8-2019

Effect of Cross-Validation on the Output of Multiple Testing Procedures

Josh Dallas Price
University of Arkansas, Fayetteville

Effect of Cross-Validation on the Output of Multiple Testing Procedures

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Statistics and Analytics

by

Josh Price
University of Central Arkansas
Bachelor of Science in Mathematics, 2012

August 2019
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

Jyotishka Datta, Ph.D.
Thesis Director

Qingyang Zhang, Ph.D.
Committee Member

John Tipton, Ph.D.
Committee Member

Abstract

High dimensional data with sparsity is routinely observed in many scientific disciplines. Filtering out the signals embedded in noise is a canonical problem in such situations requiring multiple testing. The Benjamini–Hochberg procedure (BH) using False Discovery Rate control is the gold standard in large scale multiple testing. In Majumder *et al.* (2009) an internally cross-validated form of the procedure is used to avoid a costly replicate study and the complications that arise from population selection in such studies (i.e. extraneous variables). I implement this procedure and run extensive simulation studies under increasing levels of dependence among parameters and different data generating distributions and compare results with other common techniques. I illustrate that the internally cross-validated Benjamini–Hochberg procedure results in a significantly reduced false discovery rate, while maintaining a reasonable, though increased, false negative rate, and in a reduction to inherent variability under strong dependence structures when compared with the usual Benjamini–Hochberg procedure. In the discussion section, I describe some possibilities for relevant applications and future studies.

Acknowledgements

I must begin by thanking my thesis advisor Dr. Datta. Throughout the entire writing process he has been readily available and incredibly helpful. Whenever I would get stuck he always had a suggestion. Whenever I had questions he would have an explanation and references for further reading. Most of all, he was patient for which I am very thankful.

I would also like to thank my committee members, Dr. Tipton and Dr. Zhang. Both professors have provided thoughtful and critical feedback, as well as encouragement throughout writing process. Their insights into improving the quality of my writing and probing questions challenged me to become a better student and Statistician.

I am also thankful for my former coworkers at ASU-Beebe who encouraged me to return to college. The work we did compiling survey data during the grant writing process is part of the reason I decided to pursue this degree.

I must also thank my family and friends. You all put up with me throughout my college career, and you've all encouraged me along the way. Pursuing a graduate degree is a lot of work, but it's those people outside of school that are needed to keep students like myself grounded and sane. Thank you all for that.

Josh Price

Table of Contents

Chapter 1

Introduction

Simultaneous hypothesis testing refers to situations in which a multitude of hypothesis tests are simultaneously evaluated on a data set. The number of hypotheses to be tested can range from just three or four (e.g. testing the effect of extracurricular activities on school performance) to well over a million (e.g. testing for significant genes), each incurring a similar problem. When performing a singular hypothesis test, a significance level is set, e.g. $\alpha = 0.05$, and the test proceeds with the consideration that the probability of a type I error is controlled at 5%. By contrast, when twenty such tests are performed, it can be expected that at least one of those tests are likely to show significance by pure chance. Furthermore, it is easy to see that as the number of tests performed increases, so too does the probability of at least one false rejection occurring. Understanding that a false rejection occurs when the null hypothesis for such tests is rejected when the null hypothesis is actually true, i.e. a type I error.

There are many reasons why researchers will want to run simultaneous hypothesis tests. To begin with, it is cheaper to run one test involving a multitude of variables than it would be to run many tests while controlling for other variables. For data collection, it is often easier to ask a few more questions than it is to collect a second sample. In the case of genomic data, a researcher may be interested in finding significant genes among thousands or millions (Efron, 2010)[15-17]. However, we still need to be able to trust the results of our hypothesis tests which has lead statisticians to develop multiple testing methods for controlling simultaneous hypothesis testing error rates.

1.1 Family Wise Error Rate

The classic approach to this problem is to control the Family Wise Error Rate (FWER). The FWER is defined as the probability of making at least one false-rejection in a family of multiple testing problems. Consider such a family of *N* hypothesis tests, the null of which are denoted as H_{0i}, with $i \in 1, 2, ..., N$. Then we can write the FWER as in equation 1.1.

FWER control is achieved by creating an rejection rule such that the probability of at least one false rejection is bounded by α. The most well known method of FWER control is the Bonferroni correction. To make the Bonferroni correction, statisticians divide α by *N* and use the result to carry out the simultaneous hypothesis tests as usual. Thus, letting p_i denote the p-value of the *i*th hypothesis test, each p_i will be compared with $\frac{\alpha}{N}$ to determine whether to reject or fail to reject H_{0i}. Letting N_0 be the number of true null hypothesis, the FWER can be shown to be $N_0 \frac{\alpha}{N} \leq \alpha$, meaning that the Bonferroni correction does control the FWER (Efron, 2010).

$$\text{FWER} = P(\text{Reject any true } H_{0i}) \tag{1.1}$$

While FWER control is widely used and easy to implement, it is not always the most appropriate multiple testing procedure to use. Considering the Bonferroni correction, when 10 tests are ran simultaneously at an $\alpha = 5\%$ level, in order for any test to reject the null the p-value must be less than 0.005, which is already small. Suppose instead there are to be 6,359 tests as in Van Steen *et al.* (2005) where researchers are interested in finding significant genes in mice, then the significance level for each individual test drops to $\frac{\alpha}{N} = \frac{0.05}{6{,}359} = 7.86 \times 10^{-6}$. Clearly as $N \to \infty$, then $\frac{\alpha}{N} \to 0$, and the probability of rejecting any null hypothesis goes to 0. Consequentially, this also leads to a dramatic loss in power for such tests. So as general rule, as *N* grows large FWER control methods are seen to be too conservative in their approach. The more tests there are, the more tests that are likely to be rejected. However, there is another multiple testing method that corrects for this problem.

1.2 False Discovery Rate

What makes FWER control methods so conservative is that these methods attempt to control the probability of there being any false rejections among all parameters to be tested. If this requirement is loosened a little to allow some proportion of our rejections to be false, then we are more likely to capture more true rejections as well. This insight is the motivation behind the False Discovery Rate (hereafter FDR) Control method proposed in Benjamini & Hochberg (1995).

Table 1.1 summarizes the outcome of some multiple testing procedure with N tests and N_0 true null hypotheses. On table 1.1, the number of tests where H_0 is true and not rejected is given by the random variable U, where H_0 is false and not rejected is given by the random variable T, where H_0 is true and rejected is given by the random variable V, and where H_0 is false and is rejected is given by the random variable S. Then $R = S + V$ gives us the total number of tests where the H_0 is rejected, $N - S$ gives the total number of tests where the H_0 is not rejected, and $N - N_0$ gives the total number of false null hypotheses. All of U, V, T, S, R, N, and N_0 are in $\mathbb{N} \cup \{0\}$. In practice, R is an observable random variable, while U, V, T, and S are unobservable random variables. A false discovery is defined as an erroneously rejected true null hypothesis. The proportion of false discoveries among all discoveries, often referred to as the False Discovery Proportion (FDP hereafter), is then given by the random variable $Q = \frac{V}{R}$, with $Q = 0$ when $R = 0$. The FDR is defined as $E(Q)$.

Table 1.1: Hypothesis Test Results

	Fail to Reject H_0	Reject H_0	Total
H_0 True	U	V	N_0
H_0 False	T	S	$N - N_0$
	$N - R$	R	N

Consider the multiple testing setup from before, N tests each with corresponding null hypotheses, H_{0i}, and p-values, P_i. Let $P_{(1)} \leq P_{(2)} \leq ... \leq P_{(N)}$ be the ordered p-values and $H_{0(i)}$ be the null hypothesis corresponding to $P_{(i)}$. Consider the following multiple testing procedure, known as the Benjamini–Hochberg (abbreviated as BH hereafter) procedure:

Let k be the largest i such that

$$P_{(i)} \leq \frac{i}{N} q. \tag{1.2}$$

Then the BH procedure rejects all $H_{0(i)}$ for which $i \leq k$. Otherwise it fails to reject $H_{0(i)}$.

Theorem 1.2.1 (Benjamini & Hochberg (1995))**.** *For independent test statistics and for any configurations of false null hypotheses, the BH procedure controls the FDR at* q*.*

I refer readers to the original paper by Benjamini and Hochberg, Benjamini & Hochberg (1995), for a proof of this theorem. Furthermore, when all null hypotheses are true, FDR control is equivalent to FWER control. And any implementation of FWER control also controls FDR, though FDR control only controls FWER when all null hypotheses are true. In general, FDR control is less stringent, thus an increase in power can be over FWER control can be expected (Benjamini & Hochberg, 1995).

1.3 Dependence in Multiple Testing

Dependence in multiple testing, specifically correlated test statistics, was recognized as a potential problem in Yekutieli & Benjamini (1999) where researches used a resampling-based p-value adjustment method to ensure FDR control and improve test power. Through proof and simulation studies, the resampling-based p-value adjustment is shown to control FDR under the condition of dependent test statistics. (Yekutieli & Benjamini, 1999) Furthermore, in Benjamini & Yekutieli (2001) researchers show that the Benjamini–Hochberg procedure controls the FDR under the assumption of dependent test statistics. Other notable works in this area are (Storey, 2003) and (Genovese & Wasserman, 2002).

Thus we see that FDR control is possible even when we have correlation among our variables. However, the above mentioned studies leave out a crucial detail in the problem of multiple testing with dependence. In none of these studies do the authors consider the variance of the FDR in their proofs or simulation studies. High variance in the FDR can cause considerable problems for researchers, as a researcher could draw drastically different inference using different data drawn from the same distribution with an arbitrary dependence structure.

Fan & Han (2017) developed a method for predicting achieved FDP for a given study that incorporates the dependence structure of the data. Depending on how accurate their estimation is, this result would allow other researchers to better estimate their achieved FDP under dependence

and make better decisions when it comes to seeking out alternative multiple testing procedures.

In the following simulation studies, I show a significant increase in the variance of the FDR for the Benjamini–Hochberg procedure when applied to data sets with highly correlated variables. The methods I propose seek to reduce the variance of the FDR under assumption of dependence among variables.

1.4 Motivation

The first method I implement was proposed in Majumder *et al.* (2009), and is a variation on the Benjamini–Hochberg procedure (abbreviated as BH hereafter). In the study, researchers were interested in uncovering which single nucleotide polymorphisms (SNPs) are significant in the immune response triggered by a typhoid vaccine. SNPs are changes in single base pairs in DNA sequences that are responsible for many phenotypic traits. The study ultimately used 2,040 SNPs from 283 genes of 984 participants. Measurements for were taken before the vaccine was administered and 28 days later, and the differences were then recorded as the antibody response (AR). Researchers then used a log transform to induce Normality in the AR data.

Once the data had been converted to an approximately Normal form, the data was randomly split into two groups, each containing 492 subjects. BH was then applied to the first group of subjects to obtain a set of potentially significant SNPs. This smaller set of SNPs are then passed on to another BH procedure to be ran on the second group of subjects. Thus another BH is ran on this second group, but only on those SNPs that resulted in significance from the procedure ran on the first group. By using this internally cross-validated Benjamini–Hochberg (ICV-BH) procedure the authors of Majumder *et al.* (2009) intend to avoid the costs and complications associated with fully replicating a separate study on a different population, and reference simulation studies that show there is little loss of statistical power in doing so. The authors attribute the size of the study to keeping the loss of power small.

My objective is to consider what effect the ICV-BH has on the FDR for multiple testing, particularly at different levels of dependence among parameters. In Majumder *et al.* (2009) their

approach is practical for their purposes, but perhaps there could be a significant reduction to the FDR without the tests becoming too conservative, as in FWER control, and without significant loss of power. Furthermore, dependence that may be present among SNPs is not considered in Majumder *et al.* (2009), but could be present in similar applications that could make use of this approach. Thus, I conduct a simulation study using the ICV-BH approach and compare results with the usual BH under assumptions of independence and increasing levels of dependence between parameters. In the simulation study, I show that this procedure does reduce the variance of the FDR under increasing levels of independence.

Since the method above has shown promising results, I have also implemented a similar method which I will call intersection Benjamini–Hochberg (I-BH). In this method, begin by spliting the data into two split groups, as above. Next, run BH on each of the split groups to gain two sets of rejected H_0's. Finally, select the intersection of H_0's rejected in both tests to be the final set of rejected H_0's.

In simulation, I-BH also reduces the FDR and the variance of the FDR. However, I-BH does appear to be the more conservative than the ICV-BH.

The following flow charts, Fig. 1.1 and Fig. 1.2, provide a visual representation for the ICV-BH procedure and the I-BH procedure, respectively.

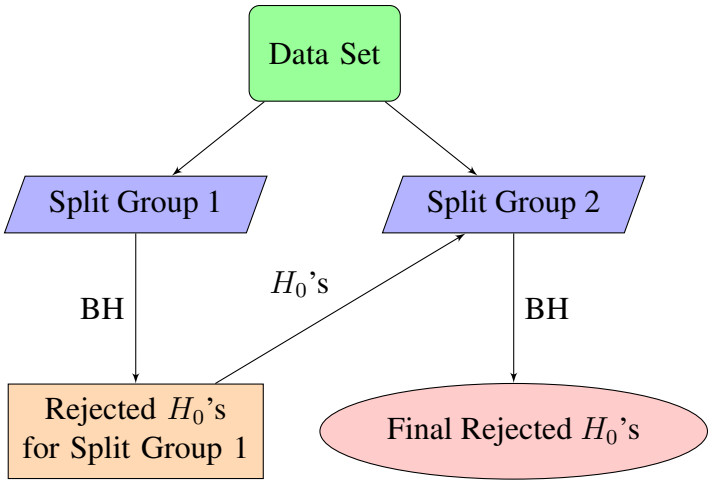

Figure 1.1: Internally Cross-Validated Benjamini–Hochberg

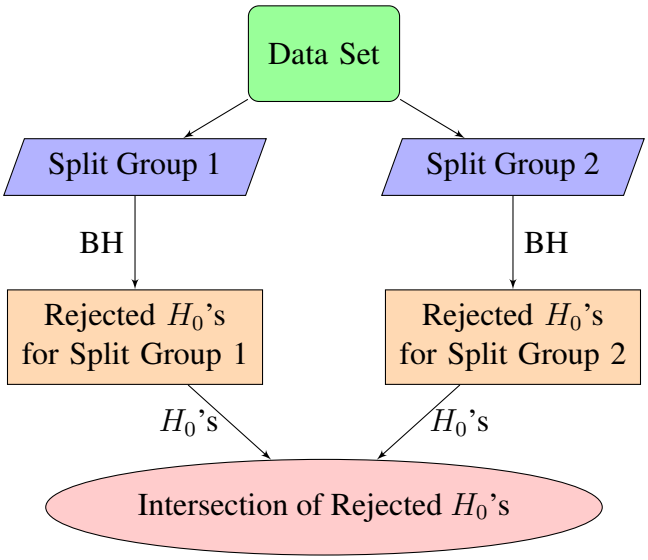

Figure 1.2: Intersection Benjamini–Hochberg

Chapter 2

Methods

The goal of my simulation study is to evaluate the effect that internal cross-validation of BH has on the FDR and FNR under increasing levels of dependence in the data. To accomplish this I have simulated data sets that resemble typical gene expression data with increasing levels of dependence. To each data set I apply the following multiple testing methods:

1. Benjamini–Hochberg
2. Internally Cross-Validated Benjamini–Hochberg
3. Intersection Benjamini–Hochberg

I then compare the results in terms of the following metrics:

1. False Discovery Rate
2. False Negative Rate
3. Misclassification Probability

All of the following simulations and analysis described were implemented using statistical software (R Core Team, 2017). (Codes are provided in the appendix.)

2.1 Data Simulation

To generate my data, I simulate an experiment comparing two groups. In group 1 there are subjects with some disease caused by a genetic abnormality, while group 2 subjects do not have this disease and are assumed to have "normal" genes. As is common with gene expression data, my data will have far more "genes" than subjects. This is often denoted in the literature as a "large p, small n" problem or "wide" data. Furthermore, my simulated data will be sparse. That means my generated data will have a greater number of parameters when compared to subjects and only a few of those parameters will be significant. This construction was chosen because it

resembles some real life genetic expression data and assumptions about that data, in particular I seek to emulate the prostate cancer dataset from Singh *et al.* (2002).

In my simulation, the number of subjects for each group is represented by n_1 for group 1 and n_2 for group 2. The simulated gene expression levels for those subjects will be our variables in the study, the number of which is given by p. Thus, each subject will have p gene expression levels which are given as vectors $\underline{s}_j$ with dimension $p \times 1$ and $j \in \{1, 2, ..., n_1 + n_2\}$. Thus, when the data is simulated the result is a $p \times (n_1 + n_2)$ matrix, call it $X_{p\times(n_1+n_2)}$.

Each element of $\underline{s}_j$ then is a random variable given by either $\theta_k^\star$ for signals or θ_i for noise, with $k \in \{1, 2, ..., p^\star\}$, where $p^\star$ is the number of signals, and $i \in \{1, 2, ..., p\}$. Signals $\theta_k^\star \sim N(\mu_k^\star, 1)$ with hyperparameter $\mu_k^\star \sim N(0, \psi^2)$, while the noise $\theta_i \sim N(0, 1)$. For my signal variables this choice of $\mu_k^\star$ implies that $\theta_k^\star \sim N(0, \sigma^2 + \psi^2)$. This construction was chosen as it is an established simulation model for genetics studies. (Scott & Berger, 2010) Then for subjects in group 1, $j \in \{1, 2, ..., n_1\}$, let $\underline{\theta}_1^T = \{\theta_1^\star, \theta_2^\star, ..., \theta_{p^\star}^\star, \theta_{p^\star+1}, \theta_{p^\star+2}, ..., \theta_p\}$, while for subjects in group 2, $j \in \{n_1 + 1, n_1 + 2, ..., n_1 + n_2\}$, let $\underline{\theta}_2^T = \{\theta_1, \theta_2, ..., \theta_p\}$. Now $\underline{\mathrm{s}}_j \sim MVN(\underline{\theta}_h, \Sigma_{p\times p})$ where $h = 1$ if $j \leq n_1$ and $h = 2$ otherwise, and $\Sigma_{p\times p}$ is the dispersion matrix of $\underline{\mathrm{s}}_j$.

To generate data for X in the independent case (that is when $\Sigma = I_{p\times p}$), I perform the following steps:

1. Generate the hyperparameters $\mu_k^\star$ and construct $\underline{\mu}_j$ as described below:
   (a) For $j \leq n_1$, $\underline{\mu}_j^T = \{\mu_1^\star, \mu_2^\star, ..., \mu_{p^\star}^\star, 0, 0, ..., 0\}$ with dimension $1 \times p$.
   (b) For $j > n_1$, $\underline{\mu}_j^T = \{0, 0, ..., 0\}$ with dimension $1 \times p$
2. For each $\underline{s}_j$, draw p samples from $N(0, 1)$ to generate for the column vector z_j.
3. Set $\underline{s}_j = \underline{\mu}_j + \underline{z}_j$.
4. Set $X = [\underline{s}_1\ \underline{s}_2\ ...\ \underline{s}_{n_1+n_2}]$.

And the X matrix, with $x_{ij} \in \underline{s}_j$ for $i \in \{1, 2, ..., p\}$ and $j \in \{1, 2, ..., n_1 + n_2\}$, is given by the

following:

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1(n_1+n_2)} \\ x_{21} & x_{22} & \dots & x_{2(n_1+n_2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{p1} & x_{p2} & \dots & x_{p(n_1+n_2)} \end{bmatrix} \tag{2.1}$$

For the dependent case, $\underline{s}_j \sim \text{MVN}(\underline{\mu}_j, \Sigma_{p\times p})$ (Note: $\underline{\mu}_j = \underline{\theta}_h$), where $\underline{\mu}_j$ is the vector of means as previously stated. Σ is assigned based on the desired dependence structure for our simulation. To simulate X under this condition I use the Cholesky decomposition, $\Sigma = LL^T$. Then X is generated as follows:

1. Calculate L using the Cholesky decompostion.
2. Generate the mean vector, $\underline{\mu}_j$ as described above.
3. For each $\underline{s}_j$, draw p samples from $N(0, 1)$ to generate for the column vector $\underline{z}_j$.
4. Set $\underline{s}_j = \underline{\mu}_j + L\underline{z}_j$.
5. Set $X = [\underline{s}_1|\underline{s}_2|\dots|\underline{s}_{n_1+n_2}]$.

The X matrix thus generated has a similar form to the X matrix given above. Using the methods of simulation described above, I will sample 200 data sets under 21 levels of increasing dependence for a total of 4,200 data sets. Having 200 data sets at each level of dependence will allow for reliable calculations of FDR, FNR, and misclassification probability at each level. Each data set will have 1,000 parameters and 100 subjects ($n_1 = 50$ and $n_2 = 50$), with sparsity set to 10%. These values mirror a typical scenario under which the BH procedure would be used without being so large that replicated simulations become too time consuming. The dependence

structure chosen is given by the following:

$$\Sigma = \begin{bmatrix} 1 & \rho & \rho^2 & \dots & \rho^{p-1} \\ \rho & 1 & \rho & \dots & \rho^{p-2} \\ \rho^2 & \rho & 1 & \dots & \rho^{p-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho^{p-1} & \rho^{p-2} & \rho^{p-3} & \dots & 1 \end{bmatrix} \tag{2.2}$$

Where $\rho = \{0, 0.05, 0.1, ..., 0.95, 0.99\}$ to give the 21 levels of dependence, with $\rho = 0$ being the independent case. This dependence structure was chosen as neighboring genes tend to have some correlation with one another. Values for ρ were chosen to allow for tracking changes in the FDR, FNR, and misclassification probability at a reasonable resolution.

2.2 Analysis

With data sets simulated as described in the previous section, I now apply the multiple testing methods described in Chapter 1 to them. For each method, two sample t-tests are used to obtain p-values for variables $\underline{\theta}_1 - \underline{\theta}_2$, where $\underline{\theta}_1$ represents the vector of population means for the "cancer" group and $\underline{\theta}_2$ represents the vector of population means for the "control" group. t-tests should be used in this case as we are looking for any significant difference between the means of the two groups. As the simulation is meant to mimic a real life experiment I use the t-test instead of a normal test, as the variance is not typically known.

Note also that in the simulations the columns are the observations and the rows relate to the variables of interest. This should follow from how the matrices were constructed, and mimic the prostate cancer dataset in Efron (2010). So the tests will compare row means for the columns belonging to the two groups.

Benjamini–Hochberg: For the BH procedure, each data set is split into two groups with the first $n_1 = 50$ columns in group 1, and the next $n_2 = 50$ columns in group 2. Then t-tests are performed on each row with the following hypothesis, $H_{0i} : \theta_{1i} - \theta_{2i} = 0$ and $H_{1i}\theta_{1i} - \theta_{2i} \neq 0$,

for i in $\{1, 2, .., p\}$, to obtain p-values for each of the p variables. Using these p-values, the BH procedure is applied and significant variables are identified. The following values are then calculated: false discovery proportion (FDP), false negative proportion (FNP), misclassification proportion (MP), and number of significant variables (NSV). Finally, for each level of dependence, these values are averaged across the 200 hundred data sets to arrive at estimates for the false discovery rate (FDR), false negative rate (FNR), misclassification rate (MR), and average number of significant variables (ANSV).

Internally Cross-Validated Benjamini–Hochberg: For the ICV-BH procedure, each data set is split into two groups with the first $n_1 = 50$ columns in group 1, and the next $n_2 = 50$ columns in group 2. These groups are then split into two sub-groups with 25 columns each, which I will refer to as S_1, S_2, S_3, and S_4. Then sub-groups S_1 and S_2 are matrices formed by selecting 25 columns each from group 1, without replacement. Similarly, sub-groups S_3 and S_4 are matrices formed by selecting 25 columns each from group 2, without replacement. Then t-tests comparing S_1 and S_3 are performed on each row with the following hypothesis, $H_{0i} : \theta_{1i} - \theta_{2i} = 0$ and $H_{1i}\theta_{1i} - \theta_{2i} \neq 0$, for i in $\{1, 2, .., p\}$, to obtain p-values for each of the p variables. Using these p-values, the BH procedure is applied and significant variables are identified. Let $\underline{p}^\star$ be the vector of row indices for the variables thus identified. Now, t-tests comparing S_2 and S_4 are performed on the $\underline{p}^\star$ rows with hypotheses as described above to obtain p-values for each variable referenced by the indices $\underline{p}^\star$. Using these p-values, the BH procedure is applied a second time and the final round of significant variables are identified. The following values are then calculated: FDP, FNP, MP, and NSV. Finally, for each level of dependence, these values are averaged across the 200 hundred data sets to arrive at estimates for the FDR, FNR, MR, and ANSV.

Intersection Benjamini–Hochberg: For the I-BH procedure, each data set is split into two groups with the first $n_1 = 50$ columns in group 1, and the next $n_2 = 50$ columns in group 2. These groups are then split into two sub-groups with 25 columns each, which I will refer to as S_1, S_2, S_3, and S_4. Then sub-groups S_1 and S_2 are matrices formed by selecting 25 columns each from group 1, without replacement. Similarly, sub-groups S_3 and S_4 are matrices formed by selecting

25 columns each from group 2, without replacement. Then t-tests comparing S_1 and S_3 are performed on each row with the following hypothesis, $H_{0i} : \theta_{1i} - \theta_{2i} = 0$ and $H_{1i}\theta_{1i} - \theta_{2i} \neq 0$, for i in $\{1, 2, .., p\}$, to obtain p-values for each of the p variables. Using these p-values, the BH procedure is applied and significant variables are identified. Let $\underline{p}_1^\star$ be the vector of row indices for the variables thus identified. Then, t-tests comparing S_2 and S_4 are performed on each row with hypotheses as described above to obtain p-values for each of the p variables. Using these p-values, the BH procedure is applied a second time and significant variables are identified. Let $\underline{p}_2^\star$ be the vector of row indices for the variables thus identified. Finally, let $\underline{p}_1^\star \cap \underline{p}_2^\star = \underline{p}^\star$ be the indices for the selected variables by the I-BH procedure. The following values are then calculated: FDP, FNP, MP, and NSV. Finally, for each level of dependence, these values are averaged across the 200 hundred data sets to arrive at estimates for the FDR, FNR, MR, and ANSV.

Choice of FDR Control: For this simulation study, I am choosing to control the FDR at a rate of $q = 0.3$. The reason for this choice stems from work in Datta (2014), where it is argued that for the ICV-BH that the FDR is controlled at a rate of q^2. The choice of $q = 0.3$ was made after some trial and error with the simulations along with desire to use a common FDR control of $q = 0.1$. Since the choice of $q = 0.3$ produced meaningful results from our simulated data and $q^2 = 0.3^2 = 0.09$ was close to our desired FDR control for the ICV-BH, $q = 0.3$ was used.

Calculations: The following calculations are for FDP, FNP, MP, and NSV given using variables from Table 1.1. Recall that the total number of tests is given by N, the number of tests where H_0 is true and not rejected is given by the random variable U, where H_0 is false and not rejected is given by the random variable T, where H_0 is true and rejected is given by the random variable V, and where H_0 is false and is rejected is given by the random variable S. Then $R = S + V$ gives us the total number of tests where the H_0 is rejected, $N - S$ gives the total number of tests where the H_0 is not rejected, and $N - N_0$ gives the total number of false null hypotheses. All of U, V, T, S, R, N, and N_0 are in $\mathbb{N} \cup \{0\}$.

$$\text{FDP} = \begin{cases} \frac{V}{R} & R \neq 0 \\ 0 & R = 0 \end{cases} \tag{2.3}$$

$$\text{FNP} = \begin{cases} \frac{T}{N-R} & R \neq N \\ 0 & R = N \end{cases} \tag{2.4}$$

$$\text{MP} = \frac{T+V}{N} \tag{2.5}$$

$$\text{NSV} = R \tag{2.6}$$

As stated above, FDR, FNR, MR, and ASNV are calculated by taking averages of FDP, FNP, MP, and SNV, respectively, across the 200 data sets for each dependence level.

Visualization: Finally, all of these calculated values are used to create graphs of the FDR, FNR, MR, and ASNV across the different levels of dependence with the FDP, FNP, MP, and SNV calculations, respectively, being used to generate confidence bands for each estimate. Furthermore, box plots of the FDP, FNP, MP, and SNV are also produced to show the spread of those values along with the extremes that are produced in the simulation study.

Software Utilized: All computation was done using R Core Team (2017) and the following packages: Microsoft & Weston (2017), Corporation & Weston (2017), Wickham (2009), and Auguie (2017)

2.3 Parallel Processing

To reduce computation time for this simulation study, I have made use of parallel processing packages in R. The computer I've used for all my computations is a Surface Pro 4 with an Intel Core i5-6300U Processor. As an example of the performance increase when parallel processing, I

compare the processing times for the parallel and nonparallel versions of the “foreach” function when generating the independent data sets (as previously described) and analyzing all data sets generated using the BH (as previously described). All time measurements are in seconds; user and system time relate to CPU processing time, while elapsed time is the total time taken to run a procedure. Note that when user + system ¿ elapsed, then there is likely something other than computations slowing the process, and further troubleshooting would be required to find ways to reduce the elapsed time.

For the data generation the times are:

Table 2.1: **Data Generation:** Processing times for generating 200 datasets, as described in Section 2.1, under the assumption of independent variables.

	User	System	Elapsed
Non-parallel	2.57 s	0.17 s	3.54 s
Parallel	0.39 s	0.22 s	2.77 s

For data analysis the times are:

Table 2.2: **Data Analysis:** Processing times for analyzing all 4200 generated data sets using the BH, as described in Section 2.2.

	User	System	Elapsed
Non-parallel	114.33 s	3.32 s	123.00 s
Parallel	39.2 s	24.98 s	154.75 s

Chapter 3

Results

3.1 An Example Data Set

For a quick glimpse at how the different procedures perform, I first consider how each procedure performs on a single data set. I chose to use the first data set generated using the simulation methods described in Section 2.1. That is, the first data set generated with independent variables. The false discovery proportion (FDP), false negative proportion (FNP), misclassification proportion (MP), and number of significant variables (NSV) found by my analysis are reported in Table 3.1. At first glance, it would seem that the internally cross-validated Benjamini–Hochberg (ICV-BH) is outperforming both the Benjamini–Hochberg procedure (BH) and the Intersection Benjamini–Hochberg (I-BH) with the lowest FNP and MP, however I will show later that the FNR on average actually increases for both the ICV-BH and I-BH methods.

Table 3.1: **Results for Single Data Set Generated using $\rho = 0$:** This table includes the calculated FDP, FNP, MP, and NSV for a single data set with independent variables using calculations described in the Section 2.2.

	FDP	FNP	MP	NSV
Benjamini–Hochberg	0.237	0.015	0.040	114
Internally Cross-Validated	0.011	0.0143	0.014	88
Intersection	0.000	0.0217	0.020	80

Interestingly, the I-BH method resulted in FDP $= 0$ in this example. This occurred because the split groups for this method did not share any false discoveries. Table 3.2 provides a list of indices for falsely rejected θ_i's in each of the split groups. Notice that there are no shared false rejections. While this did occur in some data sets, the majority of the data sets produced FDP values greater than zero. This particular problem was more pervasive when smaller values of q for the BH procedures were chosen, hence the choice of $q = 0.3$ was made in part to address this issue.

Table 3.2: **False Discoveries in Split Groups:** This table lists the row indices for the false discoveries found in each split group in the I-BH. Notice that no indices are common between the two groups.

False Discoveries for Split Group 1
140, 168, 197, 218, 258, 270, 293, 363, 415, 445, 462, 519, 527, 562, 599, 614, 697, 706, 708, 715, 806, 812, 938, 993
False Discoveries for Split Group 2
108, 146, 180, 187, 221, 262, 264, 354, 378, 393, 400, 412, 468, 503, 524, 526, 533, 548, 557, 586, 603, 605, 628, 636, 658, 679, 711, 712, 752, 753, 804, 842, 854, 870, 913, 963, 981

3.2 Simulation Study Results

Moving on from the example data set, the following figures summarize my analysis of the simulated data sets. For each figure, graphs are given for the above calculations after performing each procedure. For each graph, the calculated values from the data sets are plotted against increasing levels of dependence, ρ. These graphs present how the calculated values change relative to changes in ρ. The 90% confidence bands are given to provide a visual representation of the variance of each value.

For each graph in Fig. 3.1 the False Discovery Rate (FDR) is plotted as the solid line, and the 90% confidence bands are constructed by the 95% and 5% quantiles of the FDP calculations at each ρ. Figure 3.2 provides plots of the maximum and minimum FDP values obtained by each procedures at each value of ρ.

Comparing the graphs in Fig. 3.1, the confidence bands for the ICV-BH and I-BH are much narrower than that of the BH. This suggests that the variance of the FDR is being greatly reduced. Furthermore, in both the ICV-BH and I-BH the FDR is much lower than that of the BH. This reduction in FDR is not problematic, so long as there is not too large a decrease in power. Finally, we see that the I-BH, having the most reduced FDR, is the most conservative test of the three.

For the minimum and maximum lines in Fig. 3.2, note that the maximum values obtained by the ICV-BH and I-BH remain much lower than the BH. This suggests that even in extreme cases, the FDR of the ICV-BH and I-BH should be more reliable than the BH. However, the minimum lines for the FDR of the ICV-BH and I-BH are constant at zero, while the minimum line for the

BH only reaches zero when $\rho > 0.9$. This result further suggests these methods are more conservative than the BH.

For each graph in Fig. 3.3 the False Negative Rate (FNR) is plotted as the solid line, and the 90% confidence bands are constructed by the 95% and 5% quantiles of the FNP calculations at each ρ.

Comparing the graphs in Fig. 3.3, the confidence bands for the ICV-BH and I-BH are wider than that of the BH. This suggests that the variance of the FNR is larger for the new methods. Also, the FNR for the BH is smaller than that of the ICV-BH and I-BH, with the entire confidence band of the I-BH being greater than the confidence band of the BH. This confirms the decrease in power anticipated by the reduced FDR of the ICV-BH and I-BH, with the I-BH having the greatest reduction in power of the two. I consider whether or not this decrease in power is acceptable in the Discussion chapter 4.

For each graph in Fig. 3.4 the Misclassification Rate (MR) is plotted as the solid line, and the 90% confidence bands are constructed by the 95% and 5% quantiles of the MP calculations at each ρ.

Comparing the graphs in Fig. 3.4, the confidence bands for the ICV-BH and I-BH are much narrower than that of the BH. This suggests that the variance of the MR is much smaller for the new methods. Given the analysis of the other figures, this result is not surprising. Since the data I've simulated is sparse, more misclassifications could occur from false discoveries than from false negatives. Since the new methods reduce the FDR, then the FDR's contribution to the MR would be reduced, and since the number of true negatives is small (sparse data), the increase in FNR would have little effect on the MR.

For each graph in Fig. 3.5 the average number of significant variables (ANSV) found is plotted as the solid line, and the 90% confidence bands are constructed by the 95% and 5% quantiles of the SNV calculations at each ρ.

For Fig. 3.5, observe that the ANSV for the ICV-BH and I-BH is underestimating the true NSV, while the BH is overestimating the true NSV. However, the confidence bands for the

ICV-BH and I-BH are more narrow than those for the BH. This suggests that while the ICV-BH and I-BH will likely miss some significant variables, there is less variation in the number of variables selected when compared with the results of the BH. This result considered with the reduction in FDR and increase in FNR suggests, that a researcher may more reliably avoid false discoveries with the ICV-BH and I-BH when compared to the BH, but he would be more likely to miss some significant variable. Indeed, since the confidence band for I-BH and almost all the confidence band for ICV-BH sit below 100, these methods, for my simulated data, will not be able to detect all of the significant variables, while the BH still could based on its ANSV confidence band.

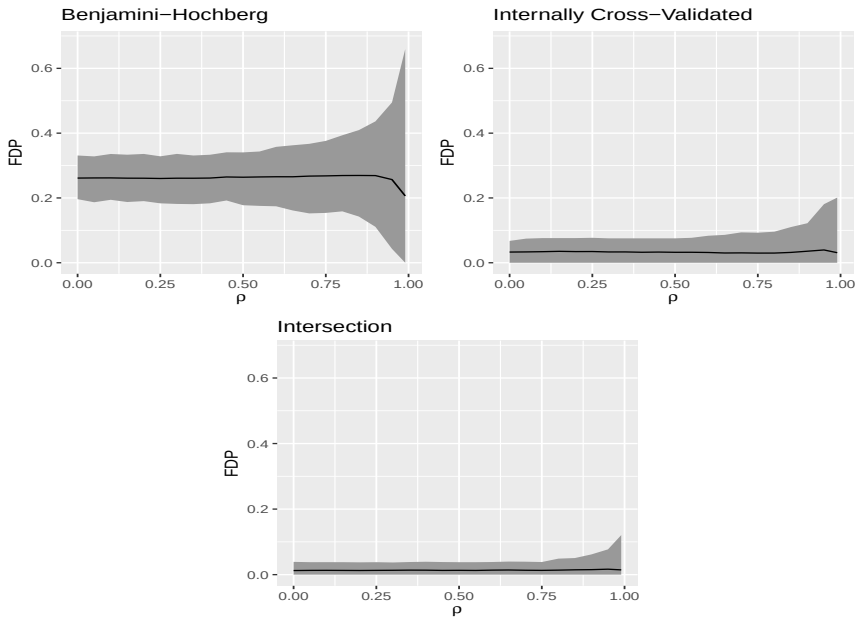

Figure 3.1: **False Discovery Rate:** False discovery rate with 90% Confidence Band for the BH, ICV-BH, and I-BH. ρ refers to the value in $[0, 1)$ representing correlation between variables, and FDP is the calculated false discovery proportion. The solid line is the FDR for simulations at each ρ, and the shaded area depicts the area between the 5% and 95% percentile for FDP for the simulations at each ρ.

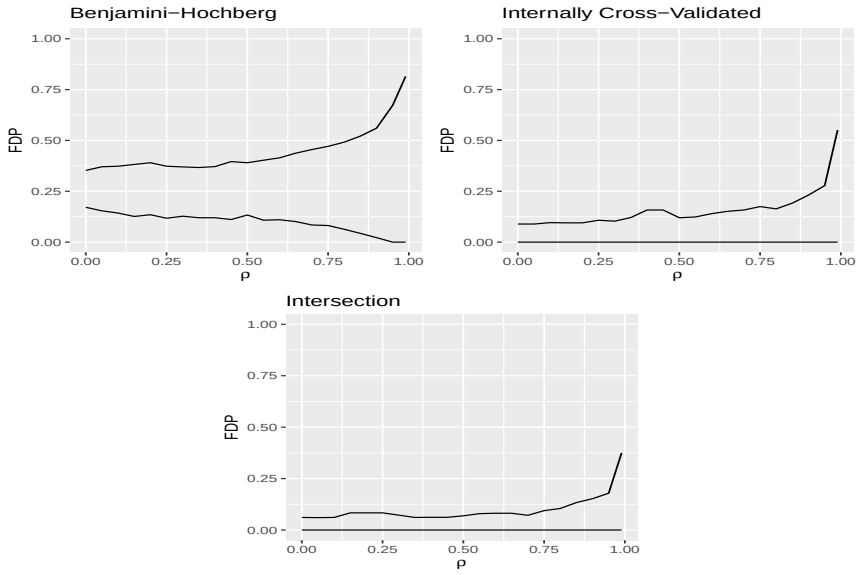

Figure 3.2: **False Discovery Proportion Extreme Values:** Extreme values for the false discovery proportion for the BH, ICV-BH, and I-BH. ρ refers to the value in $[0, 1)$ representing correlation between variables, and FDP is the calculated false discovery proportion. The top solid line represents the max FDP calculated among simulations at values for ρ, while the bottom solid line represents the minimum FDP calculated among simulations at values for ρ.

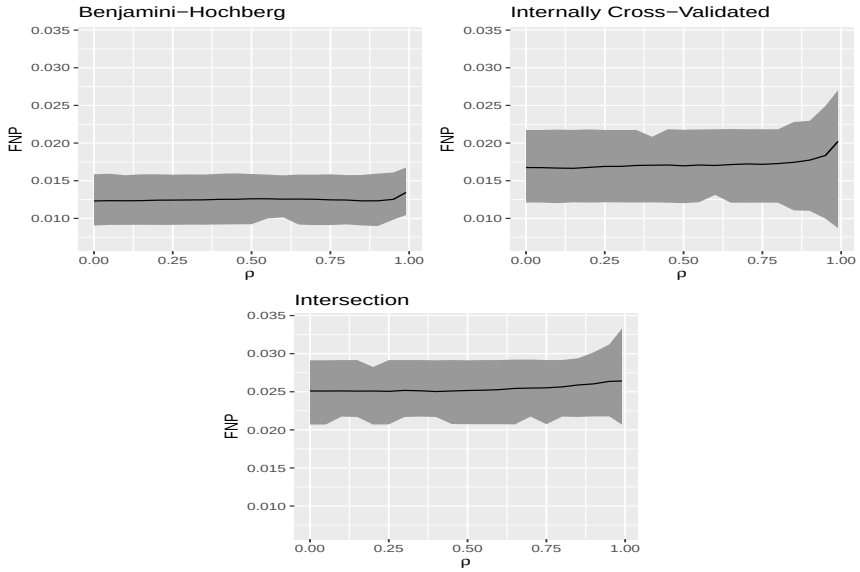

Figure 3.3: **False Negative Rate:** False negative rate with 90% Confidence Band for the BH, ICV-BH, and I-BH. ρ refers to the value in $[0, 1)$ representing correlation between variables, and FNP is the calculated false negative proportion. The solid line is the FNR for simulations at each ρ, and the shaded area depicts the area between the 5% and 95% percentile for FNP for the simulations at each ρ.

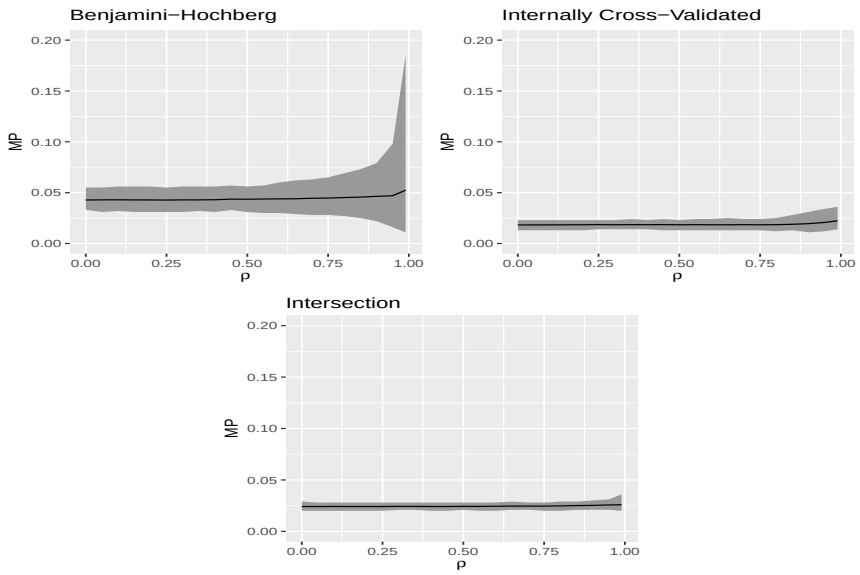

Figure 3.4: **Misclassification Rate:** Misclassification rate with 90% Confidence Band for the BH, ICV-BH, and I-BH. ρ refers to the value in $[0, 1)$ representing correlation between variables, and MP is the calculated misclassification proportion. The solid line is the MR for simulations at each ρ, and the shaded area depicts the area between the 5% and 95% percentile for MP for the simulations at each ρ.

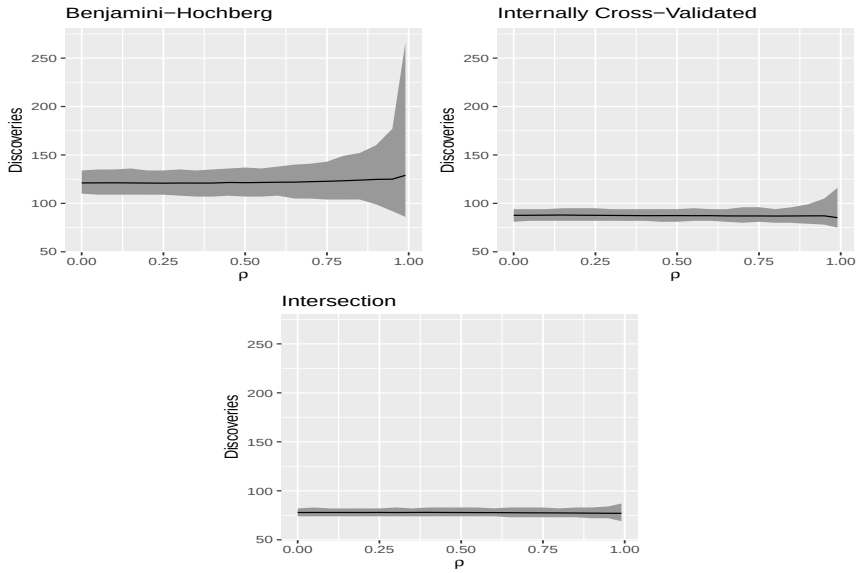

Figure 3.5: **Number of Significant Variables:** Number of significant variables with 90% Confidence Band for the BH, ICV-BH, and I-BH. ρ refers to the value in $[0, 1)$ representing correlation between variables, and discoveries are the number of significant variables detected. The solid line is the average number of significant variables for simulations at each ρ, and the shaded area depicts the area between the 5% and 95% percentile for number of significant variables for the simulations at each ρ.

3.3 Application

Since my data simulations were based upon the prostate cancer data set from Singh *et al.* (2002), it would be appropriate to compare performance of the different procedures on this data set. The prostate cancer dataset is a 6033×102 matrix, organized so that each of the 6033 gene expression levels recorded are along the rows, and the 102 subjects are along the columns. Of the 102, 50 subjects are the controls (non-cancer patients), while 52 subjects have cancer. Thus our objective will be to find significantly different genes between the two groups. So I first ran the regular BH, as described above, with $q = 0.3$, on the dataset. This resulted in the choice of 150 significant genes.

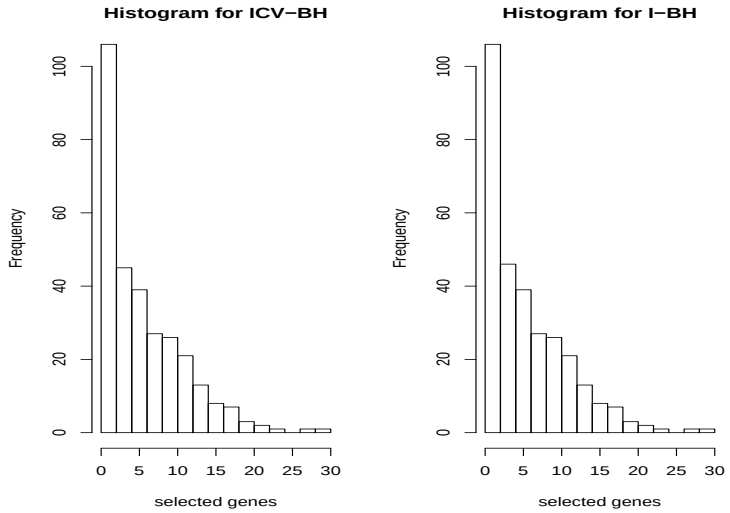

Figure 3.6: **Histograms for Selected Genes:** Histograms for the number of selected genes determined by applying the ICV-BH and the I-BH to the prostate cancer dataset. Since both procedures require the construction of subgroups before selection of significant genes, the subjects that make up those subgroups were randomly assigned to appropriate subgroups as described in Section 2.2. This procedure was random assignment to subgroups was then replicated 300 times, with the procedures being ran each time, to produce these histograms. Results are sorted into bins based on the number of genes selected by the procedure ran.

Recall that to apply the ICV-BH and the I-BH to the data set, the two groups of subjects, the control group and cancer group, must be split into two subgroups. In the simulations, I decided on the same split for each dataset, but for this application, I can consider multiple splits and look at the distribution of the number of selected genes from those splits. So the 50 subjects of the

control group were then randomly assigned to exactly one of two subgroups, such that each subgroup had 25 members. The same was done for the 52 subjects of the cancer group, except that each subgroup had 26. After the subgroups were made, the ICV-BH and the I-BH were ran on the data as described above, with $q = 0.3$. Since we could randomly assign these groups in many different ways, this process of subgroup selection was repeated 300 times and results are presented as a histogram in Fig. 3.6.

On average, the ICV-BH selected 5.97 genes, and the I-BH selected 5.96 genes. Much like their averages, the histograms for the ICV-BH and the I-BH look very similar to one another. While this result is a bit surprising, it is observed that, as expected, the resulting number of selected genes is much more conservative with these procedures, when compared to the BH.

Chapter 4

Discussion

As stated in chapter 1, the Benjamini–Hochberg procedure (BH) is currently the gold standard for large scale simultaneous hypothesis testing of sparse data. Especially if independence between variables may be assumed. However, I have shown that under increasing levels of dependence, the BH's results become less precise. That is, the variance of the False Discover Rate (FDR) greatly increases as the correlation, ρ, between variables increases.

For my simulation study, the variance of the FDR for the BH begins to noticeably increase as ρ grows larger than 0.5. In order to reduce this variance I have proposed using the Internally Cross-Validated Benjamini–Hochberg procedure (ICV-BH) and the Intersection Benjamini–Hochberg procedure (I-BH). The results of my simulation study show that the variance of the FDR was reduced for all values of ρ, but with an increase to False Negative Rate (FNR) and the variance of the FNR. This suggests that under the proposed procedures, a researcher may be more sure of the procedures' selected variables, but there is a greater risk that some variable that should have been selected was not selected.

What is clear from my simulation study is that both the ICV-BH and the I-BH are more conservative methods. Both methods trade a lower FDR with a lower variance for a larger FNR with a larger variance, with the I-BH generating a larger FNR, but with the ICV-BH leading to a larger variance for FNR. Thus a researcher using these methods should be less confident that all significant variables were found.

So researchers considering these methods should weigh the impact of making these two types of errors. Under the same FDR control, q, both of the new methods give strong control of the FDR and it's variance, but the price is an increase FNR and it's variance. Should a researcher need more certainty regard the FDR among highly correlated data, then both methods are strong candidates for the job.

Furthermore, should a researcher be willing to accept a larger FDR, then q could be increased further. Both methods, as shown, lead to a larger FNR and much lower FDR than the original BH,

thus one may consider increasing q for both the ICV-BH and I-BH so that the desired FDR is more inline with what is expected in the BH. My choice of $q = 0.3$ was made considering that the FDR should be controlled at approximately q^2 for the ICV-BH and the I-BH. This choice of q should be lead these procedures to control FDR at $q^2 \approx 0.1$. With this in mind, it may also be useful to compare the ICV-BH and I-BH with $q = 0.3$ against the BH with $q = 0.1$.

4.1 Future Work

Thus far I have focused on comparing the proposed methods, the ICV-BH and I-BH, with the BH. As the BH is the most widely used multiple testing procedure, this comparison is appropriate, however there are other methods drawing on Bayesian statistics that would be worthwhile to consider. Typically Bayesian methods are applied more for estimation than for detecting significant variables, but variable selection in a particular model can be seen as similar to testing for variable significance. Thus in the future comparing the outcomes of variable seleciton with Bayesian technique to the outcomes of the proposed methods would be useful.

Furthermore, many of the Bayesian techniques also rely independent variable assumption, thus adapting those techniques to dependent variable cases should be very interesting. There is theoretical support for taking this approach, however computation involving dependent variables in the Bayesian framework can become very cumbersome. If this computational problem can be overcome, developing such methods looks very promising.

Bibliography

Auguie, Baptiste. 2017. *gridExtra: Miscellaneous Functions for "Grid" Graphics*. R package version 2.3.

Benjamini, Yoav, & Hochberg, Yosef. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 289–300.

Benjamini, Yoav, & Yekutieli, Daniel. 2001. The control of the false discovery rate in multiple testing under dependency. *Annals of Statistics*, 1165–1188.

Corporation, Microsoft, & Weston, Steve. 2017. *doParallel: Foreach Parallel Adaptor for the 'parallel' Package*. R package version 1.0.11.

Datta, Jyotishka. 2014 (8). *Some Theoretical and Methodological Aspects of Multiple Testing, Model Selection and Related Areas*. Ph.D. thesis, Purdue University, West Lafayette, Indiana.

Efron, Bradley. 2010. *Large-scale inference*. Institute of Mathematical Statistics (IMS) Monographs, vol. 1. Cambridge University Press, Cambridge. Empirical Bayes methods for estimation, testing, and prediction.

Fan, Jianqing, & Han, Xu. 2017. Estimation of the false discovery proportion with unknown dependence. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **79**(4), 1143–1164.

Genovese, Christopher, & Wasserman, Larry. 2002. Operating characteristics and extensions of the false discovery rate procedure. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **64**(3), 499–517.

Majumder, Partha P, Staats, Herman F, Sarkar-Roy, Neeta, Varma, Binuja, Ghosh, Trina, Maiti, Sujit, Narayanasamy, K, Whisnant, Carol C, Stephenson, James L, & Wagener, Diane K. 2009. Genetic determinants of immune-response to a polysaccharide vaccine for typhoid. *The HUGO Journal*, **3**(1-4), 17–30.

Microsoft, & Weston, Steve. 2017. *foreach: Provides Foreach Looping Construct for R*. R package version 1.4.4.

R Core Team. 2017. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Scott, James G, & Berger, James O. 2010. Bayes and empirical-Bayes multiplicity adjustment in the variable-selection problem. *Annals of Statistics*, 2587–2619.

Singh, Dinesh, Febbo, Phillip G, Ross, Kenneth, Jackson, Donald G, Manola, Judith, Ladd, Christine, Tamayo, Pablo, Renshaw, Andrew A, D'Amico, Anthony V, Richie, Jerome P, Lander, Eric S, Loda, Massimo, Kantoff, Phillip W, Golub, Todd R, & Sellers, William R. 2002. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, **1**(2), 203–209.

Storey, John D. 2003. The positive false discovery rate: a Bayesian interpretation and the q-value. *Annals of Statistics*, **31**(6), 2013–2035.

Van Steen, Kristel, McQueen, Matthew B, Herbert, Alan, Raby, Benjamin, Lyon, Helen, DeMeo, Dawn L, Murphy, Amy, Su, Jessica, Datta, Soma, Rosenow, Carsten, Christman, Michael, Silverman, Edwin K, Laird, Nan M, Weiss, Scott T, & Lange, Christoph. 2005. Genomic screening and replication using the same data set in family-based association testing. *Nature Genetics*, **37**(7), 683.

Wickham, Hadley. 2009. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.

Yekutieli, Daniel, & Benjamini, Yoav. 1999. Resampling-based false discovery rate controlling multiple test procedures for correlated test statistics. *Journal of Statistical Planning and Inference*, **82**(1-2), 171–196.

Appendix A

R Code

```
## Load Libraries to be used.

library("parallel")     ##  For parallel processing
library("foreach")      ##
library("doParallel")   ##
library("ggplot2")      ##  For data visualization
library("gridExtra")    ##

## Set Working Directory

setwd("Your_Directory")

## Generate random means - mu

set.seed(13)
randmu <- function(p, prop, var){
  ## p - number of parameters
  ## prop - proportion of non-null parameters
  ## var - variance for mean generation

  mu <- c(rnorm(p*prop, 0, sqrt(var)), rep(0, p*(1-prop)))
  return(mu)
}

## Independent Data Generator

indydata <- function(N, n, sig_means){
  ## N - number of data sets
  ## n - number of samples
  ## sig_means - vector of signal means

  p <- length(sig_means)

  ## Generate sample from mu + N(0,Ip)

  ## Initialize variables
  sim <- list(rep(matrix(rep(0, n*p), nrow = p, ncol = n), N))
  samples <- matrix(rep(0, n*p), nrow = p, ncol = n)

  # Set up for parallel matrix generation
  no_cores <- detectCores()-1

  RNGkind("L'Ecuyer-CMRG")
  cl <- makeCluster(no_cores)
  registerDoParallel(cl)
  clusterSetRNGStream(cl, 13)

  # generate data
  sim <- foreach(i = 1:N)%dopar%{
    samples <- (matrix(c(rep(sig_means, n/2), rep(0, p*n/2)),
                       p, n, byrow = FALSE)
                + matrix(rnorm(p*n), p, n))
    samples
  }
  stopCluster(cl)

  ## Add vector to indicate which means are non-zero.

  indices <- which(sig_means != 0)
  sig_means[indices] = indices
  sim <- list(sig_means, sim)
```

```
  ## Return vector for non-zero means and generated data

  return(sim)
}

## Dependent data generator

rhotop <- function(N, n, sig_means, rho){
  ## N - number of data sets
  ## n - number of samples
  ## sig_means - vector of signal means
  ## rho - correlation

  p <- length(sig_means)

  ## Generate the dispersion matrix D

  D = rho^abs(matrix(c(0:(p^2-1)),p,p,byrow = TRUE)%%p -
                matrix(c(0:(p^2-1)),p,p,byrow = FALSE)%%p)


  ## Take the Cholesky decomposition of D

  CD <- chol(D)

  ## Generate sample from mu + CD*N(0,Ip)

  sim <- list(rep(matrix(rep(0, n*p), nrow = p, ncol = n), N))
  samples <- matrix(rep(0, n*p), nrow = p, ncol = n)

  no_cores <- detectCores()-1

  RNGkind("L'Ecuyer-CMRG")
  cl <- makeCluster(no_cores)
  registerDoParallel(cl)
  clusterSetRNGStream(cl, 13)

  sim <- foreach(i = 1:N)%dopar%{
    samples <- (matrix(c(rep(sig_means, n/2), rep(0, p*n/2)),
                       p, n, byrow = FALSE) +
                  CD%*%matrix(rnorm(p*n), p, n))
    samples
  }
  stopCluster(cl)

  ## Add vector to indicate which means are non-zero.

  indices <- which(sig_means != 0)
  sig_means[indices] = indices
  sim <- list(sig_means, sim)

  ## Return vector for non-zero means and generated data

  return(sim)
}

## Generate and save data sets

p <- 1000
n <- 100
N <- 200
prop <- .1
var <- log(1000)

sig_means <- randmu(p, prop, var)
```

```
for(i in c(seq(0,95, by = 5), 99)){
  if(i==0){
    data_trind <- indydata(N, n, sig_means)
  }else{
    rho <- i/100
    data_trind <- rhotop(N, n, sig_means, rho)
  }
  save(data_trind, file = paste("data_rho_",i, sep = ""))
}

##  Functions to be sourced as BHFun.R

##  Perform Benjamini-Hochberg procedure

benhoch <- function(p,q){
  if(max(p)<=q){
    return(1:length(p))
  }else{
    op <- order(p)
    orderp <- c()
    for(i in 1:length(p)){
      orderp <- c(orderp, p[op[i]])
    }
    pindex <- 1
    while(orderp[pindex] <= pindex/length(p)*q){
      pindex <- pindex + 1
    }
    altindex <- c(op[1:(pindex-1)])
    return(sort(altindex))
  }
}

##  Calculate false discovery proportion.

Fdp <- function(pind, trind){
  flsdsc <- length(pind) - sum(pind %in% trind)
  proportion <- ifelse(length(pind)==0, 0, (flsdsc/length(pind)))
  return(proportion)
}

##  Calculate false negative proportion.

Fnp <- function(pind, trind){
  tralt <- max(trind)
  flsneg <- tralt - sum(pind %in% trind)
  proportion <- ifelse(length(trind)==length(pind), 0,
                       flsneg/(length(trind)-length(pind)))
  return(proportion)
}

##  Calculate misclassificaiton proportion

misclassp <- function(pind, trind){
  p <- length(trind)
  tralt <- max(trind)
  misclass <- length(pind) + tralt - 2*sum(pind %in% trind)
  proportion <- ifelse(p==0, 0, misclass/p)
  return(proportion)
}

##  Run t-test and return p-value.

tpval <- function(x, y = NULL){
  v <- x
  n <- length(v)
  if(is.null(y)){
    x <- v[1:(n/2)]
```

```
    y <- v[(n/2+1):n]
  }
  xn <- length(x)
  yn <- length(y)
  xbar <- mean(x)
  ybar <- mean(y)
  ssx <- sum((x-xbar)^2)
  ssy <- sum((y-ybar)^2)
  t <- (xbar-ybar)/sqrt((ssx+ssy)/(xn+yn-2)*(1/xn+1/yn))
  p <- 2*pt(abs(t), df = yn+xn-2, lower.tail = FALSE)
  return(p)
}


## Important! Set working directory.

setwd("C:/UARK_Work/Thesis_Project/Ben-Hoch_Project/Thesis")

source("BHFun.R")

## Initial Comparison on One Data Set

load(file = "data_rho_0")
trind <- data_trind[[1]]
data_list <- data_trind[[2]]
data <- data_list[[1]]
p <- length(data[,1])
n <- length(data[1,])
q <- .3

grp1 <- data[,c(1:(n/4),(n/2+1):(3*n/4))]
grp2 <- data[,c((n/4+1):(n/2),(3*n/4+1):n)]

## Vanilla BH

pval <- c(rep(0,p))

pval <- apply(data, 1, FUN = tpval)

pind_BH <- benhoch(pval,q)

misclass_BH <- c(Fdp(pind_BH, trind), Fnp(pind_BH, trind),
                 misclassp(pind_BH, trind), length(pind_BH))

## ICVBH

pval1 <- c(rep(0,p))
pval2 <- c()

pval1 <- apply(grp1, 1, FUN = tpval)
pind_grp1 <- benhoch(pval1,q)

pval2 <- apply(grp2[pind_grp1,], 1, FUN = tpval)
pind_ICVBH <- benhoch(pval2,q)

misclass_ICVBH <- c(Fdp(pind_ICVBH, trind), Fnp(pind_ICVBH, trind),
                    misclassp(pind_ICVBH, trind), length(pind_ICVBH))

## Intersection BH

pval1 <- c(rep(0,p))
pval2 <- c(rep(0,p))

pval1 <- apply(grp1, 1, FUN = tpval)
pind_grp1 <- benhoch(pval1,q)

pval2 <- apply(grp2, 1, FUN = tpval)
```

```
pind_grp2 <- benhoch(pval2,q)

pind_intBH <- unique(sort(c(pind_grp1[which(pind_grp1 %in% pind_grp2)],
                            pind_grp2[which(pind_grp2 %in% pind_grp1)])))

rmpind <- sort(c(pind_grp1[-which(pind_grp1 %in% pind_grp2)],
                 pind_grp2[-which(pind_grp2 %in% pind_grp1)]))

misclass_intBH <- c(Fdp(pind_intBH, trind), Fnp(pind_intBH, trind),
                    misclassp(pind_intBH, trind), length(pind_intBH))

## Results for One Data Set

misclass <- rbind(misclass_BH, misclass_ICVBH, misclass_intBH)
colnames(misclass) <- c("FDP", "FNP", "Misclassification", "Discoveries")
rownames(misclass) <- c("BH", "ICV", "Intersection")
as.table(misclass)

pindsp1 <- c(rep(0, p))
pindsp2 <- c(rep(0, p))
pindsp1[pind_grp1] <- 1
pindsp2[pind_grp2] <- 1

int_splits <- data.frame(
  x=1:p,
  pind = c(pindsp1, - pindsp2),
  overlap = factor(ifelse(pindsp1 == 1 & pindsp2 == 1,
                          "overlap", "no overlap")))

## Visualization for single data set (consider putting in Thesis proper)

ggplot(data=int_splits, aes(x=x, y=pind, color=overlap)) +
  geom_step() +
  xlab("Variable") +
  ylab("Significance") +
  theme_bw() +
  geom_vline(xintercept = 100, linetype = "dashed", color = "navy") +
  ggtitle("Significant variables in split group 1 are positive.
          \nSignificant variables in split group 2 are negative.")

dev.copy(pdf, "int_splits.pdf")
dev.off()

## Vanilla BH

## The regular BH procedure is applied to all data sets.

rho <- c(seq(0,95, by=5), 99)
load(file = "data_rho_0")
Data <- data_trind[[2]]
N <- length(Data)
q <- .3

misclass_index_BH <- lapply(rho, FUN = function(x){
                            matrix(rep(0,N*4), nrow = N, ncol = 4)})

for(i in rho){
  load(file = paste("data_rho_",i, sep = ""))
  trind <- data_trind[[1]]
  Data <- data_trind[[2]]
  N <- length(Data)
  p <- length(Data[[1]][,1])
  pval <- list(rep(c(rep(0,p)),N))

  no_cores <- detectCores()-1

  cl <- makeCluster(no_cores)
```

```
  registerDoParallel(cl)

  system.time(
    res <- foreach(j = 1:N)%dopar%{
      pval[[j]] <- apply(Data[[j]], 1, FUN = tpval)

      pindices <- benhoch(pval[[j]], q)

      cbind(Fdp(pindices, trind), Fnp(pindices, trind),
            misclassp(pindices, trind), length(pindices))
    })
  stopCluster(cl)

  misclass_index_BH[[which(rho==i)]] <- matrix(unlist(res),
                                               nrow = N, ncol = 4, byrow = TRUE)
}

layout(matrix(1:4, nrow = 2, ncol = 2))

plot(rho/100, lapply(misclass_index_BH,
                     FUN = function(x){mean(x[,1])}), type = 'l')

plot(rho/100, lapply(misclass_index_BH,
                     FUN = function(x){mean(x[,2])}), type = 'l')

plot(rho/100, lapply(misclass_index_BH,
                     FUN = function(x){mean(x[,3])}), type = 'l')

plot(rho/100, lapply(misclass_index_BH,
                     FUN = function(x){mean(x[,4])}), type = 'l')

## Intersection BH

## The Intersection BH procedure is applied to all data sets.

rho <- c(seq(0,95, by=5), 99)
load(file = "data_rho_0")
Data <- data_trind[[2]]
N <- length(Data)
q <- .3

misclass_index_int <- lapply(rho, FUN = function(x){
                             matrix(rep(0,N*4), nrow = N, ncol = 4)})

for(i in rho){
  load(file = paste("data_rho_",i, sep = ""))
  trind <- data_trind[[1]]
  Data <- data_trind[[2]]
  N <- length(Data)
  p <- length(Data[[1]][,1])
  n <- length(Data[[1]][1,])

  no_cores <- detectCores()-1

  cl <- makeCluster(no_cores)
  registerDoParallel(cl)

  system.time(
    res <- foreach(j = 1:N)%dopar%{
      grp1 <- Data[[j]][,c(1:(n/4),(n/2+1):(3*n/4))]
      grp2 <- Data[[j]][,c((n/4+1):(n/2),(3*n/4+1):n)]

      pval1 <- c(rep(0,p))
      pval2 <- c(rep(0,p))

      for(k in 1:p){
        pval1[k] <- tpval(grp1[k,1:(n/4)], grp1[k,(n/4+1):(n/2)])
```

```
      }

      pindg1 <- benhoch(pval1, q)

      for(k in 1:p){
        pval2[k] <- tpval(grp2[k,1:(n/4)], grp2[k,(n/4+1):(n/2)])
      }

      pindg2 <- benhoch(pval2, q)

      pindices <- unique(sort(c(pindg1[which(pindg1 %in% pindg2)],
                                pindg2[which(pindg2 %in% pindg1)])))

      cbind(Fdp(pindices, trind), Fnp(pindices, trind),
            misclassp(pindices, trind), length(pindices))
    })
  stopCluster(cl)

  misclass_index_int[[which(rho==i)]] <- matrix(unlist(res), nrow = N,
                                                ncol = 4, byrow = TRUE)
}

layout(matrix(1:4, nrow = 2, ncol = 2))

plot(rho/100, lapply(misclass_index_int,
                     FUN = function(x){mean(x[,1])}), type = 'l')

plot(rho/100, lapply(misclass_index_int,
                     FUN = function(x){mean(x[,2])}), type = 'l')

plot(rho/100, lapply(misclass_index_int,
                     FUN = function(x){mean(x[,3])}), type = 'l')

plot(rho/100, lapply(misclass_index_int,
                     FUN = function(x){mean(x[,4])}), type = 'l')

## ICVBH

## Internally Cross-Validated BH procedure is applied to all data sets.

rho <- c(seq(0,95, by=5), 99)
load(file = "data_rho_0")
Data <- data_trind[[2]]
N <- length(Data)
q <- .3

misclass_index_ICV <- lapply(rho, FUN = function(x){
                             matrix(rep(0,N*4), nrow = N, ncol = 4)})

for(i in rho){
  load(file = paste("data_rho_",i, sep = ""))
  trind <- data_trind[[1]]
  Data <- data_trind[[2]]
  N <- length(Data)
  p <- length(Data[[1]][,1])
  n <- length(Data[[1]][1,])

  no_cores <- detectCores()-1

  cl <- makeCluster(no_cores)
  registerDoParallel(cl)

  system.time(
    res <- foreach(j = 1:N)%dopar%{
      grp1 <- Data[[j]][,c(1:(n/4),(n/2+1):(3*n/4))]
      grp2 <- Data[[j]][,c((n/4+1):(n/2),(3*n/4+1):n)]
```

```
      pval1 <- c(rep(0,p))
      pval2 <- c()

      for(k in 1:p){
        pval1[k] <- tpval(grp1[k,1:(n/4)], grp1[k,(n/4+1):(n/2)])
      }

      pindg1 <- benhoch(pval1, q)

      if(length(pindg1)==0){
        pindices <- pindg1
      }else{
        for(k in pindg1){
          pval2 <- c(pval2, tpval(grp2[k,1:(n/4)], grp2[k,(n/4+1):(n/2)]))
        }
        pindices <- benhoch(pval2, q)
      }

      cbind(Fdp(pindices, trind), Fnp(pindices, trind),
            misclassp(pindices, trind), length(pindices))
    })
  stopCluster(cl)

  misclass_index_ICV[[which(rho==i)]] <- matrix(unlist(res), nrow = N,
                                                ncol = 4, byrow = TRUE)
}

layout(matrix(1:4, nrow = 2, ncol = 2))

plot(rho/100, lapply(misclass_index_ICV,
                     FUN = function(x){mean(x[,1])}), type = 'l')

plot(rho/100, lapply(misclass_index_ICV,
                     FUN = function(x){mean(x[,2])}), type = 'l')

plot(rho/100, lapply(misclass_index_ICV,
                     FUN = function(x){mean(x[,3])}), type = 'l')

plot(rho/100, lapply(misclass_index_ICV,
                     FUN = function(x){mean(x[,4])}), type = 'l')


## Converting Results to Data Frames

## Vanilla BH

df_fdr_BH <- as.data.frame(matrix(unlist(lapply(
  misclass_index_BH, FUN = function(x){x[,1]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_fdr_BH) <- rho/100

df_fnr_BH <- as.data.frame(matrix(unlist(lapply(
  misclass_index_BH, FUN = function(x){x[,2]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_fnr_BH) <- rho/100

df_mis_BH <- as.data.frame(matrix(unlist(lapply(
  misclass_index_BH, FUN = function(x){x[,3]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_mis_BH) <- rho/100

df_dis_BH <- as.data.frame(matrix(unlist(lapply(
  misclass_index_BH, FUN = function(x){x[,4]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_dis_BH) <- rho/100

sum_fdr_BH <- as.data.frame(matrix(c(rho/100,
```

```
                  apply(df_fdr_BH, 2, FUN = function(x){mean(x)}),
                  apply(df_fdr_BH, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_fdr_BH, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_fdr_BH, 2, FUN = function(x){var(x)}),
                  apply(df_fdr_BH, 2, FUN = function(x){min(x)}),
                  apply(df_fdr_BH, 2, FUN = function(x){max(x)})),
                nrow = 21, ncol = 7, byrow = FALSE),
                col.names = c("rho", "FDR","Q5","Q95"))
sum_fnr_BH <- as.data.frame(matrix(c(rho/100,
                  apply(df_fnr_BH, 2, FUN = function(x){mean(x)}),
                  apply(df_fnr_BH, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_fnr_BH, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_fnr_BH, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))
sum_mis_BH <- as.data.frame(matrix(c(rho/100,
                  apply(df_mis_BH, 2, FUN = function(x){mean(x)}),
                  apply(df_mis_BH, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_mis_BH, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_mis_BH, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))
sum_dis_BH <- as.data.frame(matrix(c(rho/100,
                  apply(df_dis_BH, 2, FUN = function(x){mean(x)}),
                  apply(df_dis_BH, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_dis_BH, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_dis_BH, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))

## Intersection

df_fdr_int <- as.data.frame(matrix(unlist(lapply(
  misclass_index_int, FUN = function(x){x[,1]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_fdr_int) <- rho/100

df_fnr_int <- as.data.frame(matrix(unlist(lapply(
  misclass_index_int, FUN = function(x){x[,2]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_fnr_int) <- rho/100

df_mis_int <- as.data.frame(matrix(unlist(lapply(
  misclass_index_int, FUN = function(x){x[,3]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_mis_int) <- rho/100

df_dis_int <- as.data.frame(matrix(unlist(lapply(
  misclass_index_int, FUN = function(x){x[,4]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_dis_int) <- rho/100

sum_fdr_int <- as.data.frame(matrix(c(rho/100,
                  apply(df_fdr_int, 2, FUN = function(x){mean(x)}),
                  apply(df_fdr_int, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_fdr_int, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_fdr_int, 2, FUN = function(x){var(x)}),
                  apply(df_fdr_int, 2, FUN = function(x){min(x)}),
                  apply(df_fdr_int, 2, FUN = function(x){max(x)})),
                nrow = 21, ncol = 7, byrow = FALSE))
sum_fnr_int <- as.data.frame(matrix(c(rho/100,
                  apply(df_fnr_int, 2, FUN = function(x){mean(x)}),
                  apply(df_fnr_int, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_fnr_int, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_fnr_int, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))
sum_mis_int <- as.data.frame(matrix(c(rho/100,
                  apply(df_mis_int, 2, FUN = function(x){mean(x)}),
                  apply(df_mis_int, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_mis_int, 2, FUN = function(x){quantile(x, probs = .95)}),
```

```
                  apply(df_mis_int, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))
sum_dis_int <- as.data.frame(matrix(c(rho/100,
                  apply(df_dis_int, 2, FUN = function(x){mean(x)}),
                  apply(df_dis_int, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_dis_int, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_dis_int, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))

## Internal Cross-Validation

df_fdr_ICV <- as.data.frame(matrix(unlist(lapply(
  misclass_index_ICV, FUN = function(x){x[,1]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_fdr_ICV) <- rho/100

df_fnr_ICV <- as.data.frame(matrix(unlist(lapply(
  misclass_index_ICV, FUN = function(x){x[,2]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_fnr_ICV) <- rho/100

df_mis_ICV <- as.data.frame(matrix(unlist(lapply(
  misclass_index_ICV, FUN = function(x){x[,3]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_mis_ICV) <- rho/100

df_dis_ICV <- as.data.frame(matrix(unlist(lapply(
  misclass_index_ICV, FUN = function(x){x[,4]})),
  ncol = 21, nrow = 200, byrow = FALSE))
colnames(df_dis_ICV) <- rho/100

sum_fdr_ICV <- as.data.frame(matrix(c(rho/100,
                  apply(df_fdr_ICV, 2, FUN = function(x){mean(x)}),
                  apply(df_fdr_ICV, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_fdr_ICV, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_fdr_ICV, 2, FUN = function(x){var(x)}),
                  apply(df_fdr_ICV, 2, FUN = function(x){min(x)}),
                  apply(df_fdr_ICV, 2, FUN = function(x){max(x)})),
                nrow = 21, ncol = 7, byrow = FALSE))
sum_fnr_ICV <- as.data.frame(matrix(c(rho/100,
                  apply(df_fnr_ICV, 2, FUN = function(x){mean(x)}),
                  apply(df_fnr_ICV, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_fnr_ICV, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_fnr_ICV, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))
sum_mis_ICV <- as.data.frame(matrix(c(rho/100,
                  apply(df_mis_ICV, 2, FUN = function(x){mean(x)}),
                  apply(df_mis_ICV, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_mis_ICV, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_mis_ICV, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))
sum_dis_ICV <- as.data.frame(matrix(c(rho/100,
                  apply(df_dis_ICV, 2, FUN = function(x){mean(x)}),
                  apply(df_dis_ICV, 2, FUN = function(x){quantile(x, probs = .05)}),
                  apply(df_dis_ICV, 2, FUN = function(x){quantile(x, probs = .95)}),
                  apply(df_dis_ICV, 2, FUN = function(x){var(x)})),
                nrow = 21, ncol = 5, byrow = FALSE))

## Visualization

## For Saving
setwd("Set appropriate directory")
## For Saving

## FDR

gg_fdr_BH <- ggplot(data = sum_fdr_BH, aes(x= V1, y = V2)) + ylim(0, 0.68) +
```

```
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) + labs(x = expression(rho), y = "FDR",
                                        title = "Benjamini-Hochberg")
gg_fdr_ICV <- ggplot(data = sum_fdr_ICV, aes(x= V1, y = V2)) + ylim(0, 0.68) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) + labs(x = expression(rho), y = "FDR",
                                        title = "Internally Cross-Validated")
gg_fdr_int <- ggplot(data = sum_fdr_int, aes(x= V1, y = V2)) + ylim(0, 0.68) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) + labs(x = expression(rho), y = "FDR",
                                        title = "Intersection")
grid.arrange(gg_fdr_BH, gg_fdr_ICV, gg_fdr_int, widths = c(.5,.5,.5,.5),
             layout_matrix = rbind(c(1,1,2,2), c(NA, 3,3, NA)))

dev.copy(pdf, "fdr_ribbon.pdf")
dev.off()

gg_fdr_BH <- ggplot(data = sum_fdr_BH, aes(x= V1, y = V2)) +
  ylim(0, 1) + geom_line(aes(x = V1, y = V6)) +
  geom_line(linetype = "dashed", aes(x = V1, y = V7))+
  labs(x = expression(rho), y = "FDR", title = "Benjamini-Hochberg")
gg_fdr_ICV <- ggplot(data = sum_fdr_ICV, aes(x= V1, y = V2)) +
  ylim(0, 1) + geom_line(aes(x = V1, y = V6)) +
  geom_line(linetype = "dashed", aes(x = V1, y = V7)) +
  labs(x = expression(rho), y = "FDR", title = "Internally Cross-Validated")
gg_fdr_int <- ggplot(data = sum_fdr_int, aes(x= V1, y = V2)) +
  ylim(0, 1) + geom_line(aes(x = V1, y = V6)) +
  geom_line(linetype = "dashed", aes(x = V1, y = V7)) +
  labs(x = expression(rho), y = "FDR", title = "Intersection")
grid.arrange(gg_fdr_BH, gg_fdr_ICV, gg_fdr_int, widths = c(.5,.5,.5,.5),
             layout_matrix = rbind(c(1,1,2,2), c(NA, 3,3, NA)))

dev.copy(pdf, "fdr_minmax.pdf")
dev.off()

## FNR

gg_fnr_BH <- ggplot(data = sum_fnr_BH, aes(x= V1, y = V2)) + ylim(0.007, 0.034) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) + labs(x = expression(rho), y = "FNR",
                                        title = "Benjamini-Hochberg")
gg_fnr_INV <- ggplot(data = sum_fnr_ICV, aes(x= V1, y = V2)) + ylim(0.007, 0.034) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) + labs(x = expression(rho), y = "FNR",
                                        title = "Internally Cross-Validated")
gg_fnr_int <- ggplot(data = sum_fnr_int, aes(x= V1, y = V2)) + ylim(0.007, 0.034) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) + labs(x = expression(rho), y = "FNR",
                                        title = "Intersection")
grid.arrange(gg_fnr_BH, gg_fnr_INV, gg_fnr_int, widths = c(.5,.5,.5,.5),
             layout_matrix = rbind(c(1,1,2,2), c(NA, 3,3, NA)))

dev.copy(pdf, "fnr_ribbon.pdf")
dev.off()

## Misclassification

gg_mis_BH <- ggplot(data = sum_mis_BH, aes(x= V1, y = V2)) + ylim(0, 0.2) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) +
  labs(x = expression(rho), y = "Misclassification Probability",
       title = "Benjamini-Hochberg")
gg_mis_INV <- ggplot(data = sum_mis_ICV, aes(x= V1, y = V2)) + ylim(0, 0.2) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) +
  labs(x = expression(rho), y = "Misclassification Probability",
       title = "Internally Cross-Validated")
```

```
gg_mis_int <- ggplot(data = sum_mis_int, aes(x= V1, y = V2)) + ylim(0, 0.2) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) +
  labs(x = expression(rho), y = "Misclassification Probability",
       title = "Intersection")
grid.arrange(gg_mis_BH, gg_mis_INV, gg_mis_int, widths = c(.5,.5,.5,.5),
             layout_matrix = rbind(c(1,1,2,2), c(NA, 3,3, NA)))

dev.copy(pdf, "mis_ribbon.pdf")
dev.off()

## Discoveries

gg_dis_BH <- ggplot(data = sum_dis_BH, aes(x= V1, y = V2)) + ylim(60, 270) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) +
  labs(x = expression(rho), y = "Discoveries", title = "Benjamini-Hochberg")
gg_dis_INV <- ggplot(data = sum_dis_ICV, aes(x= V1, y = V2)) + ylim(60, 270) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) +
  labs(x = expression(rho), y = "Discoveries", title = "Internally Cross-Validated")
gg_dis_int <- ggplot(data = sum_dis_int, aes(x= V1, y = V2)) + ylim(60, 270) +
  geom_ribbon(aes(ymin = V3, ymax = V4), fill="grey60") +
  geom_line(aes(x = V1, y = V2)) +
  labs(x = expression(rho), y = "Discoveries", title = "Intersection")
grid.arrange(gg_dis_BH, gg_dis_INV, gg_dis_int, widths = c(.5,.5,.5,.5),
             layout_matrix = rbind(c(1,1,2,2), c(NA, 3,3, NA)))

dev.copy(pdf, "dis_ribbon.pdf")
dev.off()

## Prostate Cancer Data Analysis

load("Load Data")

### For Saving
setwd("Your Directory")
### For Saving

set.seed(13)

data <- prostatedata
p <- length(data[,1])
n <- length(data[1,])
q1 <- .1
q2 <- .3

data1 <- c()
data2 <- c()

for(i in 1:n){
  if(labels(data[1,i])==1){
    data1 <- cbind(data1,data[,i])
  } else {
    data2 <- cbind(data2,data[,i])
  }
}


## Vanilla BH

pval <- c(rep(0,p))

pval <- apply(data, 1, FUN = function(x){tpval(x = x[which(labels(x)==1)],
                                               y = x[which(labels(x)==2)])})

pind_BH <- benhoch(pval,q2)
```

```
nsigBH <- length(pind_BH)

nsigBH

## ICVBH

nsigICVBH <- c()

pval1 <- c(rep(0,p))
pval2 <- c()

for(i in 1:300){
  grp1col <- c(sample(1:ncol(data1), (ncol(data1)/2)),
               sample((ncol(data1)+1):ncol(data), (ncol(data2)/2)))

  grp1 <- data[,grp1col]
  grp2 <- data[,-grp1col]

  pval1 <- apply(grp1, 1, FUN = function(x)
    {tpval(x = x[which(labels(x)==1)], y = x[which(labels(x)==2)])})

  pind_grp1 <- benhoch(pval1,q2)

  if(is.null(pind_grp1)){
    nsigICVBH <- c(nsigICVBH, length(pind_ICVBH))
  }else{
    if(length(pind_grp1)==1){
      pval2 <- tpval(x = grp2[pind_grp1,which(labels(grp2[1,])==1)],
                     y = grp2[pind_grp1,which(labels(grp2[1,])==2)])
      pind_ICVBH <- benhoch(pval2,q2)

      nsigICVBH <- c(nsigICVBH, length(pind_ICVBH))
    }else{
      pval2 <- apply(grp2[pind_grp1,], 1, FUN = function(x)
        {tpval(x = x[which(labels(x)==1)], y = x[which(labels(x)==2)])})

      pind_ICVBH <- benhoch(pval2,q2)

      nsigICVBH <- c(nsigICVBH, length(pind_ICVBH))
    }
  }
}

mean(nsigICVBH)

## Intersection BH

nsigIBH <- c()

pval1 <- c(rep(0,p))
pval2 <- c()

for(i in 1:300){
  grp1col <- c(sample(1:ncol(data1), (ncol(data1)/2)),
               sample((ncol(data1)+1):ncol(data), (ncol(data2)/2)))

  grp1 <- data[,grp1col]
  grp2 <- data[,-grp1col]

  pval1 <- apply(grp1, 1, FUN = function(x)
    {tpval(x = x[which(labels(x)==1)], y = x[which(labels(x)==2)])})

  pind_grp1 <- benhoch(pval1,q2)

  pval2 <- apply(grp2, 1, FUN = function(x)
    {tpval(x = x[which(labels(x)==1)], y = x[which(labels(x)==2)])})
```

```
  pind_grp2 <- benhoch(pval2,q2)

  pind_IBH <- unique(sort(c(pind_grp1[which(pind_grp1 %in% pind_grp2)],
                            pind_grp2[which(pind_grp2 %in% pind_grp1)])))

  nsigIBH <- c(nsigICVBH, length(pind_ICVBH))
}


layout(matrix(c(1,2), ncol = 2, nrow = 1))
hist(nsigICVBH, main = "Histogram for ICV-BH", xlab = "selected genes")
hist(nsigIBH, main = "Histogram for I-BH", xlab = "selected genes")

dev.copy(pdf, "IIBH_hist.pdf")
dev.off()

mean(nsigIBH)
```