Assignment4

Cyril 20109216

7 May 2020

```r
rm(list=ls())

# NOTE: ALL LIBRARIES ARE NOT NEEDED
list.of.packages <- c("data.table", "fasttime",'plyr',"PerformanceAnalytics",
                      'imputeTS',"parallel","doParallel","doMC",'lubridate',
                      'anytime','xts','TTR','missRanger','quantmod')
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages,repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/cyril/Documents/R/win-library/3.6'
## (as 'lib' is unspecified)

## Warning: package 'doMC' is not available (for R version 3.6.2)
```

```r
invisible(lapply(list.of.packages, require, character.only = TRUE))
```

```
## Loading required package: data.table

## Warning: package 'data.table' was built under R version 3.6.3

## Loading required package: fasttime

## Loading required package: plyr

## Loading required package: PerformanceAnalytics

## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'xts'

## The following objects are masked from 'package:data.table':
##
##     first, last

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend
## Loading required package: imputeTS

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.fracdiff    fracdiff
##   residuals.fracdiff fracdiff

##
## Attaching package: 'imputeTS'

## The following object is masked from 'package:zoo':
##
##     na.locf

## Loading required package: parallel

## Loading required package: doParallel

## Warning: package 'doParallel' was built under R version 3.6.3

## Loading required package: foreach

## Warning: package 'foreach' was built under R version 3.6.3

## Loading required package: iterators

## Loading required package: doMC

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'doMC'

## Loading required package: lubridate

##
## Attaching package: 'lubridate'

## The following object is masked from 'package:plyr':
##
##     here

## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following object is masked from 'package:base':
##
##     date

## Loading required package: anytime

## Loading required package: TTR

## Loading required package: missRanger

## Warning: package 'missRanger' was built under R version 3.6.3

## Loading required package: quantmod

## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```r
# LOAD ENVIRONMENT
if(Sys.info()['sysname'] == "Windows" ) {
  library(doParallel)
  registerDoParallel(cores=detectCores())
} else {
  library(doMC)
  registerDoMC(cores=detectCores())
}
```

Question 1

```r
M = 20000 # M is the number of hypothesis tests
L = 1000 # Number of tests
pi0 = 0.9 # proportion of null
beta_para = c(0.5,1)
uni_para = c(0,1)
thres = 0.1

gen_data = function(pi0,beta_para,uni_para) {
  pi1 = 1-pi0
  # y is the hidden variable
  # indicating null or non-null
  y = sample(0:1,M,replace = T,prob = c(pi0,pi1))
  mu = rep(0,M)
  mu[y==0] = runif(sum(y==0),min=uni_para[1],max=uni_para[2])
  mu[y==1] = rbeta(sum(y==1),shape1=beta_para[1],shape2=beta_para[2])
  mu
}

beta = function(p,para) {
  para[1]*(p^(para[1]-1)*(1-p)^(para[2]-1))
}

analytical = function(pi0,p,para) {
  pi1 = 1 - pi0
  (pi0) / (pi0 + pi1 * beta(p,para))
}

get_lfdr = function(pi0,beta_para,uni_para) {
  mu = gen_data(pi0,beta_para,uni_para)
  post = analytical(pi0,mu,beta_para)
  post = 1 - post
  post
}

get_pvalues = function(post,thres) {
  ### Bonferroni correction
  bon = pmin(1,  length(post)* post)
  bon_thres = length(which(bon < thres))
  names(bon_thres) = 'bon'
  ### Benjamini-Hochberg Procedure
  lp <- length(post)
  i <- lp:1L
  o <- order(post, decreasing = TRUE)
  ro <- order(o)
  bh = pmin(1, cummin(lp/i * post[o]))[ro]
  bh_thres = length(which(bh < thres))
  names(bh_thres) = 'bh'
  c(bon_thres,bh_thres)
}

post = get_lfdr(pi0,beta_para,uni_para)

all_experiments = function(L,pi0,beta_para,uni_para) {
  grid.param <- expand.grid(1:L)
  fe <- foreach(param = iter(grid.param, by = "row"),
                .verbose = TRUE, .errorhandling = "pass",
                .multicombine = TRUE, .maxcombine = max(2, nrow(grid.param)),
                .export=c("analytical","beta","gen_data","get_lfdr","get_pvalues","M",'thres'),
    #names(Filter(is.function, mget(ls(".GlobalEnv")[-which(ls(".GlobalEnv") == 'all_experiments')])))
                .packages="foreach")

  fe$args <- fe$args[1]
  fe$argnames <- fe$argnames[1]

  results <- fe %dopar% {
    post = get_lfdr(pi0,beta_para,uni_para)
    count = get_pvalues(post,thres)
  }
  results = do.call(rbind,results)
  results
}

results = invisible(all_experiments(L,pi0,beta_para,uni_para))
stopImplicitCluster()
```

```
print(apply(results,2,quantile))
```

```
##      bon bh
## 0%     0  0
## 25%    0  0
## 50%    0  0
## 75%    0  0
## 100%   0  0
```

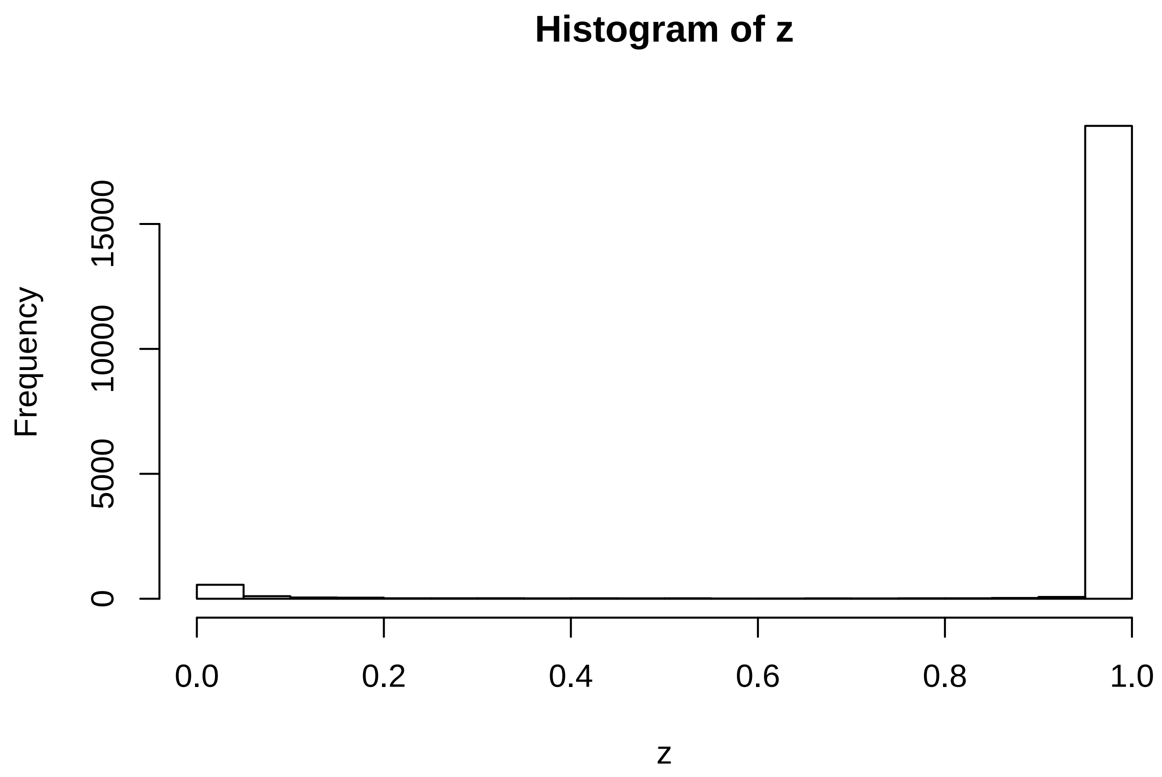
Question 2

```
M = 20000
pi0 = 0.95


gen_data= function(M,pi0) {
  pi1 = 1-pi0
  y = sample(0:1,M,replace = T,prob = c(pi0,pi1))
  mu = rep(0,M)
  mu[y==0] = rnorm(sum(y==0),0,0.5)
  mu[y==1] = rnorm(sum(y==1),2.5,0.5)
  mu
}
```

```
analytical = function(mu) {
  mu_ml = sum(mu) / length(mu)
  cov_ml = (1/length(mu)) * sum((mu - mu_ml) * (mu - mu_ml))
  1/(2 * pi *  abs(cov_ml))^0.5 * exp(-0.5 * (mu - mu_ml) * cov_ml^-1 * (mu - mu_ml))
}

convo = function(mu,z) {
  exp(-0.5 * (mu - z) * (var(mu) + var(z))^-1 * (mu - z)) / (2 * pi *  abs(var(mu) + var(z))^0.5)
}

conv_variance = function(pi0) {
  pi0* 0.5 + (1-pi0) * 0.5 + (pi0 * 0^2 + (1-pi0)*2.5^2 - (pi0*0+(1-pi0)*2.5)^2)
}

analytical_better = function(pi0,z) {
  pi0 * dnorm(z,0,sqrt(conv_variance(pi0))) + (1- pi0) * dnorm(z,2.5,sqrt(conv_variance(pi0)))
}


get_lfdr = function(pi0,mu) {
  pi1 = 1 - pi0
  (pi1 * dnorm(mu,2.5,sqrt(0.5))) / (pi0 * dnorm(mu,0,sqrt(0.5)) + pi1 * dnorm(mu,2.5,sqrt(0.5)))
}

# E(mu | z)
posterior = function(pi0,z) {
  pi1 = 1 - pi0
  (pi1 * dnorm(z,2.5,0.5)) / (pi0 * dnorm(z,0,0.5) + pi1 * dnorm(z,2.5,0.5))
}

# Part A
## sanity check: should be equal to 0
mu = gen_data(M,pi0)
print(var(mu) - conv_variance(pi0))
```

```
## [1] -0.2577144
```

```
# Part B
z =  1 - get_lfdr(pi0,mu)
hist(z)
```



```
# Part C
post = 1 - posterior(pi0,mu)
```

Question 3

```
### EM
M = 10000
alpha = 3^-1
iteration =100
for(i in 1:iteration) {
  mu = rnorm(M,0,sqrt(alpha))
  z =rnorm(M,mu,1)
  alpha_new = M / sum(1/(1+alpha) + z^2/(1+alpha)^2)
  alpha = alpha_new
}
print(alpha)
```

```
## [1] 1.005354
```

```
## James Stein
alpha = 3^-1

js = function(draw) {
  l2 <- sum(draw^2)
  return((1 - (length(draw) - 2) / l2) *draw)
}
mu = rnorm(M,0,sqrt(alpha))
mu_js = js(mu)
```

```r
print(var(mu_js))
```

```
## [1] 1.296584
```

Hence, we conclude James Stein estimator have a larger variance

Question 4

I did not have enough time I am affraid, I took 7courses this semester. My apologies.