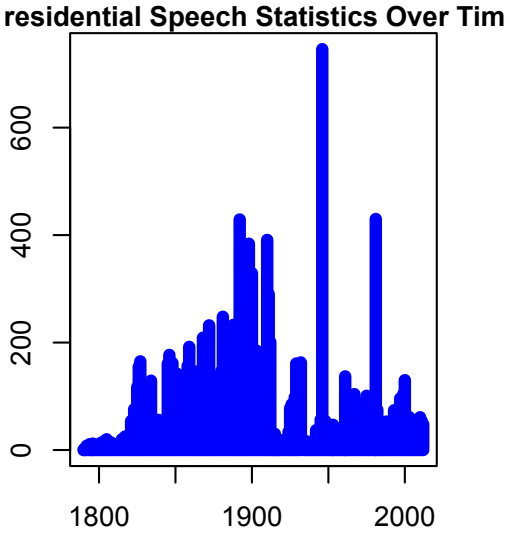

```
# extra

par(mfrow = c(6, 1), cex = 0.5, cex.main = 1)
typ = "h"
col = "blue"
lwd = 6
par(mar = c(0.5, 6, 4, 6))
plot(date, strIwe, type = typ, col = col, lwd = lwd, main = plotTitle1, ylab = "'I''We' Count")
par(mar = c(0.5, 6, 0.5, 6))
plot(date, strAme, type = typ, col = col, lwd = lwd, ylab = "'America' Count")
par(mar = c(0.5, 6, 0.5, 6))
plot(date, strFree, type = typ, col = col, lwd = lwd, ylab = "'Free/dom' Count")
par(mar = c(0.5, 6, 0.5, 6))
plot(date, strWar, type = typ, col = col, lwd = lwd, ylab = "'War' Count")
par(mar = c(0.5, 6, 0.5, 6))
plot(date, strGod, type = typ, col = col, lwd = lwd, ylab = "'God' Count")
par(mar = c(4, 6, 0.5, 6))
plot(date, strWoman, type = typ, col = col, lwd = lwd, xlab = xlab, ylab = "'Woman' Count")
```




```
par(mfrow = c(2, 2), mar = c(1, 2, 1, 2))
cex = 1
col = "blue"
boxplot(strRep[repY], col = col, ylim = c(0, 15))
legend("topleft", "Rep. 'Rep' Count", cex = cex)
col = "purple"
boxplot(strDem[repY], col = col, ylim = c(0, 15))
legend("topleft", "Rep. 'Dem' Count", cex = cex)
boxplot(strRep[demY], col = col, ylim = c(0, 15))
legend("topleft", "Dem. 'Rep' Count", cex = cex)
col = "brown"
boxplot(strDem[demY], col = col, ylim = c(0, 15))
legend("topleft", "Dem. 'Dem' Count", cex = cex)
```




```
getDigts <- function(x) {
    x <- gsub("(\\d)[,\\.](\\d)", "\\1\\2", x, perl = TRUE)
    x <- gsub("\\D", "\n", x, perl = TRUE)
    xs <- unlist(strsplit(x, "\n", perl = TRUE))
    return(xs[xs != ""])
}
listSpeech$dc <- sapply(speechVec, function(x) {
    return(length(getDigts(x)))
}, USE.NAMES = FALSE)
typ = "h"
col = "blue"
lwd = 6
plot(date, listSpeech$dc, type = typ, col = col, lwd = lwd, main = plotTitle1,
    xlab = xlab, ylab = "Digits Count")
```



And for presidents since Franklin Roosevelt in 1932, comparison between Republican presidents (Eisenhower, Nixon, Ford, Reagan, G. Bush, G.W. Bush) and Democratic presidents (Roosevelt, Truman, Kennedy, Johnson, Carter, Clinton, Obama) are also given.

Some additional research and/or additional thinking to come up with additional variables that quantify speech in interesting ways. Do some plotting that illustrates how the speeches have changed over time.