PS06

Yingyue Luan

October 24 2017

Problem 1

1. What are the goals of their simulation study and what are the metrics that they consider in assessing their method?

   - Their simulation study is trying to answer two questions: how fast does test statistics converge in distribution to asymptotic distribution and what is the power
   - They run simulation under two cases to show the likelihood ratio statistic of the null hypothesis against the alternative hypothesis of two different component normal mixture distribution is asymptotically distributed as a weighted sum of independent chi-squared random variables with one degree of freedom. In the first case, the random samples are drawn from a single normal distribution for null hypothesis and from a mixture of two normal distribution for alternative hypothesis. In the second case, the random samples are from mixtures of two and three normal distributions.
   - The metrics are the separation between two component D, sample size, mixing proportion (the height of different Gaussian distribution), and the number of components.

2. What choices did the authors have to make in designing their simulation study? What are the key aspects of the data generating mechanism that likely affect the statistical power of the test? Are there data-generating scenarios that the authors did not consider that would be useful to consider?

   - The authors have to choose whether to use equal variance, the number of replication, and significant levels.
   - Key aspects affecting the statistical power of the test include the spacing between two component D, sample size and mixing proportion.
   - The authors could have explained why they used 1000 replications or conducted other number of simulations.

3. Do their tables do a good job of presenting the simulation results and do you have any alternative suggestions for how to do this?

   - The simulation results are well presented. A minor suggestion is that they can present the simulated significance levels in the same way as simulated power using LR and LR* and listing different sample sizes in column.

4. Interpret their tables on power (Tables 2 and 4) - do the results make sense in terms of how the power varies as a function of the data generating mechanism?

    - Table 2 and table 4 measures the probability of rejection given the alternative hypothesis is true, while table 1 and table 3 presents the probability of rejection under null hypothesis. For example in table 2, when mixing proportion is 0.5, sample size is 50, nominal level is 0.01, and D = 1, the power for unadjusted test is 1.8, meaning 18/1000 reject null hypothesis give the alternative is true.
    - In table 2, for sample size less than 200, the power for D = 1 or 2 is low and for sample size larger than 100, the power of D = 3 becomes large.
    - In table 4, for sample size less than 200, low power when one of the D is less than 4 and the other D is less than 2. When both D large than 3, power is reasonable for sample size larger than 100.

5. How do you think the authors decided to use 1000 simulations. Would 10 simulations be enough? How might we decide if 1000 simulations is enough?

    - The authors decided to use 1000 simulations because the largest sample size is 1000. 10 simulations are not enough and we check if 1000 simulation is enough by looking at the number of rejection and precision.
    - Benefits of simulation include no asymptopia and quick to generate data compared to deriving analytical result. Disadvantages can be infeasible to understand full parameter space and the influence on computational power.

Problem 2

```
library(RSQLite)
drv <- dbDriver("SQLite")
dir <- '/Users/lunaluan/Documents/Academic/Berkeley/Fall 2017/243/HW/ps06'
dbFilename <- 'stackoverflow-2016.db'
db <- dbConnect(drv, dbname = file.path(dir, dbFilename))
dbListFields(db, "questions_tags")
```

```
## [1] "questionid" "tag"
```

```
dbListFields(db, "questions")
```

```
## [1] "questionid"   "creationdate" "score"        "viewcount"
## [5] "title"        "ownerid"
```

```
dbListFields(db, "users")
```

```
##  [1] "userid"         "creationdate"   "lastaccessdate" "location"
##  [5] "reputation"     "displayname"    "upvotes"        "downvotes"
##  [9] "age"            "accountid"
```

```
dbGetQuery(db, "select * from questions_tags limit 5")
```

```
##   questionid        tag
## 1   34552711         c#
## 2   34552711      razor
## 3   34552711      flags
## 4   34552829 javascript
## 5   34552829       rxjs
```

```
result <- dbGetQuery(db,
  "select distinct U.userid from users U
  join questions Q on U.userid = Q.ownerid
  join questions_tags T on Q.questionid = T.questionid
  where T.tag = 'r' and
  userid not in (select userid from users U2
  join questions Q2 on U2.userid = Q2.ownerid
  join questions_tags T2 on Q2.questionid = T2.questionid
  where T2.tag = 'python')")
length(result$userid) #18611
```

```
## [1] 18611
```

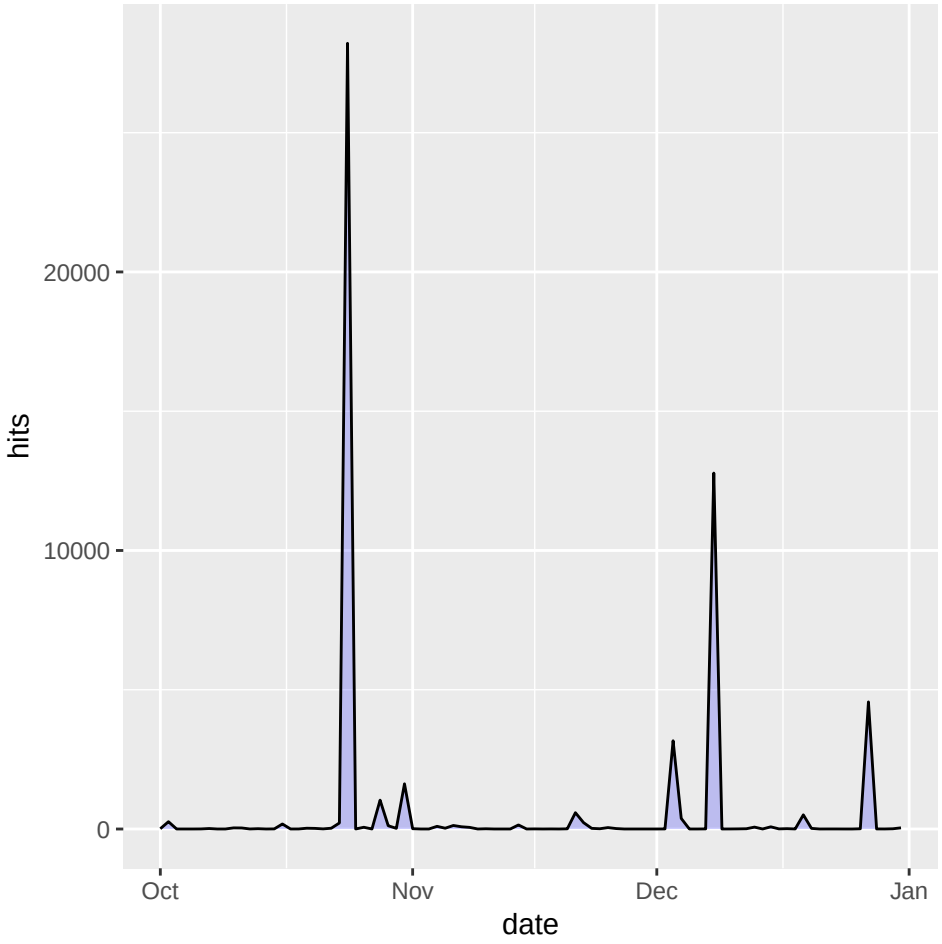
```
dbDisconnect(db)
```

Problem 3

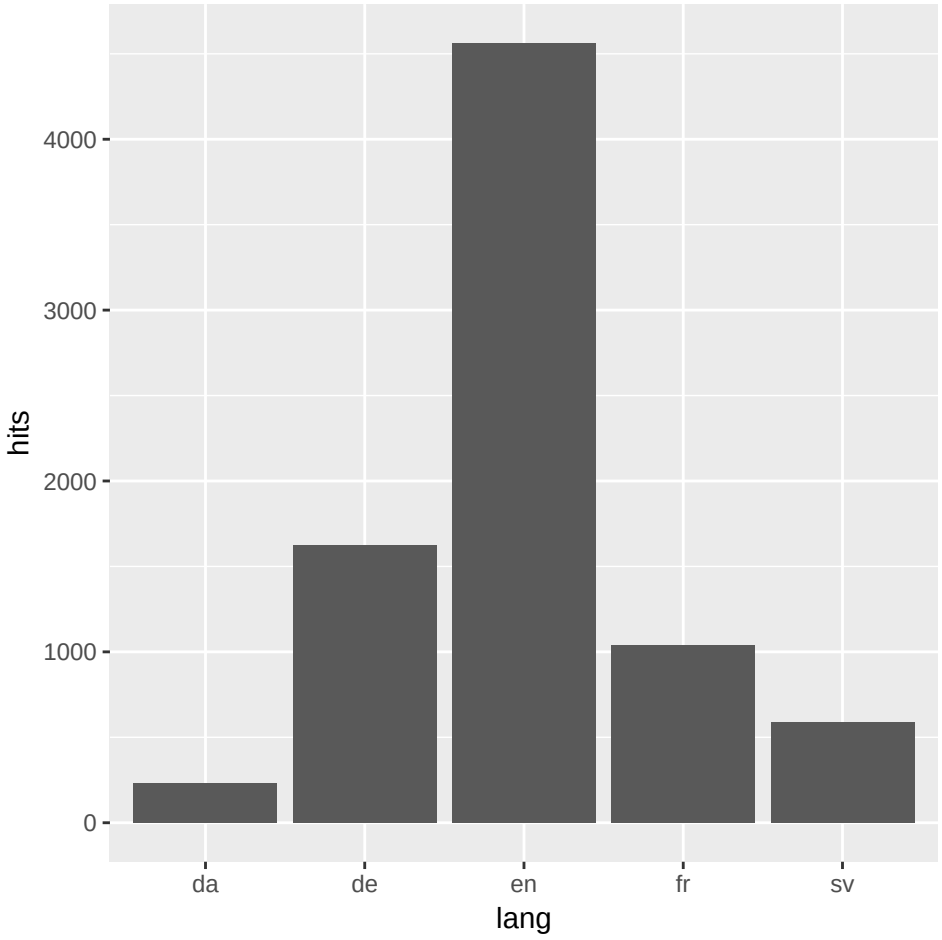
```
dir = '/global/scratch/paciorek/wikistats_full'
lines = sc.textFile(dir + '/' + 'dated')
lines.getNumPartitions() # 16800
lines.count() # 9467817626

import re
from operator import add

# find all webpage related to thanksgiving
def find(line, regex = "Thanksgiving", language = None):
  vals = line.split(' ')
  if len(vals) < 6:
    return(False)
  tmp = re.search(regex, vals[3])
  if tmp is None or (language != None and vals[2] != language):
    return(False)
  else:
    return(True)
ts = lines.filter(find).repartition(480)

def computeKeyValue(line):
  # create key-value pairs where:
  # key = date and language
  # value = number of website hits
  vals = line.split(' ')
  return(vals[0] + '-' + vals[2], int(vals[4]))

counts = ts.map(computeKeyValue).reduceByKey(add)

def transform(vals):
  # split key info back into separate fields
  key = vals[0].split('-')
  return(",".join((key[0], key[1], str(vals[1]))))
  ### output to file ###

dir2 = '/global/home/users/yingyueluan'
outputDir = dir2 + '/' + 'output'
output = counts.map(transform).repartition(1).saveAsTextFile(outputDir)
```

```
suppressWarnings(library(readr))
suppressWarnings(library(sqldf))

## Loading required package:  gsubfn
```

```
## Loading required package:  proto
## Could not load tcltk.  Will use slower R code instead.

library(ggplot2)
a = read_delim("/Users/lunaluan/Desktop/part-00000", delim = ",", col_names = c("date",
    "lang", "hits"))

## Parsed with column specification:
## cols(
##  date = col_integer(),
##  lang = col_character(),
##  hits = col_integer()
## )

# formatting date as yyyy-mm-dd
a$date = as.Date(as.character(a$date), "%Y%m%d")

## Warning in strptime(x, format, tz = "GMT"):  unknown timezone 'default/America/Los_Angeles'

# getting the number of hits everyday from Oct 1 to Dec 31
b = sqldf("select date, hits from a group by date")
ggplot(b, aes(x = date, y = hits)) + geom_area(fill = "blue", alpha = 0.2) +
    geom_line()
```




```
# find the five languages having the most hits
c = sqldf("select lang, hits from a group by lang order by hits desc limit 5")
ggplot(c, aes(x = lang, y = hits)) + geom_bar(stat = "identity")
```



From the plots, we can see that hits peak at the end of October and at early December. Also, People search "Thanksgiving" the most in English and then in German.

Problem 4

a

```
if (!require(readr)) {
    install.packages("readr")
    spark_install(version = "2.2.0")
}
if (!require(sqldf)) {
    install.packages("sqldf")
    spark_install(version = "2.2.0")
```

```
}
library(doParallel)
library(foreach)
library(readr)
library(sqldf)
nCores <- as.numeric(Sys.getenv("SLURM_CPUS_ON_NODE"))
registerDoParallel(nCores)

index = sprintf("%0.3d", 0:199)
dir = "/global/scratch/paciorek/wikistats_full/dated_for_R/part-00"

# create a function getting the data frame for each file and filter out
# webpages unrelated to Barack_Obama
getInfo <- function(part) {
    data <- read_delim(part, delim = " ", col_names = c("date", "time", "lang",
        "webpage", "hits", "size"))
    dataBO <- sqldf("select * from data where webpage LIKE '%Barack_Obama%'")
}

# for each file, combine the information into a single data frame
results <- foreach(i = index, .combine = rbind, .verbose = TRUE) %dopar% {

    cat("Starting ", i, "th job.\n", sep = "")
    part = paste(dir, i, sep = "")
    tmp = getInfo(part)
    cat("Finishing ", i, "th job.\n", sep = "")
    tmp
}
save(results, file = "/global/home/users/yingyueluan/results.Rda")
```

```
> proc.time()
     user    system   elapsed
7601.575  1389.044   521.975
```

```
load("/Users/lunaluan/Desktop/results2.Rda")
x = head(results)
print(x, right = FALSE)

##   date     time   lang
## 1 20081129 210000 pt
## 2 20081014 190000 en
## 3 20081108 190000 no
## 4 20081128 190001 en
## 5 20081110 160000 et
## 6 20081101 110000 fr
```

```
##   webpage
## 1 Barack_Obama
## 2 Special:AllPages/I_ran_Project_Vote_voter_registration_drive_in_Illinois,_ACORN_was_smack_dab_i
## 3 Bilde:Barack_Obama_2004.jpg
## 4 Early_life_and_career_of_Barack_Obama
## 5 Barack_Obama
## 6 Discuter:Barack_Obama
##   hits size
## 1 86   2032215
## 2  2     25520
## 3  1      7825
## 4 16    760462
## 5  4     55875
## 6  1     20922
```

b

I used parallelized R to create the filtered dataset that just has the Obama-related webpage traffic for the first 200 files. The time elapsed is about 522 seconds. For 960 files, $522 \div \frac{200}{960} \approx 2506$ seconds. Consider I used only one core, it would be $\frac{2506}{96} \approx 26$ seconds.

c

```
if (!require(readr)) {
    install.packages("readr")
    spark_install(version = "2.2.0")
}
if (!require(sqldf)) {
    install.packages("sqldf")
    spark_install(version = "2.2.0")
}
library(doParallel)
library(foreach)
library(readr)
library(sqldf)
nCores <- as.numeric(Sys.getenv("SLURM_CPUS_ON_NODE"))
registerDoParallel(nCores)

index = sprintf("%0.3d", 0:199)
dir = "/global/scratch/paciorek/wikistats_full/dated_for_R/part-00"

getInfo <- function(part) {
    data <- read_delim(part, delim = " ", col_names = c("date", "time", "lang",
        "webpage", "hits", "size"))
```

```
    dataBO <- sqldf("select * from data where webpage LIKE '%Barack_Obama%'")
}

results <- foreach(i = index, .combine = rbind, .verbose = TRUE, .options.multicore = list(preschedu
    {

        cat("Starting ", i, "th job.\n", sep = "")
        part = paste(dir, i, sep = "")
        tmp = getInfo(part)
        cat("Finishing ", i, "th job.\n", sep = "")
        tmp
    }
save(results, file = "/global/home/users/yingyueluan/results.Rda")
```

```
.options.multicore=list(preschedule=TRUE)
> proc.time()
     user    system   elapsed
7715.725  6115.122   783.845
```

Tasks assigned dynamically is faster than prescheduling.

```
tmp = load("/Users/lunaluan/Desktop/results3.Rda")
y = get(tmp)
rm(tmp)
print(head(y), right = FALSE)
```

```
##   date     time   lang
## 1 20081129 210000 pt
## 2 20081014 190000 en
## 3 20081108 190000 no
## 4 20081128 190001 en
## 5 20081110 160000 et
## 6 20081101 110000 fr
##   webpage
## 1 Barack_Obama
## 2 Special:AllPages/I_ran_Project_Vote_voter_registration_drive_in_Illinois,_ACORN_was_smack_dab_i
## 3 Bilde:Barack_Obama_2004.jpg
## 4 Early_life_and_career_of_Barack_Obama
## 5 Barack_Obama
## 6 Discuter:Barack_Obama
##   hits size
## 1 86   2032215
## 2  2     25520
## 3  1      7825
## 4 16    760462
## 5  4     55875
## 6  1     20922
```