Summary on *Estimating False Discovery Proportion Under Arbitrary Covariance Dependence*

Liu Huihang SA18017026

11/16/2019

Theoretic Summary

This paper focuses on estimating FDP in high-dimensional multiple testing under complex covariance structures. They assume that the covariance structures and variance of noise are known and $\hat{\beta}$ is unbiased estimator. Then the test statistics have explicit distribution

$$(Z_1, \ldots, Z_p)^T \sim N\left((\mu_1, \ldots, \mu_p)^T, \mathbf{\Sigma}\right) \tag{1}$$

To handle the complex covariance structure, they decompose the covariance matrix Σ by eigen vectors γ_i and eigen values λ_i as

$$\mathbf{\Sigma} = \sum_{i=1}^{k} \lambda_i \gamma_i \gamma_i^T + \sum_{i=k+1}^{p} \lambda_i \gamma_i \gamma_i^T = \mathbf{L}\mathbf{L}^T + \mathbf{A} \tag{2}$$

and the test statistics can be written as

$$Z_i = \mu_i + \mathbf{b}_i^T \mathbf{W} + K_i, \quad i = 1, \ldots, p. \tag{3}$$

where $\mathbf{b}_i$ is a known $k \times 1$ matrix given by γ, $\mathbf{W}$ and K_i are unknown random variables from $N(0, \mathbf{I}_k)$ and $N(0, \mathbf{A})$.

This is a smart method, it decompose the covariance matrix into two part, one is W who contains the most information of $\mathbf{\Sigma}$, another one is insignificant K_i who is a weakly dependent vector if k is chosen appropriately. We know that mutiple test problem with weakly dependent covariance structure is handled before. And unknown W can be obtained by linear regression, which is a data driven method. Thus, the problem can be handled appropriately.

The paper gives the following reuslt under some conditions

$$\lim_{p \to \infty} \left\{ \mathrm{FDP}(t) - \frac{\sum_{i \in \text{ litre null}} \left[\Phi\left(a_i\left(z_{t/2} + \eta_i\right)\right) + \Phi\left(a_i\left(z_{t/2} - \eta_i\right)\right)\right]}{\sum_{i=1}^{p} \left[\Phi\left(a_i\left(z_{t/2} + \eta_i + \mu_i\right)\right) + \Phi\left(a_i\left(z_{t/2} - \eta_i - \mu_i\right)\right)\right]} \right\} = 0 \text{ a.s.} \tag{4}$$

To calculate $\mathbf{W}$, I use the L1 regression as following

$$\widehat{\mathbf{w}} \equiv \operatorname{argmin}_{\beta \in R^k} \sum_{i=1}^{m} \left| Z_i - \mathbf{b}_i^T \beta \right| \tag{5}$$

More results are shown in the paper, including choosing k, calculating $\hat{\mathbf{W}}$, esitmating realized FDP and asymptotic justification to $\hat{\mathbf{W}}$. I didn't focus on them, but I took a lot time on simulation and real data analysis.

Real Data

I download the data from *ftp://ftp.sanger.ac.uk/pub/genevar/*, and put them in the compressed file.

The structure of the data does not match the statistician's habits. So I took a lot of time to study them, reorganize them, but without result.

I think it's interesting, although it seems in vain.

Simulation Settings

In the simulation studies, we consider $p = 2000$, $n = 100$, $\sigma = 2$, the number of false null hypotheses $p_1 = 10$, and the nonzero $\beta_i = 1$, unless stated otherwise. We will present six different dependence structures for Σ of the test statistics $(Z_1, \ldots, Z_p)T \sim N((\mu_1, \ldots, \mu_p)^T, \Sigma)$. Σ is the correlation matrix of a random sample of size n of p-dimensional vector $\mathbf{X}_i = (X_{i1}, \ldots, X_{ip})$, and $\mu_j = \sqrt{n}\beta_j\hat{\sigma}_j/\sigma, j = 1, \ldots, p$. The data-generating process vector X_i's are as follows.

Equal correlation Let $\mathbf{X}^T = (X_1, \ldots, X_p)^T \sim N_p(0, \Sigma)$, where Σ has diagonal element 1 and off-diagonal element 1/2.

Fan and Song's model For $\mathbf{X} = (X_1, \ldots, X_p)$, let $\{X_k\}_{k=1}^{1900}$ be iid $N(0, 1)$ and

$$X_k = \sum_{l=1}^{10} X_l(-1)^{l+1}/5 + \sqrt{1 - \frac{10}{25}}\epsilon_k, k = 1901, \ldots, 2000 \tag{6}$$

where $\{\epsilon_k\}_{k=1901}^{2000}$ are standard normally distributed.

Independent Cauchy For $\mathbf{X} = (X_1, \ldots, X_p)$, let $\{X_k\}_{k=1}^{2000}$ be iid. Cauchy random variables with location parameter 0 and scale parameter 1.

Three factor model For $\mathbf{X} = (X_1, \ldots, X_p)$, let

$$X_j = \rho_j^{(1)}W^{(1)} + \rho_j^{(2)}W^{(2)} + H_j \tag{7}$$

where $W^{(1)} \sim N(2, 1)$, $W^{(2)} \sim N(1, 1)$, $W^{(3)} \sim N(4, 1)$, $\rho_j^{(1)}$, $\rho_j^{(2)}$, $\rho_j^{(3)}$ are iid $U(1, 1)$, and H_j are iid $N(0, 1)$.

Two factor model For $\mathbf{X} = (X_1, \ldots, X_p)$, let

$$X_j = \rho_j^{(1)}W^{(1)} + \rho_j^{(2)}W^{(2)} + H_j \tag{8}$$

where $W^{(1)}$ and $W^{(2)}$ are iid $N(0, 1)$, $\rho_j^{(1)}$ and $\rho_j^{(2)}$ are iid $U(1, 1)$, and H_j are iid $N(0, 1)$.

Nonlinear factor model For $\mathbf{X} = (X_1, \ldots, X_p)$, let

$$X_j = \sin\left(\rho_j^{(1)}W^{(1)}\right) + sgn\left(\rho_j^{(2)}\right)\exp\left(\left|\rho_j^{(2)}\right|W^{(2)}\right) + H_j \tag{9}$$

where $W^{(1)}$ and $W^{(2)}$ are iid $N(0, 1)$, $\rho_j^{(1)}$ and $\rho_j^{(2)}$ are iid $U(1, 1)$, and H_j are iid $N(0, 1)$.

I listed them here, because the covariance structure listed in the article is very detailed and be worthy of marking.

My code and results

In the following, I try to repeat the result in simulation 1 and theorem 1 of paper. I use the same setting in the following as described in the paper.

To get the distribution of FDR and $\widehat{FDR}$, I generate $X \sim N(0, \Sigma)$ in Equal correlation structure. Then calcuate $\mathbf{Z}$ by $Z_i = \frac{\hat{\beta}_i}{\sigma/(\sqrt{n}\hat{\sigma})}$.

By equation (10) in the paper, we can write Z_i as

$$Z_i = \mu_i + \mathbf{b}_i^T\mathbf{W} + K_k, \quad i = 1, \ldots, p. \tag{10}$$

To calculate $\mathbf{W}$, I use the L1 regression as following

$$\widehat{\mathbf{w}} \equiv \text{argmin}_{\beta \in R^k} \sum_{i=1}^{m} \left|Z_i - \mathbf{b}_i^T\beta\right| \tag{11}$$

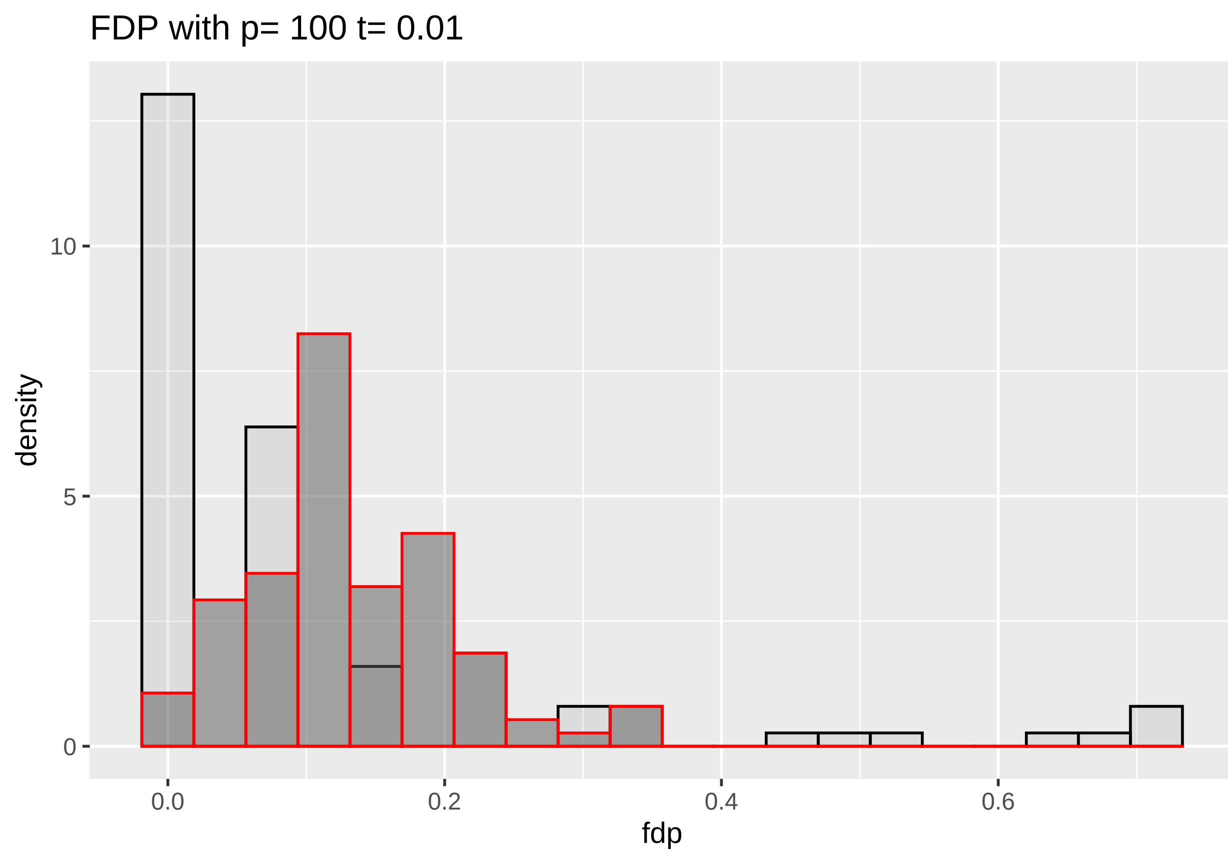
which is robust. And L1 regression is done by **l1fit** defined in package **L1pack**.

To accelerate the computation, I use 40 CUPs working paralleled supported by package **snowfall**. So, the following code will not take a long time.

```r
library(MASS, snowfall, ggplot2, L1pack)
# snowfall for parallel computation, L1pack for L1 regression

set.seed(12345)

n <- 100; rho <- 0.5; sig <- 2; p.nonzero <- 10; beta.nonzero <- 1

# FDP and FDP_lim at t
fdp <- function(t){
  ##  FDP
  Z <- MASS::mvrnorm(1, mu, Sigma)
  pvalue <- unlist(base::lapply(X=1:p, FUN=function(ii) 1-pnorm(abs(Z[ii]))))
  tmp.pvalue <- pvalue[(1+p.nonzero):p]
  re1 <- length(which(tmp.pvalue < t)) / length(which(pvalue < t))

  ##  FDP_lim
  # k is dimension of W
  k <- 2
  # m.idx contains smallest 90% of |zi|'s indexes
  m.idx <- order(abs(Z), decreasing=TRUE)[(0.1*p+1):p]
  # x is the first k cols, eq(22)
  x.tmp <- (diag(sqrt(Sigma.eigen$values)) %*% Sigma.eigen$vectors)[m.idx, 1:k]
  y.tmp <- Z[m.idx]
  # L1 regression by eq(23)
  W <- L1pack::l1fit(x=x.tmp, y=y.tmp, intercept=FALSE)$coefficients
  # b is given by eq(22)
  b <- diag(sqrt(Sigma.eigen$values)) %*% Sigma.eigen$vectors[, 1:k]
  # numerator is given by eq(12)
  numerator <- sum(unlist(base::lapply(X=1:p.nonzero, FUN=function(ii) {
    ai <- (1 - sum((b[ii, ])^2))^(-0.5)
    pnorm(ai*(qnorm(t/2) + b[ii,] %*% W)) + pnorm(ai*(qnorm(t/2) -
                                                    b[ii,] %*% W))})))
  # eq(12)
  denominator <- sum(unlist(base::lapply(X=1:p, FUN=function(ii) {
    ai <- (1 - sum((b[ii, ])^2))^(-0.5)
    pnorm(ai*(qnorm(t/2) + b[ii,] %*% W + mu[ii])) +
      pnorm(ai*(qnorm(t/2) - b[ii,] %*% W - mu[ii]))})))
  # eq(12)
  re2 <- numerator / denominator
  return(rbind(re1, re2))
}

my.fun <- function(p, t){
  # Equal correlation
  beta <- c(rep(beta.nonzero, p.nonzero), rep(0, p-p.nonzero))
  Sigma <- matrix(rep(rho, p*p), p, p); diag(Sigma) <- rep(1, p)
  dat <- MASS::mvrnorm(n, rep(0,p), Sigma)
  Sigma.eigen <- eigen(Sigma)
  mu <- unlist(base::lapply(X=1:p, FUN=function(ii)
    sqrt(n)*beta[ii]*sqrt(var(dat[, ii]))/sig))
```

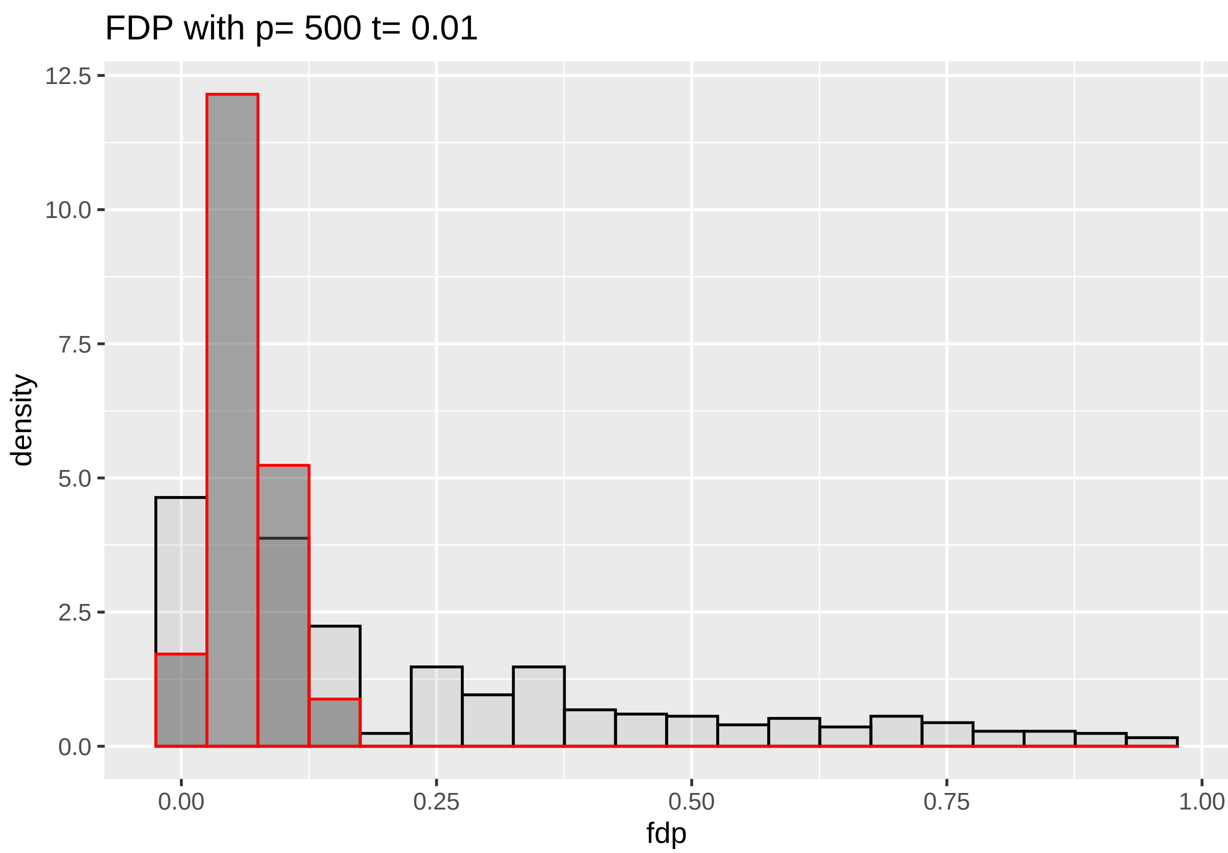
```r
  # parallel calculation
  snowfall::sfInit(parallel = TRUE, cpus = 40)
  snowfall::sfLibrary(MASS)
  snowfall::sfLibrary(L1pack)
  snowfall::sfExport("p", "mu", "Sigma", "t", "p.nonzero", "Sigma.eigen", "rho")
  fdp.repeat <- unlist(snowfall::sfLapply(rep(0.01, 1000), fdp))
  snowfall::sfStop()

  # figure
  tmp.data <- data.frame(fdp=fdp.repeat[(1:p)*2-1])
  tmp.data.lim <- data.frame(fdp=fdp.repeat[(1:p)*2])
  pic <- ggplot()
  pic <- pic + geom_histogram(data=tmp.data, aes(fdp, y=..density..), bins=20,
                              color=1, alpha=0.1)
  pic <- pic + geom_histogram(data=tmp.data.lim, aes(fdp, y=..density..),
                              bins=20, color=2, alpha=0.5)
  pic <- pic + ggtitle(paste("FDP with p=", p, "t=", t, sep=' '))
  plot(pic)
}

my.fun(p=100, t=0.01)
```

```
## Warning in searchCommandline(parallel, cpus = cpus, type = type, socketHosts =
## socketHosts, : Unknown option on commandline: rmarkdown::render('/home/huihang/
## Documents/GWAS/Simulation.Rmd',~+~~+~encoding~+~

## R Version:  R version 3.6.1 (2019-07-05)

## snowfall 1.84-6.1 initialized (using snow 0.4-3): parallel execution on 40 CPUs.

## Library MASS loaded.

## Library MASS loaded in cluster.

## Library L1pack loaded.

## Library L1pack loaded in cluster.

##
## Stopping cluster
```



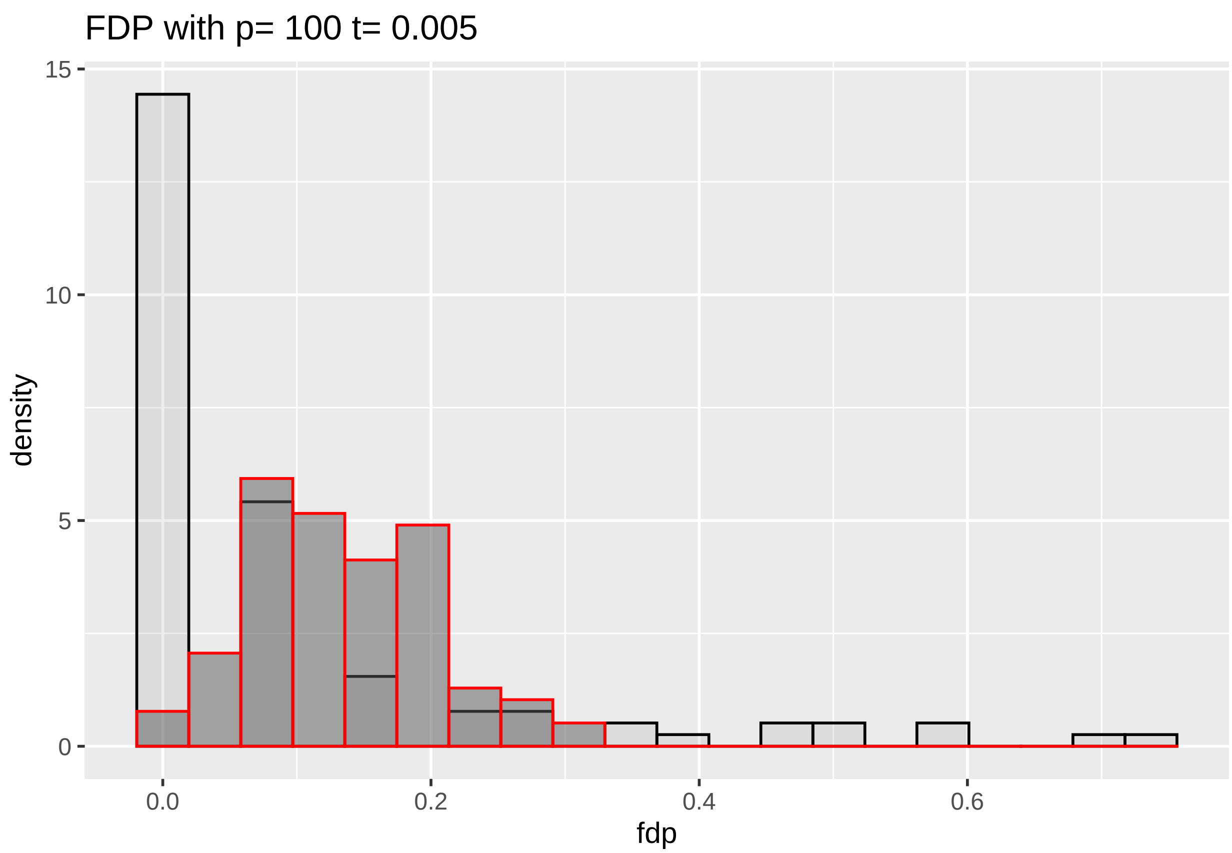

```r
my.fun(p=500, t=0.01)
```

```
## Warning in searchCommandline(parallel, cpus = cpus, type = type, socketHosts =
## socketHosts, : Unknown option on commandline: rmarkdown::render('/home/huihang/
## Documents/GWAS/Simulation.Rmd',~+~~+~encoding~+~

## snowfall 1.84-6.1 initialized (using snow 0.4-3): parallel execution on 40 CPUs.

## Library MASS loaded.

## Library MASS loaded in cluster.

## Library L1pack loaded.

## Library L1pack loaded in cluster.
##
##
## Stopping cluster
```



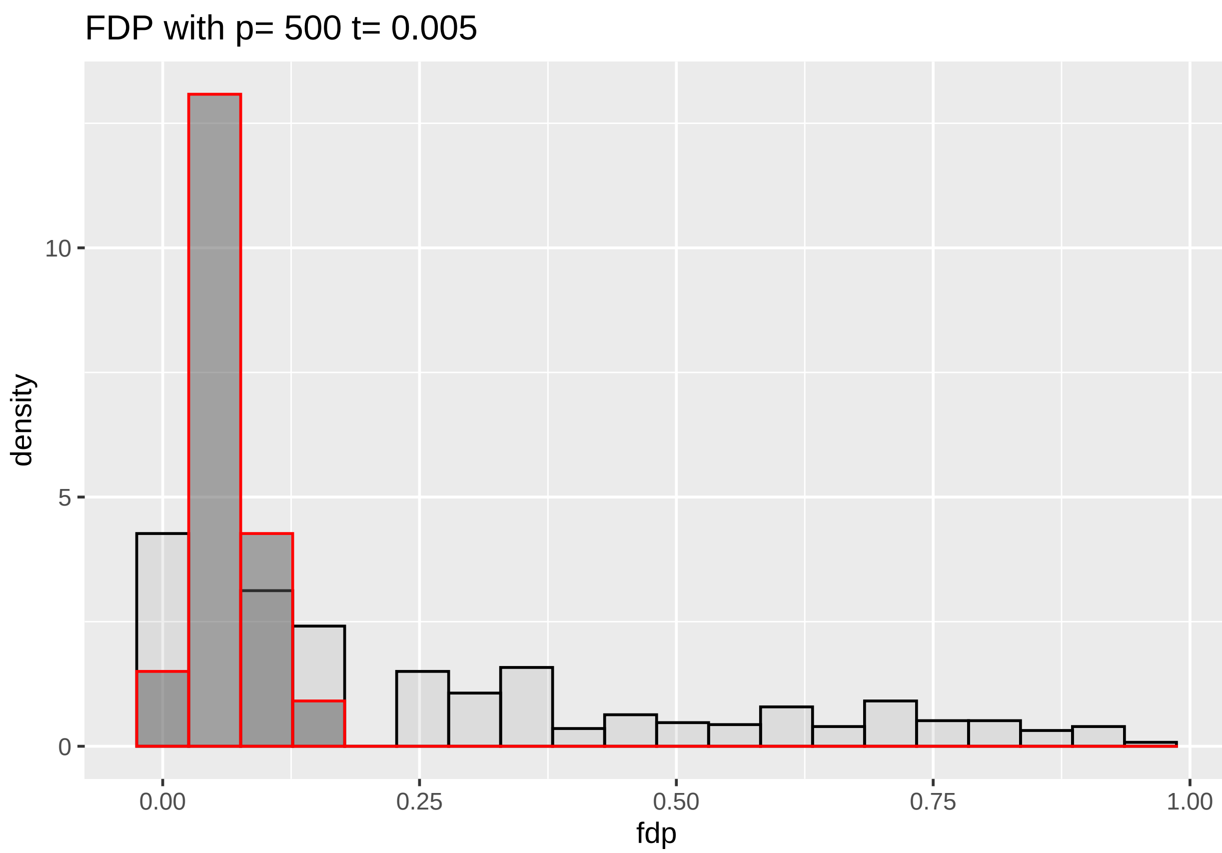

```
my.fun(p=100, t=0.005)
```

```
## Warning in searchCommandline(parallel, cpus = cpus, type = type, socketHosts =
## socketHosts, : Unknown option on commandline: rmarkdown::render('/home/huihang/
## Documents/GWAS/Simulation.Rmd',~+~~+~encoding~+~

## snowfall 1.84-6.1 initialized (using snow 0.4-3): parallel execution on 40 CPUs.

## Library MASS loaded.

## Library MASS loaded in cluster.

## Library L1pack loaded.

## Library L1pack loaded in cluster.
##
##
## Stopping cluster
```




```r
my.fun(p=500, t=0.005)
```

```
## Warning in searchCommandline(parallel, cpus = cpus, type = type, socketHosts =
## socketHosts, : Unknown option on commandline: rmarkdown::render('/home/huihang/
## Documents/GWAS/Simulation.Rmd',~+~~+~encoding~+~

## snowfall 1.84-6.1 initialized (using snow 0.4-3): parallel execution on 40 CPUs.

## Library MASS loaded.

## Library MASS loaded in cluster.

## Library L1pack loaded.

## Library L1pack loaded in cluster.
##
##
## Stopping cluster
```



The grey bars in the figures is the density of FDR and the red bars in the figures represent the density of $\widehat{FDR}$.

From the figures above, I find both the true FDR and $\widehat{FDR}$ are similiar with the result in the paper. I am satisfied with the result, althought they have some differences shown in the figure. But the paper just show the result of two factor model, so I cannot compare them.

I suppose it is because my $\hat{W}$ is some kind of incorrect or biased, maybe.