HW1-TianyiFang

Tianyi Fang

August 27, 2017

Problem1:

1.install tm package

```
#install.packages("tm")
library(tm)
```

```
## Warning: package 'tm' was built under R version 3.4.1
```

```
## Loading required package: NLP
```

```
## Warning: package 'NLP' was built under R version 3.4.1
```

2.Read text-files using Corpus from tm.

```
cname <- file.path("C:","Users","Tianyi Fang","Desktop","stat545","txt")
dir(cname)
```

```
## [1] "NEWTIME.txt" "stat.txt"
```

```
corpus <- Corpus(DirSource(cname))
```

3.Look at the type of corpus

```
typeof(corpus)
```

```
## [1] "list"
```

```
typeof(corpus[1])
```

```
## [1] "list"
```

```
typeof(corpus[[1]])
```

```
## [1] "list"
```

Difference between [] and [[]]:
[[]]selects a single element of a list or data frame, while[]can select more than one element. Also, in most cases, []returns the same class(like list[]is still list), while [[]]returns the class of that element. ####4.transformations: + replace"/,@,\,-," with space to avoid two words being run into one string of characters.

```
toSpace <- content_transformer(function(x,pattern) gsub(pattern, " ",x))
corpus <- tm_map(corpus, toSpace, "/|@|\\|")
inspect(corpus[2])
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 1
##
##
## Statistics is a branch of mathematics dealing with the collection, analysis, interpretation, presenta
```

2. conversion to Lower Case so that it is easier to count the numbers of some words regardless of their Upper/Lower cases

```
corpus <- tm_map(corpus, content_transformer(tolower))
```

3. remove punctuation and numbers.

```
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, removeNumbers)
```

4. remove English Stop words since they do not provide much useful information for text mining

```
  corpus <- tm_map(corpus, removeWords, stopwords("english"))
```

5.create document term matrix

```
dtm <- DocumentTermMatrix(corpus)
inspect(dtm[1:2,1:10])
```

```
## <<DocumentTermMatrix (documents: 2, terms: 10)>>
## Non-/sparse entries: 11/9
## Sparsity           : 45%
## Maximal term length: 12
## Weighting          : term frequency (tf)
## Sample             :
##              Terms
## Docs          abandoned abbreviated abc ability able abolitionist abortion
##   NEWTIME.txt         1           1   1       1    1            1        1
##   stat.txt            0           0   0       0    0            0        0
##              Terms
## Docs          absence absolute abukhalil
##   NEWTIME.txt       1        1         1
##   stat.txt          0        1         0
```

6.Calculate the frequency

```
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 3.4.1
```

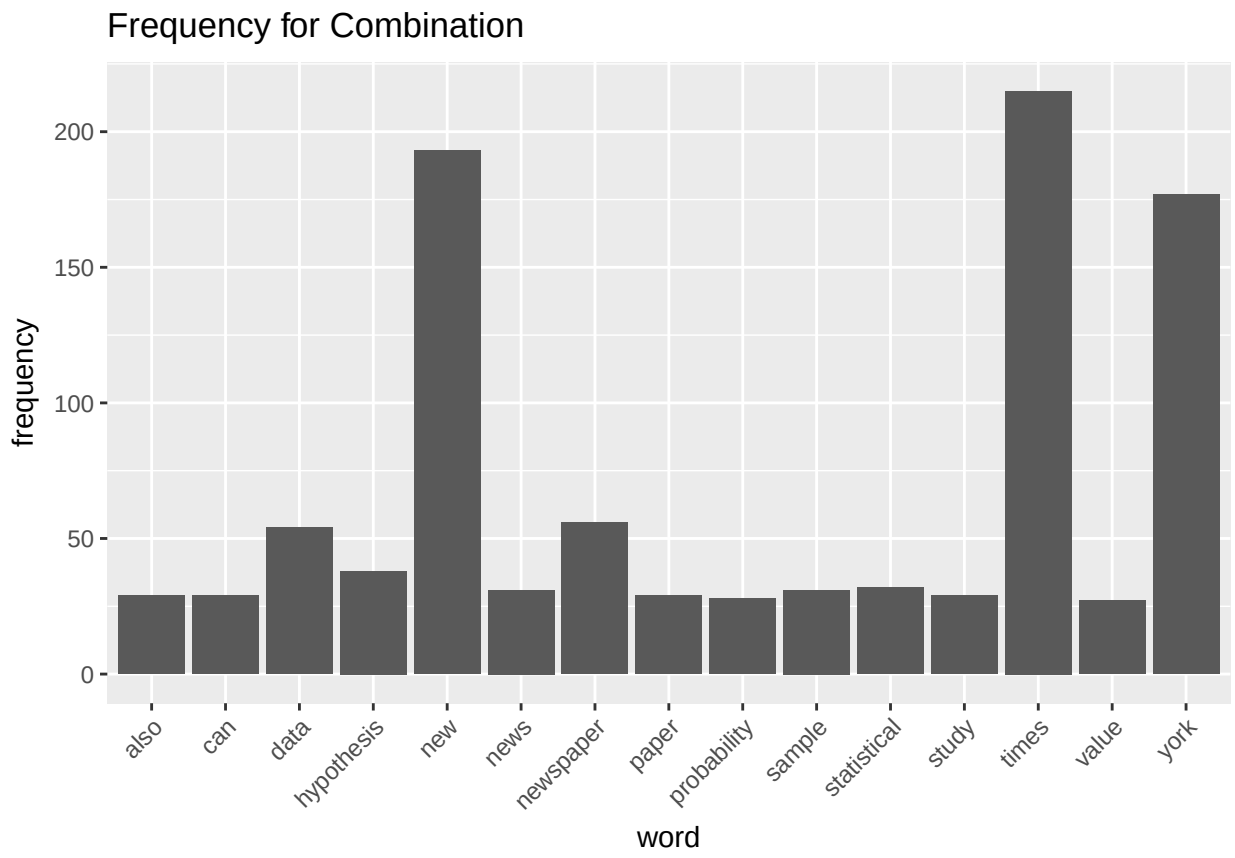
```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.1
```

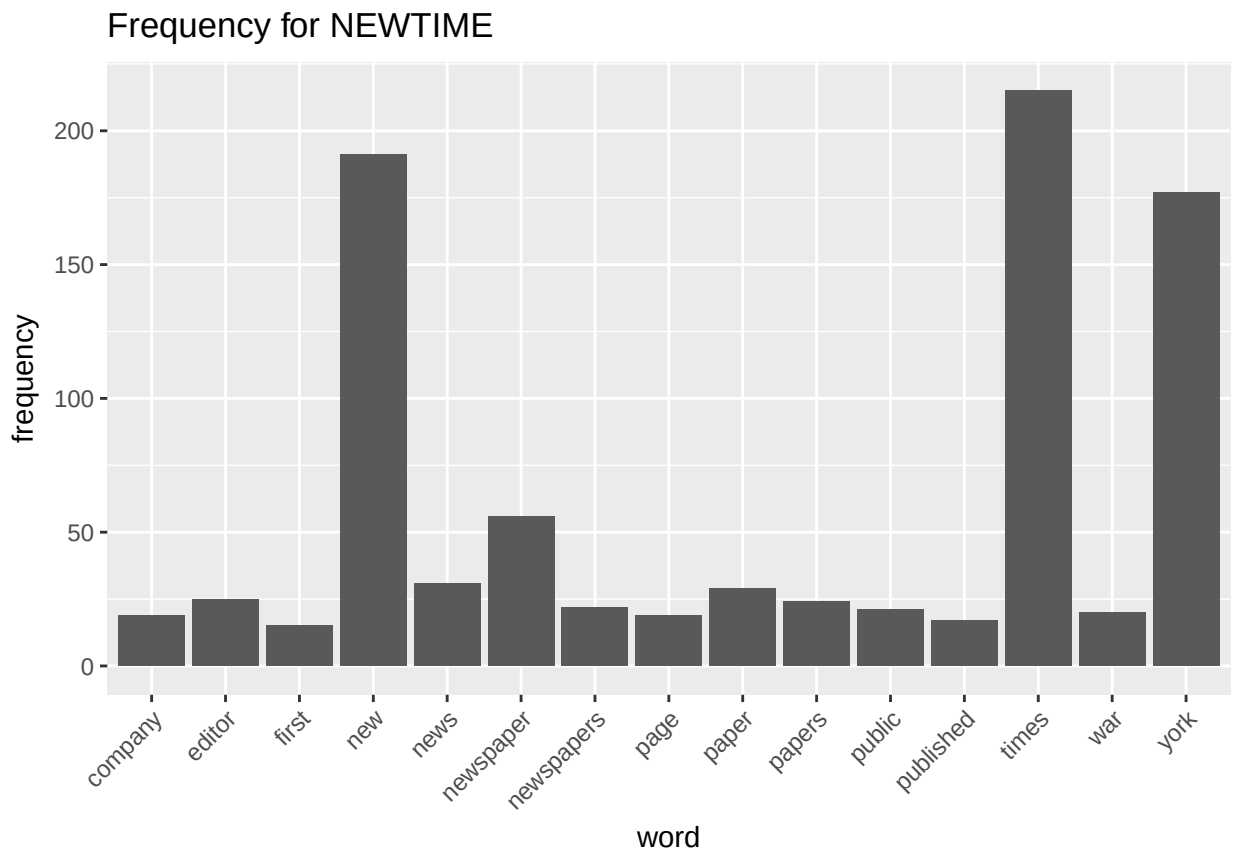
```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:NLP':
##
##     annotate
```

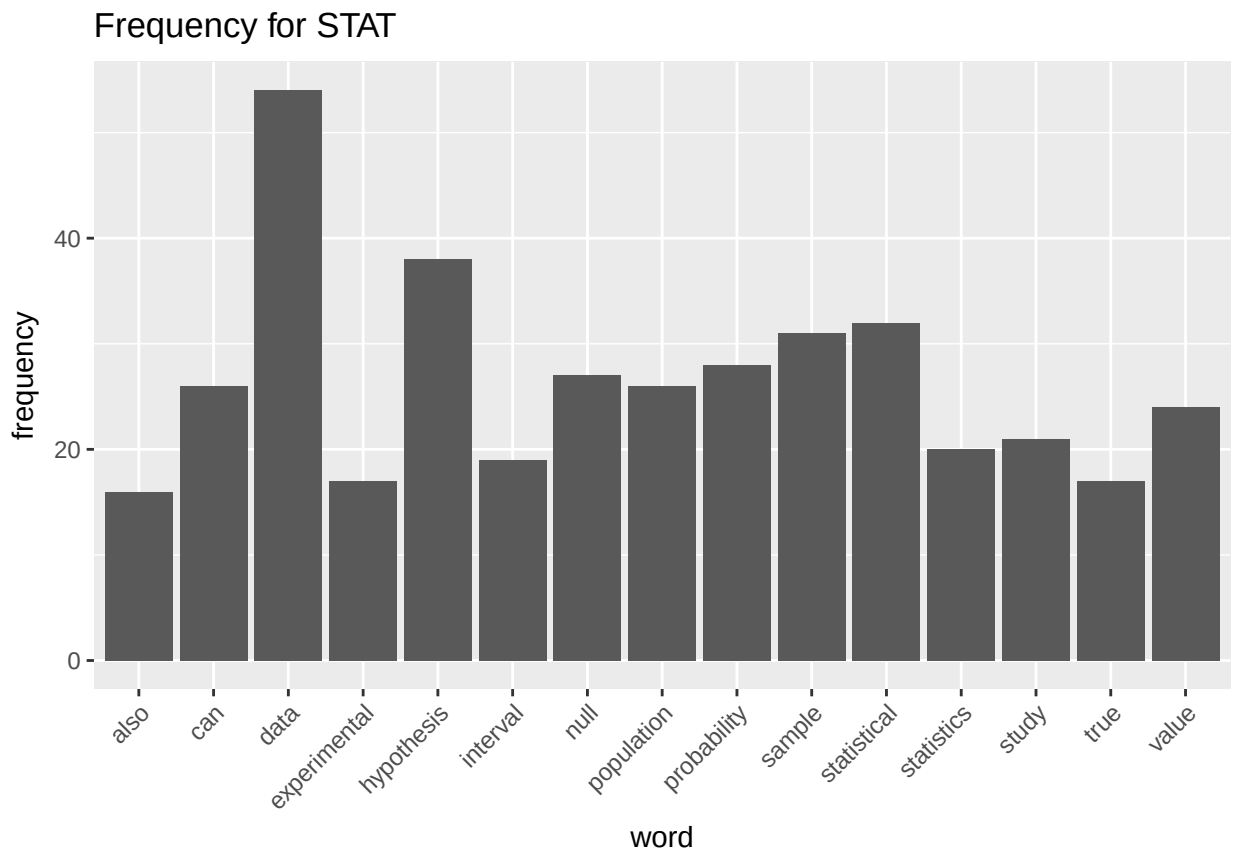
```
freq <- sort(colSums(as.matrix(dtm)), decreasing = TRUE)
wf <- data.frame(word=names(freq),freq=freq)
subset(wf[1:15,])%>%
  ggplot(aes(word, freq))+
  geom_bar(stat="identity") +
  labs(x="word", y = "frequency", title = "Frequency for Combination") +
  theme(axis.text.x = element_text(angle = 45,hjust = 1))
```




```
text1 <- dtm[1,]#count freq separately
text2 <- dtm[2,]#sort the words in order of decreasing freq
freq1 <- sort(colSums(as.matrix(text1)), decreasing=TRUE)
freq2 <- sort(colSums(as.matrix(text2)), decreasing=TRUE)
freq1df <- data.frame(freq1, word = names(freq1), freq = freq1)
freq2df <- data.frame(freq2, word = names(freq2), freq = freq2)
subset(freq1df[1:15,]) %>%
  ggplot(aes(word, freq))+
  geom_bar(stat="identity") +
  labs(x="word", y = "frequency", title = "Frequency for NEWTIME") +
  theme(axis.text.x = element_text(angle = 45,hjust = 1))
```




```
subset(freq2df[1:15,]) %>%
  ggplot(aes(word, freq))+
  geom_bar(stat="identity") +
  labs(x="word", y = "frequency", title = "Frequency for STAT") +
  theme(axis.text.x = element_text(angle = 45,hjust = 1))
```



####7.wordcloud

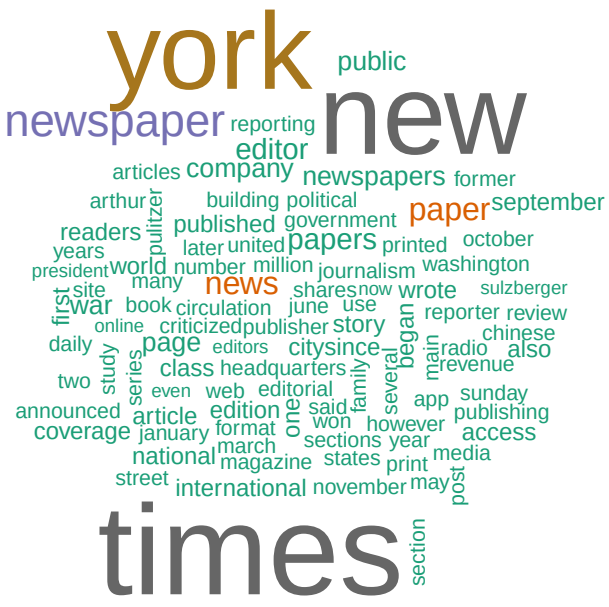
```
library(wordcloud)
```

```
## Warning: package 'wordcloud' was built under R version 3.4.1
```

```
## Loading required package: RColorBrewer
```

```
set.seed(1)
wordcloud(freq1df$word, freq1df$freq, max.word = 100, colors=brewer.pal(10,"Dark2"))
```

```
## Warning in brewer.pal(10, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

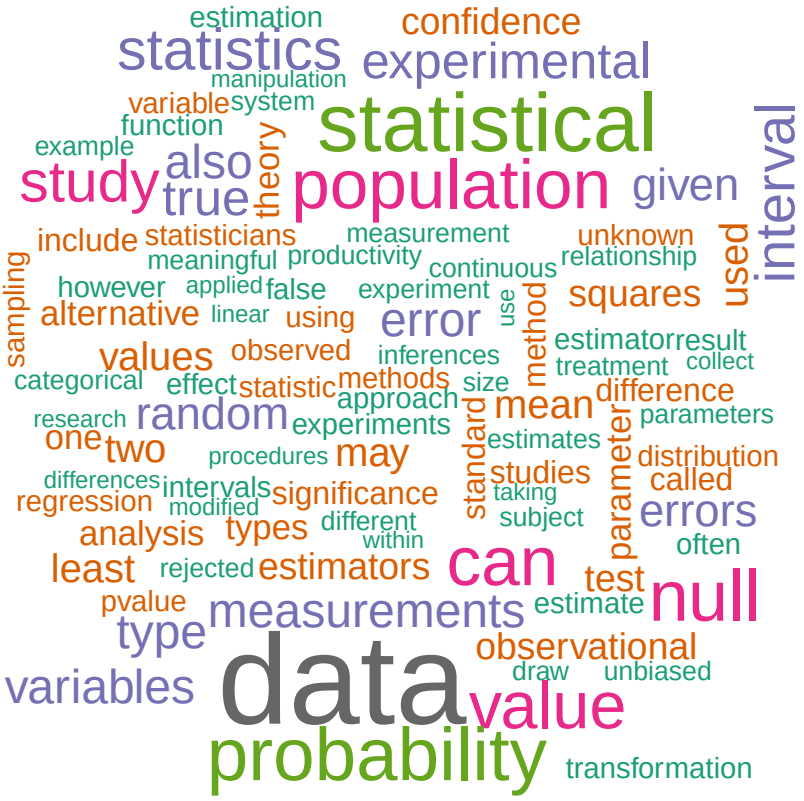



```
wordcloud(freq2df$word, freq2df$freq, max.word = 100, colors=brewer.pal(10,"Dark2"))
```

```
## Warning in brewer.pal(10, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

```
## Warning in wordcloud(freq2df$word, freq2df$freq, max.word = 100, colors
## = brewer.pal(10, : hypothesis could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(freq2df$word, freq2df$freq, max.word = 100, colors =
## brewer.pal(10, : sample could not be fit on page. It will not be plotted.
```




```
wordcloud(wf$word, wf$freq, max.word = 100, colors=brewer.pal(10,"Dark2"))
```

```
## Warning in brewer.pal(10, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

Problem2:Checking the central limit theorem

Write the my_CLT function and plot the density.

```
#follow the CLT definition. standardize:
#sum(xi)/thegma/sqrt(n)~N(0,1) when n is extremely large,
#if scaled my_matrix by sqrt(n), then mm <- rowSums(my_matrix)/thegma/n
library(ggplot2)
library(reshape2)
```

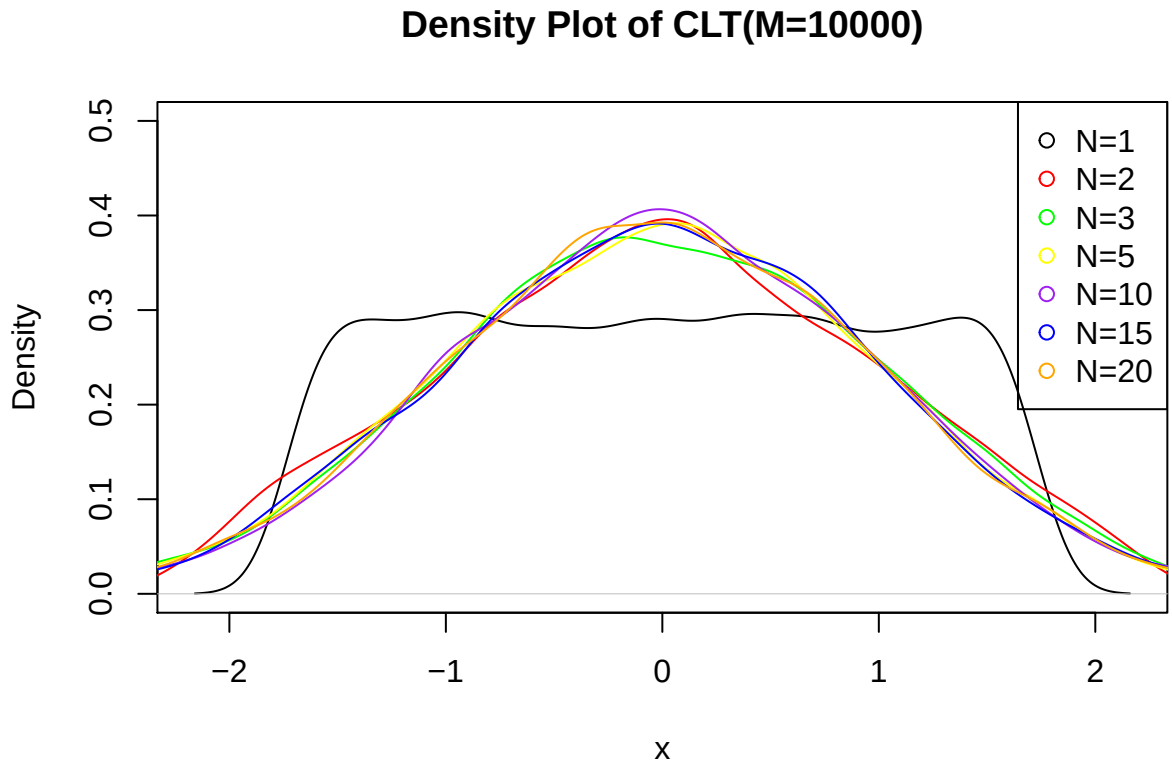
```
## Warning: package 'reshape2' was built under R version 3.4.1
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
my_CLT <-function(n,m){
  uniform <- runif(n*m,min=-1, max=1)
  my_matrix <- matrix(uniform, nrow=m, ncol = n)
  thegma <- sqrt(1/3)
  result <- rowSums(my_matrix)/thegma/sqrt(n)
 #result2 <- rowSums(my_matrix*sqrt(n))/thegma/n
  result
}
```

```
plot(density(my_CLT(1,10000)),ylim=c(0,0.5),main = "Density Plot of CLT(M=10000)",xlab = "x")
lines(density(my_CLT(2,10000)), col = "red")
lines(density(my_CLT(3,10000)), col = "green")
lines(density(my_CLT(5,10000)), col = "yellow")
lines(density(my_CLT(10,10000)), col = "purple")
lines(density(my_CLT(15,10000)), col = "blue")
lines(density(my_CLT(20,10000)), col = "orange")
legend("topright",pch = 1, col = c("black","red","green","yellow","purple","blue","orange"),
       legend = c("N=1","N=2","N=3","N=5","N=10","N=15","N=20"))
```




```
#plot(density(mm), main = "density plot")
#plot(density(mm2))
#check whether the var and mean is 1 and 0 or not.
  #print(var(mm))
  #print(mean(mm))
#x<-cbind(my_CLT(1,10000),my_CLT(2,10000),my_CLT(3,10000),my_CLT(5,10000),
#my_CLT(10,10000),my_CLT(15,10000),my_CLT(20,10000))
#data <- melt(x)
#ggplot(data, aes(x=value, fill=variable))+ geom_density(alpha=0.5)
```

problem3:The power method

1.Eigen vector/value with eigen()

```r
set.seed(1)
m1 <- runif(25,0,2)
rm <- matrix(m1,5,5) + t(matrix(m1,5,5))
eigen(rm)
```

```
## eigen() decomposition
## $values
## [1] 10.7613938  2.6551793  0.0360662 -1.0647158 -2.2100730
##
## $vectors
##            [,1]        [,2]       [,3]         [,4]       [,5]
## [1,] -0.4260525  0.06895636  0.6302662  0.043193663  0.6439123
## [2,] -0.4828835  0.73820609 -0.2773274  0.349705552 -0.1505682
## [3,] -0.4756286 -0.47386449 -0.6438601 -0.009108171  0.3668666
## [4,] -0.4816793  0.01822447  0.1284631 -0.771627510 -0.3946405
## [5,] -0.3564620 -0.47478111  0.3078886  0.529479331 -0.5218946
```

2.Power Method

```r
eigen_function <- function(matrix, iter){
  x <- c()#the vector set which will be multipled by the input matrix
  y <- c()#the result vector set
  x[[1]] <- c(1,1,1,1,1)# add the initial vector as the first element of x

  for (i in 1:iter){
    y[[i]] <- matrix %*% x[[i]]
    ynorm <- sqrt(sum(y[[i]]*y[[i]]))
    #ynorm is the scaler
    x[[i+1]] <- y[[i]] / ynorm
    eigen_vector <- x[[i+1]]
    eigen_value <- mean(matrix%*%eigen_vector/eigen_vector)

    #print(eigen_value)#print out evector/evalue for each iteration
    #print(eigen_vector)
  }
  #print out the final one
  result <- list(eigen_value, eigen_vector)
  return(result)
}
first<-eigen_function(rm,30)
first
```

```
## [[1]]
## [1] 10.76139
##
## [[2]]
##           [,1]
## [1,] 0.4260525
## [2,] 0.4828835
## [3,] 0.4756286
## [4,] 0.4816793
## [5,] 0.3564620
```

```
# as A*x1 = y1, scaled x2 = y1/max(y1)
# next A*x2 = y2, then scaled x3 = y2/max(y2)
# next A*x3 = y3, then scaled x4 = y3/max(y3)
#...
```

3.Second largest Eigen vector/value

```
#second_eigen <- function(matrixA, iter){
  #get the list of largest Eigen value/vector from the above function
  #B = A - value1*vector1*vector1^t
  matrixB <- rm - first[[1]]*(first[[2]] %*% t(first[[2]]))
  #send matrixB to the above funtion and get the largest Eigen vector/value of matrix B
  second_eigen <- eigen_function(matrixB, 30)
  second_eigen
```

```
## [[1]]
## [1] 2.679176
##
## [[2]]
##             [,1]
## [1,] -0.07015323
## [2,] -0.73792492
## [3,]  0.47318169
## [4,] -0.01749083
## [5,]  0.47575045
```

```
#}
```

Problem4.Briefly Answer

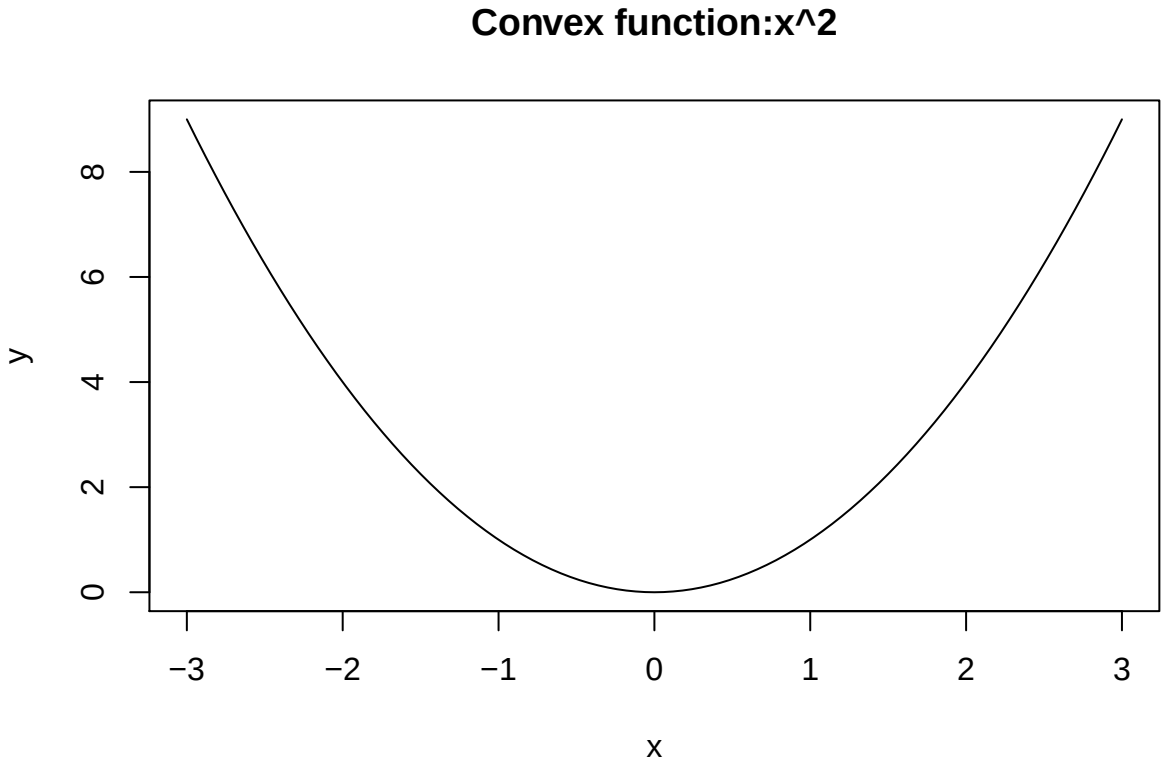
1.What is the positive-definite matrix?

In linear algebra, a symmetric n*n real matrix M is said to be positive definite if z^t*M*z is positive for every non-zero column vector z of n real numbers. Which means, all the Eigen values are positive, the associated sesquilinear form is an inner product; it is the Gram matrix of linearly independent vectors, its leading principal minors are all positive and it has a unique Cholesky decomposition.

2.What is the Hessian of a function? What is a convex function?Plot a convex function

Hessian matirx is a square matrix of second-order partial derivatives of a scalar-valued function. **Convex function** is a real-valued function if the line segment between any two points on the graph of the function lies above or on the graph, in a Euclidean space of at least two dimensions.

```
curve(x^2, -3, 3, ylab="y")
title(main="Convex function:x^2")
```



####3.Write the probability density of the multovariate Gaussian distribution.

$$f_{\mathbf{X}}(x_1, \ldots, x_k) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)}{\sqrt{(2\pi)^k|\boldsymbol{\Sigma}|}}$$

$$= \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)}{\sqrt{|2\pi\boldsymbol{\Sigma}|}};$$

image:

4.What is the law of large numbers? CLT?

In probability theory: **LLN** describe the result of performing the same experiment a large number of times, the average of the results obtained from a large number of trials should be close to the expected value, and will tend to become closer as more trials are performed. That is, the more trials we done, the more close the observed averages be to the theoretical mean.

CLT describes that when independent random variables are added, their properly normalized sum tends toward a normal distribution even if the original variables themselves are not normal distribution. That is, as n tends to infinity, the sum of a set of x will tend to follow the normal distribution, no matter what distribution x follows.

Feedback

(a) Is the material presented at an adequate pace during lecture (too slow/too fast)? I think

the pace is OK for me, but maybe professor can point out the prerequisite materials, so that we can catch up easier.

(b) What general material would you like to spend more time on? I don't have too much algorithm background, so for now I may spend more time on learning the fundamental rules and principles about algorithm, like the complexity. Also, I will spend more time practice R, hoping I could be really familiar with those useful and powerful packages.

(c) Are the homework questions generally representative of material covered in lecture? Yes. I think the homework is really helpful to not only practice my R skills, but also learning and understanding better about what we learned in the class.

(d) Any further comments/questions/feedback? If we can learn more practical problems, that will be better for us to apply what we learned in practice.