Multiple Testing

Javier Esteban Aragoneses, Mauricio Marcos Fajgenbaun, Danyu Zhang, Daniel Alonso

February 18, 2021

Introduction

In this research, we will try to study which genes are the most relevant in determining the concentration of a specific protein and get a lower bound for the false discovery rate.

We are going to use data containing information on 8 SNP (Single Nucleotid Polymorfism) of 100 individuals. The DNA is a combination of nucleobasis (cytosine, guanine,adenine and thymine) and these combination sequence are independent, so we will not have to worry about multicolinearity during this study.

Methods

First, in order to determine if a gen is relevant we will perform a simple linear regression where the dependent variable will be the concentration of a specific protein, and the independent covarite will be a specific gen. We will do this, for evert gen having in total 8 linear regressions. When doing a linear regression we have to assume the following:

a) There is linearity between the covariate and the dependent variable.

b) Normality of residuals.

c) Homocesdaticity across residuals (constant variance).

d) Independance between residuals (we already justified with by saying that the sequences are independent).

As every one of the gens can only take value of: 1,2 or 3, we will treat then as factors. This way, every single regression will be of the type:

$$Concentration = B_0 + B_2 * snp_2 + B_3 * snp_3$$

Where, B_0 is the estimation of the concentration of the protein, when the gen takes value equal to 1, B_2 is the increase or decrease (depending on the sign) in concentration when the gen takes value equal to 2, and B_3 represents the increase or decrease of concentration when the gen takes value equal to 3.

Once we get all the regressions done, we would have to check the assumptions stated before.

Then, we will have 8 different p-values. The p-values are going to be responding to the following hypothesis test:

H_0: The concentration is independent from the specifical gen (beta is significantly not different from zero).

H_1: The concentration and the specifical gen have some kind of significative relationship

As we are doing multiple testing, we have to find a better meassure, as when we do many tests, we have more change of getting a strange result. So we want to control the error, and not the p-value.

In addition, we have to get the *lower bound*. It is the probability of the null hypothesis (the target variable is independent from the particular predictor, which means that that particular *snp* does not contribute the

concentration of protein in the blood for such corresponding individual) given our specific dataset. We want this value to be *0.05*, which means that we want to reduce the probability of committing a type I error. However, by reducing this type I error incidence, it is likely that we increase the incidence of type II errors (meaning we might not take into account the relevent variables as a result of the strict criterion that we set for variable selection)

This lower bound, assuming that p-value is uniformly distributed in the interval $(0, 1)$, will be computed as follows:

$$E[H_0|data] = \frac{-e * p * log(p)}{q - e * p * log(p)}$$

We will use 5 methods, however, in order to be sufficiently sure that a gene is relevant to predict cocentration of protein in blood, we will only confidently select an *snp* if all of these methods' results coincide.

Method 1

The first process utilized to controll the error is Bonferroni. With this, we control the family wise error rate $FWER = Pr(R > 0)$.

When using this, the wrong decisions are strongly penalized (we can say it is a restrictive method).

$$p < \frac{\alpha}{m}$$

Method 2

After doing this, we will use the Benjamini-Hochberg method, that controlls the False Discovery Rate: $FDR = E(\frac{R}{M0})$.

This is a less restrictive method than Bonferroni, as now we will be multiplying every p-value by its ordered tag (after ordering them from smallest to biggest).

$$p_{(i)} < \alpha \frac{i}{m}$$

Method 3

The Benjamini-Yekutieli (BY) controls the FDR without assuming a positive correlation dependence on tests, but it is more conservative.

$$p_{(i)} \leq \frac{i}{m * c(m)}\alpha, \quad \text{where} \quad c(m) = \sum_{i=1}^{m} \frac{1}{i}$$

Method 4

The q-value of a test measures the proportion of false positives incurred (called the false discovery rate) when that particular test is called significant.

The local FDR measures the posterior probability the null hypothesis is true given the test's p-value.

Method 5

z are the data (p-value as a supposed standard normal r.v.): $z = \Phi - 1(p)$

The local FDR measures the posterior probability the null hypothesis is true given the test's p-value.

Here z is the data: the p-values under normality ($z = \pi_0$) is the prior probability of the null H0. We name $f_0(z)$ to the density, $f_1(z)$ is the density under the alternative H_1.

The predictive or marginal density of z is:

$$f(z) = \pi_0 f_0(z) + (1 - \pi_0) f_1(z)$$

Once this is defined, we can define LFDR:

$$lFDR(z) = Pr(H_0|Z = z) = \pi_0 \frac{f_0(z)}{f(z)}$$

This is a much conservative approach.

We perform a linear model using each variable as a predictor of concentration of protein in blood.

Linear models for each *snp*

Linear model for *snp1*

For the *snp1* we get a p-value of ~**0.0003**

According to the p-value criterion, and taking type I error (which is the case in which the genetic profile does not contribute), however we take that variable as relevent and equal to 0.05, the first *snp1* is significant.

Linear model for *snp2*

For the *snp2* we get a p-value of ~**0.0005**

By using the p-value criterion and taking type I error equal to 0.05, *snp2* is significant.

Linear model for *snp3*

For the *snp3* we get a p-value of ~**0.4**

By using the p-value criterion and taking type I error equal to 0.05, *snp3* is not relevant to contributing the concentration of protein in blood.

Linear model for *snp4*

For the *snp4* we get a p-value of ~**0.01**

By using the p-value criterion and taking type I error equal to 0.05, *snp4* is relevant to contributing the concentration of protein in blood.

Linear model for *snp5*

For the *snp5* we get a p-value of ~**0.2**

By using the p-value criterion and taking type I error equal to 0.05, *snp5* is not relevant at contributing the concentration of protein in blood.

Linear model for *snp6*

For the *snp6* we get a p-value of ~**0.7**

By using the p-value criterion and taking type I error equal to 0.05, *snp6* is not relevant at contributing the concentration of protein in blood.

Linear model for *snp7*

For the *snp7* we get a p-value of ~**0.8**

By using the p-value criterion and taking type I error equal to 0.05, *snp7* is not relevant at contributing the concentration of protein in blood.

Linear model for *snp8*

For the *snp8* we get a p-value of ~**0.6**

By using the p-value criterion and taking type I error equal to 0.05, *snp8* is not relevant at contributing the concentration of protein in blood.

Finally, we checked the model assumptions, and confirm there is no problem assuming them true.

p-values for each linear model

Table 1: p-vals and lower bounds

model	pval
snp1	0.00
snp2	0.35
snp3	0.96
snp4	0.05
snp5	0.90
snp6	0.74
snp7	0.68
snp8	0.23

If we were to use the p-value rule, where $p - val < 0.005$ from the performed linear models, *snp1* and *snp2* are both significant.

Multitest

We perform a multitest, where we obtain a result for each test (Bonferroni, Benjamini-Hochberg and Benjamini-Yekutieli).

Table 2: Multitest

Bonferroni	BH	BY
0.013	0.013	0.036
0.389	0.194	0.528
1.000	0.618	1.000
1.000	0.698	1.000
1.000	0.962	1.000
1.000	0.962	1.000
1.000	0.962	1.000
1.000	0.962	1.000

Assumptions

On one hand, the linear model assumptions are true. However, we can't assert that our p-values are uniformly distributed beacause we dont have enough models.

So we decide to use the empirical null distribution:

$$\underline{\alpha}^*(p) = \frac{1}{1 + B^*(p)}$$

Where:

$$\underline{B}^*(p) = \begin{cases} -\widehat{\xi}_0 p^{\widehat{\xi}_0} e \log p & p < e^{-1/\widehat{\xi}_0} \\ 1 & otherwise \end{cases}$$

With this method, our lower bound is equal to zero, perhaps we do not have a sufficient enough models/p-values in order to utilize this method. Therefore, we will assume that in the hypothetical scenario where we had more models, our p-value could be uniformly distrubuted, and it would be reasonable to use use the first method to calculate the lower bound, in order to present a more conservative (and accurate) result.

Results

The most relevant (and most of the times uniquely relevant) genetic profile marker for each method is *snp1*, we can see this as follows:

Bonferroni

Only the *snp1* is significant.

Benjamini-Hochberg test

Only the *snp1* is significant.

Benjamini-Yekutieli test

Only the *snp1* is significant.

Qvalue

For Q-value, the only significant genetic profile markers are *snp1* and *snp4*.

Local false discovery rate

For the Local false discovery rate, *snp1* and *snp3* are significant.

Final selection

We only select *snp1* as a relevant genetic profile marker because is the only one that is significant for all methods. As it is a health-related matter, we intend to be strict when it comes to selecting the gene that predicts concentration of protein in blood the most accurately.

Lower bound of false discovery rate

In order to select the lower bound, we choose the Benjamini-Yekutieli method as is the most restrictive.

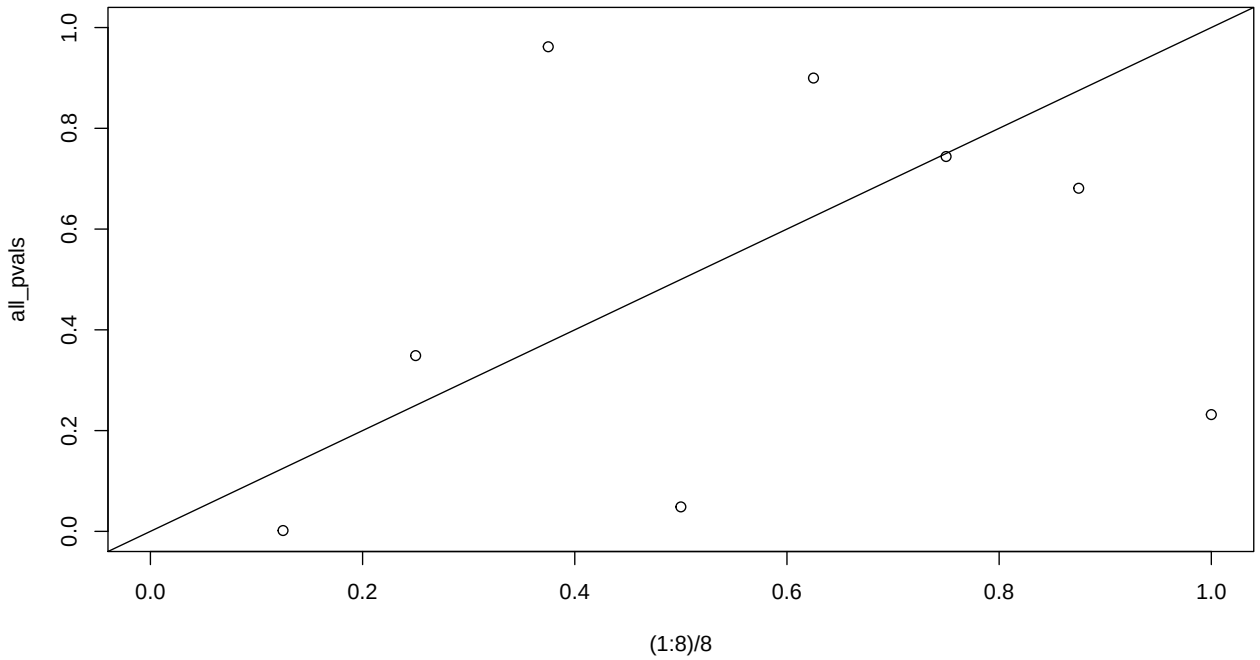
Table 3: p-vals and lower bounds

	pval	Prob..of.H_0.given.the.data
snp1	0.04	0.25

Appendix

Checking that the p-value is uniformly distributed

```
plot((1:8)/8,all_pvals, xlim=c(0,1), ylim=c(0,1))
abline(0,1)
```



We do not have enough data to confidently say that the p-value is or is not uniformly distributed.

Code used to obtain the results in the report

```
# importing libraries
library(dplyr)
library(multtest)
library(qvalue)
library(locfdr)


gen <- read.csv('./data/gendata.csv', sep=';', header=TRUE)
gen <- gen[2:length(names(gen))]
for (i in 2:length(names(gen))) {
    gen[,i] <- as.factor(gen[,i])
}

# function used for p-val
lower_bound <- function(p) {-exp(1)*p*log(p)/(1-exp(1)*p*log(p))}
```

```r
# alpha
alpha = 0.2

# LMs
snp1 <- lm(conc~snp1, gen)
snp2 <- lm(conc~snp2, gen)
snp3 <- lm(conc~snp3, gen)
snp4 <- lm(conc~snp4, gen)
snp5 <- lm(conc~snp5, gen)
snp6 <- lm(conc~snp6, gen)
snp7 <- lm(conc~snp7, gen)
snp8 <- lm(conc~snp8, gen)

# mods
models <- list(snp1, snp2, snp3, snp4, snp5, snp6, snp7, snp8)

#pvals
pvals <- c()
for (i in 1:length(models)) {
    pvals <- c(pvals, summary(models[[i]])$coefficients[2,4])
}
names(pvals) <- names(gen)[2:length(names(gen))]

# lowerbounds
lbs <- c()
for (i in 1:length(models)) {
    lbs <- c(lbs, lower_bound(pvals[i]))
}

# table for pvalues and lower bounds for pvalues
tbl <- data.frame(model = names(pvals),
        pval = round(unname(pvals), 2),
        "P(H_0 | data)" = round(lbs, 2))

knitr::kable(
    tbl,
    booktabs=TRUE,
    caption="p-vals and lower bounds",
)

# multitest
adjp <- data.frame(round(mt.rawp2adjp(pvals,alpha = alpha)$adjp,3))
adjp <- adjp %>% dplyr::select(c("Bonferroni", "BH", "BY"))

knitr::kable(
    adjp,
    booktabs=TRUE,
    caption="Multitest",
)


A.q <- qvalue(p = pvals)$qvalue < alpha
A.q[A.q == TRUE]
```

```r
# local false discovery rate
names(res.n01$fdr) <- names(pvals)
A.lfdr.n01=res.n01$fdr<alpha
A.lfdr.n01[A.lfdr.n01 == TRUE]


#pvals
pvals <- adjp$BY[1]
names(pvals) <- c("snp1")

# lowerbounds
lbs <- lower_bound(pvals)

# table
tbl <- data.frame(pval = round(pvals, 2),
        "P(H_0 | data)" = round(lbs, 2))

knitr::kable(
    tbl,
    booktabs=TRUE,
    caption="p-vals and lower bounds",
)
```