Università degli Studi di Milano–Bicocca
Scuola di Economia e Statistica

Corso di Laurea in
Scienze Statistiche ed Economiche


# Robust Variable Selection: An Approach Based on Knockoffs

Relatore: Prof. Tommaso Rigon
Correlatore: Prof. Federico Camerlenghi

Tesi di laurea di:
Federico Boiocchi
Matricola N. 895221

Anno Accademico 2024/2025

# Contents

**Acknowledgements**

Voglio ringraziare il Prof. Tommaso Rigon per i preziosi consigli, le stimolanti conversazioni, le opportunità ricevute e il tempo dedicatomi non solo durante la stesura della tesi ma soprattutto nel corso di questi tre anni.

Ringrazio poi la mia famiglia per il supporto che mi ha sempre dimostrato.

Infine ringrazio il mio amico Andrea con cui ho condiviso questo percorso universitario.

# Chapter 1

# Classical Inferential Tools

This chapter introduces foundational concepts in the theory of single hypothesis testing. The main references are the section on hypothesis testing in chapter 2 of Pace & Salvan (2001) and chapter 8 of Casella & Berger (2008).

## 1.1 Statistical models

A fundamental concept in statistical inference is to consider observed data $(y_1, \ldots, y_n)$ as realizations of an underlying random variable $Y$. This assumption allows us to link scarce empirical evidence with the random behavior of $Y$ on the whole population or domain $S_Y$. Therefore, the randomness of the phenomenon object of study is formally expressed through an unknown probability distribution $\mathbb{P}$. In this sense, we say that $Y$ is distributed according to $\mathbb{P}$ or $Y \sim \mathbb{P}$. Consequently, the purpose of inferential statistics is to extract valid information about $\mathbb{P}$ from a sample $(y_1, \ldots, y_n)$ belonging to a **sample space** $\mathcal{Y}$. The sample space is the set of all possible samples given a specific sample amplitude $n$ and a specific data generating model. In extracting information about $\mathbb{P}$, one must define the set of all probability distributions $\mathbb{P}$ to choose from and a criterion to find the best $\mathbb{P}$ based on collected data. Thus, finding the most plausible $\mathbb{P}$, according to data, can be considered as a constrained optimization problem, in which the criterion used is often based on likelihood maximization, the search space $\mathcal{F}$ is the **statistical model**, namely, the set of all probability distributions compatible with the data, while the simplifying assumptions made on the model represent the constraints of the problem. Probably the easiest approach consists of fixing the functional form for all the elements in $\mathcal{F}$ and indexing them to a parameter $\theta$ that is free to vary in a parameter space $\Theta \subseteq \mathbb{R}^p$. A model of this kind is known as a **parametric statistical model** with

the following formulation

$$\mathcal{F} = \{\mathbb{P}_\theta : \theta \in \Theta\}, \tag{1.1}$$

where $\theta$ is an unknown but deterministic population parameter, namely, a parameter characterizing the universal behavior of the phenomenon $Y$; $\Theta$ is a subset of the Euclidean space such as $\mathbb{R}^p$ and $p \in \mathbb{N}^+$; lastly, $\mathcal{F}$ is a set of probability mass functions (PMFs), in the case $Y$ was a discrete random variable, or a set of probability density functions (PDFs) when $Y$ is a continuous random variable. Clearly, fixing a specific functional form or distribution class for all $\mathbb{P}_\theta$ represents another assumption that, in this basic framework, is not considered as an object of inference. In this context, the need to introduce a parametric simplification is immediately clear, since it allows to simplify dramatically the inferential phase. Actually, at the root of this simplification stands a fundamental property of $\theta$ called **identifiability**.

**Definition 1.1.** A parametric statistical model (1.1) is said to be identifiable if any pair of values $\theta^{'}$, $\theta^{''}$ of the parameter $\theta$, (both $\in \Theta$) define the same probability law if and only if $\theta^{'} = \theta^{''}$. Namely:

$$\mathbb{P}_{\theta'} = \mathbb{P}_{\theta''} \iff \theta^{'} = \theta^{''}.$$

In other words, $\theta$ is said to be identifiable if there exists a one-to-one correspondence between $\Theta$ and $\mathcal{F}$. As a result, inference will be carried out on a subset $\Theta$ of the Euclidean space, called the parameter space, rather than a space of functions $\mathcal{F}$. However, parameter estimation is not the only interest of inferential statistics. In fact, scientists are often interested in testing the coherence of empirical evidence with certain mathematically formulated "hypotheses" on the data distribution; this alternative way of extracting information from empirical evidence is called hypothesis testing.

## 1.2 Hypothesis testing

In real-world problems, it is often of interest to test whether a conjecture about the distribution of the data is validated by empirical evidence or not. In statistical language, this is formalized through a hypothesis.

**Definition 1.2.** A statistical hypothesis $H$ is a conjecture on the probability distribution of a random variable.

Actually, considering only a single statement $H$ turns out to be not that interesting. Indeed, in many real-world scenarios, one would like to choose between two alternatives. For this reason, two complementary hypotheses are often defined, and the goal of a hypothesis testing procedure is to choose which of the two is more supported by the data. In this setting, the hypothesis to validate is called the **null hypothesis** and it is denoted by $H_0$, while the complementary one is known as the **alternative hypothesis** or $H_1$. These attributes arise because $H_0$ is often the simple hypothesis representing no effect, while $H_1$ is the alternative hypothesis that assumes a specific effect. Therefore, once $\theta$ is defined as a population parameter in a statistical model, such as in (1.1), and identifiability is satisfied, the general structure of a null versus alternative hypothesis test is

$$H_0 : \theta \in \Theta_0 \qquad \text{vs} \qquad H_1 : \theta \in \Theta_0^c, \tag{1.2}$$

where $\Theta_0$ is a subset of the parameter space $\Theta$ and $\Theta_0^c$ is its complementary set such that their union returns $\Theta$. An important aspect of hypothesis testing is the distinction between hypotheses that fully specify a model and those that only partially define it. The former case identifies a single probability distribution within the model, while the latter describes a subset of possible distributions. This distinction is so relevant that the first type is referred to as a **simple hypothesis**, while the second is known as a **composite hypothesis**. It is essential to clarify that it is possible to simplify the hypothesis comparison structure, by making complementary conjectures on $\Theta$, as written in (1.2), only thanks to the identifiability of the model. Otherwise, only a specification of two complementary subsets of probability functions would have been allowed. It is also worth mentioning that testing complementary hypotheses is profoundly different from doing estimation to discover the best probability distribution for a random phenomenon. Even though inference is still involved, the interest of a testing procedure is only to choose which hypothesis between $H_0$ and $H_1$ is more plausible according to data. As a consequence, the statistical models specified in the two hypotheses are not questioned. Moreover, an important observation is that, strictly speaking, a hypothesis is never truly "accepted" as this would imply that sufficient evidence has been found to prove it, something that is not possible. Instead, hypotheses are said to be "not rejected". Nonetheless, the term "accepted" is sometimes used for simplicity and to avoid misunderstanding. To conclude, it is important to note that rejected hypotheses are sometimes referred to as discoveries.

For this reason, the terms true discovery and false discovery are sometimes used for correct and incorrect rejection.

### 1.2.1 Statistical test procedure

In a parametric framework, given a statistical model such as (1.1), and a testing structure as (1.2), other tools are required to choose between $H_0$ and $H_1$.

**Definition**: A **test statistic** $t$ is a function of both the data and the parameter under the null hypothesis. It measures the gap between the statement in $H_0$ and the observed sample.
Therefore, $t$ is a function that shows which model, between the one specified under $H_0$ and the one under $H_1$, is more reasonable based on the information in the sample. It is defined as follows:

$$t = t(y_1, \dots, y_n, \theta) \quad \Longrightarrow \quad t : \mathcal{Y} \times \Theta_0 \to \mathcal{T}. \tag{1.3}$$

Where $\mathcal{Y}$ is the sample space, $\Theta_0$ the subset of the parameter space specified under $H_0$, and $\mathcal{T}$ the codomain of the test statistic. It is immediately clear how mapping the sample space into the codomain $\mathcal{T}$, reduces the complexity of the problem; indeed, when $Y$ is a continuous random variable, the sample space is a subset of $\mathbb{R}^n$ while $\mathcal{T}$ is a subset of $\mathbb{R}$. Therefore, the decision between the two hypotheses will be based on a summary $t$ of the sample rather than the sample itself; in this way, we are able to reduce from $n$ to 1 the dimensionality of the problem. Besides that, $t$ has to satisfy two properties in order to be a proper test statistic. The first one is the definition (1.3), with a stress on the fact that $t$ has to be a function also of $\theta$ while assuming $H_0$ to be true. The second one requires that the distribution of $t$ under $H_0$ must not depend on $\theta$. In this context, it is meaningful to talk about the distribution of $t$ because the repeated sampling principle guarantees that the test statistic is a random variable since it is a function of random data. In this regard, when $t$ is considered as a random variable, it is often used the notation $T = t(Y)$ instead of a small $t$. Coming to the actual decision phase, the test statistic (1.3) suggests rejecting or accepting the null hypothesis by partitioning the sample space into two disjoint sets,

$$R = \{y \in \mathcal{Y} : t(y, \theta) \text{ suggests rejecting } H_0\},$$

that is the **rejection region**, namely the set of sample values that would make us reject $H_0$ and

$$A = \{y \in \mathcal{Y} : t(y, \theta) \text{ suggests accepting } H_0\},$$

which is the **acceptance region**, namely, the subset of the sample space that would lead towards accepting $H_0$. $R$ and $A$ are two disjoint sets, and their union forms the sample space. Given the complementary structure of the regions, it is usually enough to specify only the rejection region $R$ of the test. In this framework, whenever the sample belongs to $R$, or equivalently, the test statistic takes on values in the mapped version of the rejection region, $H_0$ will be rejected; otherwise, $H_0$ will be accepted. As defined in (1.3), a test statistic quantifies the discrepancy between the observed data and the null hypothesis. Intuitively, large absolute values of the test statistic suggest stronger evidence against $H_0$, making it reasonable to reject the null hypothesis in these cases. Based on this idea, three types of tests can be defined. The first type rejects $H_0$ for large positive values of the test statistic; this is known as a **right-tailed test**, due to the location of the critical region on the right side of the distribution of $t$. The second type rejects $H_0$ for large negative values of t, and is called a **left-tailed test**. Finally, the third one, known as a **two-tailed test**, rejects the null hypothesis for both large positive and large negative values of $t$. At this point, it is essential to understand that a statistical test is not an infallible mechanism to decide between a null and an alternative hypothesis; as a result, it commits errors. Below is shown a confusion matrix displaying the four possible error cases.

**Table 1.1:** Hypothesis Testing Confusion matrix

| | $H_0$ false | $H_0$ true |
|---|---|---|
| reject $H_0$ | correct decision | type I error |
| accept $H_0$ | type II error | correct decision |

The columns are labeled with the ground truth, while the rows are labeled with the choices made. A **type I error** is committed when the null hypothesis is rejected, but it was actually true. This error is also known as a false positive or false discovery, since the procedure is declaring the presence of an effect while in reality there is none. On the other side, a **type II error** happens when the null hypothesis is accepted, but it was actually false. This type of error is also known as a false negative, meaning that the procedure has failed to detect an effect that was actually present. Thus, it

becomes relevant to understand how to design statistical tests that control these two errors while being useful in detecting signals. But first, it is essential to introduce, more precisely, how these error measures are computed.

### 1.2.2 Significance Level

It is called the **significance level** of the test with rejection region $R$ the value

$$\alpha = \sup_{\theta \in \Theta_0} \mathbb{P}_\theta(Y \in R).$$

First of all, a key concept in this definition is that computing a probability based on the distribution specified under $H_0$ is equivalent to assuming $H_0$ is true. Therefore, given the general hypothesis test structure as in (1.2), and since $Y \in R$ means rejecting $H_0$, the quantity $\mathbb{P}_\theta(Y \in R)$ represents the probability of committing a type I error, for a fixed $\theta$. Consequently, by letting vary $\theta$ in $\Theta_0$ and taking the sup, we obtain the maximum probability of committing a type I error, also known as the significance level of the test. It could also happen that the probability of making a type I error is the same for all values of $\theta$ under the null hypothesis; in that case, the test is said to have a constant significance level $\alpha$. Given a test statistic $t$, a criterion to design $R$ could be to arbitrarily fix $\alpha$ in advance and then find the upper and lower bounds of $R$ in order to obtain the pre-determined significance level. Hence, the resulting rejection region will be denoted by $R_\alpha$. Intuitively, the choice of the significance level stems from the amount of evidence we require to reject the null hypothesis. In other words, and high $\alpha$ would mean more probability mass is assigned to the rejection region; namely, it will be sufficient less evidence to reject $H_0$. Conversely, a low $\alpha$ means less probability mass is assigned to $R$, and then more evidence is required to reject the null hypothesis. As a result, the choice of the significance level is crucial in determining how strict (low $\alpha$) or loose (high $\alpha$) the test will be in declaring discoveries. In the scientific literature and applications, the most commonly used significance level values are $\alpha = 0.1, 0.05, 0.01$.

### 1.2.3 Power function

The **power function** of a test with rejection region $R$ is

$$\pi(\theta) = \mathbb{P}_\theta(Y \in R), \quad \theta \in \Theta, \quad \Longrightarrow \quad \pi : \Theta \to [0, 1]. \tag{1.4}$$

Thus, it is the probability of rejecting the null hypothesis as a function of $\theta$. Clearly, when $\theta \in \Theta_0$, the power function represents the probability of committing a type I error as a function of $\theta$, or $\alpha(\theta)$. Vice versa, when $\theta \in \Theta_0^c$, $\pi(\theta)$ represents the power of the test. In other terms, the power measures the ability of the test to reject $H_0$ when it is actually false, namely, detecting an effect when it is actually present. Hence, depending on the value of $\theta$ provided, the power function can yield either the probability of a type I error or the power of the test. The latter is related to the probability of committing a type II error $\beta$, indeed, we have:

$$\beta(\theta) = 1 - \pi(\theta) = \mathbb{P}_\theta(Y \in A), \quad \theta \in \Theta_0^c,$$

that means $\beta(\theta)$, for a specific $\theta$, is the probability of accepting $H_0$ when it is actually false. This follows from the fact that the event $Y \in A$ means accepting $H_0$, and its probability is assigned assuming the alternative hypothesis is true. Therefore, a desirable $\pi(\theta)$ is 0 over $\Theta_0$, so that the probability of detecting false positives is null, while on $\Theta_0^c$ is 1, meaning that the test can detect perfectly true signals. However, given the fallibility of the test, this behavior is not achievable in practice; nonetheless, if the test is well specified, $\pi(\theta)$ will show a sigmoid behavior, in the case of a one-tailed test, or a u-shape behavior for a two-tailed test; while at the same time potentially reaching 0 over $\Theta_0$ and 1 over $\Theta_0^c$. Further details about its representation are presented in the example section of this chapter. Besides that, one would ideally aim to design a test that minimizes both $\alpha$ and $\beta$. However, this is not an easy task, as these two probabilities are inversely related. For this reason, a comprehensive theory has been developed to guide the construction of optimal tests; however, this topic falls outside the scope of this thesis. At this point, before introducing the next topic, it is useful to examine the probability confusion matrix of a testing procedure, which is similar to Table 1.1, but it is expressed in terms of the probabilities associated with the four possible outcomes.

**Table 1.2:** Hypothesis Testing Probability Confusion matrix

| | $H_0$ false | $H_0$ true |
|---|---|---|
| reject $H_0$ | $\pi = 1 - \beta$ | $\alpha$ |
| accept $H_0$ | $\beta$ | $1 - \alpha$ |

Theoretically, for a fixed $\theta$, these four quantities are well-defined with closed formulas. However, to deeply understand their meaning, it could be useful to present how they can be estimated through simulation via Monte Carlo. It is worth mentioning that, in practice, we are not necessarily interested in approximating these quantities via simulation since, as presented in this chapter, in simple contexts, there exist analytical formulas to compute them. In this context, simulating data is essential because it allows us to know the true quantities from which we are generating, enabling a direct comparison between the true values and their approximations. In this regard, we consider the approximation of the power function, whose pseudo-code is detailed in Algorithm 1. In this way, we are estimating both the approximated

**Algorithm 1:** Power function Approximation via Monte Carlo (Single Hypothesis Testing)

1. Fix a true significance level $\alpha \in [0, 1]$ and a single hypothesis test structure $H_0 : \theta \in \Theta_0$ vs $H_1 : \theta \in \Theta_0^c$.
2. Fix a number $M$ of Monte Carlo iterations.
3. Fix a grid $\{\theta_1, \ldots, \theta_p\} \subset \Theta$ of parameter values on which to evaluate the approximated power $\hat{\pi}(\theta_j)$ for all $j \in \{1, \ldots, p\}$.
4. **For** $j$ in $1 : p$ do:
   (a) Fix a specific $\theta = \theta_j$.
   (b) **For** $k$ in $1 : M$ do:
      i. Simulate data $D$ from $f(D, \theta_j)$
      ii. Compute the corresponding test statistic $T_k$
      iii. Reject or accept $H_0$ based on the test statistic $T_k$ and the test structure
      iv. store the value $W_k = \mathbb{1}(H_0 \text{ is rejected})$
   (c) **End For**
   (d) Compute $\hat{\pi}(\theta_j) = \dfrac{\sum_{i=1}^{M} W_k}{M}$
5. **End For**
6. Plot $\{\theta_1, \ldots, \theta_p\}$ vs $\{\hat{\pi}(\theta_1), \ldots, \hat{\pi}(\theta_p)\}$.

significance level $\hat{\alpha}(\theta) = \hat{\pi}(\theta)$ for $\theta \in \Theta_0$, and the power of the test for different values of $\theta \in \Theta_0^c$. Globally, $\hat{\pi}(\theta)$ represents a discrete version of the power function $\pi(\theta)$ for all $\theta \in \Theta$ . More precisely, $\hat{\alpha}(\theta)$ represents the probability of having a

false positive for a fixed value of $\theta \in \Theta_0$ because we are computing the frequency of discoveries under the null hypothesis, namely, false discoveries over the total number of true nulls. On the other hand, $\hat{\pi}(\theta)$ is the frequency of true discoveries among true alternatives, hence it approximates the test power. Clearly, to perform the testing, the true significance level must be determined in advance to allow the construction of the rejection region. Consequently, by the law of large numbers, we are guaranteed that, as $M \to +\infty$, the approximated power function converges to the real one. The same could be done with the probability of committing a type II error $\beta(\theta)$. This idea of approximating probabilities via Monte Carlo will turn out to be highly useful in multiple testing contexts, where, only through simulation, we will be able to compute average power functions to compare different multiple testing approaches. In that case, we will compute the average power function of each multiple comparison procedure by averaging true positive rates (TPRs) over $M$ Monte Carlo iterations across different ground truths.

### 1.2.4 Observed significance level

In a statistical test, the probability distribution of the test statistic $T$ is sensitive to a change of the probabilistic model underlying the data from $H_0$ towards $H_1$. Considering, for example, as a test statistic the sample mean, it is clear how changing the distribution of data will shift the distribution of the resulting sample mean. In simple cases, this sensitivity is shown in a stochastic ordering of distributions of $T$, namely, by moving $\theta$ from $\Theta_0$ to $\Theta_1$, we obtain cumulative distribution functions of $T$ that could be ordered. Under $H_0$, the distributions of the univariate test statistic $T$ fall into the set $\{\mathbb{P}_\theta^T, \theta \in \Theta_0\}$, while, under $H_1$, one of the following situations could happen:

1. for a test with right-unilateral critical region: $T$ will have all probability distributions that will be stochastically bigger than those under $H_0$, namely, $\mathbb{P}_1(T_1 > t) \geq \mathbb{P}_0(T_0 > t)$ for all t, where $T_0$ is the test statistic under the null hypothesis with distribution $\mathbb{P}_0$ and $T_1$ is the test statistic under the alternative hypothesis with distribution $\mathbb{P}_1$

2. for a test with left-unilateral critical region: T will have all probability distributions that will be stochastically smaller. Namely, $\mathbb{P}_1(T_1 > t) \leq \mathbb{P}_0(T_0 > t)$

3. for a test with bilateral-critical region: T will have all probability distributions either stochastically smaller or stochastically bigger.

Considering a right-unilateral critical region, we can define the test statistic observed value as $t^{\text{obs}} = t(y^{\text{obs}})$. Thus, since high values of $t$ suggest a disagreement between $y^{\text{obs}}$ and $H_0$, a synthetic measure of this gap is given by:

$$\alpha^{\text{obs}} = \sup_{\theta \in \Theta_0} \mathbb{P}_\theta(T \geq t^{\text{obs}}).$$

The quantity $\alpha^{\text{obs}}$ is said to be the **observed significance level** or **p-value** of the test. Clearly, we have $0 \leq \alpha^{\text{obs}} \leq 1$. If $\alpha^{\text{obs}} \approx 0$ then $t^{\text{obs}}$ is significantly large with respect of all distributions of $T$ under $H_0$. In other words, even if one takes the $\mathbb{P}_\theta$ under $H_0$ that maximizes the probability assigned to the event $T \geq t^{\text{obs}}$, the resulting probability will still be small; this suggest a very low accordance between $y^{\text{obs}}$ and $H_0$. On the other hand, if one considers a left-unilateral critical region, the observed significance level will be:

$$\alpha^{\text{obs}} = \sup_{\theta \in \Theta_0} \mathbb{P}_\theta(T \leq t^{\text{obs}}).$$

If $\alpha^{\text{obs}} \approx 0$ then $t^{\text{obs}}$ is significantly small with respect of all distributions of $T$ under $H_0$. As a result, there is a huge gap between the sample and the hypothesis $H_0$. A slightly more complicated case happens with a bilateral test, in which the observed $\alpha$ is defined as follows:

$$\alpha^{\text{obs}} = 2 \cdot \sup_{\theta \in \Theta_0} \min \{\mathbb{P}_\theta(T \leq t^{\text{obs}}), \mathbb{P}_\theta(T \geq t^{\text{obs}})\}. \tag{1.5}$$

In this case, it is used the function $\min\{\cdot\}$ to place $t^{\text{obs}}$ in the more critical tail; whereas the sup over $\Theta_0$ helps to consider the law $\mathbb{P}_\theta$ of $T$ under $H_0$ for which $t^{\text{obs}}$ is less critical. Then, the product by 2 is essential to maintain $0 \leq \alpha^{\text{obs}} \leq 1$. In this regard, the observed significance level $\alpha^{\text{obs}}$ can be used as a test statistic; indeed, if $T$ has a continuous distribution, both in the unilateral and in the bilateral case, we have:

$$\alpha^{\text{obs}}(Y) \overset{H_0}{\sim} U(0,1),$$

namely, the observed $\alpha$ is distributed under $H_0$ according to a uniform continuous distribution in $[0,1]$. Therefore, the rejection and acceptance regions could be redefined based on the p-value. A statistical test with significance level $\alpha$ will have the following regions:

$$R_\alpha = \{y \in \mathcal{Y} : \alpha^{\text{obs}} < \alpha\} \qquad A_\alpha = \{y \in \mathcal{Y} : \alpha^{\text{obs}} \geq \alpha\}.$$

This last definition allows us to directly use the observed significance level as a proxy of whether the sample falls in the rejection or acceptance region, and consequently, whether to reject or accept the null hypothesis. The advantage of this approach is that it does not require any explicit computation of the critical region, and it is identical regardless of its structure. Conversely, a disadvantage of using p-values is that we need to know the distribution of the test statistics to compute them, which is not always possible. The following is an example of a well-known bilateral test.

### 1.2.5 Two-tailed Z-test

Given $n$ independent and identically distributed (i.i.d) observations from a Normal distribution $N(\mu, \sigma_0^2)$, namely, $(Y_1, \dots, Y_n) \overset{iid}{\sim} N(\mu, \sigma_0^2)$, it is known that the sample mean $\overline{Y}_n \sim N(\mu, \sigma_0^2/n)$. Considering a known variance $\sigma_0^2$, we would like to test the following system of hypotheses:

$$H_0 : \mu = \mu_0 \qquad \text{vs} \qquad H_1 : \mu \neq \mu_0.$$

Then a test statistic can be defined based on a **pivotal quantity** that, in this case, is taken as the standardization of $\overline{Y}_n$. This choice is made since $\bar{y}_n$ well summarizes the information carried by data, while standardizing allows for simplifying the test statistic distribution; more precisely, the test statistic will be:

$$Z = \frac{\overline{Y}_n - \mu}{\sigma/\sqrt{n}} \overset{H_0}{\sim} N(0,1).$$

In other terms $Z$ measures the discrepancy between the sample and $H_0$. It is worth noting that the denominator of $Z$ is the standard deviation of the sample mean and follows from both the property of linear combination of Gaussian random variables and the scaling property of the variance. Furthermore, we can also specify the critical region $R_\alpha$, where $\mathbf{y} = (y_1, \dots, y_n)$ :

$$R_\alpha = \left\{ \mathbf{y} \in \mathbb{R}^n : \overline{Y}_n < \mu_0 - z_{1-\frac{\alpha}{2}} \cdot \frac{\sigma_0}{\sqrt{n}} \right\} \cup \left\{ \mathbf{y} \in \mathbb{R}^n : \overline{Y}_n > \mu_0 + z_{1-\frac{\alpha}{2}} \cdot \frac{\sigma_0}{\sqrt{n}} \right\}.$$

In this case, $R_\alpha$ is found by following the sequence of equalities below:

$$\begin{aligned}\alpha &= \mathbb{P}_{H_0}((y_1,\ldots,y_n)\in R_\alpha)\\ &= \mathbb{P}_{H_0}(\overline{Y}_n \in R_\alpha)\\ &= \mathbb{P}_{H_0}(\overline{Y}_n < l) + \mathbb{P}_{H_0}(\overline{Y}_n > u)\\ &= \Phi_{H_0}\left(\frac{\overline{Y}_n-\mu_0}{\sigma_0/\sqrt{n}} < l\right) + \Phi_{H_0}\left(\frac{\overline{Y}_n-\mu_0}{\sigma_0/\sqrt{n}} > u\right).\end{aligned}$$

The first equality is an imposition in which it is fixed the significance level of the test. The subscript $H_0$ of $\mathbb{P}$ means assuming the null hypothesis to be true. The second equality stems from having defined, through a test statistic, the mapped version of the rejection region. Hence, if the sample belongs to $R_\alpha$, then the sample mean belongs to the transformation of the rejection region through the test statistic. The third step is a consequence of considering a two-tailed test, in which $R_\alpha$ includes both high negative and high positive values. To conclude, the last equality derives from the standardization of the arguments within the two probability distributions. Specifically, in the last step $\Phi_{H_0}$ represents the cumulative distribution function of a standardized normal $N(0,1)$. Consequently, by imposing the first and second term of the last step respectively equal to $\alpha/2$ we obtain the two symmetric quantiles of $N(0,1)$ of order $1-\alpha/2$, namely, $l=-z_{1-\frac{\alpha}{2}}$ and $u=z_{1-\frac{\alpha}{2}}$. As a result, the definition of $R_\alpha$ is just a consequence of having rewritten the argument of both probability distributions and having isolated on the left the test statistic $\overline{Y}_n$. Regarding the observed significance level or p-value, in this test, it is defined as follows:

$$\alpha^{\text{obs}} = 2\cdot\min\left(\Phi_{H_0}\left(\frac{\overline{Y}_n-\mu_0}{\sigma_0/\sqrt{n}}\right), 1-\Phi_{H_0}\left(\frac{\overline{Y}_n-\mu_0}{\sigma_0/\sqrt{n}}\right)\right).$$

This definition stems from having a null hypothesis that specifies a unitary set, namely, $\Theta_0=\{\mu_0\}$, and for this reason, there is no need to take the sup over $\Theta_0$. The rest is just a consequence of the standard definition of a two-tailed p-value. We could also have defined the critical and acceptance regions based on the observed significance level:

$$\begin{aligned}R_\alpha &= \{\mathbf{y}\in\mathbb{R}^n : \alpha^{\text{obs}} < \alpha\},\\ A_\alpha &= \{\mathbf{y}\in\mathbb{R}^n : \alpha^{\text{obs}} \geq \alpha\}.\end{aligned}$$

### 1.2.6 Examples

In this example, a right-tailed Z-test on the mean of a Gaussian population with known variance $\sigma^2 = 1$ has been performed, considering a composite null hypothesis. More precisely, the two hypotheses being tested are:

$$H_0 : \mu \leq \mu_0 \qquad \text{vs} \qquad H_1 : \mu > \mu_0,$$

with $\Theta_0$ equal to the left ray $\{\mu \in \mathbb{R} : \mu \leq \mu_0\}$, while the alternative hypothesis is the right ray $\{\mu \in \mathbb{R} : \mu > \mu_0\}$. In this case, the test statistic $Z$ is taken to be the standardization of the sample mean. As in a two-tailed Z-test, the test statistic under $H_0$ is distributed according to a standard Normal. In order to have a graphical representation of this hypothesis test, in Figure 1.1 are shown two distributions: on the left, the probability density of the sample mean under a specific $\mu \in \Theta_0$, namely, $\mu_0 = 0$; whereas on the right, it is drawn a generic distribution under $H_1$ with $\mu_1 = 0.08$. Although both the null and the alternative hypotheses define an entire ray of values, a single $\mu_1 \in \Theta_1$ and $\mu_0 \in \Theta_0$ must be selected to enable an effective illustration. The number of observations is $n = 1000$, while $\alpha$ has been fixed to 0.05. The critical region has been computed as follows:

$$\begin{aligned} R_{0.05} &= \{\overline{X} \in \mathbb{R} : \overline{X} > \mu_0 + z_{1-0.05} \cdot \frac{\sigma}{\sqrt{n}}\} \\ &= \{\overline{X} \in \mathbb{R} : \overline{X} > 0 + 1.644 \cdot \frac{1}{\sqrt{1000}}\} \\ &= \{\overline{X} \in \mathbb{R} : \overline{X} > 0.0520\}. \end{aligned}$$

It is important to note that the x-axis is labeled with the sample mean $\overline{X}$, and not with the raw data $X$. Therefore the two Normals are $N(\mu_0, \sigma/\sqrt{n})$ and $N(\mu_1, \sigma/\sqrt{n})$. The acceptance region $A$ is colored in light blue, and it represents the subset of the sample mean space for which $H_0$ is not rejected; while on the right, colored in purple, $R$ represents the rejection region. Dashed lines represent the means $\mu_o$ and $\mu_1$ of the two Normals, while the continuous vertical line marks the so-called **critical threshold**, that is the boundary between $R$ and $A$. The area colored in red represents the probability of committing a type I error, since it is the probability of being in the critical region while assuming $H_0$ is true. On the other hand, the area colored in blue represents the probability of committing a type II error, namely, the probability of accepting $H_0$ but assuming $H_1$ is true. To conclude, the area colored in green represents the power of the test because it is the area above the rejection region

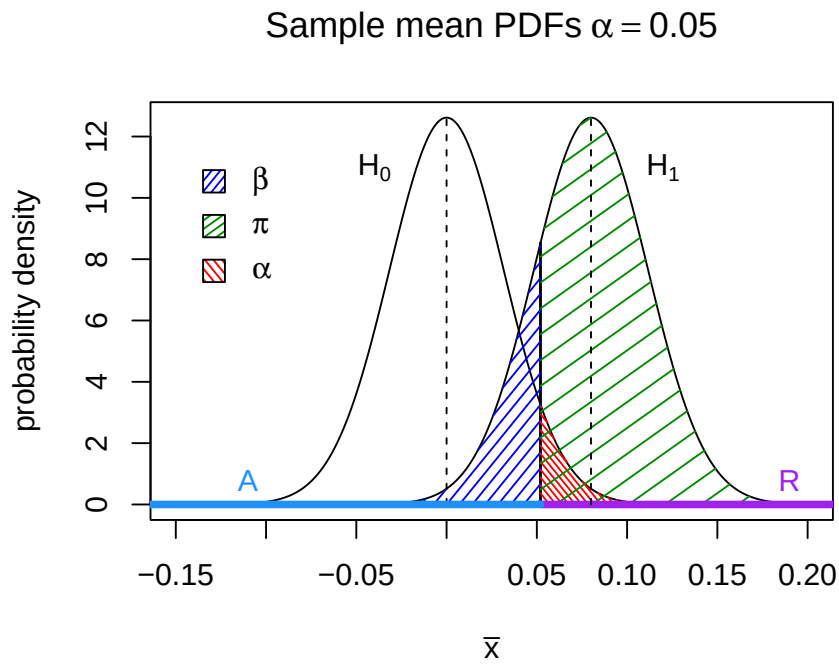


**Figure 1.1:** Sample mean PDFs under $H_0$, $N(\mu_0, \sigma/\sqrt{n})$ ,on the left, and under $H_1$, $N(\mu_1, \sigma/\sqrt{n})$ on the right . The areas underlying the two curves have been colored differently to show the power of the test $\pi$, and the probability of type I and II errors, respectively $\alpha$ and $\beta$. $\alpha$ is fixed to 0.05.

and under the sample mean PDF when $H_1$ is true. Another interesting illustration concerns the power function of this example represented in Figure 1.2; although it is unusual, an $\alpha = 0.25$ has been chosen to ease the readability of the visualization. Considering the example above, the power function has been computed as follows:

$$\begin{aligned}
\pi(\mu) &= \mathbb{P}(\text{reject } H_0 : \mu \in \Theta) \\
&= \mathbb{P}((X_1, \ldots, X_n) \in R_\alpha : \mu \in \mathbb{R}) \\
&= \mathbb{P}\left(\overline{X} > \mu_0 + z_{1-\alpha} \cdot \frac{\sigma}{\sqrt{n}} : \mu \in \mathbb{R}\right) \\
&= 1 - \mathbb{P}\left(\overline{X} < \mu_0 + z_{1-\alpha} \cdot \frac{\sigma}{\sqrt{n}} : \mu \in \mathbb{R}\right) \\
&= 1 - \mathbb{P}\left(\frac{\overline{X} - \mu}{\frac{\sigma}{\sqrt{n}}} < \frac{\mu_0 + z_{1-\alpha} \cdot \frac{\sigma}{\sqrt{n}} - \mu}{\frac{\sigma}{\sqrt{n}}} : \mu \in \mathbb{R}\right) \\
&= 1 - \Phi\left(\frac{\mu_0 - \mu}{\frac{\sigma}{\sqrt{n}}} + z_{1-\alpha}\right).
\end{aligned}$$

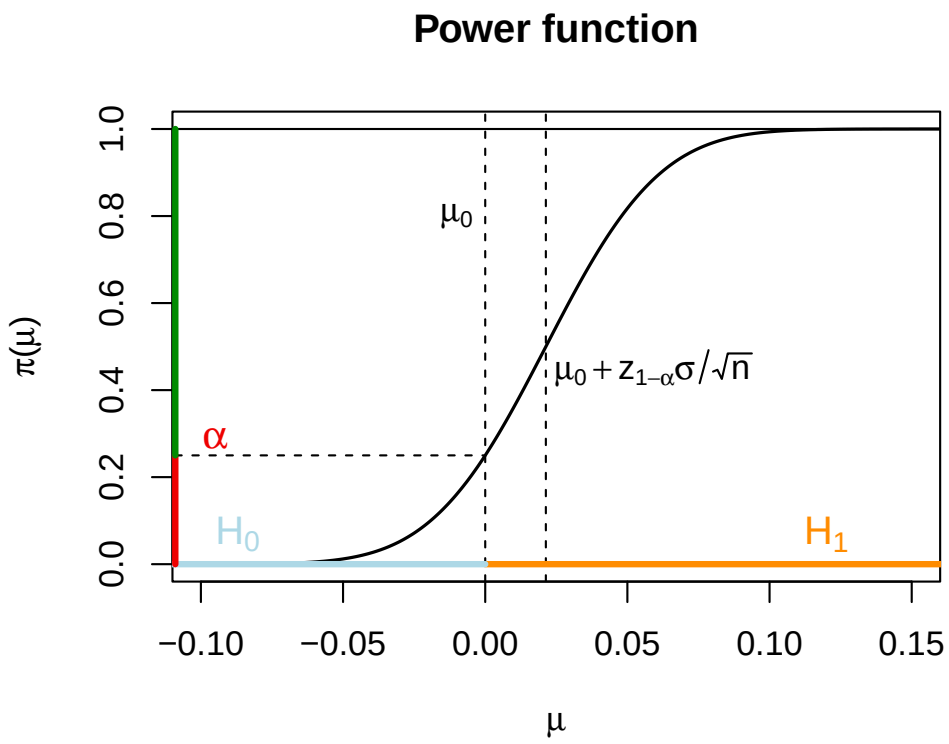


**Figure 1.2:** Power function of a right-tailed Z-test. The parameter space $\mathbb{R}$ on the x-axis is partitioned into two rays representing the null and alternative hypotheses. On the y-axis, are plotted in red, the values of the probability of type I error up to the significance level $\alpha = 0.25$. Whereas, in green are shown values of the power of the test for different $\mu$ specified under the alternative hypothesis. It is also shown $\mu_0$ and the critical threshold that corresponds to the inflection point of the curve.

The power function has a sigmoid behavior with domain the whole parameter space $\mathbb{R}$ and codomain the interval $[0, 1]$ since it is a probability by definition. Besides that, the domain of the function can be partitioned according to the hypothesis test structure; therefore, it is possible to visualize the significance level of the test as

$$\alpha = \sup_{\mu \leq \mu_0} \mathbb{P}(\text{reject } H_0 : \mu \in \mathbb{R}) = \sup_{\mu \leq \mu_0} \pi(\mu),$$

indeed, in the graph above, $\alpha$ is the maximum of all values of the probability of rejecting $H_0$ over $\Theta_0 = \{\mu \in \mathbb{R} : \mu \leq \mu_0\}$. On the opposite, values greater than $\alpha$ represent the power of the test for different mean values specified under the alternative hypothesis. Lastly, It can be proved that the x coordinate of the inflection point is the critical threshold of the test. However, having analyzed a single example

does not capture the behavior of the test across different values of $\alpha$. For this reason, it is useful to compare the graphs of the PDFs of the sample mean under $H_0$ and $H_1$ across different values of the probability of committing a type I error. Below and on the next page are shown two versions of the sample mean PDF graph from the previous example, identical in all aspects except for the significance level. In the first illustration, Figure 1.3, a level of $\alpha = 0.01$ has been chosen, namely, the probability of declaring false positives is very low; then the test will be very strict in declaring discoveries. An additional consequence is that the power of the test (the green area) is reduced because the ability to detect true signals is lowered by the strictness of the test. The second illustration, Figure 1.4, represents the same

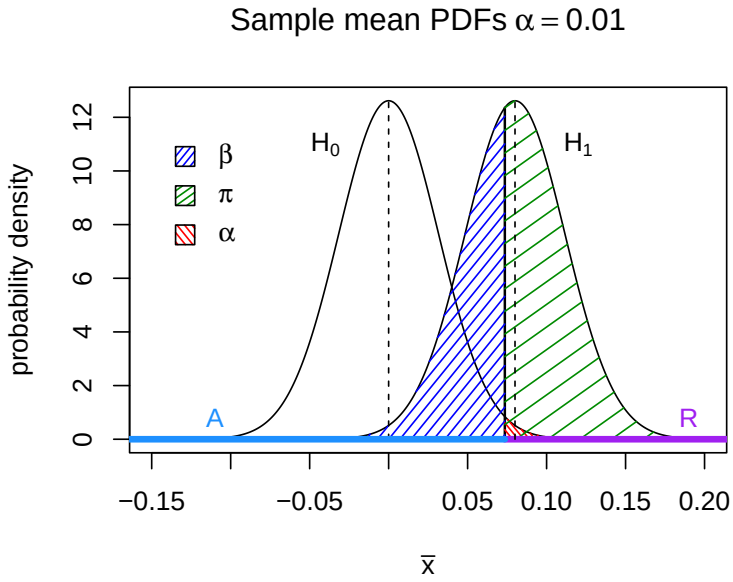


**Figure 1.3:** Probability density functions of the sample mean for a test with $\alpha = 0.01$.

graph for a less strict test. In the following case, a significance level of $\alpha = 0.2$ has been chosen; although it is rather unusual to select an $\alpha$ that high, this choice has been made to stress the difference with the more severe case. As can be seen, raising the probability of committing a type I error decreases the critical value, since it is proportional to $z_{1-\alpha}$; at the same time, $\pi$ is increased at the cost of making false discoveries. The probability of committing a type II error is reduced, since accepting a null hypothesis that is actually true is harder if the procedure is more likely to reject $H_0$. To conclude, another useful comparison to understand how tests with different significance levels behave consists of examining the power functions of the

previous example across different $\alpha$. In Figure 1.5, they have all been plotted in the same graph.

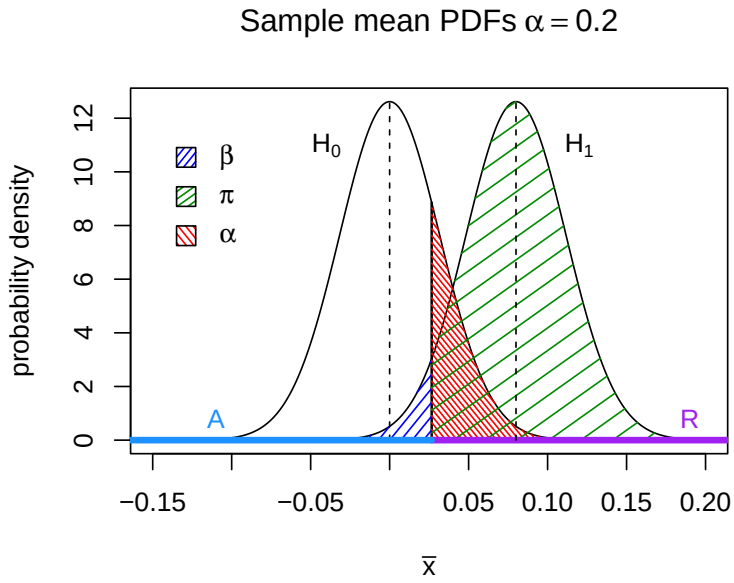


**Figure 1.4:** Probability density functions of the sample mean for a test with $\alpha = 0.2$.

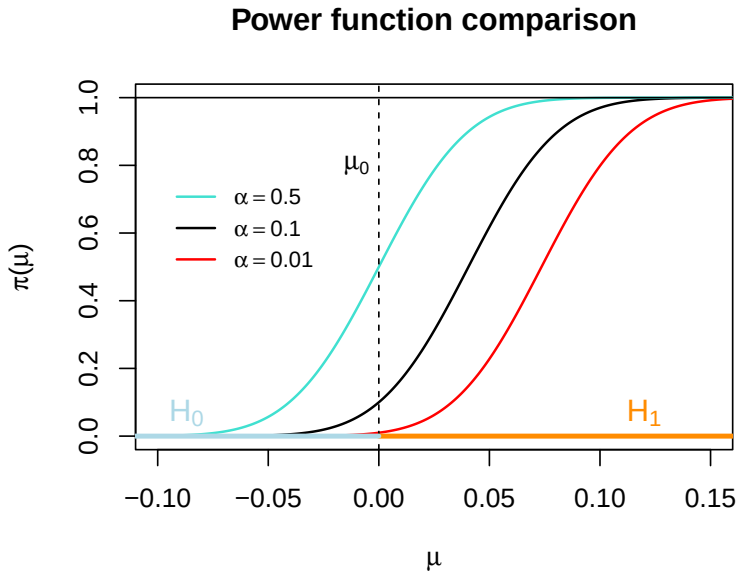


**Figure 1.5:** Power function comparison for the right-tailed Z-test of the previous example with different significance level values. From left to right, the strictness of the test in declaring positives increases.

It can be seen how decreasing the probability of declaring false positives shifts the power function to the right. Indeed, the significance level is represented by the intercept of the power function with the line $\mu = \mu_0$. Hence, shifting the function to the right means decreasing the intercept. As we expected, given the same fixed value of $\mu$ under the alternative hypothesis, the three tests have three distinct powers, which increase as $\alpha$ is decreased. In general, procedures with a power function shifted on the left will be looser in declaring discoveries; vice versa, power functions shifted to the right will determine more conservative choices. This behavior also persists in the average power functions of multiple testing procedures.

## 1.3 Multiple Hypothesis Testing

### 1.3.1 Introduction

Hypothesis testing could be interpreted as the mathematical framework underlying pretty much all scientific discoveries. In this sense, almost every breakthrough in science stems from some form of hypothesis testing, whether to validate a result or to guarantee its reproducibility. In this context, with the advent of high-dimensional data, the need to develop a theory of multiple hypothesis testing was essential to detect significant signals among many noises while controlling a global measure of error. This field of research, while being inherently a milestone of science, has reached huge popularity following the so-called Replication crisis, namely, the widespread overestimation of scientific discoveries in academic papers due, in part, to the application of single-hypothesis testing procedures to multiple contexts. More details about the reproducibility crisis are provided in (Ioannidis, 2005) from a medical perspective and (Gelman & Loken, 2014) from a statistical point of view. Over the years, different robust detection strategies have been proposed, and despite their differences, they maintain a level of similarity that allows for a meaningful comparison. Throughout chapters 2 and 3, we will introduce these strategies and compare their performances. We start by introducing the framework of union-intersection and intersection-union tests, which provides an effective link between single and multiple hypothesis testing.

### 1.3.2 Union-Intersection and Intersection-Union Tests

The following presentation is based on section 8.2.3 of Casella & Berger (2008). A first example of multiple testing methods includes the **Union-Intersection** and

**Intersection-Union** tests. The idea is that, in some cases, a complicated single null hypothesis can be expressed as a union or an intersection of simpler individual hypotheses. We first consider the union-intersection method that turns out to be useful when a complex null hypothesis is conveniently expressed as an intersection of simpler sets. The required structure for the global hypothesis $H_0$ is specified as follows:

$$H_0 : \theta \in \bigcap_{\gamma \in \Gamma} \Theta_\gamma. \tag{1.6}$$

In this case, the approach involves decomposing a difficult to describe single hypothesis into several simpler ones. Particularly, $\Gamma$ is an arbitrary index set that could be finite or infinite, usually we have $\Gamma = \{1, \ldots, m\}$, while $\Theta_\gamma$ are simpler sets for which hypothesis tests are available, namely, for each $\gamma$ we have:

$$H_{0\gamma} : \theta \in \Theta_\gamma \qquad \text{vs} \qquad H_{1\gamma} : \theta \in \Theta_\gamma^c. \tag{1.7}$$

Therefore, we have reduced the complexity of the problem by increasing the number of hypotheses being jointly tested and having simplified their structures. At this point, we can define the rejection region for the $\gamma^{\text{th}}$ test specified in (1.7) as follows

$$R_\gamma = \{y \in \mathcal{Y} : T_\gamma(y) \text{ suggest rejecting } H_{0\gamma}\}, \tag{1.8}$$

where $T_\gamma(y)$ is the test statistic computed on the sample $y$ for the $\gamma^{\text{th}}$ hypothesis. As a result, we can also define the rejection region for the union intersection test in (1.6), which is

$$R = \bigcup_{\gamma \in \Gamma} R_\gamma.$$

In this case, the rationale is rather simple because we will reject the complex hypothesis $H_0$ if any one of the hypotheses $H_{0\gamma}$ is rejected. Therefore, $H_0$ will be accepted only if all $H_{0\gamma}$ are accepted. Sometimes a simple expression for the rejection region of union-intersection test can be defined. More precisely, if we suppose to have a rejection region for each simpler hypothesis $H_{0\gamma}$ of the following kind

$$R_\gamma = \{y \in \mathcal{Y} : T_\gamma(y) > c\}, \tag{1.9}$$

then the global rejection region will be defined as:

$$R = \bigcup_{\gamma \in \Gamma} \{y \in \mathcal{Y} : T_\gamma(y) > c\} = \{y \in \mathcal{Y} : \sup_{\gamma \in \Gamma} T_\gamma(y) > c\}.$$

Hence, the global test statistic is $T(y) = \sup_{\gamma \in \Gamma} T_\gamma(y)$; intuitively, we need to check whether the most extreme test statistic exceeds the threshold, since rejecting even one individual hypothesis is sufficient to reject the global hypothesis. Therefore, we have understood how the union-intersection test provides a first example of how single and multiple testing are related to each other. The second case, as mentioned above, is the intersection-union method and it is detailed below.
The intersection-union approach is used when the global null hypothesis can be conveniently expressed as a union of simpler hypotheses, namely:

$$H_0 : \theta \in \bigcup_{\gamma \in \Gamma} \Theta_\gamma.$$

Then, after having defined the same multiple hypothesis structure as in (1.7), with the only difference of how minor hypotheses are aggregated to form the global $H_0$, and after having defined rejection regions $R_\gamma$ with the exact same form of (1.8), we can construct the global rejection region

$$R = \bigcap_{\gamma \in \Gamma} R_\gamma.$$

Thus, $H_0$ will be rejected if each of the $H_{0\gamma}$ is rejected as well. This behavior follows from the definition of the global rejection region as an intersection of simple $R_\gamma$. In particular, if the single rejection region has the same form as in (1.9), except for the sign that is $\geq$, then the global $R$ will be:

$$R = \bigcap_{\gamma \in \Gamma} \{y \in \mathcal{Y} : T_\gamma(y) \geq c\} = \{y \in \mathcal{Y} : \inf_{\gamma \in \Gamma} T_\gamma(y) \geq c\}, \tag{1.10}$$

in other words, if the smallest test statistic exceeds the threshold c, then it is guaranteed that all the others $T_\gamma$ will do the same and the global hypothesis will be rejected.
These two approaches have been introduced as a bridge between single and multiple testing frameworks, mainly to understand how they are inherently related to each other. However, in many cases, the goal is not to simplify a complex single hypothesis, but rather to test multiple hypotheses directly. In other words, our interest often lies in testing multiple hypotheses on their own without considering them as a tool to tackle a single more complicated $H_0$. For this reason, we now introduce some additional notation to present the problem of multiple testing more clearly and regardless of a global $H_0$.

### 1.3.3 Multiple testing

The core idea of multiple testing is to jointly perform many hypothesis tests, instead of a single one. Intuitively, we are interested in testing $m$ distinct, independent and easy to describe null hypotheses against their respective alternatives, more precisely, the $\gamma^{\text{th}}$ hypothesis test is:

$$H_{0\gamma} : \theta \in \Theta_{0\gamma} \qquad \text{vs} \qquad H_{1\gamma} : \theta \in \Theta_{0\gamma}^{c} \qquad \text{for} \quad \gamma = 1, \dots, m.$$

This is done by computing $m$ distinct test statistics $T_\gamma$ that once compared with a shared critical threshold, naturally determine either to reject or fail to reject $H_{0\gamma}$ for $\gamma = 1, \dots, m$. This could be done equivalently using p-values instead of the test statistics. More specifically, we assume to have a collection $\mathcal{H} = \{H_1, \dots, H_m\}$ of $m$ null hypotheses. Within this collection, an unknown number $m_0$ of null hypotheses are true, while the remaining $m_1 = m - m_0$ are false. Besides being unknown, both $m_0$ and $m_1$ are also unobservable. In this regard, a quantity that will turn out to be highly useful in Chapter 2 is the proportion of null hypothesis $\pi_0$ that is equal to $m_0/m$. The subset of true hypotheses is called $\mathcal{T} \subseteq \mathcal{H}$, whereas the subset of false hypotheses is $\mathcal{F} = \mathcal{H} \setminus \mathcal{T}$. Clearly, both these sets are unobservable and unknown, since we cannot know the true reality. Consequently, the goal of a multiple testing procedure is to choose a subset $\mathcal{R}$ of $\mathcal{H}$ composed by hypotheses to reject. If we have p-values $p_1, \dots, p_m$ for $H_1, \dots, H_m$, a natural way to define $\mathcal{R}$ is:

$$\mathcal{R} = \{H_\gamma : p_\gamma \leq c\}, \tag{1.11}$$

specifically, the set of rejected hypotheses consists of those $H_\gamma$ whose p-values are below a critical probability level $c$, indicating that the corresponding test statistics are notably distant from the null hypothesis. In an ideal situation, one would like to have the set of rejected hypotheses $\mathcal{R}$ to coincide as much as possible with the set of false null hypotheses $\mathcal{F}$. This because we would like to reject null hypotheses that are actually false.

### 1.3.4 Errors

In practice, the multiple testing procedure commits two types of errors. The first ones, in analogy with the single hypothesis framework, are known as type I errors and they represent the set of true null hypotheses that are rejected, namely, $\mathcal{R} \cap \mathcal{T}$. The second ones are the type II errors, which are accepted null hypotheses that were

actually false; they are denoted by $\mathcal{F} \setminus \mathcal{R}$. In other words, we are considering, within the set of actually false hypotheses, those that have not been rejected. Since scientific discoveries often result from rejecting null hypotheses, and new breakthroughs are built on previous ones, the scientific community generally considers making a false discovery more harmful than failing to detect a real one. This is because many false discoveries would undermine the whole structure of science, if not contained; conversely, failing to detect a true discovery could be unfortunate for a scientist's career but would not harm the whole scientific community. For this reason, type I errors, whether in single or multiple testing, are considered more problematic than type II errors. In multiple testing, in analogy with the one-hypothesis framework, it is possible to summarize correct and wrong decisions through a contingency table. Therefore, if we repeat a multiple test only once, we are able to define the following contingency table. In Table 1.3, the rows represent the decisions based on the results

**Table 1.3:** Multiple test contingency table

| | false null | true null | total |
|---|---|---|---|
| null rejected | TP | FP | $R$ |
| null accepted | FN | TN | $m - R$ |
| total | $m_0$ | $m - m_0$ | $m$ |

of the multiple test, rejecting or accepting each null hypothesis, while the columns indicate the true status of the $m$ null hypotheses. Therefore, in this case, we are jointly testing $m$ hypotheses, with a number of rejections $R = |\mathcal{R}|$. The only known quantities are the number of rejections $R$ and the number of hypotheses $m$ to be tested. $R$ is an observable random variable, while $m$ is a parameter that needs to be fixed in advance. All the remaining quantities are unobservable and unknown random variables, except for $m_0$, which is an unobservable and unknown, non-random parameter. Furthermore, TP is the number of true positives, namely, the number of null hypotheses correctly rejected, while FP represents the number of false positives, which is the number of null hypotheses rejected that were actually true. TP and FP sum up to R, which is the already mentioned total number of rejections. Instead, coming to the accepted null hypotheses, they are divided into false negatives FN, namely, the number of null hypotheses that have been accepted while being false, and true negatives TN, which is the number of accepted null hypotheses that were actually true; FN and TN add up to $m - R$. Moreover, FP is also known as the number of type I errors, while FN can be interpreted as the number of type II errors.

### 1.3.5 Family wise error rate

This section and the next one are based on chapter 15 of Efron & Hastie (2021). In multiple testing, it is not immediately clear which error measure to use when designing tests aimed at finding a balance between detecting true discoveries and avoiding false ones. This also happens because the significance level and the probability of committing a type II error are naturally defined only for a single hypothesis framework and not directly extended to multiple testing. For this reason, it has been introduced the so-called **family-wise error rate** or FWER whose definition is the following:

$$\text{FWER} = \mathbb{P}(\text{FP} > 0), \tag{1.12}$$

namely, the probability of making at least one false positive, or in other words, the probability that the collection of rejected hypotheses contains any type I error. As mentioned above, since FP is a random variable, it is meaningful to write the probability of the event $\text{FP} > 0$.
In addition to the intuitive definition in (1.12), assuming to test $m$ hypotheses each one at $\alpha$, the family-wise error rate can also be formulated as follows:

$$\text{FWER} = \mathbb{P}\left\{\bigcup_{\gamma \in \mathcal{T}} (p_\gamma \leq \alpha)\right\}. \tag{1.13}$$

This quantity represents the probability of the union of events corresponding to false positives. In probability theory, such a union is interpreted as the probability that at least one of these events occurs. For this reason, it represents the probability of committing at least one false discovery. More precisely, the individual events in the union correspond to false positives because we are considering only hypotheses $H_{0\gamma}$ with $\gamma \in \mathcal{T}$ that is the set of indices of true null hypotheses, and we are rejecting this subset of hypotheses based on p-values. Using p-values to reject hypotheses means considering $H_{0\gamma}$ and declaring a rejection if the corresponding p-value is less than $\alpha$. Therefore, we are effectively computing the probability of having at least one false positive, by testing each $H_{0\gamma}$ in $\mathcal{T}$ at level $\alpha$. However, especially in (1.12), the family-wise error rate appears to be defined in a top-down manner. As a consequence, one might wonder why do we need to define it in the first place and how do we obtain it. Actually, the family-wise error rate can be more clearly defined as a classical probability of committing a type I error for the global null hypothesis in a union-intersection test. To some extent, we could have an intuition of why it is

the case by looking at the definition (1.13). However, a more detailed explanation is needed to prove that FWER is just a significance level of a union-intersection test. In this regard, a bottom-up approach can be taken: first, define a collection of $m$ simpler hypotheses $H_{0\gamma}$, then aggregate them into a single global hypothesis $H_0$, by doing a proper intersection, and finally compute the usual probability of making a type I error for $H_0$. It turns out that the significance level associated with the resulting global hypothesis $H_0$ corresponds exactly to the definition of the family-wise error rate associated to the collection of simpler hypotheses $\{H_{01}, \dots, H_{0m}\}$; this holds true assuming a global hypothesis $H_0$ with $\Theta_0 \neq \varnothing$. To better explain this definition of FWER, we could consider the case of a multiple test composed of $m$ two-tailed Z-tests, each one testing the the expected value of a Gaussian population with known variance

$$H_{0\gamma} : \mu = \mu_0 \qquad \text{vs} \qquad H_{1\gamma} : \mu \neq \mu_0, \quad \gamma = 1, \dots, m$$

or, equivalently, using set notation, which is more appropriate in this context, we would have:

$$H_{0\gamma} : \mu \in \{\mu_0\} \qquad \text{vs} \qquad H_{1\gamma} : \mu \in \{\mu_0\}^c. \tag{1.14}$$

Consequently, the global hypotheses structure of this union-intersection test will be:

$$H_0 : \mu \in \bigcap_{\gamma \in \Gamma} \{\mu_0\} \qquad \text{vs} \qquad H_1 : \mu \in \left( \bigcap_{\gamma \in \Gamma} \{\mu_0\} \right)^c,$$

thus, if we solve the intersection, this hypothesis structure turns out to be exactly equal to (1.14), since the intersection of many identical singletons results in the singleton itself. consequently, we can assume a rejection region for the $\gamma^{\text{th}}$ test specified as follows

$$R_\gamma = \{y \in \mathcal{Y} : |T_\gamma(y)| > c\},$$

which is the natural way of defining a rejection region for a two-tailed normal test; equivalently, using the p-value and an arbitrarily chosen significance level $\alpha$, we would have:

$$R_\gamma = \{y \in \mathcal{Y} : p_\gamma \leq \alpha\}.$$

Therefore, the global rejection region for the null hypothesis $H_0$ of the union-intersection test will be:

$$R = \bigcup_{\gamma \in \Gamma} \{y \in \mathcal{Y} : |T_\gamma(y)| > c\} = \bigcup_{\gamma \in \Gamma} \{y \in \mathcal{Y} : p_\gamma \leq \alpha\}. \tag{1.15}$$

Hence, we are now able to compute a classic significance level for the global hypothesis $H_0$ and show how this quantity is exactly equal to FWER for hypotheses $H_{01}, \ldots, H_{0m}$.

$$\begin{aligned}
\alpha &= \sup_{\mu \in \Theta_0} \mathbb{P}(Y \in R) = \sup_{\mu \in \{\mu_0\}} \mathbb{P}(Y \in R) \\
&= \mathbb{P}(Y \in R \mid \mu = \mu_0) \\
&= \mathbb{P}(\bigcup_{\gamma \in \Gamma} \{y \in \mathcal{Y} : p_\gamma \leq \alpha\} \mid H_0 \text{ true}) \\
&= \mathbb{P}(\bigcup_{\gamma \in \Gamma} \{y \in \mathcal{Y} : p_\gamma \leq \alpha\} \mid H_{01}, \ldots, H_{0m} \text{ are true}) \\
&= \mathbb{P}(\bigcup_{\gamma \in \mathcal{T}} \{y \in \mathcal{Y} : p_\gamma \leq \alpha\}) \\
&= \text{FWER}.
\end{aligned}$$

The first equality follows from the definition of significance level. The second and third equalities result from the two-tailed structure of the tests used in the chosen example. The fourth step is obtained by explicitly expanding the rejection region of a union-intersection test within the probability measure $\mathbb{P}$. The fifth equality relies on the definition of the global null hypothesis $H_0$ as the intersection of simpler null hypotheses. Therefore, conditioning to $H_0$ is equivalent to conditioning to all individual null hypotheses $H_{0\gamma}$. Hence, computing the probability assuming that $H_0$ is true is the same as conditioning on the truth of all individual hypotheses. Finally, conditioning on the true nulls means computing the probability of the union of events corresponding to false discoveries only. As a result, this leads us to the definition of the family-wise error rate.

### 1.3.6 FWER Controlling procedures

However, in the context of multiple testing, our goal is not to accept or reject the global null hypothesis $H_0$, as we would have done in a union-intersection test. Instead, we aim to construct a multiple comparison procedure that allows us to assess each

individual hypothesis $H_{01}, \ldots, H_{0m}$ regardless of the global $H_0$. Therefore, we need to define multiple comparison procedures capable of determining a rejection region as specified in (1.11), while also controlling a global error measure. In this section, we introduce two different approaches to multiple testing that aim to control the family-wise error rate. In other words, when applied, these procedures ensure that the FWER is bounded by a predetermined significance level denoted by $\alpha$. Before introducing these controlling procedures, it is helpful to first consider the most intuitive, yet incorrect, approach to test m distinct hypotheses. This method could be called **Naive**, since it applies a single-hypothesis approach to multiple testing without caring about generalization or global error controlling. In practice, using this method, we test each $H_{0\gamma}$ at significance level $\alpha$, as a result, we end up with a FWER that is not bounded by $\alpha$. Algorithm 2 performs Multiple testing using the naive method:

**Algorithm 2:** Naive approach

1. fix an arbitrary level $\alpha \in (0, 1)$ (upper bound of FWER)
2. compute m distinct p-values one for each $H_{0\gamma}$: $p_1, p_2, \ldots, p_m$
3. reject all $H_{0\gamma}$ such that $p_\gamma \leq \alpha$.

It is straightforward to verify how this Naive procedure does not control the family-wise error rate:

$$\text{FWER} = \mathbb{P}\left\{\bigcup_{\gamma \in \mathcal{T}} (p_\gamma \leq \alpha)\right\} \leq \sum_{\gamma \in \mathcal{T}} \mathbb{P}\left\{p_\gamma \leq \alpha\right\} = m_0 \cdot \alpha.$$

The first equality stems from the definition of FWER (1.13), that is the probability of committing at least one false discovery. The second step originates from the finite case of Boole's Inequality. Ultimately, the third equality relies on the key assumption that the data are continuous, and therefore the test statistic is also continuous. Therefore, under the true null $p_\gamma \sim U(0, 1)$ and then $\mathbb{P}\left\{p_\gamma \leq \alpha\right\} = \int_0^\alpha dp_\gamma = \alpha$; consequently doing the sum of the term $\alpha$ over the true null indexes we obtain $m_0 \cdot \alpha$ that is $\geq \alpha$. Moreover, assuming independence among p-values, we would have:

$$\text{FWER} = \mathbb{P}\left\{\bigcup_{\gamma \in \mathcal{T}} (p_\gamma \leq \alpha)\right\} = \sum_{\gamma \in \mathcal{T}} \mathbb{P}\left\{p_\gamma \leq \alpha\right\} = m_0 \cdot \alpha.$$

This follows from the absence of the intersection terms among the sum of probabilities $\mathbb{P}\{p_\gamma \leq \alpha\}$ with $\gamma \in \mathcal{T}$. For this reason, both under independence and dependence among hypotheses, the naive approach is not guaranteed to control the family-wise error rate. Besides that, it is worth mentioning that in real-world problems when $m$ is particularly high, and independence is not satisfied, the number of true nulls $m_0$ is approximately equal to $m$; as a result, the FWER is bounded by a quantity that is $\geq 1$, which clearly means that FWER is not bounded at all, since being a probability implies that FWER is by definition $\leq 1$. Even more evident is the case when p-values are independent, which leads to FWER $= m_0 \cdot \alpha$.
At this point, we introduce the first FWER controlling procedure, known as **Bonferroni approach**. Intuitively, the Bonferroni method is just a correction of the naive algorithm used to test $m$ distinct hypotheses. The pseudocode for this FWER controlling procedure is defined in Algorithm 3:

**Algorithm 3:** Bonferroni approach

1. fix an arbitrary level $\alpha \in (0, 1)$ (upper bound of FWER)
2. compute m distinct p-values one for each $H_{0\gamma}$: $p_1, p_2, \ldots, p_m$
3. reject all $H_{0\gamma}$ such that $p_\gamma \leq \dfrac{\alpha}{m}$.

In this way, by testing each hypothesis $H_{0\gamma}$ at $\alpha/m$ instead of $\alpha$, we are guaranteed to control FWER with $\alpha$. The reason why this is true is provided below:

$$\text{FWER} \leq \sum_{\gamma \in \mathcal{T}} \mathbb{P}\left\{p_\gamma \leq \frac{\alpha}{m}\right\} = \frac{m_0 \cdot \alpha}{m}.$$

These steps are identical to the ones used to show that FWER was not controlled using the Naive approach. With the only difference that by isolating $1/m$, we are now able to show that FWER is bounded by $\alpha$ since $\pi_0 = m_0/m$ is by definition less than 1, indeed $m_0$ is always less than $m$. The fact that under the Bonferroni approach each $H_{0\gamma}$ is rejected only if $p_\gamma \leq \alpha/m$ will inevitably lead to a lower number of discoveries and to detecting only extremely significant effects. In this context, it is evident the presence of a trade-off between the conservativeness of the Bonferroni bound, which controls type I errors, and its ability to detect signals, which is reduced by its strictness. At this point, one might wonder if there exists a procedure that still controls the FWER but is slightly more generous in declaring discoveries; the

answer is Holm's approach introduced in (Holm, 1979). The **Holm's method** is the second and the last FWER controlling procedure presented in this chapter. The steps of this approach are described in Algorithm 4.

**Algorithm 4:** Holm's approach

1. fix an arbitrary level $\alpha \in (0, 1)$ (upper bound of FWER)
2. compute m distinct p-values one for each $H_{0\gamma}$: $p_1, p_2, \ldots, p_m$
3. order (and relabel) the m p-values in ascending order such that: $p_1 \leq p_2 \leq \cdots \leq p_m$
4. find the index $\gamma_0 = \min \left\{ \gamma \in \{1, \ldots, m\} : p_\gamma > \dfrac{\alpha}{m - \gamma + 1} \right\}$
5. reject all $H_{0\gamma}$ with $\gamma < \gamma_0$ and fail to reject otherwise.

The first two steps are identical to other common approaches, while from the third step on, Holm's method differs from previously exposed procedures. Specifically, the third step involves ordering the p-values and relabeling them accordingly. Successively in the fourth step, the goal is to find a threshold, denoted as $\gamma_0$, which corresponds to the smallest index for which the associated p-value exceeds the adjusted critical value $\alpha/(m - \gamma + 1)$. Finally, all null hypotheses $H_{0\gamma}$ corresponding to indices before this threshold are rejected, while the remaining ones are accepted. It can also be shown that Holm's procedure controls the family-wise error rate. The following is the proof based on the one provided in Holm (1979).
We first assume to reject at least one true null hypothesis and have reordered hypotheses in ascending order by their p-values. Then, we consider the minimum index associated with a rejected true null, and we call it $l$:

$$l = \min\{\gamma \in \mathcal{T} : H_{0\gamma} \text{ is rejected}\}.$$

Consequently, if $H_{0l}$ is the first true null to be rejected, we have $l - 1$ false nulls that have been correctly rejected before $H_{0l}$. Since the total number of false nulls is $m_1 = m - m_0$, we can claim that

$$l - 1 \leq m - m_0,$$

because the number of correct rejections before $l$ is necessarily smaller than the number of total false nulls. Thus, by isolating $m_0$ on the right and taking the inverse,

we obtain

$$\frac{1}{m-l+1} \leq \frac{1}{m_0}. \tag{1.16}$$

At this point, we recognize that using the Holm's step-down procedure means rejecting those hypotheses $H_{0_\gamma}$ such that:

$$p_\gamma < \frac{\alpha}{m-\gamma_0+1},$$

where $\gamma_0$ has been properly defined in Algorithm 4. This also means that for all indices $\gamma < \gamma_0$ it holds true the following claim:

$$p_\gamma < \frac{\alpha}{m-\gamma+1}.$$

Consequently, combining this last claim with (1.16), we obtain the following for all rejected hypotheses

$$p_\gamma < \frac{\alpha}{m_0}. \tag{1.17}$$

Hence, the definition of FWER based on this threshold is equivalent to computing the FWER of Holm's procedure. Since (1.17) is especially true for rejected true nulls, we have

$$\text{FWER} = \mathbb{P}\left(\bigcup_{\gamma\in\mathcal{T}}\left\{p_\gamma < \frac{\alpha}{m_0}\right\}\right) \leq \sum_{\gamma\in\mathcal{T}}\mathbb{P}\left\{p_\gamma < \frac{\alpha}{m_0}\right\} = m_o \cdot \frac{\alpha}{m_0} = \alpha.$$

The first equality derives from the definition of family-wise error rate and having used claim (1.17). The second step is given by Boole's inequality; the last one assumes continuous test statistics for the $m$ tests, and therefore, continuous uniform distributions in $[0,1]$ for the p-values. Consequently, by computing the probability of the event $p_\gamma < \alpha/m_0$, we obtain that FWER is bounded by $\alpha$.

Without going into the details, it is worth mentioning that multiple testing can also be approached from another perspective, by fixing a significance threshold $\alpha$ and adjusting the p-values according to a multiple comparison procedure. If the adjustment is done appropriately, this is exactly equivalent to computing raw p-values and comparing each of them to a threshold that is appropriately defined based on the chosen method and the selected $\alpha$. In this regard, the `stats` package of the `R` software provides the function `p.adjust()` that can perform such p-value adjustments directly. In the last section of this chapter, we have seen two examples of family-wise error rate

controlling procedures; however, in the scientific literature, other global measures of error have been proposed with their own controlling procedures. More sophisticated approaches will be presented in the next chapter.

# Chapter 2

# False Discovery Rate Controlling Procedures

The two main references for this chapter are: chapter 15 of Efron & Hastie (2021) for the part on false discovery rate and Benjamini-Hochberg procedure, and the article Barber & Candès (2015a) for the Knockoff procedure.

## 2.1 False Discovery Rate

In the literature of multiple testing, there is extensive debate over which global error measure to control in order to define a good multiple comparison procedure. One influential alternative to the family-wise error rate is the **False Discovery Rate** (FDR). In order to define the FDR we first need to introduce the **False Discovery Proportion** (FDP).
Using the notation in Table 1.3, the false discovery proportion is defined as:

$$\text{FDP} = \frac{\text{FP}}{\max(\text{R}, 1)} = \begin{cases} \text{FP}/\text{R} & \text{if R} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

In other words, the false discovery proportion is a random variable equal to the proportion of false discoveries among all discoveries. If no discoveries are made, to avoid division by zero, the denominator is set equal to 1; consequently, yielding to $\text{FDP} = 0$ in such cases.
The false discovery rate is defined as follows:

$$\text{FDR} = \mathbb{E}(\text{FDP}), \tag{2.1}$$

namely, it is the expected proportion of false discoveries among all discoveries. Here, the expectation is taken with respect to the underlying data. Indeed, once both

the testing procedure and the system of hypotheses are fixed, the false discovery proportion becomes a random variable as the data vary, so it is natural to consider its expectation. As we defined FWER controlling procedures in Chapter 2, we can now introduce FDR controlling procedures to test multiple hypotheses. More precisely, a testing procedure is said to control FDR if the resulting rejection set $\mathcal{R}$ is chosen in such a way that FDR $\leq \alpha$. It is worth noting that the family-wise error rate and the false discovery rate are closely related, and so are the methods used to control them. Indeed, we can show that:

$$\text{FDP} = \frac{\text{FP}}{\max(\text{R}, 1)} \leq \begin{cases} 1 & \text{if FP} > 0, \\ 0 & \text{otherwise.} \end{cases} = \mathbb{1}(\text{FP} > 0).$$

This is just a consequence of $0 \leq \text{FDP} \leq 1$; while $\mathbb{1}$ represents the indicator function. As a result, if we take the expectation on both sides, we obtain:

$$\begin{aligned} &\mathbb{E}(\text{FDP}) \leq \mathbb{E}(\mathbb{1}(\text{FP} > 0)) \\ &\text{FDR} \leq \mathbb{P}(\text{FP} > 0) \\ &\text{FDR} \leq \text{FWER}. \end{aligned}$$

The second inequality stems from observing that $\mathbb{1}(\text{FP} > 0)$ can be considered a Bernoulli random variable with a success probability equal to its expected value, that is $\mathbb{P}(\text{FP} > 0)$. Therefore, we have proved that FWER control implies FDR control. Hence, all the FWER controlling procedures also control the FDR, while the reverse is not true. Intuitively, this connection is supported by the fact that FDR and FWER are proper generalizations of the concept of type I error to multiple hypotheses. In fact, in the case where the number of hypotheses to be tested is $m = 1$, the two error rates are identical and equal to the type I error rate.
The following sections present two false discovery rate controlling procedures, with particular emphasis on the Knockoff filter, which is the central topic of this thesis.

## 2.2 Benjamini-Hochberg approach

In High-dimensional contexts where the number of true alternatives $m_1$ is very low and the number of true nulls $m_0$ is approximately equal to the total number of hypotheses tested $m$, the conservative bounds yielded by FWER controlling procedures risk producing always approximately 0 discoveries. For this reason, in 1995 Benjamini and

Hochberg introduced the first FDR controlling procedure (Benjamini & Hochberg, 1995). This paradigm shift has been highly influential both inside and outside the field of mathematical statistics, primarily for its power and simplicity. The framework is quite similar to the one used in previously discussed FWER controlling procedures, with a few key differences. First of all, the Benjamini-Hochberg (BHq) procedure is guaranteed to control the FDR but not the FWER. Secondly, the rejection threshold is clearly defined in a different way. Finally, we require independence among hypotheses, a request not made for FWER controlling procedures, which is crucial to guarantee FDR control for BHq. In this regard, the independence request is highly unrealistic and not satisfied in many real cases, due to this limit other procedures have been proposed, such as modified versions of BHq and the Knockoffs. In Algorithm 5 is detailed the **Benjamini-Hochberg** approach:

**Algorithm 5:** Benjamini-Hochberg approach

1. Fix an arbitrary level $q \in (0, 1)$
2. Compute m distinct p-values one for each $H_{0\gamma}$
3. Order the m p-values in ascending order such that: $p_1 \leq p_2 \leq \cdots \leq p_m$
4. If $p_\gamma \leq \dfrac{\gamma}{m} \cdot q \quad \forall\, \gamma = 1, \ldots, m$ reject nothing
5. Otherwise, find the index $\gamma_{\max} = \max\left\{\gamma \in \{1, \ldots, m\} : p_\gamma \leq \dfrac{\gamma}{m} \cdot q\right\}$
6. Reject all $H_{0\gamma}$ with $\gamma \leq \gamma_{\max}$ and fail to reject otherwise.

The logic is very similar to Holm's method, but with two key differences: firstly, we use a different rejection threshold that is based on $p_{\gamma_{\max}}$, secondly, the BHq procedure controls the FDR rather than the FWER. It is also interesting to note that, unlike the Bonferroni approach, which tests each hypothesis at level $\alpha/m$, the BHq procedure progressively assigns more weight to the hypotheses through the parameter $\gamma$. This helps in counterbalancing the conservativeness of testing all $H_{0\gamma}$ at the uniform level $\alpha/m$. The goal, then, is to identify the largest p-value that remains below its corresponding threshold determined by $\gamma$, and subsequently test each hypothesis at the level $p_{\gamma_{\max}}$.

Defining $\mathcal{D}_q$ as the decision rule described above to either reject or fail to reject hypotheses $H_{0\gamma}$, one could denote the false discovery rate yielded by this procedure as $\text{FDR}(\mathcal{D}_q)$. The following theorem states how BHq is guaranteed to control the $\text{FDR}(\mathcal{D}_q)$ at level $q$. The proof is based on Solari (2023)

**Theorem 2.1** (Benjamini & Hochberg (1995) FDR Control)**.** *if the p-values corresponding to valid null hypotheses are independent of each other; then*

$$FDR(\mathcal{D}_q) = \pi_0 \cdot q \leq q \qquad where \quad \pi_0 = \frac{m_0}{m}. \tag{2.2}$$

*Proof.* When $m_0 = 0$ the conclusion is obvious, FDR is exactly 0 since there are no true nulls. Hence, it is obviously controlled at level $q$ since it is exactly 0. Therefore, let us assume $m_0 \geq 1$.
For each $\gamma \in \mathcal{T}$ with $\mathcal{T} = \{\gamma \in \{1, \dots, m\} : H_{0\gamma} \text{ is actually true}\}$, define $V_\gamma$ such that:

$$V_\gamma = \mathbb{1}\{H_{0\gamma} \text{ rejected}\} = \begin{cases} 1 & \text{if } H_{0\gamma} \text{ is rejected} \\ 0 & \text{if } H_{0\gamma} \text{ is not rejected.} \end{cases}$$

Namely, $V_\gamma$ is a random variable that takes values in $\{0, 1\}$ and signals when a rejection is a false positive ($V_\gamma = 1$) or a true negative ($V_\gamma = 0$). The false discovery proportion could then be expressed as follows:

$$\text{FDP} = \sum_{\gamma \in \mathcal{T}} \frac{V_\gamma}{R \vee 1} \qquad \text{where} \quad R \vee 1 = \max\{R, 1\}.$$

Hence, the numerator counts only false discoveries, while the denominator counts the maximum between the total number of discoveries $R$ and 1, this is done to avoid division by 0 if there are no positives. At this point, we could make the following statement (it will be proved later):

$$\mathbb{E}\left(\frac{V_\gamma}{R \vee 1}\right) = \frac{q}{m} \quad \text{with} \quad \gamma \in \mathcal{T}. \tag{2.3}$$

Based on which it is possible to write that:

$$\text{FDR} = \mathbb{E}(\text{FDP}) = \sum_{\gamma \in \mathcal{T}} \mathbb{E}\left(\frac{V_\gamma}{R \vee 1}\right) = \sum_{\gamma \in \mathcal{T}} \frac{q}{m} = \pi_0 \cdot q, \tag{2.4}$$

where the first equality stems from the definition of FDR, the second equality is given by the linearity of the expected value $\mathbb{E}(\cdot)$ and the third step is a consequence of the claim (2.3) yet to be proved. The last step derives from adding $q/m$ $m_0$ times with $m_0$ the cardinality of $\mathcal{T}$. Clearly, (2.4) is the core result of the theorem, however, to conclude the proof it remains to show that the statement (2.3) is true. Let's say that there are $R = k$ rejections using BHq, then $\gamma_{max} = k$ and each $H_{0\gamma}$

is rejected if and only if $p_\gamma \leq (k \cdot q)/m$ ; Therefore using the notation introduced before we have that:

$$V_\gamma = \mathbb{1}\{H_{0\gamma} \text{ rejected}\} = \mathbb{1}\left\{p_\gamma \leq \frac{k}{m} \cdot q\right\}. \tag{2.5}$$

Suppose now that $p_\gamma \leq (k \cdot q)/m$, hence $H_{0\gamma}$ is rejected. Let us take $p_\gamma$ and set its value to 0, and denote the new number of total rejections by $R(p_\gamma \downarrow 0)$. This new number of rejections is exactly k because we have only reordered the first k p-values all of which remain below the threshold $(k \cdot q)/m$. On the other hand, if $p_\gamma \leq (k \cdot q)/m$ we do not reject $H_{0\gamma}$ and so $V_\gamma = 0$. As a consequence we have:

$$V_\gamma \cdot \mathbb{1}\{R = k\} = V_\gamma \mathbb{1}\{R(p_\gamma \downarrow 0) = k\}. \tag{2.6}$$

This means that we are considering $V_\gamma$ conditioned to the case in which we have observed $R = k$ rejections. At this point, in order to prove (2.3), we need the following result:

$$\frac{V_\gamma}{R \vee 1} = \sum_{k=1}^{m} \frac{V_\gamma}{k} \mathbb{1}\{R = k\}. \tag{2.7}$$

(2.7) holds because when a specific number of rejections $R = k$ is made, given the presence of indicator functions as weights of the linear combination on the right, all indices different from $k$ are sent to 0 and the realization of the random variable is exactly $R/k$. Combining the observation above and taking the expectation with respect of all p-values except for $p_\gamma$, and defining the sigma-algebra of p-values $\mathcal{F}_j = \{p_1, \ldots, p_{\gamma-1}, p_{\gamma+1}, \ldots, p_m\}$, we obtain:

$$\begin{aligned}
\mathbb{E}\left(\frac{V_\gamma}{R \vee 1} \mid \mathcal{F}_\gamma\right) &= \mathbb{E}\left(\sum_{k=1}^{m} \frac{V_\gamma}{k} \mathbb{1}\{R = k\} \mid \mathcal{F}_\gamma\right) \\
&= \sum_{k=1}^{m} \mathbb{E}\left(\frac{V_\gamma}{k} \mathbb{1}\{R = k\} \mid \mathcal{F}_\gamma\right) = \sum_{k=1}^{m} \frac{\mathbb{E}(V_\gamma \mathbb{1}\{R = k\} | \mathcal{F}_\gamma)}{k} \\
&= \sum_{k=1}^{m} \frac{\mathbb{E}(\mathbb{1}\{p_\gamma \leq \frac{k}{m} \cdot q\} \mathbb{1}\{R(p_\gamma \downarrow 0) = k\} | \mathcal{F}_\gamma)}{k} \\
&= \sum_{k=1}^{m} \frac{(qk)/m \mathbb{1}\{R(p_\gamma \downarrow 0) = k\}}{k} = \frac{q}{m} \sum_{k=1}^{m} \mathbb{1}\{R(p_\gamma \downarrow 0) = k\} \\
&= \frac{q}{m}
\end{aligned} \tag{2.8}$$

The first equality follows from having replaced (2.7) within the expected value. The second equality is a consequence of the linearity of the expectation while the third one follows from the number of rejections $k$ being deterministic. Next, the fourth step stems from replacing $V_\gamma$ and $\mathbb{1}(R = k)$ respectively with the expressions in (2.5) and the indicator function in (2.6). The fifth equality derives from the p-values being independent and uniformly distributed and $\mathbb{1}\{R(p_\gamma \downarrow 0) = k\}$ being deterministic. Finally, by recognizing that the sum of indicator functions $\mathbb{1}\{R(p_\gamma \downarrow 0) = k\}$ is 1 since we can only have a fixed number of rejections, we obtain the final result.
To conclude the proof, we apply the tower property, which allows us to explicitly express the FDR as follows:

$$\text{FDR} = \sum_{\gamma \in \mathcal{T}} \mathbb{E}\left(\frac{V_\gamma}{R \vee 1}\right) = \sum_{\gamma \in \mathcal{T}} \mathbb{E}\left[\mathbb{E}\left(\frac{V_\gamma}{R \vee 1} \mid \mathcal{F}_\gamma\right)\right] = \sum_{\gamma \in \mathcal{T}} \frac{q}{m} = \pi_0 \cdot q$$

The first equality is the definition of FDR given in (2.4). The second equality is a consequence of the tower rule, whose statement is: $\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X \mid Y]]$. The third equality is given by replacing the inner expected value with the result proved in (2.3), and finally, by applying the expectation on a constant, we obtain the sum of $q/m$ $m_0$ times, which is $\pi_0 \cdot q$. □

## 2.3 Variable Selection

### 2.3.1 The two cultures

Performing variable selection is probably one of the most crucial steps in constructing an appropriate regression model. As with many foundational statistical concepts, the process of model building and variable selection has been the subject of extensive debate. In this context, the highly influential paper by Breiman (2001) highlights two opposing statistical cultures that have developed over the years. The first culture, emerging from the high-dimensional data setting and focused on prediction, prioritizes predictive accuracy over significance and interpretability, as the primary criterion for model evaluation. However, this approach often lacks inferential guarantees regarding the significance of the estimated parameters and the selected variables. In contrast, the second culture, often referred to as the classical or "old-fashioned" approach, follows more thoroughly traditional inferential principles. Consequently, this second perspective emphasizes significance, robustness, and interpretability over predictive accuracy. In this sense, multiple testing can be considered as an "old-fashioned"

approach to perform variable selection since it is rooted in classical inference and can have the purpose of building robust regression models. Indeed, the multiple testing framework introduced in Section (1.3.3) is particularly useful for assessing the significance of coefficients in a linear regression model. While in the classical setting we were interested in testing hypotheses about parameters of the data-generating model, our interest here lies in conducting multiple hypothesis tests on the regression coefficients $\boldsymbol{\beta} = (\beta_1, \dots, \beta_p)$ of the linear regression model applied to $n$ statistical units and $p$ explanatory variables:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} \ + \ \mathbf{z}, \tag{2.9}$$

in this case we are assuming $\mathbf{z} \sim N(\mathbf{0}, \sigma^2\mathbf{I})$ to be a Gaussian n-dimensional noise, $\mathbf{X} \in \mathbb{R}^{n \times p}$ a known and deterministic design matrix with $n$ rows and $p$ columns and $\mathbf{y} \in \mathbb{R}^n$ a vector of responses. While a linear regression model is often specified for a single statistical unit, in (2.9) it is expressed for the entire sample using matrix notation. Specifically, the response variable $Y$ is a univariate random variable assumed to follow a Gaussian distribution, while $\mathbf{y}$ denotes its realizations across $n$ observations. As already mentioned, our interest lies in testing regression coefficients not per se, but to perform variable selection supported by a robust inferential procedure. Namely, for each $j \in \{1, \dots, p\}$, we select the variable $X_j$ if the test statistic suggests rejecting the corresponding null hypotheses $H_{0j} : \beta_j = 0$.
On the other side, there exist techniques such as stepwise regression, best subset selection or cross validation, to name a few, that can select variables based on the overall performance of the model in terms of bias-variance tradeoff; In doing so, these procedures often do not care about the significance of the variables selected. In this regard, the Lasso can be seen as a compromise between these two cultures because, while it does not provide inferential guarantees about the significance of the selected variables, it promotes interpretability by creating sparse solutions. The following is a brief description of the Lasso.

### 2.3.2 The Lasso

The Least Absolute Shrinkage and Selection Operator (Lasso) is a penalized regression technique that promotes sparse estimates for the regression coefficients $\boldsymbol{\beta}$ in a linear model. For this reason, it turns out to be highly useful in sparse regression problems where the number of covariates is high and only a few variables are truly relevant. In practice, the Lasso differs from ordinary least squares since it leads to sparse

solutions, namely, it estimates many $\beta_j$ with $\hat{\beta}_j = 0$; in this way, it performs both shrinkage and variable selection. Particularly, the shrinkage effect consists in the reduction in magnitude of the regression coefficients aimed at satisfying the constraint that guarantees sparsity in $\boldsymbol{\beta}$. On the other hand, variable selection is a consequence of estimating many $\beta_j$'s with 0, meaning that the respective $X_j$'s will not appear in the final model. Hence, in the context of a linear model (2.9) which is assumed to be sparse, to find the regression coefficients yielded by the Lasso $\hat{\boldsymbol{\beta}}_{\text{Lasso}}$, we need to find the minimizer of the following constrained optimization problem:

$$\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \qquad \text{subject to} \quad \|\boldsymbol{\beta}\|_1 \leq t, \tag{2.10}$$

namely, we want to find the vector of regression coefficients $\hat{\boldsymbol{\beta}}$ that minimizes the sum of squared residuals, subject to the constraint that the sum of the absolute values of the components of $\boldsymbol{\beta}$ is bounded by a threshold $t$. Clearly, decreasing the value of $t$ will tighten the constraint, thus increasing the amount of shrinkage on the regression coefficients. Conversely, raising $t$ will loosen the constraint and lead towards an ordinary least squares estimate. Often, this minimization problem is expressed in its more compact Lagrangian form specified as follows:

$$\min_{\boldsymbol{\beta}} \{\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1\}, \tag{2.11}$$

where $\lambda$ is the Lagrangian multiplier, while the whole expression is just a compact form to incorporate the constraint within the minimization problem. Consequently, $\lambda$ plays the opposite role of $t$, as it represents the strength of the penalty applied to the regression and is proportional to the amount of shrinkage obtained. As a result, if $\lambda = 0$, we end up having an ordinary least squares (OLS) estimate. Conversely, as $\lambda$ increases, the OLS estimates are progressively shrunk; in other words, $\hat{\boldsymbol{\beta}}_{\text{lasso}}$ will be increasingly composed of zeros as the penalty parameter is raised. Unfortunately, there is no closed-form expression for the Lasso solution; therefore, numerical methods such as Least Angle Regression or Coordinate Descent must be used to approximate it. It is worth noting that any monotonic transformation of the target function leaves the minimizer $\hat{\boldsymbol{\beta}}_{\text{lasso}}$ unchanged. At the same time, certain transformations can ease the minimization process. For this reason, it is common practice not to minimize the expression in (2.11) directly, but rather a scaled version obtained by multiplying it by a factor such as $1/2n$ or $1/2$. Thus, the minimizer of the Lasso regression could

be defined as follows:

$$\hat{\boldsymbol{\beta}}_{\text{lasso}}(\lambda) = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \left\{ \frac{1}{2}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{p}|\beta_j| \right\}; \tag{2.12}$$

in this case, both the $L^2$ norm, representing the residual sum of squares, and the $L^1$ norm, representing the lasso constraint, have been expanded to ease readability. Moreover, as can be seen, the vector of estimated coefficients is a function of the complexity parameter $\lambda$; since, as explained earlier, changing the penalty modifies the final Lasso estimate of the coefficients. Furthermore, a concept that lies at the heart of the Lasso is the so-called Lasso path.

The Lasso path can be intuitively defined as the graphical representation of the evolution of regression coefficients $\{\beta_1, \ldots, \beta_p\}$ as functions of the penalization parameter $\lambda$. It is usually read from left to right, focusing on the evolution of $\hat{\boldsymbol{\beta}}_{\text{lasso}}(\lambda)$ as $\lambda$ increases; in this case, we start with the OLS estimates, which are progressively shrunk toward zero. Conversely, it can also be read in the opposite direction if we are interested in assessing how the regression coefficients behave as the constraint is progressively loosened. Therefore, if we read the Lasso path, in the first way, from left to right, looking at the evolution of $\hat{\boldsymbol{\beta}}_{\text{lasso}}$ from $\hat{\boldsymbol{\beta}}_{\text{ols}}$ to $\underline{0}$, we observe that the order in which the variables are shrunk - more precisely, the order in which their respective regression coefficients are shrunk to zero - is a proxy for each variable's importance in the linear model. For example, in an orthogonal design, the solution $\hat{\boldsymbol{\beta}}_{\text{lasso}}$ is exactly a function of $\lambda$, making the complexity parameter more than just a proxy for variable importance. This is based on the assumption that variables that shrink to zero at low values of $\lambda$ are less likely to be informative, whereas those that persist until larger values of $\lambda$ are likely to be more influential. In other words, using an analogy, we can think of the complexity parameter as time, and define the exit time for each variable as the value of $\lambda$ at which the corresponding $\beta_j$ reaches zero. Intuitively, the higher the exit time, the more important the variable; this is because even under a strong penalization, the model continues to estimate a nonzero coefficient for that variable, suggesting that its contribution to the model outweighs the penalty. In this sense, since we are constrained by the $L^1$ norm, retaining a variable in the model comes at the expense of others. Therefore, to justify this cost, the variable being kept must be important; otherwise, we would have chosen another one.

### 2.3.3 Limitations of standard procedures

As we have already mentioned, in a response-covariate setting, the theory of multiple testing can be seen as an effective variable selection tool that not only identifies relevant variables but also provides inferential guarantees about their significance. In this context, to overcome the limitations of existing approaches - such as the lack of inferential guarantees in the Lasso, the conservativeness of methods like Bonferroni and Holm, and the strong independence assumption underlying the Benjamini-Hochberg (BHq) procedure, which is typically violated in linear models due to non-orthogonal covariates - Barber & Candès (2015a) introduced the Knockoff filter. It is worth observing that if the covariates are assumed to be normally distributed, an orthogonal design matrix, which means zero correlation among variables, implies that the features are independent, and so are their p-values. This follows from the fact that linear dependence is the only form of dependence among Gaussian random variables. On the contrary, if no assumptions are made on the distribution of the features, then uncorrelated (orthogonal) variables are not necessarily independent, and their p-values may still be dependent due to the presence of non-linear relationships. For this reason, the BHq independence assumption becomes even harder to verify. Moreover, since the primary aim of the knockoff method is to relax the assumption of independence among hypotheses while still controlling the FDR, this new approach is not supposed to perform better than BHq in terms of power under the assumption of independence. Furthermore, knockoffs do not use p-values to perform variable selection, which is profoundly advantageous since, many times, we do not know the distribution of test statistics. Another interesting property is that knockoffs can work with a broad class of test statistics, well beyond the classical pivotal quantities; this provides additional flexibility to the method. Ultimately, since the Knockoff approach, unlike Bonferroni or Holm, does not control the FWER, it will have a higher ability to detect true signals. In the following section it is detailed the whole construction of the Knockoff filter and its FDR controlling property.

## 2.4 Knockoff filter

### 2.4.1 Introduction

The Knockoff method, introduced in Barber & Candès (2015a), enables variable selection through multiple testing without requiring independence among hypotheses. It also provides inferential guarantees on the selected variables, and it effectively

controls the false discovery rate at an arbitrary level $q$; moreover, by not controlling FWER, the knockoff approach is likely to be more powerful than previously described FWER controlling procedures such as Bonferroni and Holm that also do not require independence among the hypotheses. The intuition behind this method is to generate an artificial copy of the original design matrix, mimicking in this way a simulation framework in which we actually know what variables are relevant and which are not. Consequently, we can calibrate a data-dependent threshold $T$ used to select variables via multiple testing. In practice, calibrating the threshold means defining $T$ as the most permissive threshold possible while controlling an estimate of the false discovery proportion. Computing the estimate of FDP is feasible only through the pseudo-simulation context in which we are operating, thanks to the knockoffs. The overall context in which this method is applied is a linear regression setting as defined in (2.9), with the caveat that $n$ has to be larger than or equal to $2p$. This requirement is crucial to properly define the matrix of knockoffs. However, it is worth noting that in modern applications we often deal with high-dimensional data, where the ratio $p/n$ is large and $n \geq 2p$ is not necessarily satisfied. For this reason, there exist more sophisticated versions of knockoffs that deal with this problem; however, in this work, we will only consider the case where $n \geq 2p$. As mentioned above, the Knockoff filter is an effective variable selection tool that performs multiple testing on the regression coefficients of a linear model. In this context, given a total of $p$ covariates and denoting $\hat{S} \subset \{1, \dots, p\}$ as the set of indices of chosen variables, we can express the FDR associated with the selection procedure as follows:

$$\text{FDR} = \mathbb{E}\left[\frac{\#\{j : \beta_j = 0 \text{ and } j \in \hat{S}\}}{\#\{j : j \in \hat{S}\} \ \vee \ 1}\right]. \tag{2.13}$$

This is the same definition as in (2.1) adapted to a variable selection context, where the numerator represents the number of variables declared significant that do not appear in the true model, while the denominator represents the total number of variables declared significant. Besides that, since we are testing the nullity of regression coefficients, the subscript of the hypotheses tested will change from $\gamma$ to $j$, because $j$ is commonly used to index covariates. Additionally, the denominator uses the notation $a \vee b = \max(a, b)$ in order to set FDR $= 0$ when zero variables are selected. Given the analogy between rejecting $H_{0j} : \beta_j = 0$ and selecting the variable $X_j$, this terminology will be used interchangeably in this section.

Coming to the controlling property, as mentioned in the introduction to the false

discovery rate, since the Knockoff filter is an FDR controlling procedure, it is guaranteed to control the FDR at any arbitrarily chosen level $q$. In this context, we will denote the upper bound of the FDR by $q$ instead of $\alpha$, which is typically used to bound the FWER; moreover, the values of $q$ are usually set to a different order of magnitude than those of $\alpha$, such as $q = 0.1, 0.2$.

The Knockoff procedure is guaranteed to work under any deterministic design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, which is the matrix of recorded values of $p$ variables across $n$ statistical units; as long as the response $\mathbf{y}$ follows a Gaussian distribution and $n > p$. Actually, as mentioned earlier, we will consider the stricter case where $n \geq 2p$; the rationale for this choice is clarified in A.1.1. Moreover, this procedure does not require the design matrix to be orthogonal nor any knowledge about the noise variance $\sigma^2$, which can be considered unknown.

The core idea of this method is to create "knockoff" variables, $\{\tilde{X}_1, \ldots, \tilde{X}_p\}$, which are artificial copies of the original variables designed to mimic the correlation structure of the original data $\{X_1, \ldots, X_p\}$. Consequently, they are used to calibrate a data-dependent threshold $T$ that will be used to perform multiple testing on the regression coefficients of the linear model (2.9).

Knockoffs work well in combination with a broad class of test statistics used to test the significance of each regression coefficient $\beta_j$. However, these test statistics have to satisfy two properties, detailed in A.1.5, to be compatible with the knockoff procedure. Coming to the actual exposition of this FDR controlling approach, we only present it in combination with statistics from the Lasso, following the outline of the paper that has introduced this tool, Barber & Candès (2015a).

### 2.4.2 Construction

*Step 1: Construct Knockoffs.*

To construct Knockoffs, we first need to compute the Gram Matrix or Correlation Matrix $\mathbf{\Sigma}$ of the original data. The detailed construction of the Correlation matrix $\mathbf{X}^\top \mathbf{X}$ is given in A.1.2. However, we are not interested in computing the Gram matrix per se; instead, we need it to determine the constraints that define the matrix of knockoff variables $\tilde{\mathbf{X}}$. In this regard, the first equality that the knockoff features have to satisfy is:

$$\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}} = \mathbf{\Sigma}, \tag{2.14}$$

which means that the correlation matrix of the knockoff features has to be identical to that of the original data. This follows from implicitly considering $\tilde{\mathbf{X}}$ as

the standardized (through deviance) version of the original knockoff matrix, and consequently having $\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}}$ as the correlation matrix of original knockoffs. From a geometrical viewpoint, considering the observation space (where each axis is a statistical unit and each of the $p$ variables is represented by an $n$-dimensional vector), the constraint in (2.14) means that the cosines of the angles between knockoff variables are equal to those between respective original features. This analogy holds because, in the observation space, the cosines of the angles between variables correspond to their linear correlations. As previously stated, while the first equality involves the correlation of knockoffs with each other, the second constraint involves cross-correlation between knockoff and original variables. The second equality that has to be satisfied by the knockoff features is the following:

$$\mathbf{X}^\top \tilde{\mathbf{X}} = \mathbf{\Sigma} - \text{diag}\{\mathbf{s}\}, \tag{2.15}$$

where $\mathbf{s}$ is a $p$-dimensional nonnegative vector, and $\text{diag}\{\mathbf{s}\}$ denotes a diagonal matrix with $\mathbf{s}$ as entries. This second constraint implies that the correlation structure between the knockoff variables and the original variables must match that of the knockoffs with each other and the original variables with each other. The only exception is the correlation between each knockoff and its corresponding original variable, which should be as low as possible. In fact, $\mathbf{\Sigma}$ and $\mathbf{\Sigma} - \text{diag}\{\mathbf{s}\}$ are equal on off-diagonal entries:

$$\mathbf{X}_j^\top \tilde{\mathbf{X}}_k = \mathbf{X}_j^\top \mathbf{X}_k \qquad \text{for all} \quad j \neq k,$$

while on diagonal entries, we have that

$$\mathbf{X}_j^\top \tilde{\mathbf{X}}_j = \Sigma_{jj} \ - s_j = 1 - s_j.$$

Intuitively, we require this behavior because we want to create fake artificial copies of the original variables with an identical covariance structure, with the exception of correlations with the respective original features; otherwise, we wouldn't be able to distinguish real from fake variables. For this reason, to ensure that our method will have a good statistical power in detecting significant regression coefficients, we should choose entries of $\mathbf{s}$ as high as possible. However, these two equalities don't provide a direct operational definition of the matrix of knockoffs. For this reason, we need to define an explicit formula to compute $\tilde{\mathbf{X}}$. In this regard, it can be proved that there exists a closed formula that allows us to construct the matrix of knockoffs

while satisfying both the constraints in (2.14) and in (2.15); A.1.3 contains the proofs that (2.16) satisfies both these constraints. The formula is stated as follows:

$$\tilde{\mathbf{X}} = \mathbf{X}(\mathbf{I} - \boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\}) + \tilde{\mathbf{U}}\mathbf{C}. \tag{2.16}$$

In this case, $\mathbf{X}$ is the matrix of standardized data using the deviance, while $\boldsymbol{\Sigma}$ is the associated correlation matrix of $\mathbf{X}$; whereas, $\mathbf{s} \in \mathbb{R}^p_+$ is chosen in order to be as large as possible and to satisfy the condition:

$$2\boldsymbol{\Sigma} - \text{diag}\{\mathbf{s}\} \succeq 0, \tag{2.17}$$

which means that the matrix $2\boldsymbol{\Sigma} - \text{diag}\{\mathbf{s}\}$ must be positive semidefinite. The reason behind the requirement in (2.17) lies in the definition of the matrix $\mathbf{C}$. In this regard, $\mathbf{C}$ is defined as the upper triangular matrix resulting from the Cholesky decomposition of the following matrix:

$$2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} = \mathbf{C}^\top\mathbf{C}. \tag{2.18}$$

The Cholesky decomposition is just a way of rewriting a complicated matrix as the product of an upper triangular matrix and its transpose; this decomposition is feasible only if the matrix to decompose is symmetric and positive semidefinite. In this regard, it can be proved that if (2.17) holds, then also the matrix to decompose to obtain $\mathbf{C}$ is positive semidefinite. Besides that, the existence of $\mathbf{C}$ is also guaranteed by the symmetry of the matrix $2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\}$. The condition of symmetry and the equivalence between the expression specified in (2.17) and requiring that the matrix to decompose is positive semidefinite are detailed and proved in A.1.4. Finally, the last quantity not yet defined in (2.16) is $\tilde{\mathbf{U}}$, which is an orthonormal matrix, namely, its column and row vectors have unit length, and they are orthogonal to each other. Therefore, we have that:

$$\tilde{\mathbf{U}}^\top\tilde{\mathbf{U}} = \tilde{\mathbf{U}}\tilde{\mathbf{U}}^\top = \mathbf{I}.$$

Moreover, $\tilde{\mathbf{U}}$ has to be orthogonal to the span of standardized features $\mathbf{X}$. In this regard, it is relevant to mention that the definition of orthogonality can be naturally extended from being a property of a single matrix to a relationship between two matrices; indeed, we say that a matrix $\mathbf{A}$ is orthogonal to $\mathbf{B}$ if their row and column

vectors are perpendicular, namely

$$\mathbf{A}^\top \mathbf{B} = \mathbf{A}\mathbf{B}^\top = \mathbf{0},$$

where $\mathbf{0}$ is the null matrix. On the other hand, the span of a matrix $\mathbf{B}$ is defined as the subspace composed of linear combinations of columns of $\mathbf{B}$. For this reason, saying that $\tilde{\mathbf{U}}$ is orthogonal to the span of $\mathbf{X}$ means that all columns of $\tilde{\mathbf{U}}$ are perpendicular to the column space of $\mathbf{X}$. It is worth noting that the requirement $n \geq 2p$ is strongly related to the construction of $\tilde{\mathbf{U}}$. For this reason, in A.1.1 is provided an extended explanation of this relationship and why $n \geq 2p$ is a crucial requirement to construct classical knockoffs. To conclude, by summing and multiplying the quantities just described as stated in (2.16), we obtain the matrix of knockoff variables.

Another important aspect is how $\mathbf{s}$ is defined. This choice is so crucial that, depending on the method used, the entire knockoff construction will take different names. The first and cheapest method determines the so-called *Equicorrelated knockoffs*, which are the ones used in the simulations of Chapter 3. In this case the vector $\mathbf{s}$ is composed by identical entries $s_j$ defined as:

$$s_j = \min\{2 \cdot \lambda_{\min}(\Sigma), 1\},$$

where $\lambda_{\min}(\Sigma)$ is the minimum eigenvalue of $\Sigma$. In other terms, we are assigning to all pairs $(X_j, \tilde{X}_j)$ the same correlation $1 - s_j$. An alternative that offers greater power but comes at a higher computational cost is to define knockoffs such that the total correlation between original and knockoff variables is minimized. These are called *SDP knockoffs* and are defined by a vector $\mathbf{s} = \{s_1, \ldots, s_p\}$ that solves the following minimization problem:

$$\min_{\mathbf{s} \in \mathbb{R}^p_+} \sum_{j=1}^{p} (1 - s_j) \qquad \text{subject to} \quad 0 \leq s_j \leq 1,\ \operatorname{diag}\{\mathbf{s}\} \preceq 2\Sigma.$$

This optimization problem can be solved efficiently using techniques from linear programming. SDP stems from the minimization problem being a semidefinite program. However, we will not deepen this construction since for the entire thesis, the simpler equi-variant version of the knockoff will be used. Once we have properly defined the matrix of knockoffs, we can go through the second step in which we will construct statistics $W_j$'s. As stated in the introduction, we will present the knockoff

filter in combination with the Lasso.

*Step 2: Compute statistics $W_j$'s for each pair of original variable and knockoff copy.* In the second step, we introduce the statistics $W_j$'s one for each $\beta_j$ with $j \in \{1, \dots, p\}$; these statistics will help to decide which variables are likely to belong to the true model and which are not. Specifically, these $W_j$'s are constructed so that high positive values suggest rejecting the null hypotheses $H_{0j} : \beta_j = 0$ for $j = \{1, \dots, p\}$, while large negative values are evidence in support of the null hypotheses. Since we are presenting the Knockoff filter with statistics from the Lasso, we can consider the previously defined "exit times" of the Lasso path as proxies for variable importance; therefore, we can use them as statistics to assess the significance of $\beta_j$'s. This naturally leads to the definition of a statistic with one of the desired properties mentioned above, namely, that high positive values provide evidence for the significance of $X_j$; we define these statistics as follows:

$$Z_j = \sup\{\lambda : \hat{\beta}_j(\lambda) \neq 0\} \qquad \text{for } j \in \{1, \dots, p\}, \tag{2.19}$$

in this case $\hat{\boldsymbol{\beta}}_{\text{lasso}}(\lambda)$ defined as in (2.12) is replaced with a more concise notation $\hat{\boldsymbol{\beta}}(\lambda)$. $Z_j$ is what we have earlier defined as the exit time for variable $X_j$, namely, the highest value of the penalization parameter for which $\hat{\beta}_j(\lambda)$ is different from 0. In this regard, the true signals are likely to be associated with high positive values of $Z_j$, whereas for most of the null variables, $Z_j$ will likely be small. However, the statistics $W_j$'s, we are interested in defining, need to be calibrated using the knockoff variables; therefore, we compute the statistics in (2.19) not only for original variables $\{X_1, \dots, X_p\}$ but also for knockoffs; this implies estimating a $2p$-dimensional vector $\hat{\boldsymbol{\beta}}_{\text{lasso}}(\lambda)$, that is the minimizer of a Lasso regression, with a design matrix that is the column-wise concatenation of $\mathbf{X}$ and the matrix of knockoffs $\tilde{\mathbf{X}}$. The following is the minimizer of the Lasso regression applied on the concatenated matrix:

$$\hat{\boldsymbol{\beta}}_{\text{lasso}}(\lambda) = \underset{\boldsymbol{\beta} \in \mathbb{R}^{2p}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathbf{y} - [\mathbf{X}\ \tilde{\mathbf{X}}]\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 \right\},$$

that could be rewritten in expanded form as:

$$\hat{\boldsymbol{\beta}}_{\text{lasso}}(\lambda) = \underset{\boldsymbol{\beta} \in \mathbb{R}^{2p}}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^{n} (\underset{1\times 1}{y_i} - \underset{1\times 2p}{[\mathbf{x}_i\ \tilde{\mathbf{x}}_i]^\top} \underset{2p\times 1}{\boldsymbol{\beta}})^2 + \lambda \sum_{j=1}^{2p} |\beta_j| \right\};$$

therefore, we can easily define the statistics in (2.19) for the augmented $2p$-dimensional vector of regression coefficients for both original variables and knockoffs. Consequently, we end up having a corresponding $2p$-dimensional vector of importance scores $(Z_1, \dots, Z_p, \tilde{Z}_1, \dots, \tilde{Z}_p)$. At this point, we are ready to define the statistics $W_j$'s for each pair of original and knockoff variables. Thus, for each $j \in \{1, \dots, p\}$ we define:

$$W_j = \max(Z_j, \tilde{Z}_j) \cdot \begin{cases} +1, & Z_j > \tilde{Z}_j, \\ 0, & Z_j = \tilde{Z}_j, \\ -1, & Z_j < \tilde{Z}_j. \end{cases} \tag{2.20}$$

In this regard, if we look at the lasso path from right to left, namely, from $\underline{0}$ towards $\hat{\boldsymbol{\beta}}_{\text{ols}}$, high positive values of $W_j$ indicates that the original variable $X_j$ enters the lasso path earlier than its knockoff copy $\tilde{X}_j$, or equivalently looking at the lasso path from $\hat{\boldsymbol{\beta}}_{\text{ols}}$ to $\mathbf{0}$, this means that the exit time of the original variable $Z_j$ is greater than the one of its knockoff $\tilde{Z}_j$. Hence, considering a variable $X_j$, if it enters the Lasso path before its knockoff copy, this can be seen as a genuine signal that the variable truly belongs to the model. Conversely, if the Lasso shrinkage is misled by the knockoff, causing the knockoff to enter before the original variable, this can be interpreted as a symptom that the variable is not part of the true model. Intuitively, since in practice we do not know what the true model is, we create artificial copies that allow us to tease apart significant variables from irrelevant ones. At this point, based on $W_j$'s, we can define the actual data-dependent threshold $T$ that will be used to perform multiple testing.

*Step 3: Calculate a data-dependent threshold for the statistics $W_j$'s.*
As said earlier, we would like to select variables such that $W_j \geq t$ for some $t > 0$. At the same time, the purpose of the knockoff filter is to introduce an alternative procedure that both performs variable selection and controls the false discovery rate at level $q$. Thus, letting $q$ be the arbitrarily chosen upper bound of the FDR, we define the critical threshold $T$ as the statistic $W_j$ such that, if selected, it leads to an estimated false discovery proportion bounded by $q$; namely,

$$T = \min \left\{ t \in \mathcal{W} : \widehat{\text{FDP}}(t) \leq q \right\},$$

where $\mathcal{W} = \{|W_j| : j = 1, \dots, p\} \setminus \{0\}$ is the set of unique non null statistics $W_j$'s in absolute value. Thus, by taking the minimum over $\mathcal{W}$, we define $T$ as the loosest

threshold among the $W_j$'s, corresponding to the setting where we achieve the highest number of discoveries while controlling the FDP at level $q$. Intuitively, large values of $W_j$ strictly control the FDP; in the extreme case where $T = +\infty$ - which occurs when $\mathcal{W}$ is empty - the FDP is exactly 0, but no discoveries are made. Since our goal is to maximize the number of discoveries while keeping the proportion of false discoveries under control, we select the threshold $T$ corresponding to the smallest $W_j$ that ensures FDP control. In other words, since lower thresholds lead to more discoveries - but also to more false ones - we define the lowest threshold $T$ such that the resulting estimated false discovery proportion $\widehat{\text{FDP}}(t)$ remains bounded by $q$. In this regard, if we explicitly write the Knockoff estimate of the false discovery proportion, we obtain:

$$T = \min\left\{t \in \mathcal{W} : \frac{\#\{j : W_j \leq -t\}}{\#\{j : W_j \geq t\} \vee 1} \leq q\right\}. \tag{2.21}$$

We have already explained why a large positive value of $W_j$ brings evidence against $H_{0j} : \beta_j = 0$; however, it is not immediately clear how the estimated FDP has been computed. In this regard, the mathematical details about the computation of the knockoff estimate of the false discovery proportion are provided in the intuitive result that follows from Lemma A.1 in Appendix A.

*Step 4: Perform multiple testing*

As explained in the introduction, we are interested in the following multiple comparison problem:

$$H_{0j} : \beta_j = 0 \qquad \text{vs} \qquad H_{1j} : \beta_j \neq 0,$$

with $j \in \{1, \ldots, p\}$. Consequently, to perform multiple testing, we first compute for each hypothesis the statistic $W_j$ as explained in Step 2; Secondly, we define the common threshold $T$ that will be used to decide whether a regression coefficient $\beta_j$ is significant or not. Hence, the multiple testing rejection region $\mathcal{R}$ - with the structure specified in (1.11) but using test statistics instead of p-values - will have the following form:

$$\mathcal{R} = \{H_{0j} : W_j \geq T\},$$

namely, it represents the set of rejected null hypotheses using the knockoff filter. However, since we are working within a linear model framework, and since we have noted that multiple testing on regression coefficients corresponds to performing

variable selection on the model, we can define the set of indexes of selected variables using the Knockoffs procedure as follows:

$$\hat{S} = \{j : W_j \geq T\};$$

It is worth noting that both the set $\hat{S}$ and the data-dependent threshold $T$ depend on the arbitrarily chosen upper bound of the false discovery rate $q$.

### 2.4.3 FDR control

As stated in the introduction, the Knockoff filter is designed to control the false discovery rate; However, up to this point, we have defined the threshold $T$ for multiple testing based solely on false discovery proportion control. For this reason, we now introduce the main result of Barber & Candès (2015a), which establishes that the Knockoff procedure controls a quantity nearly equal to the FDR at the desired level $q$.

**Theorem 2.2.** *Knockoff FDR control (Barber & Candès, 2015a)*
*For any $q \in [0, 1]$, the knockoff method satisfies*

$$\mathbb{E}\left[\frac{\#\{j : \beta_j = 0 \text{ and } j \in \hat{S}\}}{\#\{j : j \in \hat{S}\} + q^{-1}}\right] \leq q,$$

*where the expected value is taken over the Gaussian noise $\mathbf{z}$ in the model (2.9), while treating $\mathbf{X}$ and $\tilde{\mathbf{X}}$ as deterministic and fixed.*

The modified version of FDR that is controlled in Theorem 2.2 is very close to the real false discovery rate when a large number of features is selected; indeed, in that case, adding $q^{-1}$ to the denominator has little effect. However, in some cases, we would like to control exactly the FDR and not its modified version; for this reason, a slightly more conservative procedure has been proposed, which, through a little modification of the data-dependent threshold, guarantees exact FDR control.

**Definition 2.1.** (Knockoff+ Barber & Candès (2015a))
The Knockoff+ is a method that performs variable selection as in the standard Knockoff procedure, but instead of performing multiple testing using the threshold

defined in (2.21), it uses its modified version $T$ which can be defined as follows:

$$T = \min\left\{t \in \mathcal{W} : \frac{1 + \#\{j : W_j \leq -t\}}{\#\{j : W_j \geq t\} \vee 1} \leq q\right\}. \tag{2.22}$$

An important remark is that the critical threshold $T$ in the Knockoff+ procedure is always greater than or equal to that of the classical Knockoff filter. This is because of the additional $+1$ in the numerator of the constraint, which forces us, assuming the same target level $q$, to compensate it by selecting a higher (and therefore more conservative) threshold to reduce the estimated FDP. As a result, Knockoff+ tends to be slightly more stringent in declaring discoveries compared to the standard Knockoff procedure. Consequently, the exact FDR control theorem using Knockoff+ is the following:

**Theorem 2.3.** *Knockoff+ FDR control (Barber & Candès, 2015a)*
*For any $q \in [0, 1]$, the knockoff+ method satisfies*

$$FDR = \mathbb{E}\left[\frac{\#\{j : \beta_j = 0 \text{ and } j \in \hat{S}\}}{\#\{j : j \in \hat{S}\} \vee 1}\right] \leq q,$$

*where the expected value is taken over the Gaussian noise $\mathbf{z}$ in the model (2.9), while treating $\mathbf{X}$ and $\tilde{\mathbf{X}}$ as deterministic and fixed.*

At this point, we can prove the main result of the paper Barber & Candès (2015a) that is expressed in Theorem 2.3. The following is a proof sketch where many details are deferred to Appendix A, which covers Theoretical Details.

*Proof Sketch of Knockoff+ FDR-Control (Barber & Candès, 2015a)*
To prove how the knockoff+ method controls the false discovery rate, we first need to focus our attention on the knockoff estimate of the false discovery proportion, which is crucial in defining the data-dependent threshold $T$. In this regard, we can

observe that:

$$\begin{aligned}
\text{FDP} &= \frac{\#\{j : \beta_j = 0 \text{ and } W_j \geq T\}}{\#\{j : W_j \geq T\} \vee 1} \\
&\leq \frac{1 + \#\{j : W_j \leq -T\}}{1 + \#\{j : \beta_j = 0 \text{ and } W_j \leq -T\}} \cdot \frac{\#\{j : \beta_j = 0 \text{ and } W_j \geq T\}}{\#\{j : W_j \geq T\} \vee 1} \\
&\leq \frac{1 + \#\{j : W_j \leq -T\}}{\#\{j : W_j \geq T\} \vee 1} \cdot \frac{\#\{j : \beta_j = 0 \text{ and } W_j \geq T\}}{1 + \#\{j : \beta_j = 0 \text{ and } W_j \leq -T\}} \\
&\leq q \cdot \frac{\#\{j : \beta_j = 0 \text{ and } W_j \geq T\}}{1 + \#\{j : \beta_j = 0 \text{ and } W_j \leq -T\}};
\end{aligned} \tag{2.23}$$

where the first equality follows from the definition of false discovery proportion applied to the multiple testing of regression coefficients $\beta_j$'s. Indeed, we are computing the ratio between the number of incorrect rejections (namely, null hypotheses declared significant, since $W_j \geq T$, while being false, given that $\beta_j = 0$), and the total number of discoveries. In the second step, the FDP is multiplied on the left by a new quantity greater than 0, thus changing the equality to an inequality. The fact that the new quantity is greater than 1 is a consequence of having an additional constraint $\beta_j = 0$ specified in the set at the denominator of the fraction, which reduces the cardinality of the set itself, and being greater than the numerator, makes the whole ratio $\geq 1$. In the third step, we simply switch the denominators of the two ratios. As a result, in the last step, we recognize that having defined $T$ as in (2.22) guarantees that the quantity on the left is exactly bounded by $q$. Therefore, through this chain of inequalities, we have shown that the FDP is bounded by $q$ multiplied by the ratio of false discoveries over 1+ the number of true negatives. At this point, by taking the expected value on both sides of the inequality (2.23), we have:

$$\begin{aligned}
\mathbb{E}(\text{FDP}) &\leq \mathbb{E}\left[q \cdot \frac{\#\{j : \beta_j = 0 \text{ and } W_j \geq T\}}{1 + \#\{j : \beta_j = 0 \text{ and } W_j \leq -T\}}\right] \\
\text{FDR} &\leq q \cdot \mathbb{E}\left[\frac{\#\{j : \beta_j = 0 \text{ and } W_j \geq T\}}{1 + \#\{j : \beta_j = 0 \text{ and } W_j \leq -T\}}\right];
\end{aligned} \tag{2.24}$$

the first inequality holds due to the monotonicity of the expectation. Whereas, in the second inequality, the left side follows from the definition of false discovery rate (2.1), while the right side is a consequence of the linearity of $\mathbb{E}(\cdot)$. Hence, proving the FDR controlling property of the knockoff+ boils down to showing that:

$$\mathbb{E}\left[\frac{\#\{j : \beta_j = 0 \text{ and } W_j \geq T\}}{1 + \#\{j : \beta_j = 0 \text{ and } W_j \leq -T\}}\right] \leq 1, \tag{2.25}$$

because in this way, we would have shown that defining the threshold $T$ using the knockoff+ approach would lead to FDR $\leq q$. Therefore, to conclude the proof, we need to demonstrate that the claim in (2.25) is true. To do so, we need to introduce a couple of new quantities. The first quantity $V^+(t)$, for a fixed $t$, is a random variable, due to the randomness of the underlying data, and it is defined as follows:

$$V^+(t) = \#\{j : \beta_j = 0 \text{ and } W_j \geq t\}, \tag{2.26}$$

namely, it represents the number of false discoveries when performing multiple testing on $\beta_j$'s using a threshold $t$. At the same time, by considering different values of the data-dependent threshold $t$, we obtain a sequence of random variables indexed by $t$; therefore, we can define the associated stochastic process:

$$\{V^+(t)\}_{t \in \mathcal{W}}, \tag{2.27}$$

which details are given at the beginning of section A.4. Adopting the same logic, we can define the quantity $V^-(t)$, namely, the random variable depending on $t$, representing the number of true negatives:

$$V^-(t) = \#\{j : \beta_j = 0 \text{ and } W_j \leq -t\}, \tag{2.28}$$

as did before in (2.27), we can also define the stochastic process derived from the random variable $V^-(t)$, that is obtained by letting vary the index $t$ in $\mathcal{W}$. Thus, considering a threshold $t$ instead of a fixed $T$, we are now able to rewrite the argument of the expected value in the claim we wanted to prove (2.25), as follows:

$$\frac{V^+(t)}{1 + V^-(t)}. \tag{2.29}$$

At this point, the core part of the proof is to show that $T$, as defined in (2.22), is a stopping time for the super-martingale $V^+(t)/(1 + V^-(t))$. In this regard, section A.3 provides a useful background on some essential concepts in martingale theory that help to better understand this proof sketch. Moreover, the full details on why $T$ is a stopping time and (2.29) a supermartingale are provided in section A.4.1. Consequently, by applying the Optional stopping theorem, whose details are described in A.3.3, to the supermartingale in (2.29), we obtain:

$$\mathbb{E}\left[\frac{V^+(T)}{1 + V^-(T)}\right] \leq \mathbb{E}\left[\frac{V^+(0)}{1 + V^-(0)}\right]; \tag{2.30}$$

clearly, the expected value on the left of this expression is exactly equal to the expected value of the claim (2.25); thus, to show that this expectation is $\leq 1$, we need to prove that the expected value of the super-martingale at time $T = 0$ is bounded by 1. To do so, we can invoke the exchangeability Lemma A.1 whose statement guarantees that the signs of $W_j$'s for true nulls are independent and identically distributed, namely $\text{sign}(W_j) \overset{i.i.d.}{\sim} \{\pm 1\}$; this lemma together with $T = 0$ yield to exact symmetry in declaring positives and negatives; therefore, if we interpret the number of false positives as the count of successes over $p_0$ trials, we have that $V^+(0)$ is distributed according to a $\text{Bin}(p_0, 1/2)$; where $p_0$ is the total number of true nulls, and the success probability is $1/2$ due to the randomness of signs. Consequently, by employing the Proposition A.1 applied to the Binomial random variable $V^+(0)$, we can show that the right member of the inequality in (2.30) is bounded by 1.

$$\mathbb{E}\left[\frac{V^+(0)}{1+V^-(0)}\right] = \mathbb{E}\left[\frac{V^+(0)}{1+p_0-V^+(0)}\right] \leq 1;$$

where the first equality is just a consequence of $p_0$ being the sum of true negatives $V^-(0)$ and false positives $V^+(0)$, and then $V^-(0) = p_0 - V^+(0)$, while the second inequality follows from property A.1. As a result, this last step, combined with (2.24), demonstrates how the Knockoff+ method controls the false discovery rate at an arbitrarily chosen $q$.

### 2.4.4 Knockoff Extensions

In the Knockoff filter section following the presentation of the method introduced in Barber & Candès (2015a), we have presented the tool in combination with statistics from the Lasso; However, it is worth mentioning that the knockoff approach, rather than a single method, is a flexible class of procedures used to perform multiple testing. In this sense, the knockoff approach could be more generally combined with any test statistic $W_j$ that obeys the *sufficiency property* and the *antisymmetry property*, both detailed in A.1.5. The core idea is that the symmetry of test statistics $W_j$'s, together with the construction of knockoff features, allows us to prove the crucial exchangeability Lemma in A.1 that is essential in guaranteeing the FDR-control of the whole procedure. In this regard the logic behind the construction of $W_j$'s is always to define test statistics that for large positive values provide evidence that $\beta_j \neq 0$. From an intuitive viewpoint, we are seeking test statistics $W_j$'s that are able to highlight the importance gap between original and corresponding knockoff

variables. The following are just a few examples among the endless test statistics $W_j$'s that can work well in combination with Knockoffs.

1. $W_j = |\mathbf{X}_j^\top \mathbf{y}| - |\tilde{\mathbf{X}}_j^\top \mathbf{y}|$, which compares the correlation of the original variable $X_j$ with the response and the correlation of the respective knockoff variable $\tilde{X}_j$ also with the response. This holds if the original variables, the knockoffs, and the response have been properly standardized beforehand. The rationale behind this approach is that features whose correlation with the response differs significantly from that of their corresponding knockoffs are highly likely to be relevant.

2. $W_j = |\hat{\beta}_j^{\text{LS}}| - |\hat{\beta}_{j+p}^{\text{LS}}|$, which compares ordinary least squares estimates of original and knockoff variables, where $\hat{\boldsymbol{\beta}}^{\text{LS}}$ has been obtained by regressing $\mathbf{y}$ on the augmented design $[\mathbf{X}\ \tilde{\mathbf{X}}]$. The logic behind this method is to compare the effects of the original variables and their knockoffs on the response, measured by their regression coefficients.

3. In general, considering any penalized likelihood estimation procedure of the form
$$\min_{\mathbf{b}} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2^2 + \lambda P(\mathbf{b}),$$
where $P(\cdot)$ is a penalty function such as the Lasso, Ridge, or others, we can proceed in two ways. The first one is to fix the penalization parameter $\lambda$ (for example via cross-validation) and for each $j$ compare the respective regression coefficients of the original and knockoff variables at that specific $\lambda$, namely $W_j = |\hat{\beta}_j(\lambda)| - |\hat{\beta}_{j+p}(\lambda)|$. The rationale behind this first approach is identical to that of method 2, which compares Least squares estimates of the coefficients. The second approach consists of using specific $\lambda$ values of the regularization path induced by the penalized procedures as proxies of the importance gap between original and knockoff variables; namely, defining $Z_j$ as in (2.19), and then constructing test statistics such as $W_j = Z_j - Z_{j+p}$ or the one in (2.20).

# Chapter 3

# Multiple Testing Application

## 3.1 Introduction

The main focus of this last Chapter concerns applications of multiple testing procedures both on simulated and real data. In this regard, it is worth noting that adopting a simulation framework is essential to test and compare the robustness of competing multiple testing procedures. In fact, the only way of assessing the goodness of a multiple testing approach is through data simulation; in this sense, by generating data arbitrarily, the analyst is able to decide the ground truth he wants to work with and compare results yielded by competing procedures with theoretical ones. In practice, considering for example a linear model, knowing the actual truth could mean that one should generate a design matrix, then arbitrarily fix a subset of non-null regression coefficients with their amplitudes, then determine the resulting response, and finally conduct a multiple test to select significant variables. In doing so, by comparing the selected variables with the ones in the true model, it is possible to measure the accuracy of the procedure. For this reason, by knowing the real truth in advance, we are able to compute essential quantities such as the false discovery rate and the power of the test that are precluded in real-world problems. A detailed description of how these quantities are actually computed is provided in A.5. Ultimately, by comparing the trade-offs between the ability to detect true signals and the strictness of the tests, we can choose the multiple testing procedure that performs the best in terms of FDR and power.
Consequently, Chapter 3 is divided into a first section focused on data simulation, procedures comparison, and empirical validation of theoretical results, and a second section regarding the application of multiple testing to real-world data.

## 3.2 Simulation

### 3.2.1 Comparison: p-value vs rank plot

In assessing the goodness of a multiple testing procedure, a very effective and useful graph is the p-value vs rank plot. As the name suggests, the plot is a bi-dimensional graph having on the x-axis the ranked indices $\{1, \dots, p\}$ of the hypotheses being tested and on the y-axis the corresponding p-values in ascending order $p_1 \leq \cdots \leq p_p$. Moreover, the points of the resulting dispersion plot are colored based on the ground truth of associated hypotheses. In other words, all points corresponding to true nulls are colored the same, while the group of dots representing true alternatives is colored differently. A multiple testing procedure is then represented as a threshold that cuts horizontally the graph, determining, in this way, which hypotheses are rejected and which are not. Clearly, the rejected hypotheses, being associated with low p-values, will fall under the threshold; vice versa, the accepted ones will stay above the threshold. Consequently, one would like to have most of the true alternatives under the threshold while true nulls above it. In practice, especially around the boundary of the two groups, the color distinction is often blurred due to low significant true alternatives and true nulls; therefore, a good multiple testing procedure aims to appropriately cut the graph in that specific region, yielding, as a result, the highest number of rejections while controlling a global measure of error. Another interesting aspect of the p-value vs rank plot is that it allows an extremely effective graphical comparison of different multiple testing approaches. This is done by adding the critical thresholds (in terms of p-value) of different procedures on the same plot, and then eyeballing their performances. A more precise analysis consists of comparing the FDR and the average power function of competing procedures. The Knockoff method has not been considered in the following p-value vs rank plots, given the difficulty in computing the corresponding p-values due to a complex distribution of the test statistics $W_1, \dots, W_p$. Additionally, the framework considered in the next two graphs does not involve a linear model setting, which, on the contrary, is the primary field of application of knockoffs. Despite that, comparing all the procedures, including the knockoffs, will still be possible using power functions and false discovery rates which do not require computing p-values; these comparisons will be carried out in section 3.2.4.

The following graph reproduces the just described comparison among four different approaches: Naive, Bonferroni, Holm's, and Benjamini-Hochberg. It is important

to observe that we are comparing procedures that satisfy different error controlling assumptions. For this reason, It is not fair to claim that Holm and Bonferroni have poorer performances than competing procedures. On the contrary, this is a natural behavior induced by their FWER controlling property. The details about the data-generating mechanism are described below.

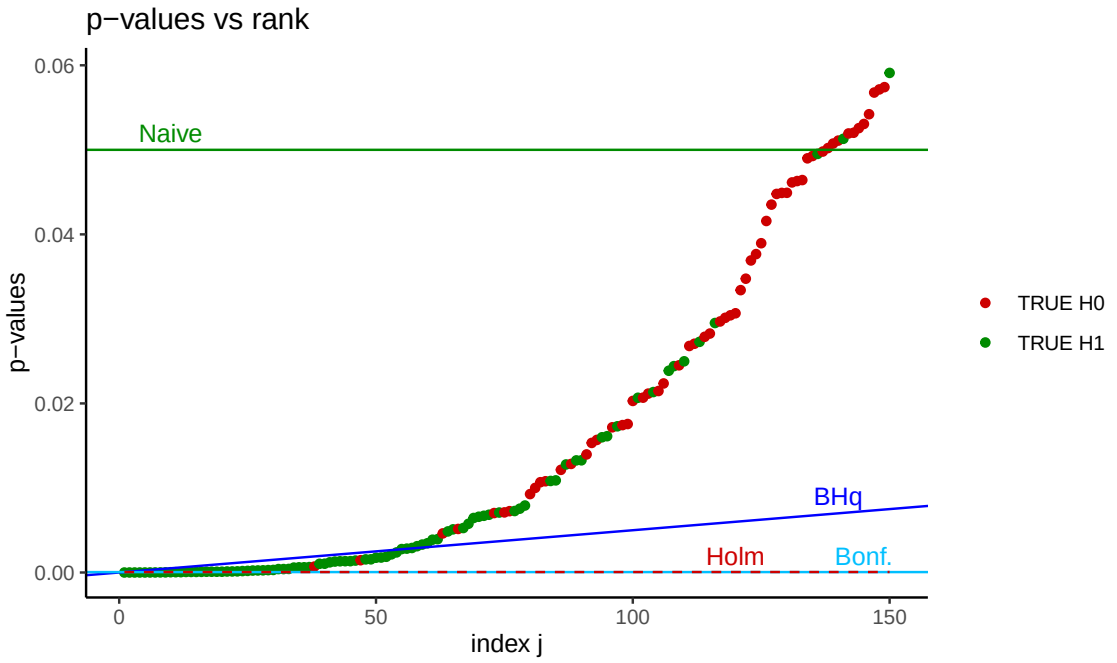


**Figure 3.1:** Ordered p-values vs rank plot. Red dots represent true nulls, while the green dots represent true alternatives. The points below the respective thresholds represent rejected hypotheses across four different methods: Naive, Holm, Bonferroni, and Benjamini-Hochberg.

In Figure 3.1, the first 150 ordered p-values of a testing procedure involving 1000 hypotheses have been plotted. The multiple testing problem that has been considered involves the multiple comparison of two-tailed Z tests, each one with the structure

$$H_{0j} : \mu_j = 0 \quad \text{vs} \quad H_{1j} : \mu_j \neq 0 \quad \text{for} \quad j = 1, \ldots, 1000.$$

The advantage of being in a simulation framework is that we can decide both the amount and the indices of true nulls and true alternatives. In this case, the number of true alternatives has been fixed to 100, and the respective indices have been sampled from $\{1, \ldots, 1000\}$. It is worth mentioning that multiple testing procedures are usually order-invariant, so sampling indices is not strictly necessary. In this case, we have skipped the actual data generation phase; instead, we have directly sampled artificial test statistics from $N(\mu, 1)$ with $\mu = 3$, which correspond to true

alternatives and the remaining test statistics from $N(0,1)$ which correspond to true nulls. Finally, the resulting sampled values $Z_1, \ldots, Z_{1000}$ have been used to compute two-tailed p-values that are shown as dots in the graph. It is important to note that by generating test statistics directly, we implicitly assume a known variance, a condition that rarely holds in real-world scenarios. Depending on the initial assumptions, this approach may be considered either acceptable or inappropriate. In the context of simulation studies, it is critical to ensure that the quantities used to generate the data are not reused in the subsequent analysis. Doing so would compromise the integrity of the simulation and falsify its results. For example, if the simulation assumes that the true variance is unknown, it is acceptable to generate data using the true variance; however, this same variance must not be used later when calculating test statistics or p-values. Using such information would invalidate the simulation by violating its assumptions. This issue highlights the importance of maintaining coherence between the simulation setup and its subsequent analysis. In this regard, a more realistic approach, namely one that does not assume a known variance, has been carried out in the section on FDR and power comparison. In that case, the data-generating mechanism is more complicated since we need to generate the underlying data and then compute the resulting test statistics. Although it comes at a greater computational cost, an approach like that allows for a relaxation of the unrealistic assumption of known variance, which, in that framework, can be estimated using the data. An idea could be, for example, to generate the underlying matrix of data and compute $p = 1000$ $T$-statistics instead of $Z$-scores; while paying attention to use the sample variance $\hat{\sigma}^2$ to standardize the test statistics instead of the known $\sigma^2$. This alternative was not adopted in this section for two main reasons. Firstly, the goal was to produce a simpler and more immediate simulation to have a first overview of the performances of competing multiple testing approaches, thus drawing directly test statistics was enough. Secondly, the performance assessment of multiple testing methods is carried out more accurately using measures such as the false discovery rate and the statistical power, rather than plots comparing p-values to ranks, hence, in this section, a realistic but more complicated simulation setup was not necessary. On the contrary, in section 3.2.2, to conduct a proper procedure comparison, a realistic simulation setup has been considered at the cost of a higher complexity in the data-generating mechanism. Returning to Figure 3.1, it is worth observing that the rejection thresholds have been computed and labeled according to the four mentioned approaches. In this regard, it is useful to note that the Naive and Bonferroni bounds are just constant scalars that differ by a factor of 1/1000;

they are respectively $\alpha = 0.05$ and $\alpha/1000$. The Holm's threshold that entirely overlaps the Bonferroni bound is actually a function of the index $j$, having the form $\alpha/(1000 - j + 1)$. Ultimately, the Benjamini-Hochberg threshold is also a function of $j$ and has been defined accordingly as $(\alpha \cdot j)/1000$. Obviously, in order to have unbroken rejection thresholds, the index $j$ is assumed to be continuous.
To have a clearer comparison of the four multiple testing procedures considered, both the p-values and the rejection threshold of Figure 3.1 can be adjusted using the log-scale. The following is the resulting plot:

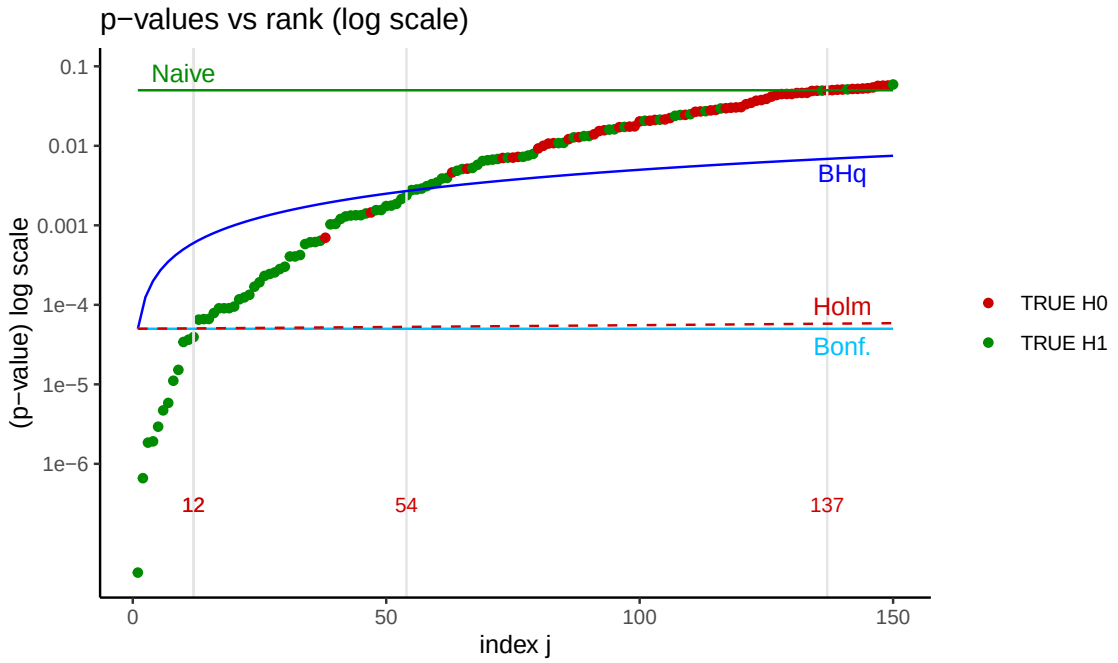


**Figure 3.2:** Ordered p-values in log-scale vs rank plot. Red dots represent true nulls, while the green dots represent true alternatives. The points below the respective thresholds represent rejected hypotheses across four different methods: Naive, Holm, Bonferroni, and Benjamini-Hochberg. The respective thresholds have been adjusted using a log-transformation.

In Figure 3.2, the distinction between performances of different approaches is more effective; in this sense, we can better appreciate the slightly lower strictness of Holm with respect to Bonferroni, the ability of Benjamini-Hochberg to cut the dots precisely in the blurred region and the exact number of rejected hypotheses for each approach that have labeled with the corresponding number colored in red. The data-generating mechanism is the same as Figure 3.1. As described at the end of Chapter 1, an alternative representation of the comparisons in Figures 3.1 and 3.2 could have been to consider a fixed threshold $\alpha$ and adjust p-values according to the

four different approaches. Obviously, the two methods must have produced the same exact discoveries.

### 3.2.2 Microarray simulation study

#### Average power comparison

A more accurate way of comparing multiple testing procedures is through their average power functions. As defined in Algorithm 6, the concept of power function can be extended to the multiple testing framework using the average true positive rate over $M$ Monte Carlo iterations; in this sense, the average power function measures the ability of a multiple comparison procedure to detect true alternatives across different signal amplitudes. Throughout the thesis, whenever the power of a multiple testing procedure is mentioned, we will mean the average true positive rate. While being highly informative, average power functions provide only a partial view of the performance of the multiple testing procedures under consideration. Specifically, they overlook the number of false discoveries produced. Therefore, a comprehensive evaluation must consider both the power of the test and an appropriate error metric, such as the false discovery rate. Figure 3.3 shows a graphical comparison of the approximated bilateral power functions averaged over 1000 trials for Naive, Benjamini-Hochberg, Bonferroni, and Holm procedures. The plot in Figure 3.3 was generated following the procedure described in Algorithm 6, with some modifications to make the simulation framework more realistic. The aim was to reproduce, within a simulation context, a microarray study inspired by the first example on prostate cancer data found in Chapter 15 of Efron & Hastie (2021). An essential difference between this analysis and the one from the book is that this one does not transform test statistics to make them Normally distributed, and it does not rely on a real dataset, but it artificially generates new datasets at each Monte Carlo iteration. The idea is to consider an $n \times p$ matrix with $p = 100$ genes' expression measured across $n = 50$ patients. These patients are divided into two known categories, namely, $n_1 = 25$ cancer patients and $n_2 = 25$ normal controls. Among the $p = 100$ genes, only 15 have been arbitrarily chosen to be truly relevant. The goal is to detect which genes have a significant difference in terms of genetic expression between the two groups, making them interesting to study since reasonably related to cancer. It is worth noting that in this problem, the focus is not to predict cancer for new or existing patients but to detect which genes are provably related to the disease.

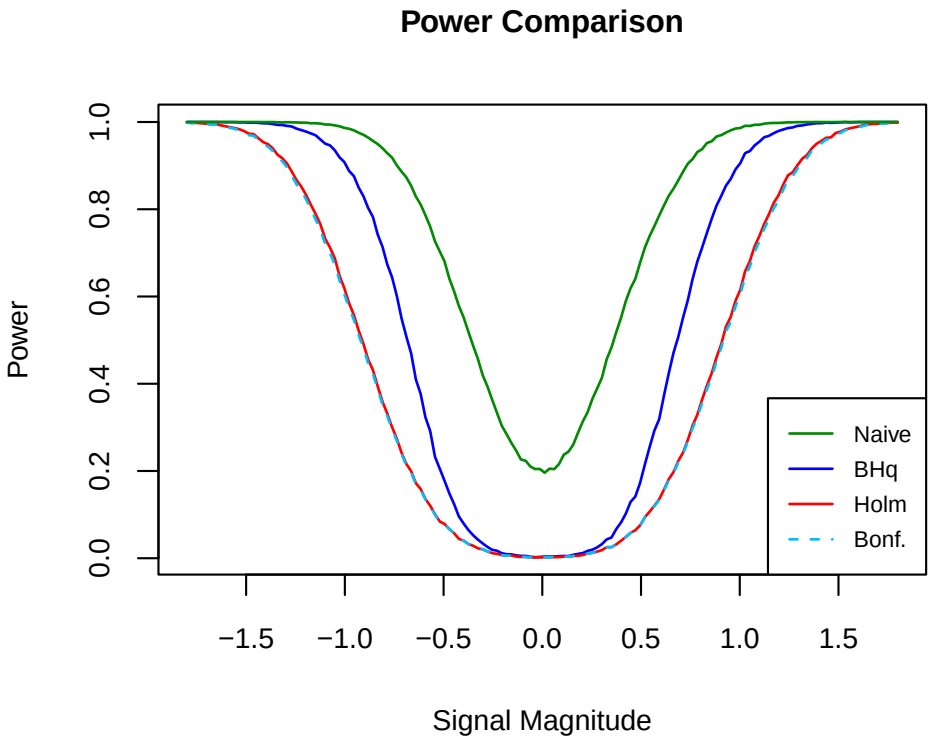


**Figure 3.3:** The graph shows the average true positive rate over $M = 1000$ Monte Carlo iterations associated with four multiple testing methods: Naive, Bonferroni, Holm, and Benjamini-Hochberg applied to a multiple Two-sample T-Test. The simulation details are provided below.

Therefore, we expect to find only a few relevant genes among a haystack of irrelevant ones. The following are the actual details of the simulation.
The test structure for each hypothesis is defined as follows:

$$H_{0j} : \mu_{1j} = \mu_{2j} \quad \text{vs} \quad H_{1j} : \mu_{1j} \neq \mu_{2j}, \quad \text{for } j = 1, \ldots, p,$$

where we essentially want to test whether the means of the two groups are different or not across all genetic expressions. However, considering the substitution $\delta = \mu_{1j} - \mu_{2j}$, and assuming that the effect $\delta$, if significant, is equal across all $j \in \{1, \ldots, p\}$ the structure of the test can be further simplified as follows:

$$H_{0j} : \delta = 0 \quad \text{vs} \quad H_{1j} : \delta \neq 0.$$

Therefore, we end up performing a multiple Two-sample T test to compare the expected values of homoscedastic Gaussian populations with unknown variance;

hence, we are also assuming that the variance is equal among variables and among groups and that the genetic expression is normally distributed. Consequently, we will test these $p$ hypotheses using Two-sample T statistics. Defined as:

$$T_j = \frac{\bar{X}_{j1} - \bar{X}_{j2}}{S_{jp}\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \mid H_{0j} \sim t_{n_1+n_2-2}, \tag{3.1}$$

where the label 1 marks the group of cancer patients with size $n_1$ and 2 marks the group of normal controls having size $n_2$. The overall test statistic is distributed following a Student T with $n_1 + n_2 - 2$ degrees of freedom, thus two tailed p-values will be computed accordingly. $\bar{X}_{j1}$ is the sample mean of genetic expression for gene $j$ among cancer patients and $\bar{X}_{j2}$ the same but for normal controls. $S_{jp}$ is the square root of the pooled variance, which is the appropriate estimator for the shared $\sigma$. Specifically, $S_{jp}$ is defined as:

$$S_{jp} = \sqrt{\frac{(n_1 - 1)S_{j1}^2 + (n_2 - 1)S_{j2}^2}{n_1 + n_2 - 2}}. \tag{3.2}$$

In this case, $S_{j1}^2$ and $S_{j2}^2$ are the unbiased sample variances of gene $j$ expression within each of the two groups.

Coming to the numerical details of the simulation, we have considered a synthetic matrix of data with dimensions $50 \times 100$, each column has been independently generated from $N(\delta, 1)$ for cancer patients and from $N(0, 1)$ for normal controls. Then, having homoscedasticity between the two groups and assuming an unknown variance, a two-sample T statistic (3.1) has been computed for each column. In this phase, to avoid invalidating the simulation, it was crucial not to use 1 as standard deviation but to estimate it through the square root of the pooled variance (3.2). Another crucial remark is that the multiple tests have been carried out independently, given that the data have been sampled column by column without any dependence. This has been done to guarantee FDR-control for the Benjamini-Hochberg procedure. Consequently, $p = 100$ two-sample T statistics have been computed, leading to the respective two-tailed p-values (1.5). Then, the four earlier mentioned procedures: Naive, Bonferroni, Holm's and BHq have been used to declare discoveries leading, as a result, to four different true positive rates. At this point, the whole multiple comparison procedure involving $p = 100$ two-sample T-tests has been repeated for $M = 1000$ Monte Carlo iterations. This has been done in order to average the

resulting true positive rates of each procedure to obtain the approximated powers across the four approaches. Lastly, this whole procedure was repeated for a grid of 100 equally spaced signal amplitudes $\delta \in \{-1.8, \ldots, 1.8\}$. Finally, the four average power functions have been plotted in Figure 3.3. Coming to the actual comments of the graph, in Figure 3.3, we can observe a similar power trend across different procedures, namely a power function that for weak signals stays around 0 and grows monotonically to 1 as the signal gets stronger (on average). An interesting fact is that these averaged power functions show an increasing first derivative up to around a power of 0.5 and then a decreasing derivative. In other words, there is an initial phase (shifted across procedures) where increasing the signal strength, even by a few amount, leads to a substantial gain in power, then a second phase in which, despite further increases in the signal, the power gain becomes weaker and eventually reaches a plateau. On the other hand, it is also interesting to analyze the shift of the power function across procedures. Coherently with theoretical results, the naive approach (by not controlling FWER or FDR) always has the highest power; this simply means that it will reject easily $H_{0j}$'s and not that it is precise, since many alleged discoveries will be false. Then, the average power of the BHq approach, as expected, appears to be a compromise between power and false discovery control. To conclude, Bonferroni and Holm's confirm their conservative bounds, by showing a very weak power, which is a consequence of their "tendency" to declare only extremely significant discoveries. It is worth noting the slightly looser behaviour of Holm with respect to Bonferroni that has been explained in the theory. Consequently, the goal is not to find the most powerful procedure (otherwise naive will always be the best), but to find the approach that, while being powerful, is able to control an error measure (such as the FDR) across different signal amplitudes. In this sense, it can be intuitively defined a notion of trade-off power-FDR control.

### FDR comparison

As already explained, assessing the procedures' control of the false discovery rate is essential in selecting a robust multiple testing approach. For this reason, in this section, we compare the same four approaches, Naive, Bonferroni, Holm's and Benjamini-Hochberg, using their respective FDRs as functions of the signal amplitudes in Figure 3.4. The framework is the microarray simulation study detailed

in the previous section, with the only difference that this time the construction of the FDR functions is provided in Algorithm 7. Therefore the data-generating mechanism is exactly equal to the one used in Figure 3.3 with the exception that, instead of computing true positive rates, at each iteration and across the four approaches, we compute the false discovery proportion (2.1) and then by averaging them we obtain the estimated FDR. In Figure 3.4, coherently with the theory of multiple testing, we can observe empirically how the Naive approach does not control the FDR while the other methods do. It is worth observing that, among the FDR-controlling procedures,

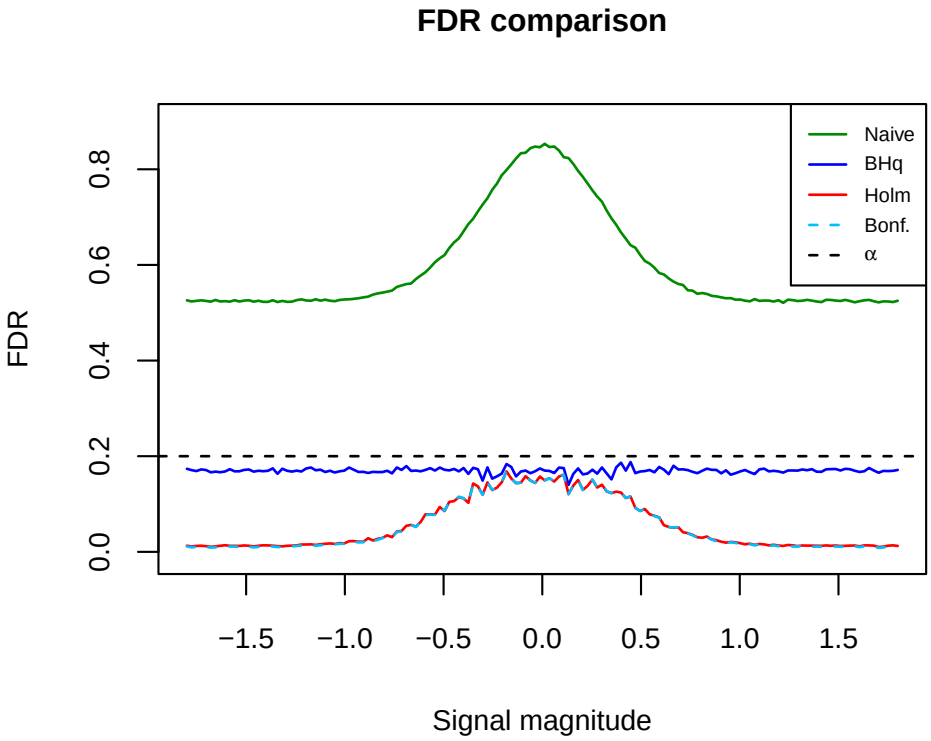


**Figure 3.4:** Approximated FDR via Monte Carlo as a function of the signal amplitude across the Naive, Bonferroni, Holm's, and BHq methods. The data-generating mechanism is identical to the one in Figure 3.3 with a few differences detailed in the extended graph description. The upper bound of FDR is fixed at $\alpha = 0.2$

the main difference lies in how the FDR is controlled when the signal varies. Indeed, this very property can be used as a watershed between FDR controlling procedures to choose the best approach. In this sense, we can see how, regardless of the signal amplitude, the FDR of Benjamini-Hochberg stays just below the threshold, which is an optimal behavior since it maximizes the number of discoveries while keeping FDR under control; whereas Bonferroni and Holm's have a decreasing FDR as the signal

increases in absolute value, namely, they will tend to produce more false positives, when the signal is low and less false positives when the signal gets stronger. In other words, Bonferroni and Holm are more conservative when they could be looser since the signal is stronger, and vice versa when they should be more conservative, because the signal is weaker, they are looser. We can easily understand that a desirable behaviour for a multiple testing procedure is the exact opposite, namely, being looser at high signal amplitudes and stricter at low signal, while controlling the FDR. In this regard, one could show that, in certain contexts, the Knockoff method introduced in Barber & Candès (2015a) satisfies this property. In other terms, in high-dimensional and nearly orthogonal settings, the Knockoff procedure shows an increasing FDR as the signal gets stronger, while maintaining it below the upper bound q. The next section analyzes in a simulation framework the theoretical properties of the Knockoff approach, while a comprehensive comparison of the five approaches will be made in the final section of this Chapter.

### 3.2.3 The Knockoff filter

In the context of variable selection for a linear model, using the Knockoffs filter turns out to be a good strategy that balances power with the FDR control, while providing inferential guarantees on the chosen signals. However, the most relevant breakthrough of the Knockoffs, which makes them more general than BHq, is that, while controlling FDR, they do not require independence among hypotheses being tested. Since in practice, unless artificially built designs, variables are never completely independent, using the knockoffs turns out to be a very useful approach. Moreover, another interesting property of Knockoffs is that they can be combined with a broad class of non-conventional test statistics that can increase the overall performance of the method. In this regard, a rather effective graph that allows both to easily communicate the results of the Knockoff method and to better understand the structure of its data-dependent threshold $T$ is the plot of the pairs of Lasso path entries of original variables and corresponding Knockoffs. In Figure 3.5 is provided a representation of this graph using simulated data. Since we are in a simulation setting, we can color the points according to the ground truth: red squares denote true alternatives (signals), while black points indicate true nulls. The goal of the procedure is to maximize the rejection of true alternatives while minimizing the rejection of true nulls. In Figure 3.5, the x-axis represents the values of the penalization parameter $\lambda$ in the Lasso path (which is read from $\underline{0}$ to $\hat{\boldsymbol{\beta}}^{\text{LS}}$) at which

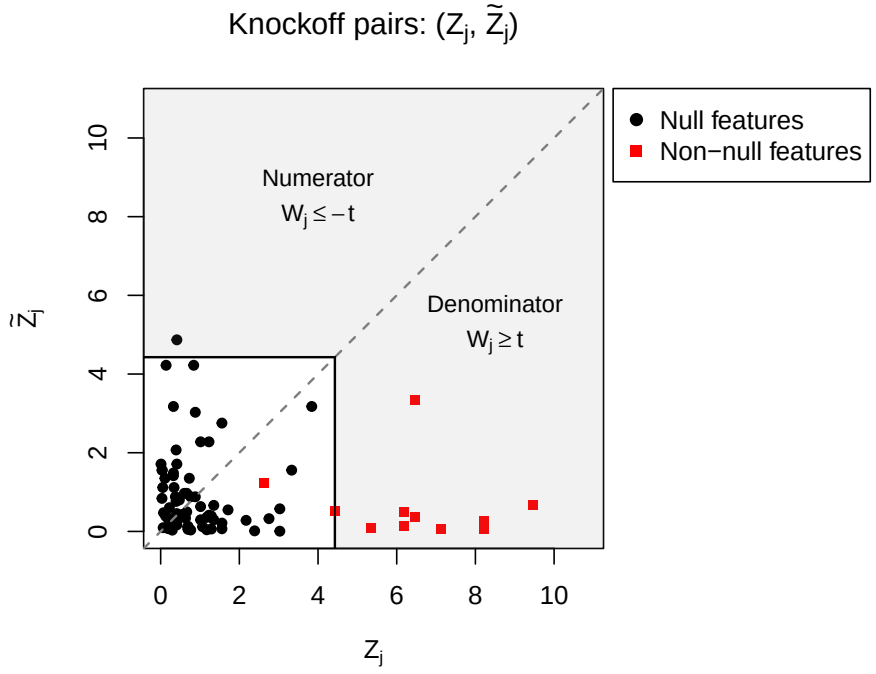


**Figure 3.5:** Scatterplot of the pairs of Lasso entries $(Z_j, \tilde{Z}_j)$ for $j = 1, \ldots, p$ for original and knockoff variables. Red squares represent true alternatives while black points represent true nulls. Dots below the diagonal line and out of the white square represent selected variables. $p = 80$ two-tailed hypothesis tests on the regression coefficients of a linear model have been performed. The number of observations is $n = 1500$, while the number of true positives has been fixed at 12. The signal amplitude $A = 3$ is equal across the $p$ coefficients. Each of the $n$ rows of the design matrix has been drawn from $N(\underline{0}\,, \Sigma)$ with $\Sigma_{jj} = 1$ for all $j = 1, \ldots, p$ and $\Sigma_{jk} = 0.3$ for all $j \neq k$. The FDR upper bound has been fixed to $q = 0.15$. Continuous black lines represent the data-dependent threshold found by the Knockoff method. The labels numerator and denominator refer to the conditions that define the quantities in the ratio inside the definition of the threshold $T$ in (2.21), here we have their graphical representation; they are crucial in defining the Knockoff estimate of FDP.

each original variable $X_j$ first enters the model, denoted by $(Z_1, \ldots, Z_p)$ as defined in (2.19). Similarly, the y-axis represents the corresponding $\lambda$ values for knockoff variables $\tilde{X}_j$, namely, $(\tilde{Z}_1, \ldots, \tilde{Z}_p)$, also defined in (2.19). As described in *Step 2* of the Knockoff construction, points below the diagonal line satisfy $Z_j > \tilde{Z}_j$, suggesting that the original variable $X_j$ is more important than its knockoff and therefore likely belongs to the true model. However, selecting all variables corresponding to points below the diagonal line would result in many false discoveries (black points), as illustrated in the figure. Often, most false discoveries are concentrated near the origin, whereas most true positives (red squares) also lie below the diagonal but are farther from the origin. This observation motivates the idea of choosing a threshold $t$ that defines a square around the origin on both axes to isolate variables likely to

be true nulls and control the FDR. Particularly, the continuous vertical segment represents the data-dependent threshold $T$ defined in (2.21); $T$ is constructed to ensure that, if variables with $W_j \geq T$ are selected, the FDR is controlled at level q; $W_j$ is defined in (2.20). Consequently, $W_j \geq T$ means rejecting hypotheses for which the corresponding point lies under the diagonal and after the threshold $T$. As a result, the number of points in the shaded region under the diagonal represents the total number of rejections, namely, $\#\{j : W_j \geq T\}$. Moreover, by observing that the dots are roughly symmetrically distributed with respect to the diagonal line (which is a graphical intuition behind Lemma A.1), the number of false discoveries can be estimated with $\#\{j : W_j \leq -T\}$. Hence, from a graphical viewpoint, it becomes clearer the definition of the Knockoff estimate of the false discovery proportion.

### 3.2.4 Effect of sparsity level, signal amplitude, and feature correlation

This section aims to present a comparison of the false discovery rate and the average power as functions of signal sparsity, signal amplitude, and feature correlation across the following procedures: Naive, Bonferroni, Holm, Benjamini-Hochberg, Knockoff, and Knockoff+. The effects of the signal sparsity, the signal magnitude, and the variable correlation on the FDR and power have been analyzed separately. Moreover, in this simulation, we have considered a linear regression framework, meaning that the focus is on performing multiple tests on the nullity of the regression coefficients. It is important to underline that the comments made on the graphs of this section are valid for these specific simulations; in general, the approximated behaviour of FDR and power changes due to a combination of factors that is hard to investigate in its entirety. Full details on the simulations in this section are provided in B.2.4

*(Signal Sparsity)* In this first part, the FDR and the power of the multiple tests are analyzed through simulation as functions of the signal sparsity across Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ methods. In this regard, the powers in Figure 3.6 have been generated using Algorithm 6 applied to all six approaches (using sparsity level instead of signal magnitude on the x-axis). Moreover, the design matrix of the linear model has been drawn once and kept fixed for the whole analysis, while new noise has been introduced at each Monte Carlo iteration. The following are the major details about the actual simulation. Since we are in a simulation setting, we can decide both the true values of the regression coefficients and their level of sparsity. Firstly, we fix the dimensions of the design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$,

namely, $n = 200$ observations and $p = 100$ features. Each row is then sampled from a multivariate normal $N(\mathbf{0}, \Theta)$ with $\Theta_{ii} = 1$ and $\Theta_{ij} = 0.4$ for all $i \neq j$. In doing so, we are deciding to work under a non-orthogonal design, which is likely the case in many real problems. At this point, we center and normalize the columns of $\mathbf{X}$, and we fix an arbitrary signal amplitude $A = 3.5$. We then set a grid of sparsity levels for the associated linear model $K = \{1, 2, 3 \ldots, 60\}$. In this regard, $k \in K$ measures the sparsity of the regression, namely the number of non-null variables included in the model. The notation for the sparsity level $k$, taken from Barber & Candès (2015a), could be quite confusing since high sparsity means low $k$ and vice versa. To be clear, when $k = 1$ the model is highly sparse, having all regression coefficients equal to zero except for one; on the contrary, as $k$ grows, more variables are included in the regression, making the problem increasingly less sparse. Then, after having fixed a sparsity level $k$, we fix the $k$ regression coefficients $\beta_j, \ldots, \beta_k$ all equal to A, and we define:

$$\mathbf{y} = \beta_1 \mathbf{X_1} + \cdots + \beta_k \mathbf{X}_k + \mathbf{z} \qquad \text{where } \mathbf{z} \sim N(\mathbf{0}, \mathbf{I}_n), \tag{3.3}$$

where $\mathbf{y}$ is an $n$-dimensional vector of responses, $\{\beta_1, \ldots, \beta_k\}$ are scalars , $\mathbf{X}_1, \ldots, \mathbf{X}_k$ are $n$-dimensional vector of features and $\mathbf{z}$ is an $n$-dimensional Gaussian noise. At this point, for a fixed sparsity level $k$, we test the nullity of regression coefficients $\{\beta_1, \ldots, \beta_k\}$ using all six approaches at $\alpha = 0.2$. We then repeat these same steps with the exception of the generation of $\mathbf{X}$ and $\boldsymbol{\beta}$ for $M = 2000$ Monte Carlo iterations, and we average the respective true positive rates to obtain six different powers for the same sparsity level $k$. Finally, we reiterate this whole procedure keeping fixed $\mathbf{X}$ for all sparsity levels specified in $K$ (we do it by progressively replacing zeros with A's in $\boldsymbol{\beta}$. We then draw the graph by plotting the set of sparsity levels $K$ against the resulting powers across the six multiple testing approaches considered. In Figure 3.6, the performances of competing methods are effectively displayed through their powers. As we expected from the theory, Naive method has the highest power across all sparsity level $k$; however, this behaviour has no value since it does not guarantee any error controlling property. Indeed, the outperformance is just a consequence of the Naive loose threshold, which is highly likely to produce false discoveries. It is worth mentioning that a more meaningful comparison should have included only FDR and FWER controlling procedures and not the Naive approach; however, we have kept it to make the reader ponder on the apparent good behavior of the Naive approach, which, in reality, is just producing an inflation of discoveries. More interesting are the behaviors of FDR controlling procedures. Starting from BHq, the

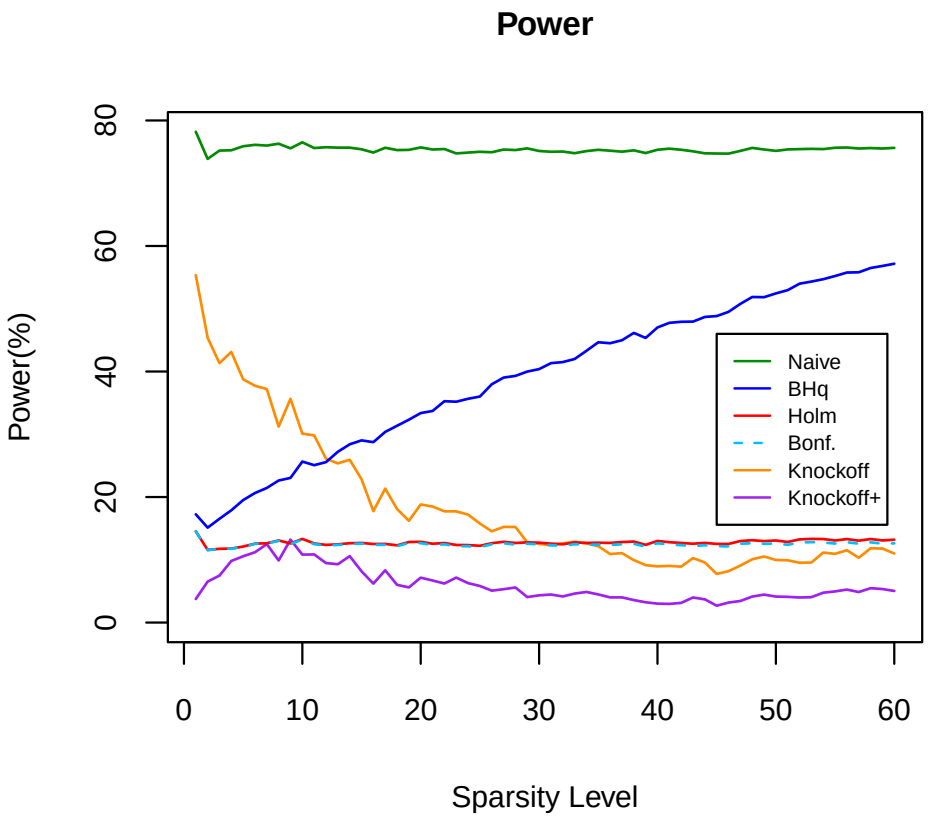


**Figure 3.6:** Power vs Sparsity Level across Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ methods. powers have been computed by averaging the respective true positive rates for $M = 2000$ Monte Carlo iterations. All multiple testing procedures have been performed using an FDR or FWER upper bound $\alpha = 0.2$.

increasing trend of its power suggests that, at least empirically, raising the number of true signals increases the power of the procedure; conversely, Classical Knockoffs, which do not control exactly the FDR but a slightly more permissive threshold, show the property of being more powerful in highly-sparse context and increasingly more conservative when more covariates are significant. Whereas, Knockoff+, given the FDR-controlling property, is in general, for all sparsity levels, more conservative than BHq, its plus version, Bonferroni, and Holm's method. As mentioned earlier, a correct multiple testing approach assessment must analyze not only the power but also an error measure such as the false discovery rate. For this reason, in Figure 3.7, the false discovery rates as functions of the sparsity levels have been plotted for the same six procedures. The graph in Figure 3.7 was created using Algorithm 7, under the same simulation settings as those described for the power. Coherently with the theory of multiple testing exposed throughout this work, the Naive method

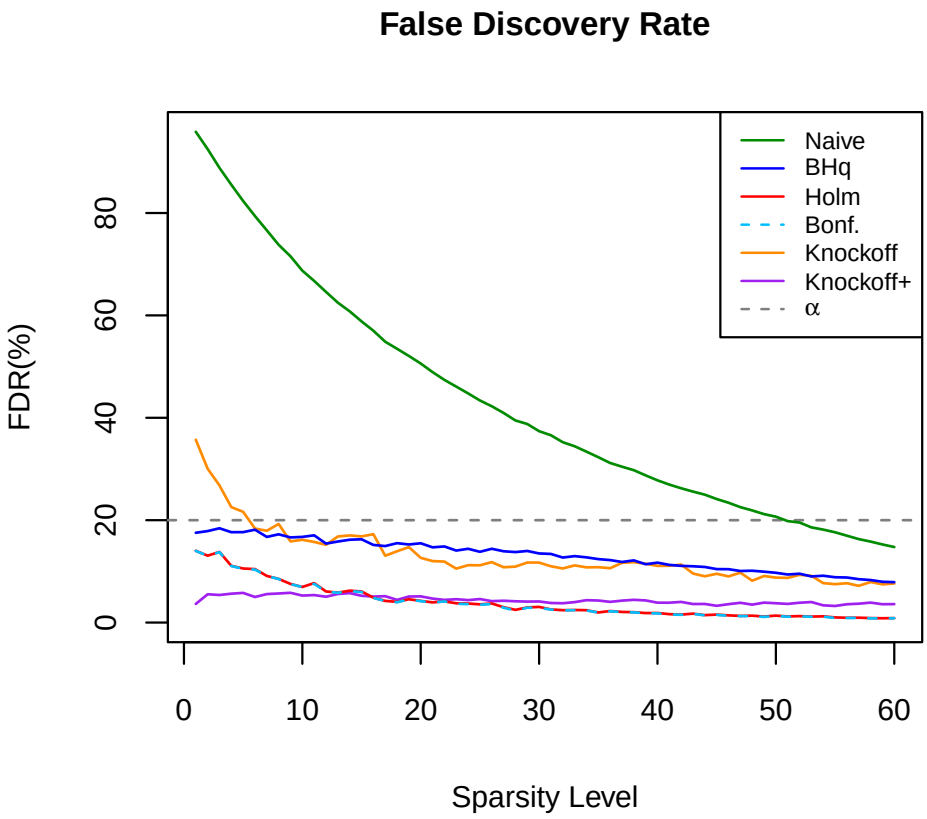


**Figure 3.7:** False Discovery Rate vs Sparsity Level across Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ methods. FDRs have been computed by averaging the respective false discovery proportions for $M = 2000$ Monte Carlo iterations. All multiple testing procedures have been performed using an FDR upper bound $\alpha = 0.2$.

does not control the FDR and it explodes under highly-sparse regression conditions; indeed, it reaches approximately FDR $= 1$ when $k$ is low, meaning that nearly all rejections are type I errors. Moreover, it can be observed that all procedures except for Knockoff+ show a decreasing FDR as sparsity is reduced. This is a reasonable behavior since it becomes harder to commit type I errors when the number of true signals increases; conversely, Knockoff+ seems not to be affected by the number of true variables in the model, showing, in this way, a robust behavior against sparsity. It is worth noting that although in this simulation BHq empirically controls the FDR at level q with a non-orthogonal design, such control is not guaranteed theoretically under dependence of the hypotheses tested.

*(Feature Correlation)* In this second part, the FDR and the average power of the multiple tests have been analyzed via simulation as functions of the feature correlation

across Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ methods. The simulation details are similar to those of the previous section, with some modifications to take into account a changing feature correlation and highlight interesting effects. Starting from the power comparison, Figure 3.8 has been generated using Algorithm 6; the following are the details of the simulation used. Firstly, we fix the dimensions of a matrix of data $\mathbf{X}^{n\times p}$ with $n = 300$ observations and $p = 150$ variables. Next, we set a grid of 30 equally spaced correlations $\mathcal{R} = \{0, \dots, 0.99\}$. Then, we choose a fixed number $k = 10$ of variables to include in the true model and a signal magnitude $A = 3.5$. We then fix a vector of regression coefficients $\beta_1, \dots, \beta_k$ all equal to A; this vector will remain the same for the whole analysis. At this point, having fixed $\rho \in \mathcal{R}$, we sample each row of the design matrix $\mathbf{X}$ from a Normal $N(\mathbf{0}, \Theta_\rho)$ with $\Theta_{ii} = 1$ and $\Theta_{ij} = \rho$ for all $i \neq j$. In doing so, we can manage the correlation structure of the design by letting $\rho$ vary in $\mathcal{R}$. Hence, fixing a specific feature correlation, we can define the regression model as:

$$\mathbf{y} = \beta_1\mathbf{X}_1 + \cdots + \beta_k\mathbf{X}_k + \mathbf{z} \qquad \text{where } \mathbf{z} \sim N(\mathbf{0}, \mathbf{I}_n), \tag{3.4}$$

where, as before, $\mathbf{y}$ is an $n$-dimensional vector of responses, $\{\beta_1, \dots, \beta_k\}$ are scalars ,$\mathbf{X}_1, \dots, \mathbf{X}_k$ are $n$-dimensional vector of features and $\mathbf{z}$ is an $n$-dimensional Gaussian noise. At this stage, for a fixed correlation of the features $\rho \in \mathcal{R}$, we test the nullity of regression coefficients $\{\beta_1, \dots, \beta_k\}$ using all six approaches and an $\alpha = 0.2$. We then repeat these same steps for $M = 800$ Monte Carlo iterations, by redrawing $\mathbf{X}$ and the response at each iteration; then we average the respective true positive rates to obtain six different powers for the same feature correlation $\rho$. Ultimately, we reiterate this whole procedure for all feature correlation $\rho \in \mathcal{R}$. We then draw the graph of powers by plotting on the x-axis the correlations in $\mathcal{R}$ and on the y-axis the corresponding averaged powers across the six multiple testing approaches considered. In this way, we obtain the plot in Figure 3.8. In this case, we can appreciate that increasing the feature correlation has a strong impact on the ability to detect true signals across all approaches. Intuitively, when variables are strongly correlated, it is more difficult to isolate relevant ones. It is worth mentioning that all procedures except for BHq and Naive are guaranteed to control FDR or its modification. This makes the comparison among approaches not that meaningful since we are comparing procedures that do not satisfy the same FDR-controlling properties. However, from this graph, we can still observe the highly desirable behavior of the Knockoff method, which, despite not controlling directly FDR (which

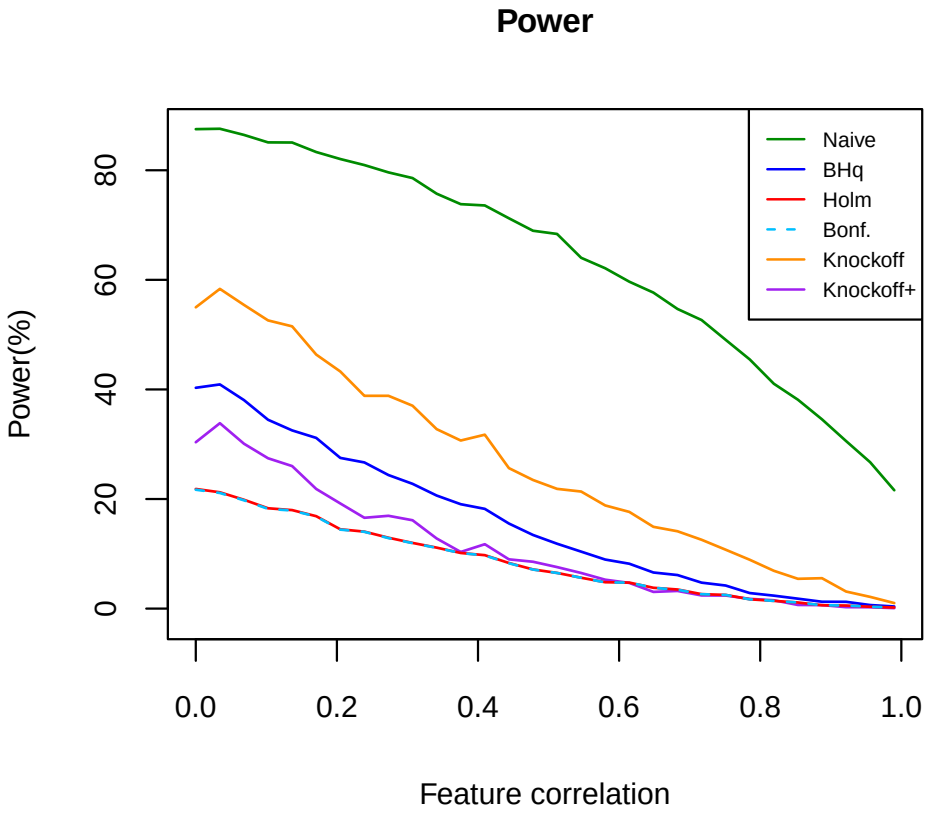


**Figure 3.8:** Power vs Feature correlation across Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ methods. powers have been computed by averaging the respective true positive rates for $M = 800$ Monte Carlo iterations. All multiple testing procedures have been performed using an FDR upper bound $\alpha = 0.2$.

is not a problem since we still know what modification of FDR we are controlling 2.2), outperforms all procedures, with the exception of Naive for understandable reasons. Therefore, in this framework, the Knockoff method is uniformly more powerful than BHq and it also guarantees modified-FDR control (which BHq does not); this means that in high-correlation contexts the Knockoff approach offers better power and FDR control, that is exactly what we are looking for; hence, in these situations it is highly preferable to Benjamini-Hochberg. Moreover, it is also worth observing that the Knockoff+ power corresponding to low feature correlation values is higher than that of strongly conservative methods. This means having higher chances of finding true positives when features have a better structure to detect signals. However, to complete the evaluation of the effects of feature correlation across multiple testing approaches, it is necessary to consider the false discovery rate as a function of $\rho$ across the six procedures; this graph has been plotted in

Figure 3.9. In this regard, the FDRs have been drawn following Algorithm 7 with a simulation setup identical to the one used for Figure 3.8. As we expected, the

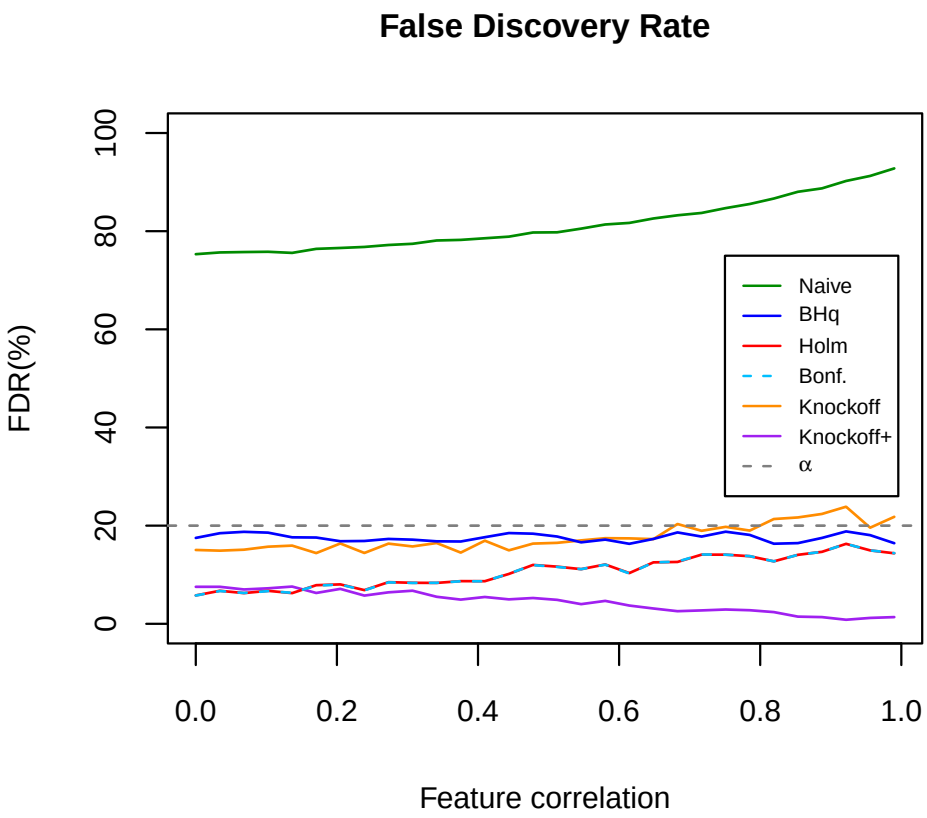


**Figure 3.9:** False Discovery Rate vs Feature correlation across Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ methods. FDRs have been computed by averaging the respective false discovery proportions for $M = 800$ Monte Carlo iterations. All multiple testing procedures have been performed using an FDR upper bound $\alpha = 0.2$.

Knockoff+ method, Bonferroni, and Holm control the false discovery rate, while other procedures do not; with the exception of classical knockoff which controls a slightly modified version of FDR. In practice, it is irrelevant to control FDR or a modified version of it as long as it is known what we are controlling; indeed, depending on the situation, we will know that Knockoff will control a certain modified upper bound of FDR while the plus version controls FDR exactly. The problem arises with other procedures, such as BHq, that, without the independence assumption, have an FDR that could potentially explode. However, Figure 3.9 could be misleading since it seems that BHq controls the FDR better than knockoffs for high correlation. In reality, knockoffs provably control a modified FDR, while the BHq method is not guaranteed to control anything. Indeed, it is more desirable to control something,

even a modified version of FDR as Knockoff+ does, rather than controlling nothing at all like BHq (in high correlation settings). A final interesting observation about Figure 3.9 is that as feature correlation increases, the FDR tends to rise across all procedures except for Knockoff+, which exhibits a more robust behavior; in fact, a slight decrease in the FDR of Knockoff+ can be observed as feature correlation increases.

*(Signal Magnitude)* In this third and last part, we have compared the actual power functions of six procedures (Naive, Bonferroni, Holm, BHq, Knockoff, Knockoff+) and the false discovery rate for different signal amplitudes. It is important to note that previous "powers" in Figure 3.6 and 3.8 were not strictly speaking power functions, since a proper definition requires the signal amplitude to be on the x-axis. Starting from Figure 3.10, average power functions have been computed using Algorithm 6. Specifically, we have first fixed the dimensions for the design matrix $\mathbf{X}^{n\times p}$, having $n = 100$ observations and $p = 50$ variables. Then, for the whole simulation, the number of truly significant features has been fixed to $k = 8$ while the feature correlation has been set to $\rho = 0.4$. In this regard, each row of the design matrix $\mathbf{X}$ has been drawn from a Normal $N(\mathbf{0}, \Theta_\rho)$ with $\Theta_{ii} = 1$ and $\Theta_{ij} = 0.4$ for all $i \neq j$. Next, we have chosen a grid of 60 equally spaced signal amplitudes $\mathcal{A} = \{-10, \dots, 10\}$, from which to compute $\beta_j$'s. Consequently, for a fixed $A \in \mathcal{A}$ we have imposed all the regression coefficients $\beta_1, \dots, \beta_k$ to be equal to A. As a result, we can define:

$$\mathbf{y} = \beta_1\mathbf{X}_1 + \cdots + \beta_k\mathbf{X}_k + \mathbf{z} \qquad \text{where} \;\; \mathbf{z} \sim N(\mathbf{0}, \mathbf{I}_n), \tag{3.5}$$

where these quantities have already been described for model (3.3) and (3.4). The idea, as before, is to test the nullity of regression coefficients $\{\beta_1, \dots, \beta_k\}$ with an $\alpha = 0.2$ using all six approaches for a fixed signal amplitude $A$. Then, the process is repeated for $M = 2000$ Monte Carlo iterations and the respective true positive rates are averaged to obtain six different powers for the same signal amplitude. In this case, as we did with signal sparsity, the design matrix $\mathbf{X}$ has been drawn once, while the responses have been sampled at each montecarlo iteration from $N(\mathbf{X}\boldsymbol{\beta}, \mathbf{I})$. Lastly, this whole procedure is reiterated for all signal magnitudes $A \in \mathcal{A}$, by changing the vector of $\boldsymbol{\beta}$ and thus leading to the graph in Figure 3.10. In this regard, power functions are obtained by plotting on the x-axis the elements of $\mathcal{A}$ and on the y-axis the corresponding averaged powers across the six multiple testing approaches. It can

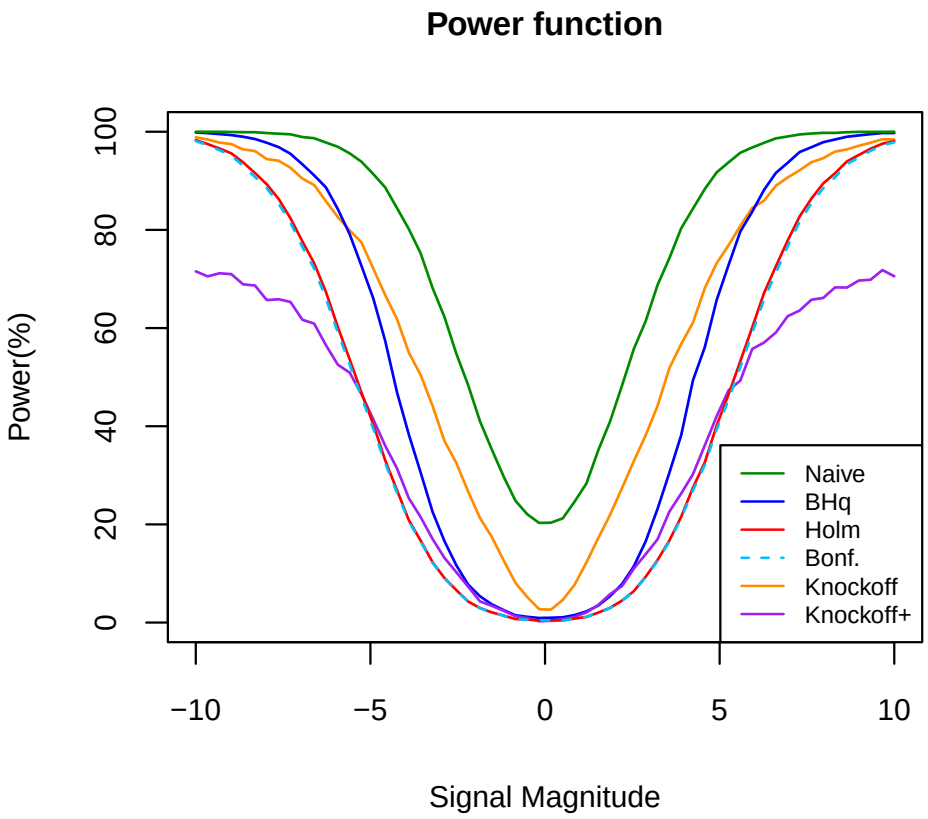


**Figure 3.10:** Power vs signal magnitude across Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ methods. power functions have been computed by averaging the respective true positive rates $M = 2000$ Monte Carlo iterations. All multiple testing procedures have been performed using an FDR upper bound $\alpha = 0.2$.

be observed that in a sparse scenario with a weak signal amplitude, Knockoff performs the best among all procedures except for Naive, which is reasonable since the Naive method does not control any error measure. Instead, restricting our attention to procedures that strictly control the false discovery rate, Knockoff plus turns out to be the best approach in terms of power, especially for weak signals. As already done in the sections on sparsity and feature correlation, to give a comprehensive evaluation of the six methods considered, we have also analyzed the respective false discovery rates across the six procedures, but this time as functions of the signal amplitude. In Figure 3.11, we can observe a highly desirable property that is typical only of the Knockoff+ method, namely, an increasing FDR as the signal amplitude is increased. This is actually a very interesting property for a multiple testing procedure since it guarantees conservativeness at low signal amplitudes and higher permissiveness at high signal amplitudes. Moreover, in this graph has been also computed the modified

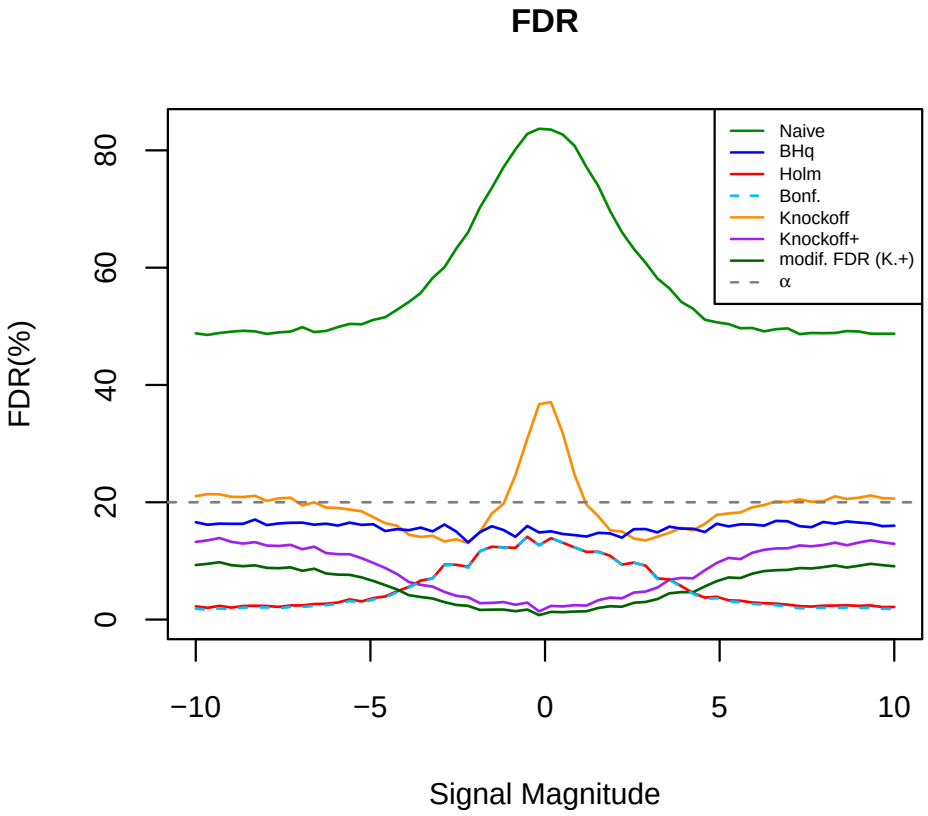


**Figure 3.11:** FDR vs signal magnitude across Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ methods. false discovery rates have been computed by averaging the respective false discovery proportions for $M = 2000$ Monte Carlo iterations. All multiple testing procedures have been performed using an FDR upper bound $\alpha = 0.2$.

FDR of the Knockoff+ method (in darkgreen) which controls the upper bound $q$ as stated in Theorem 2.3.

## 3.3 Experiment on real data: HIV-1 drug resistance

This last section deals with a real data application of the six multiple testing procedures presented throughout this work: Naive, Bonferroni, Holm, Benjamini-Hochberg, Knockoff, and Knockoff+. The idea behind this section is to provide a comprehensive and customized analysis of the HIV data proposed in (Barber & Candès, 2015a). This part will first cover the biological background needed to understand the HIV

data, secondly, the data pre-processing phase will be presented, and thirdly, we will focus on the actual application of the six methods to a cleaned dataset and comment on the final results.

The problem we want to solve using a multiple testing approach is to detect significant mutations in the Human Immunodeficiency Virus type I (HIV-1) that are associated with drug resistance. This is interesting since mutations lead to drug resistance and then to ineffective therapies. Therefore, detecting robust and provably significant mutations can help in designing new and more effective drugs.

### 3.3.1 Background

HIV-1 is a virus that attacks cells of the immune system, weakening the body's ability to fight infections and diseases. HIV-1 is primarily transmitted through blood, sexual contact, and from mother to child during pregnancy, delivery, or breastfeeding. If not treated, it can lead to AIDS (Acquired Immunodeficiency Syndrome), a disease with a life expectancy of $1-11$ years from the diagnosis. It is detected through blood tests. HIV-1 is treated using antivirals, namely, drugs that do not kill the virus directly but inhibit biochemical reactions that are essential to allow viral replication. More precisely, the virus, after having gained access inside the host cell, releases several proteins, called enzymes, that are essential in speeding up the virus's replication. If a drug is able to inhibit the action of a virus's enzyme, it can then stop the replication. Therefore, in general, antivirals are inhibitors of viruses' enzymes. For this reason, antivirals are typically known by the name of the enzyme they inhibit, followed by the term "inhibitor". In (Barber & Candès, 2015a), three classes of HIV antivirals have been considered, namely, Protease Inhibitors (PIs), Nucleoside Reverse-Transcriptase Inhibitors (NRTIs), and Nonnucleoside Reverse-Transcriptase Inhibitors (NNRTIs). In this application, we will focus only on Nelfinavir, a drug in the class of Protease Inhibitors. Protease is an HIV enzyme whose function is to cut the polyprotein produced after the translation of viral DNA. The cutting phase is essential for producing the proteins that will be processed by the Golgi apparatus and transformed into structural components of the virus. Preventing the cutting action of the Protease means terminating the replication.

However, treating viruses is not that simple. In fact, in response to the inhibitory action of the drugs, HIV-1, like all viruses, tends to mutate its enzyme structures in order to make it more difficult for drugs to recognize the enzymes to target. This phenomenon, known as drug resistance, makes drugs less effective in combating diseases. In our case, HIV-1 protease mutates in response to the action of Protease inhibitors, therefore, our interest lies in detecting relevant mutations of HIV-1 Protease associated with Nelfinavir resistance. HIV-1 Protease is a protein composed by a chain of 99 amino acids. In its non-mutated form, the enzyme is said to be wild-type. A mutation in the Protease chain means that a specific wild-type amino acid gets replaced by another one. Mutations are usually denoted with strings of three elements: the initial of the wild-type protein, the position of the mutation on the chain, and the initial of the replaced amino acid. The presence of multiple mutations within the enzyme and across different patients makes it particularly challenging to identify robust resistance-associated mutations. To conclude, I want to thank Dorian Safa, a friend of mine and medical student at the Vita-Salute San Raffaele University in Milan, for the fruitful conversations we had to understand the biological background of this analysis.

### 3.3.2 Analysis

The dataset used for the analysis is `PI_DATA`, available at Stanford HIVDB and described in (Rhee et al., 2006). In the study that created this dataset, scientists extracted from each of $n = 848$ patients positive to HIV-1 a chemical sample of HIV-1 Protease. For each sample, they recorded a measure of drug resistance and the presence of mutations at each position of the Protease chain. Specifically, the resistance $y_i$ of each chemical sample of HIV-1 Protease to Nelfinavir is measured as a log fold increase in the following way:

$$y_i = \log_{10}\left(\frac{\mathrm{IC}_{50,i}^{\mathrm{sample}}}{\mathrm{IC}_{50}^{\mathrm{WT}}}\right), \qquad \text{for } i = 1, \dots, n \tag{3.6}$$

where $\mathrm{IC}_{50,i}^{\mathrm{sample}}$ is the Inhibitory Concentration, namely, the concentration of Nelfinavir needed to inhibit 50% of the viral replication for the $i^{\mathrm{th}}$ patient Protease. Whereas, $\mathrm{IC}_{50}^{\mathrm{WT}}$ is the inhibitory concentration of the wild-type (non-mutated) HIV-1 protease chain. Intuitively, the higher the concentration of Nelfinavir needed to inhibit 50% of viral replication, the less effective the drug is. Therefore, since in (3.6)

the denominator is constant across all patients and the logarithm is a monotonic transformation, the higher the value of $y_i$, the more resistant the $i^{\text{th}}$ sample is to Nelfinavir. The `PI_DATA` dataset contains not only the response variable $y$, which measures resistance to Nelfinavir, but also the Protease mutation profiles across patients. Specifically, each of the 99 columns corresponds to a position in the HIV-1 Protease sequence. For any given cell $(i, j)$, the entry is a hyphen - if the $i^{\text{th}}$ sample has no mutation at position $j$ relative to the wild-type protease. If a mutation is present, the cell contains the initial of the amino acid that replaces the wild-type at that position. Actually, `PI_DATA` contains measurements of drug resistance for 7 different Protease inhibitors; we have only chosen Nelfinavir because it was the drug with the fewest amount of missing values, and also to simplify the analysis. Once the columns of drug resistance to Protease inhibitors, different from Nelfinavir, are removed, we end up having a dataset with $n = 848$ observations and $p = 100$ features. After having filtered patients with missing Nelfinavir resistance, the dataset has $n = 844$ observations and $p = 100$ variables. However, this is not yet the matrix of data to do multiple testing on. Following the analysis in (Barber & Candès, 2015a), we first need to consider different mutations at the same position as different variables. Moreover, we need to quantify mutations, since, up to this point, the entries of the matrix of data are either letters or hyphens. To do so, the dataset has been transformed in order to have on the columns unique mutations such as M46I, I10L etc... while on the rows the patients' IDs. To be clear, a notation like M46I means that a mutation from a wild-type Methionine to Isoleucine has occurred at position 46. In this phase, the wild-type sequence of HIV-1 Protease that is necessary to define mutations with a full notation (such as M46I) has been downloaded from Uniprot. Then, using information from the original dataset and following the approach in (Barber & Candès, 2015a), a new matrix $\mathbf{X}$ has been created with entries $X_{ij} \in \{0, 1\}$. Specifically, $X_{ij} = 0$ if sample $i$ does not show the mutation expressed in column $j$, and vice versa, $X_{ij} = 1$ if sample $i$ shows the mutation expressed in column $j$. We have then filtered out all mutations appearing in less than 3 samples and removed duplicated columns to avoid rank-deficiency problems. As a consequence, we end up with a sparse matrix of zeros and ones with $n = 844$ samples of HIV-1 protease and $p = 205$ unique mutations. We are now able to fit a linear regression model and test the significance of coefficients to detect relevant mutations using the six approaches mentioned throughout the thesis. We have considered as response variable the continuous Nelfinavir resistance $y$ and as predictors dummy variables indicating the presence or absence of unique mutations

in each sample. The logic behind this approach is to explain the drug resistance using mutations presence/absence. However, we are not interested in predicting drug resistance for new patients, but in detecting provably significant mutations. The linear regression model considered has the following structure:

$$y_i = \beta_0 + \beta_1 D_{i1} + \cdots + \beta_p D_{ip} + z \qquad \text{for } i = 1, \ldots, n \quad \text{with } z \sim N(0, \sigma^2). \tag{3.7}$$

To further clarify, we have that each dummy variable is defined as follows:

$$D_{ij} = \begin{cases} 0, & \text{if sample } i \text{ does not show mutation } j \\ 1, & \text{if sample } i \text{ shows mutation } j. \end{cases} \tag{3.8}$$

We can now perform all six multiple testing methods to jointly test the $p = 205$ regression coefficients of the linear model; as a result, finding significant regression coefficients will mean detecting relevant mutations. At this stage, in most of the real cases, the analysis would be concluded by reporting to pharmacologists or drug designers the selected mutations. however, in this specific case, we can also evaluate the accuracy of all six multiple testing approaches by comparing selected mutations with the ones in the Treatment-selected mutation (TSM) list, which provides a good approximation of the ground truth. The TSM list is built by experts such as biologists who fill it with mutations that are statistically more common in treated patients than in untreated ones. Since the TSM list for HIV -1 Protease is an approximation of the ground truth, we have compared only the positions of the mutations selected for each of the six methods with the positions of the mutations in the TSM list. Figure 3.12 is a barplot showing across the six methods how they perform in detecting relevant positions of mutations. Whereas, Figure 3.13 represents the plot of test statistics $Z_j$'s and $\tilde{Z}_j$'s used by the Knockoff procedure to tease apart relevant mutations from irrelevant ones. Red dots represent mutations whose position falls in the TSM list, while black dots are the remaining mutations. All points under the diagonal line and beyond the vertical threshold are the selected mutations by the Knockoff method. Therefore, following the Knockoff procedure, the points in the light-green area that are red represent true discoveries, while the black ones are false discoveries. On the opposite side, in the light-red area, black dots represent true negatives, while red points are false negatives. One of the strengths of the Knockoff approach lies in its ability to identify a region, colored in light blue near the origin, that is an indistinguishable mixture of true and false mutations and is therefore ignored by

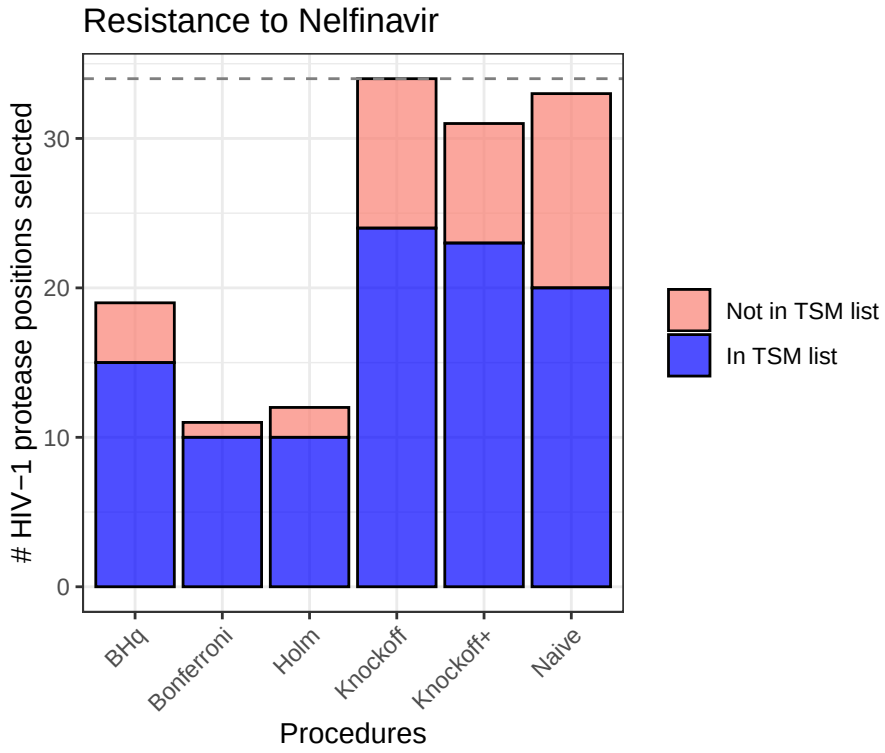


**Figure 3.12:** Results of applying Naive, Bonferroni, Holm, BHq, Knockoff, and Knockoff+ with $q = 20\%$ to detect significant mutations in a linear model having as response variable Nelfinavir resistance and as predictors Dummy variables representing mutations in the HIV-1 Protease. The bar plots show the number of unique positions selected by the six approaches. To validate the selections of the six methods, dark blue indicates Protease positions that appear in the TSM panel for the PI class of drugs, while orange indicates positions selected by the six methods that do not appear in the TSM list. The horizontal dashed line indicates the total number of HIV-1 protease positions appearing in the TSM list.

the method. It is important to notice that in the Figure 3.12 and then Figure 3.13, different quantities have been used to display the accuracy of the Knockoff filter. In fact, in Figure 3.12, only the positions of selected mutations have been compared against the positions appearing in the TSM list. On the contrary, in Figure 3.13 all unique selected mutations (even at the same position) have been plotted and colored based on whether the associated position appeared or not in the TSM list. This was to clarify the apparent incoherence in the different numbers of selected mutations and selected positions.

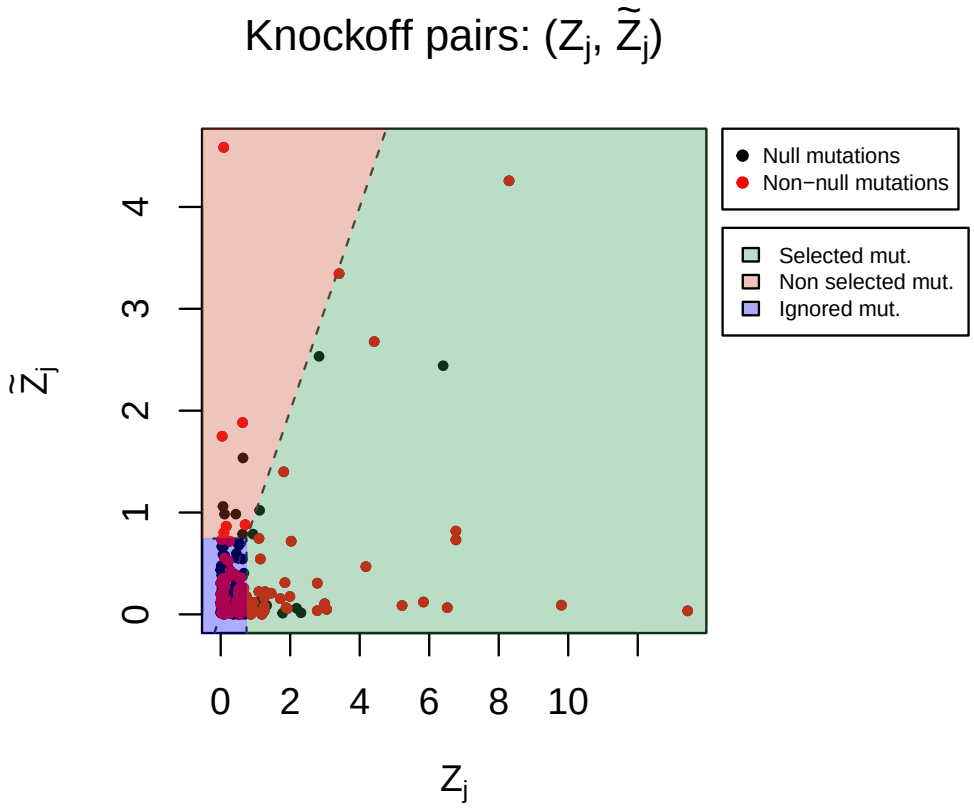


**Figure 3.13:** Scatterplot of the pairs of Lasso entries $(Z_j, \tilde{Z}_j)$ for $j = 1, \ldots, 205$ for original and knockoff variables. Red dots represent mutations whose position appears in the TSM list, black dots represent mutations whose position does not appear in the TSM list. All dots in the light-green area represent mutations selected by the knockoffs. Under this area, red points represent truly relevant mutations while black dots are false discoveries. The light blue area near the origin includes mutations that are too hard to detect and, therefore, are ignored.

# Appendix A

# Theoretical details

## A.1 Knockoff construction details

### A.1.1 $n \geq 2p$ framework

Requiring the number of observations $n$ to be larger than or equal to $2p$ is essential to define the matrix of knockoffs $\tilde{\mathbf{X}}$ in the standard way. Following the definition of $\tilde{\mathbf{X}}$ specified in (2.16), a crucial quantity that needs to be computed is the matrix $\tilde{\mathbf{U}}$. This matrix is defined as the orthogonal complement of $\mathbf{X}$, but it can also be interpreted as the Kernel of the linear application $\mathbf{X}^\top$. This because we are looking for a matrix $\tilde{\mathbf{U}}$ such that

$$\tilde{\mathbf{U}}^\top \mathbf{X} = \mathbf{0} \iff (\tilde{\mathbf{U}}^\top \mathbf{X})^\top = \mathbf{0}^\top \iff \mathbf{X}^\top \tilde{\mathbf{U}} = \mathbf{0}, \tag{A.1}$$

where the first equality means that we want $\tilde{\mathbf{U}}$ to have all columns orthogonal to the column space of $\mathbf{X}$, the second equality is just the transpose of the first one and the third equality, which is equivalent to the first one, requires $\tilde{\mathbf{U}}$ to be the Null space of the transformation $\mathbf{X}^\top$. At this point, using the well-known Rank-Nullity Theorem, page 61 Lang (1987), on the linear application $\mathbf{X}^\top$, we have that:

$$\dim(\mathbf{X}^\top) = \operatorname{rank}(\mathbf{X}^\top) + \dim(\operatorname{Ker}(\mathbf{X}^\top)). \tag{A.2}$$

Therefore, since $\operatorname{Ker}(\mathbf{X}^\top) = \tilde{\mathbf{U}}$, the $\dim(\mathbf{X}^\top) = n$ and the $\operatorname{rank}(\mathbf{X}^\top) = p$, we are left with $n - p$ dimensions for the null space. Hence, given that $\tilde{\mathbf{U}}$ must be an orthogonal $n \times p$ matrix, we want the Kernel to be at least $p$ dimensional in order to properly define $\tilde{\mathbf{U}}$; Thus $n - p$ must be larger than or equal to $p$, which means $n \geq 2p$. Another reason why $n$ must be greater than $2p$ is that some test statistics require the full rank of the columns-wise concatenation matrix $[\mathbf{X}\ \tilde{\mathbf{X}}]$.

### A.1.2 Gram Matrix construction

The construction of the correlation matrix $\mathbf{X}^\top\mathbf{X}$ associated to the original data, which is also a Gram matrix given the standardization procedure using the deviance, starts from centering the design matrix. The following is the extended formula of the linear transformation $\mathbf{H}$ that, once applied to the matrix of original features, centers it.

$$\mathbf{X}_c = \left(\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top\right)\mathbf{X} = \mathbf{H}\mathbf{X}, \tag{A.3}$$

where $\mathbf{H}$ is the centering matrix and $\mathbf{X}_c$ is the matrix having mean equal 0 for each variable $X_{cj}$. More precisely, in the expanded formula of the centering matrix, $\mathbf{I}$ is a squared $n \times n$ identity matrix, while $\mathbf{1}$ is a column vector of ones and $\mathbf{1}^\top$ is its transpose. As a result, the effect of $\mathbf{H}$ is to subtract the respective column mean from every entry of the design matrix. Once $\mathbf{X}$ has been centered, we can standardize it in order to obtain unit variance for each variable. This can be done by multiplying on the right $\mathbf{X}_c$ by the diagonal matrix whose elements are the inverse of the standard deviations of each original variable,

$$\mathbf{Z} = \mathbf{X}_c \ \mathrm{diag}\left(\frac{1}{\hat{\sigma}_{11}}, \ldots, \frac{1}{\hat{\sigma}_{pp}}\right) = \mathbf{X}_c\mathbf{D}^{-1/2}, \tag{A.4}$$

where $\hat{\sigma}_{jj}$ is the sample standard deviation computed on the values of the variable $X_j$. In this way, we obtain $\mathbf{Z}$ which is the matrix of standardized data having mean 0 and variance 1 for each variable. The standardization procedure could equivalently be done by dividing each column of $\mathbf{X}_c$ by the Euclidean norm associated with the respective variable; with the only difference that, strictly speaking, in that case, we would have standardized using the deviance rather than the standard deviation. At this point, we can compute the correlation matrix of the original variables, which is equal to the covariance matrix of the standardized data:

$$\mathbf{\Sigma} = \frac{1}{n}\mathbf{Z}^\top\mathbf{Z}. \tag{A.5}$$

An alternative way in which we could have obtained the correlation matrix $\mathbf{\Sigma}$ would have been through the Gram matrix. Indeed, as mentioned above, by normalizing each column such that $\|\mathbf{X}_j\|_2^2 = 1$ which corresponds to standardizing using the deviance of each variable, the division by n would not have been needed and the resulting Gram Matrix would have been equal to the correlation matrix of original data.

### A.1.3 Proof of Knockoffs constraints

In this section it is provided an extended proof that demonstrates how the formula of knockoffs proposed in Barber & Candès (2015a), and detailed in (2.16), complies with the constraints in (2.14) and (2.15).

*Proof of constraint* (2.14)
We want to show that $\tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} = \boldsymbol{\Sigma}$. Firstly, we expand both $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{X}}^\top$

$$\tilde{\mathbf{X}} = \mathbf{X}(\mathbf{I} - \boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\}) + \tilde{\mathbf{U}}\mathbf{C} = \mathbf{X} - \mathbf{X}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} + \tilde{\mathbf{U}}\mathbf{C},$$
$$\tilde{\mathbf{X}}^\top = \mathbf{X}^\top - (\mathbf{X}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\})^\top + (\tilde{\mathbf{U}}\mathbf{C})^\top = \mathbf{X}^\top - \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\mathbf{X}^\top + \mathbf{C}^\top\tilde{\mathbf{U}}^\top.$$

The second line follows from the properties of the transpose and observing that a diagonal matrix is identical to its transpose. Then we proceed to show that $\tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} = \boldsymbol{\Sigma}$, by replacing the matrices with their explicit formulations and compute the one by one term product.

$$\begin{aligned}\tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} &= (\mathbf{X}^\top - \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\mathbf{X}^\top + \mathbf{C}^\top\tilde{\mathbf{U}}^\top)(\mathbf{X} - \mathbf{X}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} + \tilde{\mathbf{U}}\mathbf{C}) \\ &= \mathbf{X}^\top\mathbf{X} - \mathbf{X}^\top\mathbf{X}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} + \mathbf{X}^\top\tilde{\mathbf{U}}\mathbf{C} - \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\mathbf{X}^\top\mathbf{X} \\ &+ \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\mathbf{X}^\top\mathbf{X}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\mathbf{X}^\top\tilde{\mathbf{U}}\mathbf{C} + \mathbf{C}^\top\tilde{\mathbf{U}}^\top\mathbf{X} \\ &- \mathbf{C}^\top\tilde{\mathbf{U}}^\top\mathbf{X}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} + \mathbf{C}^\top\tilde{\mathbf{U}}^\top\tilde{\mathbf{U}}\mathbf{C}.\end{aligned}$$

Given the construction of $\tilde{\mathbf{U}}$, all terms including $\tilde{\mathbf{U}}^\top\mathbf{X}$ or its transpose are equal to $\mathbf{0}$ and then can be elided. Besides that, the orthonormality of $\tilde{\mathbf{U}}$ implies that $\tilde{\mathbf{U}}^\top\tilde{\mathbf{U}}$ is the Identity matrix; moreover, by replacing $\boldsymbol{\Sigma}$ with its definition $\mathbf{X}^\top\mathbf{X}$, we have that several quantities can be simplified. Then, we can write:

$$\begin{aligned}\tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} &= \mathbf{X}^\top\mathbf{X} - \mathbf{X}^\top\mathbf{X}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\mathbf{X}^\top\mathbf{X} \\ &+ \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\mathbf{X}^\top\mathbf{X}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} + \mathbf{C}^\top\mathbf{C} \\ &= \mathbf{X}^\top\mathbf{X} - \boldsymbol{\Sigma}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\boldsymbol{\Sigma} \\ &+ \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^{-1})^\top\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} + \mathbf{C}^\top\mathbf{C}.\end{aligned}$$

At this point, we observe that the transpose and the inverse of a matrix can be interchanged. Moreover, since $\boldsymbol{\Sigma}$ is symmetric, the expression can be further simplified.

Finally, we substitute $\mathbf{C}^\top\mathbf{C}$ with its definition given in (2.18).

$$\begin{aligned}\tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} &= \mathbf{X}^\top\mathbf{X} - \mathbf{I}\,\mathrm{diag}\{\mathbf{s}\} - \mathrm{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^\top)^{-1}\boldsymbol{\Sigma} + \mathrm{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^\top)^{-1}\mathbf{I}\,\mathrm{diag}\{\mathbf{s}\} + \mathbf{C}^\top\mathbf{C} \\ &= \mathbf{X}^\top\mathbf{X} - \mathrm{diag}\{\mathbf{s}\} - \mathrm{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma} + \mathrm{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\mathrm{diag}\{\mathbf{s}\} + \mathbf{C}^\top\mathbf{C} \\ &= \mathbf{X}^\top\mathbf{X} - 2\mathrm{diag}\{\mathbf{s}\} + \mathrm{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\mathrm{diag}\{\mathbf{s}\} + \mathbf{C}^\top\mathbf{C} \\ &= \mathbf{X}^\top\mathbf{X} = \boldsymbol{\Sigma}.\end{aligned}$$

Therefore, we have shown that the correlation structure of the knockoffs is identical to the one among original features.

*Proof of constraint* (2.15)
We want to show that $\mathbf{X}^\top\tilde{\mathbf{X}} = \boldsymbol{\Sigma} - \mathrm{diag}\{\mathbf{s}\}$. In this case the proof is straightforward:

$$\begin{aligned}\mathbf{X}^\top\tilde{\mathbf{X}} &= \mathbf{X}^\top(\mathbf{X}(\mathbf{I} - \boldsymbol{\Sigma}^{-1}\mathrm{diag}\{\mathbf{s}\}) + \tilde{\mathbf{U}}\mathbf{C}) = \mathbf{X}^\top(\mathbf{X} - \mathbf{X}\boldsymbol{\Sigma}^{-1}\mathrm{diag}\{\mathbf{s}\} + \tilde{\mathbf{U}}\mathbf{C}) \\ &= \mathbf{X}^\top\mathbf{X} - \mathbf{X}^\top\mathbf{X}\boldsymbol{\Sigma}^{-1}\mathrm{diag}\{\mathbf{s}\} + \mathbf{X}^\top\tilde{\mathbf{U}}\mathbf{C} = \boldsymbol{\Sigma} - \mathrm{diag}\{\mathbf{s}\}.\end{aligned}$$

In doing so, we have verified that the cross-correlation structure between the knockoffs and the original variables is identical to that within the knockoffs alone or the originals alone, except for the correlation between each knockoff and its corresponding original feature.

### A.1.4 Positive semidefinitedness and symmetry

The following is a brief definition of positive semidefinite matrix, a property which is crucial in constructing knockoff variables. In general, for a matrix $\mathbf{A}$, being positive semidefinite can equivalently be written in the following ways:

$$\mathbf{A} \succeq 0 \iff \mathbf{x}^\top\mathbf{A}\mathbf{x} \geq 0 \iff \langle\mathbf{A}\mathbf{x}, \mathbf{x}\rangle \geq 0 \iff \|\mathbf{A}\mathbf{x}\|_2 \cdot \|\mathbf{x}\|_2 \cdot \cos(\theta) \geq 0, \quad \text{(A.6)}$$

namely, the dot product between $\mathbf{x}$ and its transformation through $\mathbf{A}$ is greater or equal than 0. In other terms, the matrix $\mathbf{A}$ transforms each vector $\mathbf{x}$ into a new vector $\mathbf{A}\mathbf{x}$ having an angle less than 90° with $\mathbf{x}$. It can equivalently be shown that this coincides with the matrix $\mathbf{A}$ having all eigenvalues larger than or equal to 0, which means that in the vector space with the spectral basis, the matrix $A$ is a transformation that applies on each axis a positive scaling that is exactly equal to

the corresponding eigenvalue.
A more straightforward property a matrix can have is symmetry. In general, a matrix $\mathbf{A}$ is said to be symmetric if

$$\mathbf{A} = \mathbf{A}^\top$$

In the case of knockoffs, we are both interested in showing that $2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\}$ is a symmetric matrix and that the condition $2\boldsymbol{\Sigma} \succeq \text{diag}\{\mathbf{s}\}$ is equivalent to require that the matrix $2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\}$ is positive semidefinite. These requirements are essential to show that this matrix can be decomposed using the Cholesky algorithm, leading as a consequence to the definition of $\mathbf{C}$.
*Symmetry proof*

$$\begin{aligned}(2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\})^\top &= 2\text{diag}\{\mathbf{s}\}^\top - \text{diag}\{\mathbf{s}\}^\top(\boldsymbol{\Sigma}^{-1})^\top\text{diag}\{\mathbf{s}\}^\top \\ &= 2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}(\boldsymbol{\Sigma}^\top)^{-1}\text{diag}\{\mathbf{s}\} \\ &= 2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\}.\end{aligned}$$

This follows from the matrices $\text{diag}\{\mathbf{s}\}$ and $\boldsymbol{\Sigma}$ being symmetric. The second result consists of showing that the condition $2\boldsymbol{\Sigma} \succeq \text{diag}\{\mathbf{s}\}$ coincides with requiring that the matrix $2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\}$ is positive semidefinite.

*Positive semidefinite proposition equivalence*

$$\begin{aligned}2\text{diag}\{\mathbf{s}\} - \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} &\succeq 0 \\ 2\text{diag}\{\mathbf{s}\} &\succeq \text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} \\ 2\text{diag}\{\mathbf{s}\}^{-1}\text{diag}\{\mathbf{s}\} &\succeq \text{diag}\{\mathbf{s}\}^{-1}\text{diag}\{\mathbf{s}\}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} \\ 2\mathbf{I} &\succeq \mathbf{I}\boldsymbol{\Sigma}^{-1}\text{diag}\{\mathbf{s}\} \\ 2\boldsymbol{\Sigma} &\succeq \text{diag}\{\mathbf{s}\}.\end{aligned}$$

In this case, the equivalence of the two propositions is demonstrated by multiplying both sides by $\text{diag}\{\mathbf{s}\}^{-1}$ and then simplifying the expression accordingly.

### A.1.5 Test statistic properties

To work properly in combination with the knockoff framework, test statistics $W_j$'s have to satisfy the sufficiency and the antisymmetry property. As already mentioned,

these two properties are in place to achieve the exchangeability result in A.1, which is a crucial result to prove the FDR control of the knockoffs.

(*Sufficiency*) The statistic $\mathbf{W} = (W_1, \dots, W_p)$ is said to obey the sufficiency property if $\mathbf{W}$ depends only on the Gram matrix associated with the column-wise concatenation of the matrix of knockoff and original variables and on the feature-response inner product. More precisely, we have that:

$$\mathbf{W} = f([\mathbf{X}\ \tilde{\mathbf{X}}]^\top [\mathbf{X}\ \tilde{\mathbf{X}}], [\mathbf{X}\ \tilde{\mathbf{X}}]^\top \mathbf{y}), \tag{A.7}$$

in this case, $f : S_{2p}^+ \times \mathbb{R}^{2p} \to \mathbb{R}^p$, where $S_{2p}^+$ is the set of $2p \times 2p$ positive semidefinite matrices, containing for example $[\mathbf{X}\ \tilde{\mathbf{X}}]^\top [\mathbf{X}\ \tilde{\mathbf{X}}]$, while $\mathbb{R}^{2p}$ is the vector space of $2p$-dimensional real vector such as $[\mathbf{X}\ \tilde{\mathbf{X}}]^\top \mathbf{y}$. Therefore, the domain of $f$ is the Cartesian product of the spaces $S_{2p}^+$ and $\mathbb{R}^{2p}$. Finally, since we need $p$ test statistics to perform the knockoff procedure, the codomain of the vector function $\mathbf{W}$ is exactly $\mathbb{R}^p$. This first property is called sufficiency because under Gaussian noise $\mathbf{X}^\top \mathbf{y}$ is a sufficient statistic for $\boldsymbol{\beta}$. In general, given a parametric statistical model as in (1.1), a statistic $T(\mathbf{X})$ is said to be sufficient if the distribution of the data $\mathbf{X}$ given the statistic does not depend on the model parameters. Intuitively, this means that the statistic carries all the information needed to estimate the parameter of the statistical model of the data.

(Antisymmetry property) The statistic $\mathbf{W}$ is said to obey the antisymmetry property if swapping $\mathbf{X}_j$ and $\tilde{\mathbf{X}}_j$ has the effect of switching the signs of $W_j$?s; which means that for any subset of indices $S \subseteq \{1, \dots, p\}$, we can write:

$$W_j([\mathbf{X}\ \tilde{\mathbf{X}}]_{\text{swap}(S)}, \mathbf{y}) = W_j([\mathbf{X}\ \tilde{\mathbf{X}}], \mathbf{y}) \cdot \begin{cases} +1, & j \notin S \\ -1, & j \in S. \end{cases} \tag{A.8}$$

Here, by writing $[\mathbf{X}\ \tilde{\mathbf{X}}]_{\text{swap}(S)}$, we mean that the columns $\mathbf{X}_j$ and $\tilde{\mathbf{X}}_j$ have been swapped in the matrix $[\mathbf{X}\ \tilde{\mathbf{X}}]$ for each index $j \in S$. It can be proved that Lasso statistics obey these two properties.

## A.2 Exchangeability Lemma

The following is the Lemma about the exchangeability result that is crucial to prove both the formula of the knockoff estimate of FDP and the main FDR controlling result of the knockoff+. It will not be proved in this appendix.

**Lemma A.1.** *(i.i.d. signs for the nulls, Barber & Candès (2015a))*
*Let* $\boldsymbol{\varepsilon} \in \{\pm 1\}^p$ *be a sign sequence independent of* $\mathbf{W} = \{W_1, \dots, W_p\}$*, with* $\varepsilon_j = +1$ *for all* $j$ *such that* $X_j$ *truly belongs to the model and* $\varepsilon_j \overset{i.i.d}{\sim} \{\pm 1\}$ *for all* $j$ *such that* $X_j$ *does not belong to the true model. Then*

$$(W_1, \dots, W_p) \overset{d}{=} (W_1 \cdot \varepsilon_1, \dots, W_p \cdot \varepsilon_p).$$

In this case $\overset{d}{=}$ means equality in distribution, namely, as $n$ grows, these two vectors will tend to have the same cumulative distribution function. Therefore, the Lemma A.1 is saying that the signs of the statistics $W_j$'s are i.i.d. random for the true null hypotheses and additionally they are independent from the magnitudes of $|W_j|$ for all $j$, and from $\text{sign}(W_j)$ for the true alternatives $j$.
In order for generic test statistics $W_j$ to be compatible with the knockoff method, they must satisfy both the sufficiency and the antisymmetry property; these two properties together with knockoff construction are in place to achieve the crucial result stated in Lemma A.1 which allows us to explain why the estimated FDP has the expression in (2.21). In other terms, we can only use test statistics which satisfy both (A.7) and (A.8) because these two properties lead to satisfying Lemma A.1 on which is based the definition of FDP that is crucial in defining the knockoff data-dependent threshold $T$ (2.21). Therefore, we can informally say that this Lemma asserts that the signs of $W_j$'s are i.i.d. random for all true null hypotheses (those for which $\beta_j$ is truly 0). Consequently, we have:

$$\#\{j : \beta_j = 0 \text{ and } W_j \geq t\} \overset{d}{=} \#\{j : \beta_j = 0 \text{ and } W_j \leq -t\}, \tag{A.9}$$

Intuitively, this Lemma states that for the true null hypotheses, the number of false discoveries (on the left) and the number of true negatives (on the right) as $n$ grows tend to have the same CDF. Hence, we can estimate the false discovery proportion

at the threshold t as

$$\begin{aligned}\frac{\#\{j : \beta_j = 0 \text{ and } W_j \geq t\}}{\#\{j : W_j \geq t\} \vee 1} \approx & \frac{\#\{j : \beta_j = 0 \text{ and } W_j \leq -t\}}{\#\{j : W_j \geq t\} \vee 1} \\ \leq & \frac{\#\{j : W_j \leq -t\}}{\#\{j : W_j \geq t\} \vee 1} =: \widehat{\text{FDP}}(t),\end{aligned} \tag{A.10}$$

where the first quantity is exactly the ratio between the number of false discoveries and the number of total discoveries, given that we reject $H_{0j}$ for high positive values of $W_j$. However, since the numerator of the first ratio is unobservable, we can approximate it with the number of true nulls such that $W_j \leq -t$ using the result in A.9, while the denominator remains unchanged. Lastly, since we cannot observe which $H_{0j}$'s are true nulls, we simply remove the constraint $\beta_j = 0$ and we consider the number of hypotheses such that $W_j \leq -t$. In doing so we have defined the knockoff estimate of the false discovery proportion.

## A.3 Essential Concepts in Martingale Theory

This section does not claim to be comprehensive. Its sole purpose is to provide the essential background needed to follow the proof of the FDR control theorem for the knockoffs presented in Barber & Candès (2015a); The material of this section is primarily based on Chapter V of Çinlar (2011) and section 34 of Billingsley (1995). Moreover, appropriate excursuses will be included to bridge the knockoff framework with the newly introduced concepts on Martingale Theory.

### A.3.1 Filtration and Stopping Times

Consider a probability space $(\Omega, \mathcal{H}, \mathbb{P})$ and an index set $\mathbb{T}$. We can then define the associated stochastic process $\{X_t\}_{t\in\mathbb{T}}$, where each $X_t$ is a measurable function from the probability space to $(S_{X_t}, \mathcal{B})$, with $S_{X_t}$ denoting the support of $X_t$ and $\mathcal{B}$ the Borel $\sigma$-algebra on the support of the random variable. A filtration on $\mathbb{T}$ is an increasing family of sub-$\sigma$-algebras of $\mathcal{H}$ indexed by $\mathbb{T}$. Namely, $\mathcal{F} = (\mathcal{F}_t)_{t\in\mathbb{T}}$ is a filtration if each $\mathcal{F}_t$ is a $\sigma$-algebra on $\Omega$, $\mathcal{F}_t$ is a subset of $\mathcal{H}$, and $\mathcal{F}_s \subset \mathcal{F}_t$, whenever $s < t$. From an intuitive viewpoint, we can think of a filtration $\mathcal{F}$ as the flow of information, with $\mathcal{F}_t$ representing the body of information accumulated up to time $t$. Furthermore, to prove the FDR control property of Knockoff+, we must also introduce the concept of stopping time for a stochastic process.

**Definition A.1.** (Stopping Time) Let $\mathcal{F}$ be a filtration on the index set $\mathbb{T}$. A random time $T : \Omega \to \mathbb{T} \cup \{+\infty\}$ is called a stopping time of $\mathcal{F}$ if

$$\{T \leq t\} \in \mathcal{F}_t \qquad \text{for each} \quad t \in \mathbb{T}. \tag{A.11}$$

$T$ is a random variable whose realizations determine when the associated stochastic process is stopped. In this regard, requiring that the event $\{T \leq t\}$ belong to $\mathcal{F}_t$ means that the decision to stop the process must depend solely on the information available up to time $t$, as captured by the filtration, and not on future information. In this sense, the knockoff+ data-dependent threshold $T$ defined in (2.22) can be interpreted as a stopping time for the super-martingale in (2.29). This holds true because $T$ corresponds to the first time the knockoff estimate of FDP falls below q. Consequently, since FDP is controlled by the super-martingale in (2.29) and the stopping decision is based only the past, we have that $\{T \leq t\}$ belongs to the filtration generated by the super-martingale $V^+(T)/(1 + V^-(T))$. Finally, this interpretation allows us to apply the Optional Stopping Time Theorem to the supermartingale $V^+(T)/(1 + V^-(T))$ using $T$ as the stopping time. At this point, before proceeding, we need to introduce the concepts of supermartingales, of conditional expectation -which is essential in defining martingales- and the Optional Stopping Theorem.

### A.3.2 Conditional Expectation and Supermartingales

The concept of conditional expectation is extremely relevant in Martingale theory; at the same time, it is profoundly different from the standard notion of expectation. For this reason, it is necessary to provide an overview of its construction. The following section is based on Section 34 "conditional expectation" of the book Probability and Measure by Billingsley (1995).

We start with a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, and a random variable $X : \Omega \to \mathbb{R}^n$, with finite expectation. Then, we consider a sub $\sigma$-algebra $\mathcal{H} \subset \mathcal{F}$ that could be thought of as the partial information we know about $X$. Clearly, $\mathcal{F}$ represents the full information that, unfortunately, we do not possess. We can now observe that since $\mathcal{H}$ is smaller than $\mathcal{F}$, the random variable $X$ might not be measurable with respect of $\mathcal{H}$, namely, there could exist Borel sets whose preimages through $X^{-1}(\cdot)$ do not belong to $\mathcal{H}$. Consequently, the idea is that we cannot fully observe $X$ based only on $\mathcal{H}$. Despite that, fixing an event $H \in \mathcal{H}$, we might be interested in computing

the expected value of $X$ over $H$ given the available information in $\mathcal{H}$

$$\int_H X d\mathbb{P}|_{\mathcal{H}}, \tag{A.12}$$

where $\mathbb{P}|_{\mathcal{H}}$ is the restriction of $\mathbb{P}$ to $\mathcal{H}$; however, as explained earlier, the quantity in (A.12) cannot be defined in general since $X$ is not necessarily $\mathcal{H}$-measurable, which is a crucial request in the definition of the Lebesgue integral. Therefore, to handle this situation, we introduce the so-called conditional expectation $\mathbb{E}[X|\mathcal{H}]$, which is any $\mathcal{H}$-measurable random variable that satisfies the following equality:

$$\int_H \mathbb{E}[X|\mathcal{H}] d\mathbb{P} = \int_H X d\mathbb{P} \quad \text{for each } H \in \mathcal{H}. \tag{A.13}$$

Namely, the conditional expectation is that specific random variable that, over each event $H$ of the known $\sigma$-algebra $\mathcal{H}$, has the same expected variable of $X$; in other terms, it is a new random variable that depends only on information in $\mathcal{H}$, and it approximates $X$ as well as possible based on that information. Clearly, this is not an operational definition of $\mathbb{E}[X|\mathcal{H}]$. In this regard, it can be proved that the conditional expectation is unique up to sets with null measure. Moreover, an explicit formulation is generally difficult to define due to the various structures the conditioning $\sigma$-algebra might take. However, for the purposes of the knockoff proof, these details are secondary. In fact, to understand the following proofs, having just an intuition of what a conditional expectation is will be enough.

**Definition A.2.** (Supermartingale) Let $X = \{X_t\}_{t \in T}$ be a real-valued stochastic process on the probability space $(\Omega, \mathcal{H}, \mathbb{P})$, and $\mathcal{F} = \{\mathcal{F}_t\}_{t \in T}$ a filtration. If these conditions are satisfied

1. $X$ is adapted to $\mathcal{F}$, namely, $X_t$ is $\mathcal{F}_t$-measurable for every $t \in \mathbb{T}$,
2. each $X_t$ is integrable, meaning that $\mathbb{E}[|X_t|] < +\infty$ ,
3. $\mathbb{E}[X_t|\mathcal{F}_s] \leq X_s \qquad$ for all $t \geq s$,

then, $X$ is called a supermartingale.

It is worth noting that $\mathbb{E}[X_t|\mathcal{F}_s]$ is not a standard expected value, instead, it is exactly the above-mentioned conditional expectation of $X_t$ with respect to the $\sigma$-algebra $\mathcal{F}_s$. Moreover, the $\leq$ sign expresses a stochastic ordering between the conditional expectation of $X_t$ and $X_s$. Therefore, a super-martingale is a stochastic process that,

given all information available up to each time $s$, summarized by the filtration $\mathcal{F}_s$, does not increase in expectation over time.

### A.3.3 Optional Stopping Time Theorem

**Theorem A.1.** *(Optional Stopping Theorem, Doob Joseph.L 1949) Let $X = \{X_t\}_{t\in\mathbb{T}}$ be a supermartingale, and $T$ a stopping time, both with respect to the filtration $\{\mathcal{F}_t\}_{t\in\mathbb{T}}$. Assuming that one of the following three conditions holds:*

1. *the stopping time is almost surely bounded, namely, there exists $c \in \mathbb{R}$ such that $T < c$ a.s.*

2. *The stopping time $T$ has finite expectation $\mathbb{E}[T] < +\infty$, and the conditional expectations of the absolute value of the martingale increments are almost surely bounded, namely, $\mathbb{E}(|X_{t+1} - X_t| \ |\mathcal{F}_t) \leq c$ for all $t$.*

3. *There exists a constant $c$ such that $|X_{\min(t,T)}| \leq c$ almost everywhere for all $t \in \mathbb{T}$.*

*Then:*

$$\mathbb{E}[X_T] \leq \mathbb{E}[X_0] \tag{A.14}$$

## A.4 Knockoff+ FDR-control Proof details

Throughout the proof, a variant of the stochastic process $\{V^+(t)\}_{t\in\mathcal{W}}$ will be heavily used; therefore, it is crucial to define it carefully. The quantity $\{V^+(t)\}_{t\in\mathcal{W}}$ is a collection of functions depending on $t \in \mathcal{W}$ and $\omega \in \Omega$ defined between the following spaces $\Omega \times \mathcal{W} \longrightarrow S^{\mathcal{W}}$, where $\Omega$ denotes the sample space and $\mathcal{W}$ the index set composed by the ordered test statistics $W_j$'s taken in absolute value; while $\Omega \times \mathcal{W}$, called the state-space, represents their cartesian product and it is the set of all possible realizations of the stochastic process. For a fixed threshold $t$ and varying data points $\omega \in \Omega$, we obtain a random variable $V^+(t)$ representing the number of false discoveries. On the other side, for a fixed realization $\omega$ of the data, we can define a deterministic trajectory over $\mathcal{W}$ that represents the number of false discoveries as the threshold $t$ varies. The same logic can be applied to define the stochastic process $\{V^-(t)\}_{t\in\mathcal{W}}$ associated with the number of true negatives $V^-(t)$.

### A.4.1 Supermartingale Proof

The proof that shows why $V^+(T)/(1+V^-(T))$ can be considered a super-martingale has been taken from the supplementary materials (Barber & Candès, 2015b) of the article (Barber & Candès, 2015a). To make the proof easier, the authors have preferred to redefine the core quantities $V^+(t)$ and $V^-(t)$ based on p-values instead of test statistics $W_j$'s. In fact, as mentioned several times, performing multiple testing using properly defined p-values is equivalent to using the respective test statistics. Hence, the random variable analogous to $V^+(t) = \{j : \beta_j = 0 \text{ and } W_j \geq t\}$ can be defined as follow:

$$V^+(k) = \#\{\text{null j} : 1 \leq j \leq k,\ p_j \leq c\}, \tag{A.15}$$

which is exactly the random variable determining the number of false positives through the knockoff+ method that uses $p$-values instead of test statistics to perform tests. Analogously, the random variable representing true negatives can be rewritten using $p$-values as follows:

$$V^-(k) = \#\{\text{null j} : 1 \leq j \leq k,\ p_j \geq c\} \tag{A.16}$$

Therefore, since the rejection region for each test is determined by the condition $W_j \geq T$, with $T$ the data-dependent threshold, the corresponding $p$-values are defined as follows:

$$p_j = \sup_{\beta_j \in \Theta_{0j}} \mathbb{P}(W_j \geq T) \qquad \text{for } j = 1, \ldots, p; \tag{A.17}$$

It turns out that defining each test statistic $W_j$ as in Step 3 of the knockoff construction leads to a highly complex distribution $\mathbb{P}$, which makes p-values impossible to compute. However, this complexity poses no issue, because knowing the explicit form of $\mathbb{P}$ is completely unnecessary for proving the next Lemma. On the other side, $c$ is defined as the common $p$-value threshold determined by the knockoff procedure such that:

$$T = F_t^{-1}(c) \tag{A.18}$$

namely, $c$ is the probability value that, if plugged into the quantile function of $t$, which is $F_t^{-1}(\cdot)$, would produce $T$. Obviously, as with the definition of $p_j$'s in (A.17), this formula aims only to give an intuition of the quantities involved in the proof. In practice, we are not interested in finding $T$ through its quantile function, nor determining the form of $F_t^{-1}(\cdot)$ that is irrelevant to the proofs of the article. Thus, having defined $V^{\pm}(k)$, we can also determine their corresponding stochastic processes.

In this regard, the index set of the stochastic processes $V^{\pm}(t)$ will no longer be the set $\mathcal{W}$ of test statistics in absolute value, but the set $K = \{1, \ldots, m\}$ which represents the indexes of the hypotheses $H_{0j}$'s being tested. On top of that, throughout this proof, we will assume that the p-values associated with true nulls are independent and identically distributed, and they satisfy $p_j \geq U(0,1)$; specifically, $p_j \geq U(0,1)$ means that p-values are stochastically greater than a continuous uniform distribution in $[0,1]$, and then $\mathbb{P}(p_j \leq t) \leq t$ for all $t \in [0,1]$. Intuitively, under true nulls, p-values are less likely to be small than a continuous uniform in $[0,1]$. The following is the Lemma that allows us to demonstrate that the quantity in (2.29) is a supermartingale.

**Lemma A.2.** *(Barber & Candès, 2015a) For $k = m, m-1, \ldots, 1, 0$, define $V^+(k) = \#\{null\ j : 1 \leq j \leq k,\ p_j \leq c\}$ and $V^-(k) = \#\{null\ j : 1 \leq j \leq k,\ p_j \geq c\}$, with the convention that $V^{\pm}(0) = 0$. Let $\mathcal{F}_k$ be the filtration defined by knowing all non-null p-values, as well as, $V^{\pm}(k')$ for all $k' \geq k$. Then the process*

$$M(k) = \frac{V^+(k)}{1 + V^-(k)} \tag{A.19}$$

*is a super-martingale running backward in time with respect to $\mathcal{F}_k$.*

In this case, $V^+(k)$ represents the number of false discoveries, namely null hypotheses for which $p_j \leq c$, while $V^-(k)$ denotes the number of true negatives. These two quantities are the same random variables as those defined in (2.26) and (2.28), respectively, but in this proof they are defined in terms of p-values. Additionally, $c$ is the p-value threshold determined by the Knockoff filter, which plays a role analogous to the data-dependent threshold $T$. It is worth noting that the martingale is considered to run backward in time, from $k = m$ to $k = 0$. Furthermore, the hypotheses have been relabeled in ascending order according to p-values.
We observe that the filtration $\mathcal{F}_k$, which is the family of $\sigma$-algebras defined for each index up to $k$, informs us about whether the $k^{\text{th}}$ hypothesis is a true null or not; whereas, the non-nulls are known exactly from the assumption of the lemma A.2. Therefore, we have that if $k$ is non-null (true alternative), then $M(k-1) = M(k)$; this follows from having the $k^{\text{th}}$ null hypothesis that is false, and hence neither the number of false discoveries nor the number of true negatives is affected by a shift from $k-1$ to $k$. On the other hand, if $k$ is null, then

$$M(k-1) = \frac{V^+(k) - I}{1 + V^-(k) - (1 - I)} = \frac{V^+(k) - I}{(V^-(k) + I) \vee 1}, \qquad \text{where } I = \mathbb{1}_{p_k \leq c}$$

Specifically, $I = 1$ if the $k^{\text{th}}$ null hypothesis is rejected ($p_j \leq c$), and $I = 0$ otherwise. Since we are assuming that $H_{0k}$ is a true null, $I = 1$ corresponds to a false discovery, while $I = 0$ indicates a true negative for the $k^{\text{th}}$ test. Now, because we are analyzing the process in reverse (moving from $k$ to $k-1$), the quantity $M(k-1)$ reflects this shift. Indeed, if a false discovery occurred at step $k$, the numerator of $M(k-1)$, being the number of false discoveries at $k-1$, becomes $V^+(k)-1$ (one less false discovery), while the denominator remains unchanged at $V^-(k)-0$, since no true negatives are made in the $k^{\text{th}}$ test that need to be removed. Conversely, if the $k^{\text{th}}$ hypothesis was not rejected, resulting in a true negative, the numerator of $M(k-1)$ remains $V^+(k)$, while the denominator becomes $V^-(k)-1$, accounting for the removal of one true negative.

Additionally, we can observe that the filtration $\mathcal{F}_k$ gives no further knowledge about $I$. Consequently, by the exchangeability property of the true nulls (A.9), we have that

$$\mathbb{P}\{I = 1\} = \frac{V^+(k)}{V^+(k) + V^-(k)}, \tag{A.20}$$

namely, the probability of having a false discovery in the $k^{\text{th}}$ test is the ratio between the number of false discoveries and the total number of true nulls. Conversely, using the same exchangeability property, we can also define the complementary probability

$$\mathbb{P}\{I = 0\} = \frac{V^-(k)}{V^-(k) + V^+(k)}, \tag{A.21}$$

which is the ratio between the number of true negatives and the total number of discoveries. As a consequence, in the case where $k$ is null, we can write:

$$\begin{aligned}
\mathbb{E}[M(k-1) \mid \mathcal{F}_k] &= M(k-1|I=0) \cdot \mathbb{P}(I=0) + M(k-1|I=1) \cdot \mathbb{P}(I=1) \\
&= \frac{V^+(k)-0}{(V^-(k)+0) \vee 1} + \frac{V^-(k)}{V^+(k)+V^-(k)} \\
&+ \frac{V^+(k)-1}{V^-(k)+1} \cdot \frac{V^+(k)}{V^+(k)+V^-(k)} \\
&= \frac{1}{V^+(k)+V^-(k)} \left[ V^-(k) \cdot \frac{V^+}{V^-(k) \vee 1} + V^+(k) \cdot \frac{V^+(k)-1}{V^-(k)+1} \right];
\end{aligned} \tag{A.22}$$

where the first equality derives from having expanded the expected value of $M(k-1)$ conditioned on the filtration $\mathcal{F}_k$. The second step follows from having explicitly written the probabilities in (A.21) and in (A.20) and having replaced $I$ with its

values. Finally, the third equality follows from factoring out $\frac{1}{V^+(k)+V^-(k)}$ from the expression in the second step. At this point, through a basic simplification of the last expression in (A.22), we can write:

$$\mathbb{E}[M(k-1)|\mathcal{F}_k] = \begin{cases} \dfrac{V^+(k)}{1+V^-(k)}, & V^-(k) > 0 \\ V^+(k) - 1, & V^-(k) = 0. \end{cases} \tag{A.23}$$

As a result, we have that

$$\mathbb{E}[M(k-1)|\mathcal{F}_k] = \begin{cases} M(k), & \text{k non null,} \\ M(k), & \text{k null and } V^-(k) > 0, \\ M(k) - 1, & \text{k null and } V^-(k) = 0. \end{cases} \tag{A.24}$$

The first case follows from the fact that if $H_{0k}$ is a false null, then by definition, no false discoveries or true negatives can occur, and therefore $M(k-1) = M(k)$. The second case corresponds to the compact notation of the first expression in (A.23). Finally, the third case arises from noting that the second expression in (A.23) can equivalently be obtained by setting $V^+(k) = 0$ in $M(k) - 1$.
Consequently, through (A.24) we can show that

$$\mathbb{E}[M(k-1)|\mathcal{F}_k] \leq M(k), \tag{A.25}$$

which prove that $M(k)$ is a super-martingale, as stated in definition A.2, but running backward in time since $k-1$ comes after $k$.

### A.4.2 Binomial Property

The following is the property of Binomial random variables that, once applied to $V^+(0)$, allows us to prove the FDR control property of the Knockoff+ method.

**Proposition A.1.** *(Barber & Candès, 2015b) Given a Binomial random variable $Y \sim Bin(N, c)$, we have that:*

$$\mathbb{E}\left[\frac{Y}{1+N-Y}\right] \leq \frac{c}{1-c}.$$

In our context, the random variable considered is $V^+(0) \sim Bin(p_0, 1/2)$, namely, $Y = V^+(0)$, $N$ is equal to $p_0$ that is the number of true nulls, while $c = 1/2$; consequently, by proving proposition A.1, we are indirectly demonstrating the special

case

$$\mathbb{E}\left[\frac{V^+(0)}{1+p_0-V^+(0)}\right] \leq 1. \tag{A.26}$$

The proof is detailed in the following page and is structured as follows. The first equality follows from the observation that expanding the initial expected value with or without the addend $Y = 0$ yields the same result; therefore, conditioning on the indicator function $\mathbb{1}_{Y>0}$ leaves the expected value unchanged. In the second step, the expectation of $Y$ is expanded using the definition of the expected value for a transformation of a discrete random variable. The term corresponding to $Y = 0$ is omitted, as it is negligible in this context.

$$\begin{aligned}
\mathbb{E}\left[\frac{Y}{1+N-Y}\right] &= \mathbb{E}\left[\frac{Y}{1+N-Y}\cdot\mathbb{1}_{Y>0}\right] \\
&= \sum_{i=1}^{N}\mathbb{P}\{Y=i\}\cdot\frac{i}{1+N-i} \\
&= \sum_{i=1}^{N}c^i(1-c)^{N-i}\cdot\frac{N!}{i!(N-i)!}\cdot\frac{i}{1+N-i} \\
&= \sum_{i=1}^{N}c^i(1-c)^{N-i}\cdot\frac{N!}{\frac{i}{i}(i-1)!}\cdot\frac{1}{(N-i+1)(N-i)!} \\
&= \frac{c}{1-c}\cdot\sum_{i=1}^{N}c^{i-1}(1-c)^{N-i+1}\cdot\frac{N!}{(i-1)!(N-i+1)!} \\
&= \frac{c}{1-c}\cdot\sum_{i=1}^{N}\mathbb{P}\{Y=i-1\} = \frac{c}{1-c}\cdot\sum_{i=0}^{N-1}\mathbb{P}\{Y=i\} \\
&\leq \frac{c}{1-c}.
\end{aligned}$$

The third equality is obtained by replacing the general term $P\{Y = i\}$ with the probability mass function of a Binomial distribution with $N$ trials and success probability $c$. In the fourth step, the Binomial coefficient and the ratio of the transformation are rewritten to form a new unique Binomial coefficient $\binom{N}{i-1}$. In the fifth step, the entire expression is multiplied by $\frac{c}{1-c}\cdot\frac{1-c}{c} = 1$, thus preserving the value of the expectation. Specifically, the factor $\frac{c}{1-c}$ is left outside the summation, while the factor $\frac{1-c}{c}$ is incorporated in the sum and multiplied by the Binomial kernel. Therefore, in the sixth step, the resulting summation corresponds to the total probability mass of a Binomial distribution $\mathrm{Bin}(N, c)$ over a subset of its support.

This may not be immediately obvious due to the shift in the index by $-1$. Finally, in the sixth step, after adjusting the indices, it becomes clear that the summation covers the entire support of the binomial distribution except for the final value $N$. Therefore, the total sum is less than or equal to 1, which leads directly to the conclusion that the original expected value is bounded above by $\frac{c}{1-c}$.

## A.5 FDR and power approximation

In this last section, we have described the algorithms used to approximate via Monte Carlo the false discovery rate and the average power function of a generic Multiple testing procedure. Both the pseudo-algorithms 6 and 7 are extensions and adaptations of the Matlab code used in the simulation analysis proposed in Barber & Candès (2015a), which can be found at Knockoff Matlab Tutorial 2. It is worth mentioning that only in a simulation setting is it possible to estimate the FDR or the average power; this because in practice we never know the true status of hypotheses and we cannot compare it with empirical results. An important remark is that the order in which hypotheses are tested is irrelevant for the computation of the FDR or the power. The logic behind the approximation of FDR stems from its definition as the expected value of the false discovery proportion (FDP); therefore, it is natural to approximate it with the sample mean of FDPs. On the other hand, the rationale for the estimation of the average power function is less intuitive. In a multiple testing framework, the concept of power needs to be clarified since the definition given in (1.4) can no longer be applied. There exist different ways of defining the power in multiple comparison problems; for example one could use the FWER or some form of average of powers. In this work, the power for a specific parameter $\theta \in \Theta$ has been defined numerically as the average true positive rate over $M$ montecarlo iterations. As a result, by letting vary $\theta \in \Theta$ we can obtain a Monte Carlo approximation of the average power function. In Algorithm 6 is detailed the approximation of the average power function, whereas in Algorithm 7 is described the pseudo-code for the approximation of the false discovery rate, in both cases considering the signal amplitude on the x-axis. It is assumed the nullity of the parameters under the null hypothesis. Clearly, this algorithm provides a discrete and approximated version of the mean power function and FDR associated with the multiple testing procedure considered. Another important remark is that in the case of linear models, both algorithms are slightly different from the ones given in this section; this is because

one could consider a fixed design and redrawing only the noise and the response at each Monte Carlo iteration instead of the whole design matrix. For these details, the full R code is provided in Appendix B. However, in section 3.2.4, both approaches have been considered.

**Algorithm 6:** Average power Function Approximation in Multiple Testing via Monte Carlo

1. Fix a specific Multiple Testing approach.
2. Fix a number $m$ of total hypotheses to be tested, of which $m_0$ are true nulls and $m_1 = m - m_0$ are true alternatives. Generate the set $\mathcal{T}$ of true null indices and $\mathcal{F}$ of false null indices.
3. Fix a true significance level $\alpha \in [0, 1]$.
4. Fix the number $M$ of Monte Carlo iterations.
5. Fix a grid $\{\theta_1, \ldots, \theta_p\}$ of parameter values on which to evaluate the approximated power $\hat{\pi}(\theta_j)$ for all $j \in \{1, \ldots, p\}$.
6. **For** $j$ in $1 : p$ do:
   (a) Fix a specific $\theta = \theta_j$.
   (b) **For** $k$ in $1 : M$ do:
      i. **For** $\gamma$ in $1 : m$ do:
         A. Simulate data from $H_{1\gamma} : \theta = \theta_j$ if $\gamma \in \mathcal{F}$, or from $H_{0\gamma} : \theta = 0$ if $\gamma \in \mathcal{T}$.
         B. Compute the corresponding test statistic $T_\gamma$ for the hypothesis $H_{0\gamma}$.
         C. Reject or accept each $H_{0\gamma}$ based on the respective test statistic $T_\gamma$.
      ii. **End For**
      iii. Store the value of the $k^{\text{th}}$ true positive rate $\text{TPR}_k(\theta_j)$:

$$\text{TPR}_k(\theta_j) = \frac{\sum_{\gamma=1}^{m} \mathbb{1}(H_{0\gamma} \text{ is rejected and } \gamma \in \mathcal{F})}{m_1}$$

   .
   (c) **End For**
   (d) Compute $\hat{\pi}(\theta_j) = \dfrac{\sum_{k=1}^{M} \text{TPR}_k(\theta_j)}{M}$.
7. **End For**
8. Plot $\{\theta_1, \ldots, \theta_p\}$ vs $\{\hat{\pi}(\theta_1), \ldots, \hat{\pi}(\theta_p)\}$.

**Algorithm 7:** False Discovery Rate Approximation in Multiple Testing via Monte Carlo

1. Fix a specific Multiple Testing approach.
2. Fix a number $m$ of total hypotheses to be tested, of which $m_0$ are true nulls and $m_1 = m - m_0$ are true alternatives. Generate the set $\mathcal{T}$ of true null indices and $\mathcal{F}$ of false null indices.
3. Fix a true significance level $\alpha \in [0, 1]$.
4. Fix the number $M$ of Monte Carlo iterations.
5. Fix a grid $\{\theta_1, \ldots, \theta_p\}$ of parameter values on which to evaluate the approximated false discovery rate $\widehat{\mathrm{FDR}}(\theta_j)$ for all $j \in \{1, \ldots, p\}$.
6. **For** $j$ in $1 : p$ do:
   (a) Fix a specific $\theta = \theta_j$.
   (b) **For** $k$ in $1 : M$ do:
      i. **For** $\gamma$ in $1 : m$ do:
         A. Simulate data from $H_{1\gamma} : \theta = \theta_j$ if $\gamma \in \mathcal{F}$ or from $H_{0\gamma} : \theta = 0$ if $\gamma \in \mathcal{T}$.
         B. Compute the corresponding test statistic $T_\gamma$ for the hypothesis $H_{0\gamma}$.
         C. Reject or accept each $H_{0\gamma}$ based on the respective test statistic $T_\gamma$.
      ii. **End For**
      iii. Store the value of the $k^{\text{th}}$ false discovery proportion $\mathrm{FDP}_k(\theta_j)$:

$$\mathrm{FDP}_k(\theta_j) = \frac{\sum_{\gamma=1}^{m} \mathbb{1}(H_{0\gamma} \text{ is rejected and } \gamma \in \mathcal{T})}{\left\{ \sum_{\gamma=1}^{m} \mathbb{1}(H_{0\gamma} \text{ is rejected}) \right\} \vee 1} = \frac{\mathrm{FP}_k}{\max(\mathrm{R}_k, 1)}.$$

   (c) **End For**
   (d) Compute $\widehat{\mathrm{FDR}}(\theta_j) = \dfrac{\sum_{k=1}^{M} \mathrm{FDP}_k(\theta_j)}{M}$.
7. **End For**
8. Plot $\{\theta_1, \ldots, \theta_p\}$ vs $\{\widehat{\mathrm{FDR}}(\theta_1), \ldots, \widehat{\mathrm{FDR}}(\theta_p)\}$.

# Appendix B

# R code

## B.1 Chapter I

The following is the commented R code used to generate Figure 1.1, Figure 1.2 and Figure 1.5. Whereas, Fig. 1.3 and Fig. 1.4 have been generated by modifying the code for Figure 1.1.

```
rm(list = ls())
# Sample mean on the x axis
# arbitrarily chosen quantities to allow
# an effective representation
n <- 1000
sd <- 1
mu0 <- 0
mu1 <- 0.08
mi <- -0.15
ma <- 0.2
x <- seq(mi, ma, length.out = 10000)

# Values used to plot the densities under H_0 and H_1
y0 <- dnorm(x, mean = mu0, sd = sd / sqrt(n))
y1 <- dnorm(x, mean = mu1, sd = sd / sqrt(n))

# Critical threshold and probability of committing a first type error
alpha <- 0.2
ct <- mu0 + qnorm(1 - alpha) * (sd / sqrt(n))
```

```r
# plot of the two densities (Figure 1.1)

plot(x, y0,
  type = "l", lwd = 1, col = "black",
  xlim = c(mi, ma), ylim = c(0, max(y0, y1)),
  xlab = expression(bar(x)), ylab = "probability density",
  main = bquote("Sample mean PDFs" ~ alpha == 0.2)
)

points(x, y1, type = "l", lwd = 1, col = "black")
abline(h = 0)
segments(ct, 0, ct, max(dnorm(ct, mean = mu1, sd = sd / sqrt(n)),
↪  dnorm(ct, mean = mu0, sd = sd / sqrt(n))), lwd = 1.5)
x1 <- x[x <= ct]
x2 <- x[x >= ct]

# Probability of committing a type II error (False Negative)
polygon(c(x1, rev(x1)),
  c(dnorm(x1, mu1, sd / sqrt(n)), rep(0, length(x1))),
  density = 20, angle = 45, col = "blue", border = NA
)

# Probability of committing a type I error (False positive)
polygon(c(x2, rev(x2)),
  c(dnorm(x2, mu0, sd / sqrt(n)), rep(0, length(x2))),
  density = 40, angle = -45, col = "red", border = NA
)

# Probability of making a true discovery (True positive)
polygon(c(x2, rev(x2)),
  c(dnorm(x2, mu1, sd / sqrt(n)), rep(0, length(x2))),
  density = 10, angle = 30, col = "green", border = NA
)

segments(ct, 0, ma + 1, 0, col = "purple", lwd = 4)
segments(mu0, 0, mu0, dnorm(mu0, mu0, sd / sqrt(n)), col = "black", lwd =
↪  1, lty = "dashed")
```

```
segments(mu1, 0, mu1, dnorm(mu1, mu1, sd / sqrt(n)), col = "black", lwd =
↪  1, lty = "dashed")

text(-0.04, 11, labels = expression(H[0]))
text(0.12, 11, labels = expression(H[1]))
legend(-0.15, 12,
  fill = c("blue", "green", "red"), bty = "n",
  legend = c(expression(beta), expression(pi), expression(alpha)),
  cex = 1, density = c(30, 30, 30), angle = c(45, 30, -45)
)

text(0.19, 0.8, labels = c("R"), col = "purple")
segments(mi - 1, 0, ct, 0, col = "dodgerblue", lwd = 4)
text(-0.11, 0.8, labels = c("A"), col = "dodgerblue")

# Power function (Figure 1.2)
alpha <- 0.25

ct <- mu0 + qnorm(1 - alpha) * (sd / sqrt(n))

power <- function(mu0, mu, alpha, sd, n) {
  p <- numeric(length(mu))
  for (i in 1:length(mu)) {
    p[i] <- 1 - pnorm(((mu0 - mu[i]) / (sd / sqrt(n))) + qnorm(1 - alpha))
  }
  return(p)
}
mm <- seq(-0.5, 0.5, length.out = 2000)
pp <- power(mu0, mu = mm, alpha, sd, n)
pp
min <- -0.1
max <- 0.15
plot(mm, pp,
  type = "l", col = "black", lwd = 1.5, xlim = c(min, max),
  main = "Power function", xlab = expression(mu),
  ylab = expression(pi * "(" * mu * ")")
)
```

```
abline(v = mu0, lty = "dashed")
abline(v = ct, lty = "dashed")
abline(h = 0)
abline(h = 1)
segments(0, 0, 0.3, 0, lwd = 3, col = "darkorange")
segments(-0.3, 0, 0, 0, lwd = 3, col = "lightblue")
text(0.12, 0.07, labels = c(expression(H[1])), col = "darkorange", cex =
↪  1.2)
text(-0.087, 0.07, labels = c(expression(H[0])), col = "lightblue", cex =
↪  1.2)
text(-0.01, 0.8, labels = c(expression(mu[0])))
text(-0.095, 0.29, labels = c(expression(alpha)), col = "red2", cex = 1.2)
text(ct + 0.03, 0.45, labels = expression(mu[0] + z[1 - alpha] * sigma /
↪  sqrt(n)))
eps <- 0.009
segments(min - eps, 0, min - eps, alpha, lwd = 3, col = "red2")
segments(min - eps, alpha, 0, alpha, lty = "dashed")
segments(min - eps, alpha, min - eps, 1, lwd = 3, col = "green")

# Power comparison (Figure 1.5)

alpha <- 0.1
ct <- mu0 + qnorm(1 - alpha) * (sd / sqrt(n))

power <- function(mu0, mu, alpha, sd, n) {
  p <- numeric(length(mu))
  for (i in 1:length(mu)) {
    p[i] <- 1 - pnorm(((mu0 - mu[i]) / (sd / sqrt(n))) + qnorm(1 - alpha))
  }
  return(p)
}
mm <- seq(-0.5, 0.5, length.out = 2000)
pp <- power(mu0, mu = mm, alpha, sd, n)
pp
min <- -0.1
max <- 0.15
plot(mm, pp,
```

```
  type = "l", col = "black", lwd = 1.5, xlim = c(min, max),
  main = "Power function comparison", xlab = expression(mu),
  ylab = expression(pi * "(" * mu * ")")
)
alpha1 <- 0.01
pp1 <- power(mu0, mu = mm, alpha1, sd, n)
points(mm, pp1, col = "red", type = "l", lwd = 1.5)
alpha2 <- 0.5
pp2 <- power(mu0, mu = mm, alpha2, sd, n)
points(mm, pp2, col = "turquoise", type = "l", lwd = 1.5)
legend(
  x = -0.10, y = 0.8,
  legend = c(
    expression(alpha == 0.5),
    expression(alpha == 0.1),
    expression(alpha == 0.01)
  ),
  col = c("turquoise", "black", "red"),
  lty = 1,
  lwd = 1.5,
  bty = "n",
  cex = 0.8
)
abline(v = mu0, lty = "dashed")
abline(h = 0)
abline(h = 1)
segments(0, 0, 0.3, 0, lwd = 3, col = "darkorange")
segments(-0.3, 0, 0, 0, lwd = 3, col = "lightblue")
text(0.12, 0.07, labels = c(expression(H[1])), col = "darkorange", cex =
↪  1.2)
text(-0.087, 0.07, labels = c(expression(H[0])), col = "lightblue", cex =
↪  1.2)
text(-0.01, 0.8, labels = c(expression(mu[0])))
```

## B.2 Chapter III

### B.2.1 p-values vs rank plot

The following is the commented code used to generate Figure 3.1 and Figure 3.2.

```
# MULTIPLE TESTING PROCEDURES COMPARISON
# p-value vs rank plot

rm(list = ls())
library(tidyverse)
set.seed(123)

m <- 1000 # number of total hypotheses tested
m0 <- 900 #  number of true null H0
m1 <- m - m0 # number of true alternatives H1

# tni = true null indexes
tni <- sample(1:m, size = m0, replace = FALSE)
# tai = true alternative indexes
tai <- c(1:m)[-tni]

# the order of the hypotheses tested is not important
# they are invariant with respect to the index j

# Simulation

# under true null H0j zj ~ N(0,1)
# under the alternative H1j zj ~ N(muj,1)
# the variance is fixed at 1
# We consider muj = mu (for all j in tai)

# zj is the test statistics we will use to compute the p-value

zj <- numeric(m)

# We will fill this vector zj drawing from a N(0,1) for the indexes tni
# and from a N(muj,1) for the indexes tai. In other words we skip the
```

```
# data generation phase and we directly generate test statistics.
# we are assuming a known variance

muj <- 3
sd <- 1
zj[tni] <- rnorm(m0, mean = 0, sd)
zj[tai] <- rnorm(m1, mean = muj, sd)

# two tailed test p-value
pval <- 2 * (1 - pnorm(abs(zj)))
alpha <- 0.05
pval_asc <- sort(pval, decreasing = FALSE)
q <- alpha

upper <- m # maximum index we want to represent on the x-axis
pval_naive <- pval_asc[1:upper]
ind <- 1:upper

indx <- numeric()
for (j in 1:upper) {
  indx[j] <- which(pval_asc[j] == pval)
}
indx
color <- numeric()

for (j in 1:upper) {
  if (indx[j] %in% tai) {
    color[j] <- 1
  } else {
    color[j] <- 0
  }
}

# Holm's threshold
thr_holm <- function(index) {
  q / (m - index + 1)
}
```

```
color <- factor(color)
levels(color) <- c("TRUE H0", "TRUE H1")
data <- data.frame(ind, pval_naive)
xmax <- 150

# p-values vs rank plot (Figure 3.1)
plot1 <- ggplot(data = data[1:xmax, ], mapping = aes(
  x = ind, y = pval_naive,
  color = color[1:xmax]
), xlab = "index j", ylab = "p-value") +
  theme_light() +
  geom_point(alpha = .4, size = 2.5) +
  scale_color_manual(values = c("red", "green")) +
  geom_abline(intercept = 0, slope = q / m, color = "blue") +
  geom_text(data = data[1, ], aes(x = 140, y = 0.009, label = "BHq"),
  ↪  color = "blue", size = 4) +
  geom_abline(intercept = q / m, slope = 0, colour = "deepskyblue") +
  geom_text(data = data[1, ], aes(x = 145, y = 0.002, label = "Bonf."),
  ↪  color = "deepskyblue", size = 4) +
  geom_abline(intercept = q, slope = 0, colour = "green") +
  geom_text(data = data[1, ], aes(x = 10, y = 0.052, label = "Naive"),
  ↪  color = "green", size = 4) +
  geom_function(fun = thr_holm, color = "red", lty = "dashed") +
  geom_text(data = data[1, ], aes(x = 120, y = 0.002, label = "Holm"),
  ↪  color = "red", size = 4) +
  labs(x = "index j", y = "p-values", color = NULL, title = "p-values vs
  ↪  rank: Procedures Comparison")
plot1

# same graph in logarithmic scale base = e

logpval <- log(pval_naive)
data <- data.frame(ind, logpval)

thr_naive <- function(index) {
  log(q)
}
```

```r
thr_bonf <- function(index) {
  log(q / m)
}
thr_holm <- function(index) {
  log(q / (m - index + 1))
}
thr_BHq <- function(index) {
  log((q * index) / m)
}
# Naive
ind_naive <- which(pval <= alpha)
(TOTP_naive <- (length(ind_naive)))
# Bonferroni
ind_bonf <- which(pval <= alpha / m)
(TOTP_bonf <- (length(ind_bonf)))
padj_bonf <- p.adjust(pval, method = "bonferroni")
ind_bonf_adj <- which(padj_bonf <= alpha)
# Holm
pval_asc <- sort(pval)
indices <- 1:m
i0 <- min(which(pval_asc > alpha / (m - indices + 1)))
ind_holm <- which(pval < alpha / (m - i0 + 1))
TOTP_holm <- length(ind_holm)
padj_holm <- p.adjust(pval, method = "holm")
ind_holm_adj <- which(padj_holm <= alpha)
# BHq
imax <- max(which(pval_asc <= (indices / m) * q))
ind_bhq <- which(pval <= (imax / m) * q)
TOTP_BH <- length(ind_bhq)
padj_bhq <- p.adjust(pval, method = "BH")
ind_bhq_adj <- which(padj_bhq <= alpha)

xmax <- 150
ln_breaks <- log(c(1, 0.1, 0.01, 0.001, 0.0001, 1e-5, 1e-6))
ln_labels <- c("1", "0.1", "0.01", "0.001", "1e-4", "1e-5", "1e-6")

# p-values vs rank plot (log-scale) (Figure 3.2)
```

```
plot2 <- ggplot(data = data[1:xmax, ], mapping = aes(
  x = ind, y = logpval,
  color = color[1:xmax]
), xlab = "index j", ylab = "p-value") +
  theme_light() +
  geom_point(alpha = .4, size = 2.5) +
  scale_color_manual(values = c("red", "green")) +
  geom_function(fun = thr_BHq, color = "blue") +
  geom_text(data = data[1, ], aes(x = xmax - 10, y = log((q * xmax) / m)
  ↪  - 0.5, label = "BHq"), color = "blue", size = 4) +
  geom_function(fun = thr_bonf, colour = "deepskyblue") +
  geom_text(data = data[1, ], aes(x = xmax - 10, y = log(q / m) - 0.5,
  ↪  label = "Bonf."), color = "deepskyblue", size = 4) +
  geom_function(fun = thr_naive, colour = "green") +
  geom_text(data = data[1, ], aes(x = 10, y = log(q) + 0.5, label =
  ↪  "Naive"), color = "green", size = 4) +
  geom_function(fun = thr_holm, color = "red", lty = "dashed") +
  geom_text(data = data[1, ], aes(x = xmax - 10, y = log(q / (m - xmax +
  ↪  1)) + 0.5, label = "Holm"), color = "red", size = 4) +
  geom_vline(xintercept = TOTP_BH, lty = "dotted", alpha = 0.25) +
  geom_vline(xintercept = TOTP_bonf, lty = "dotted", alpha = 0.25) +
  geom_vline(xintercept = TOTP_holm, lty = "dotted", alpha = 0.25) +
  geom_vline(xintercept = TOTP_naive, lty = "dotted", alpha = 0.25) +
  geom_text(data = data[1, ], aes(x = TOTP_bonf, y = -15, label =
  ↪  as.character(TOTP_bonf)), color = "red", size = 4) +
  geom_text(data = data[1, ], aes(x = TOTP_holm, y = -15, label =
  ↪  as.character(TOTP_holm)), color = "red", size = 4) +
  geom_text(data = data[1, ], aes(x = TOTP_BH, y = -15, label =
  ↪  as.character(TOTP_BH)), color = "red", size = 4) +
  geom_text(data = data[1, ], aes(x = TOTP_naive, y = -15, label =
  ↪  as.character(TOTP_naive)), color = "red", size = 4) +
  labs(x = "index j", y = "(p-value) log scale", color = NULL, title =
  ↪  "p-values vs rank: Procedures Comparison(log scale)") +
  scale_x_continuous(breaks = c(seq(0, xmax, by = 50)), minor_breaks =
  ↪  NULL, ) +
  scale_y_continuous(breaks = ln_breaks, labels = ln_labels)
```

```
plot2
```

### B.2.2 Microarray simulation study

The following is the code used to generate Figure 3.3 and Figure 3.4

```
# Microarray simulation study
# Power and FDR comparison using two-sample T tests
↪  (Naive,Bonferroni,Holm,Bhq)
# No knockoffs, no linear model setting, high-dimensional data p>n
rm(list = ls())
set.seed(321)
par(mfrow = c(1, 1))
par(pty = "m")
M <- 1000 # Montecarlo iterations
AA <- seq(-1.8, 1.8, length.out = 150) # signal magnitude
len <- length(AA)
pwnaive <- numeric(len)
pwbonf <- numeric(len)
pwholm <- numeric(len)
pwbhq <- numeric(len)
fdrnaive <- numeric(len)
fdrbonf <- numeric(len)
fdrholm <- numeric(len)
fdrbhq <- numeric(len)
alpha <- 0.2 # FDR upper bound
n <- 50 # number of observations
p <- 100 # number of features
tp <- 15 # number of true significant features
fp <- p - tp
n1 <- 25 # dimension of the first group
n2 <- 25 # dimension of the second group
sd <- 1 # shared variance
# we are assuming unknown variance equal among groups
# (in simulation we fix a standard deviation of sd=1)
```

```r
for (i in 1:len) {
  # cycle for different signal amplitudes
  A <- AA[i]
  pwnaive_iter <- numeric(M)
  pwbonf_iter <- numeric(M)
  pwholm_iter <- numeric(M)
  pwbhq_iter <- numeric(M)
  fdrnaive_iter <- numeric(M)
  fdrbonf_iter <- numeric(M)
  fdrholm_iter <- numeric(M)
  fdrbhq_iter <- numeric(M)

  for (m in 1:M) {
    # cycle for Montecarlo iterations
    mu <- numeric(p)
    ind_tp <- sample(1:p, size = tp, replace = FALSE)
    ind_fp <- c(1:p)[-ind_tp]
    mu[ind_tp] <- A
    mu[ind_fp] <- 0
    pval <- numeric(p)

    # two sample T test with equal variance are done independently

    for (j in 1:p) {
      # cycle to test all p hypotheses using a Two sample T statistics
      x1 <- rnorm(n1, mean = 0, sd = sd)
      x2 <- rnorm(n2, mean = mu[j], sd = sd)
      x1bar <- mean(x1)
      x2bar <- mean(x2)
      var1 <- var(x1)
      var2 <- var(x2)
      varPool <- ((n1 - 1) * var1 + (n2 - 1) * var2) / (n1 + n2 - 2)
      T_stat <- (x1bar - x2bar) / sqrt(varPool * (1 / n1 + 1 / n2))
      pval[j] <- 2 * (1 - pt(abs(T_stat), df = n1 + n2 - 2))
    }
    pval_asc <- sort(pval)
    indices <- c(1:p)
```

```
  # Naive
  ind_naive <- which(pval <= alpha)
  totp_naive <- length(ind_naive)
  tp_naive <- sum(ind_naive %in% ind_tp)
  fp_naive <- totp_naive - tp_naive
  # Bonferroni
  ind_bonf <- which(pval <= alpha / p)
  totp_bonf <- length(ind_bonf)
  tp_bonf <- sum(ind_bonf %in% ind_tp)
  fp_bonf <- totp_bonf - tp_bonf
  # Holm
  i0 <- min(which(pval_asc > alpha / (p - indices + 1)))
  ind_holm <- which(pval < alpha / (p - i0 + 1))
  totp_holm <- length(ind_holm)
  tp_holm <- sum(ind_holm %in% ind_tp)
  fp_holm <- totp_holm - tp_holm
  # BHq
  imax <- max(which(pval_asc <= (indices / p) * alpha))
  ind_bhq <- which(pval <= (imax / p) * alpha)
  totp_bhq <- length(ind_bhq)
  tp_bhq <- sum(ind_bhq %in% ind_tp)
  fp_bhq <- totp_bhq - tp_bhq

  pwnaive_iter[m] <- tp_naive / tp
  pwbonf_iter[m] <- tp_bonf / tp
  pwholm_iter[m] <- tp_holm / tp
  pwbhq_iter[m] <- tp_bhq / tp

  fdrnaive_iter[m] <- fp_naive / max(totp_naive, 1)
  fdrbonf_iter[m] <- fp_bonf / max(totp_bonf, 1)
  fdrholm_iter[m] <- fp_holm / max(totp_holm, 1)
  fdrbhq_iter[m] <- fp_bhq / max(totp_bhq, 1)
}
pwnaive[i] <- mean(pwnaive_iter)
pwbonf[i] <- mean(pwbonf_iter)
pwholm[i] <- mean(pwholm_iter)
pwbhq[i] <- mean(pwbhq_iter)
```

```
  fdrnaive[i] <- mean(fdrnaive_iter)
  fdrbonf[i] <- mean(fdrbonf_iter)
  fdrholm[i] <- mean(fdrholm_iter)
  fdrbhq[i] <- mean(fdrbhq_iter)
}

# Power comparison (Figure 3.3)
plot(AA, pwbhq,
  type = "l", col = "blue", main = "Power Comparison", ylab = "Power",
  xlab = "Signal Magnitude", lwd = 1.25, cex.axis = 0.8, cex.lab = 0.8,
  ↪  cex.main = 0.9
)
points(AA, pwholm, type = "l", col = "red", lwd = 1.25)
points(AA, pwbonf, type = "l", col = "deepskyblue", lty = "dashed", lwd =
↪  1.25)
points(AA, pwnaive, type = "l", col = "green", lwd = 1.25)
legend("bottomright",
  legend = c("Naive", "BHq", "Holm", "Bonf."),
  col = c("green", "blue", "red", "deepskyblue"), # line colors
  lty = c(1, 1, 1, 2), # line types
  cex = 0.7,
  lwd = 1.25,
)

# FDR comparison (Figure 3.4)
plot(AA, fdrbhq,
  type = "l", ylim = c(-0.01, 0.9), col = "blue", main = "FDR
  ↪  comparison", xlab = "Signal magnitude",
  ylab = "FDR", lwd = 1.25, cex.axis = 0.8, cex.lab = 0.8, cex.main = 0.9
)
points(AA, fdrholm, , type = "l", col = "red", lwd = 1.25)
lines(AA, fdrbonf, lty = "dashed", col = "deepskyblue", lwd = 1.25)
points(AA, fdrnaive, type = "l", col = "green", lwd = 1.25)
abline(h = alpha, col = "black", lty = "dashed", lwd = 1.25)
legend(
  "topright",
```

```
  legend = c("Naive", "BHq", "Holm", "Bonf.", expression(alpha)),
  col = c("green", "blue", "red", "deepskyblue", "black"), # line colors
  lty = c(1, 1, 1, 2, 2), # line types
  cex = 0.6,
  lwd = 1.25,
)
```

### B.2.3 Knockoff filter code

The following is the core code for the entire Knockoff procedure that has been programmed from scratch following the construction detailed in (Barber & Candès, 2015a). It also includes the code to generate Figure 3.5.

```
# KNOCKOFFS: CORE CODE

rm(list = ls())
library(MASS) # to load the function mvrnorm
library(glmnet)
library(knockoff)
# artificial generation of the design matrix

# Setting: n >= 2p
n <- 1500 # number of observations
p <- 80 # number of variables
rho <- 0.3 # correlation among variables
tp <- 12 # number of true positives
A <- 3 # signal amplitude
q <- 0.15 # fdr upper bound
ind_tp <- sample(x = c(1:p), size = tp, replace = FALSE)
ind_fp <- c(1:p)[-ind_tp]

mu <- rep(0, p)
Sigma <- matrix(data = rep(rho, p^2), nrow = p, ncol = p) + diag(rep(1 -
↪  rho, p))
X <- mvrnorm(n = n, mu = mu, Sigma = Sigma)
Xc <- scale(X, center = T, scale = FALSE) # centered matrix
```

```r
Xcn <- apply(Xc, 2, function(x) x / sqrt(sum(x^2)))

# True positive variables

Xtp <- Xcn[, ind_tp]
z <- rnorm(n, mean = 0, sd = 1)

y <- A * rowSums(Xtp) + z
y <- (y - mean(y)) / sd(y)

# Equicorrelated knockoffs

I <- diag(p) # identity matrix
SIG <- crossprod(Xcn) # Gram Matrix
eig_min <- min(abs(eigen(SIG)$values))
s <- min(2 * eig_min, 1)
diags <- diag(rep(s, p))

# Inverse gram matrix
invSIG <- solve(SIG)
mat <- 2 * diags - diags %*% invSIG %*% diags + diag(1e-12, nrow = p)

# Cholesky decomposition to find C
C <- chol(mat)
# U matrix (orthonormal and orthogonal to the span of X)
U <- Null(Xcn)[, 1:p]

# apply(U,2,function(x) norm(x,type="2")) # columns are normalized
# (t(U)%*%Xcn) it can be verified it is a null matrix

# knockoff formula satisfying the two constraint on the correlation
↪  structure
Xtil <- Xcn %*% (I - invSIG %*% diags) + U %*% C
Xtilc <- scale(Xtil, center = T, scale = FALSE) # centered matrix
Xtilcn <- apply(Xtilc, 2, function(x) x / sqrt(sum(x^2)))
Xtot <- as.matrix(cbind(Xcn, Xtilcn))
```

```
nlambda <- 2 * p # number of penalization parameters to consider
lambda_max <- max(abs(crossprod(Xcn, y))) / n # maximum lambda at which
↪  all regression coefficients go to zero

lambda_min <- lambda_max / 2000 # (arbitrarily chosen)
k <- (0:(nlambda - 1)) / nlambda
lambda_val <- lambda_max * (lambda_min / lambda_max)^k
# in this way we have a logarithmic spaced penalization parameter sequence
# between lambda_min and lambda_max, namely lambdas are more densely
# packed near zero and gradually become sparser as they grow.

# fitting the lasso path through glmnet to find the statistics Z_j and
# Z_j tilde.
fit <- glmnet(Xtot, y,
  alpha = 1,
  lambda = lambda_val,
  standardize = FALSE,
  standardize.response = FALSE,
  intercept = FALSE
)
first_nz <- function(x) match(T, abs(x) > 0)
first_nz_ind <- apply(fit$beta, 1, first_nz)
sum(is.na(first_nz_ind))
Z_j <- as.numeric(ifelse(is.na(first_nz_ind), 0,
↪  fit$lambda[first_nz_ind]) * n)

# compute the statistics W_j's
W_j <- numeric(p)
ind_orig <- 1:p
W_j <- pmax(Z_j[ind_orig], Z_j[ind_orig + p]) * sign(Z_j[ind_orig] -
↪  Z_j[ind_orig + p])

# Compute the data-dependent threshold Th
W <- unique(abs(W_j))
W <- W[W != 0]
FDP <- numeric(length(W))
for (j in 1:length(W)) {
```

```
  t <- W[j]
  FDP[j] <- (sum(W_j <= -t)) / max(sum(W_j >= t), 1)
}
j <- which(FDP <= q)
if (length(j) == 0) {
  Th <- Inf
  ind_knock <- integer(0)
} else {
  Th <- min(W[j])
  ind_knock <- which(W_j >= Th)
}

ind_knock <- which(W_j >= Th) # selected indices by the knockoff method
TOTP_knockoff <- length(ind_knock) # total number of discoveries
TP_knockoff <- sum(ind_knock %in% ind_tp) # number of true discoveries
FP_knock <- sum(ind_knock %in% ind_fp) # number of false discoveries
(FDP_knock <- FP_knock / (TOTP_knockoff + 1 / q))
(PW_iter <- TP_knockoff / tp)

# Knockoff pair plot (Figure 3.5)
lim <- max(Z_j) * 8 / 7
par(pty = "s")
plot(Z_j[ind_fp], Z_j[ind_fp + p],
  pch = 19, asp = 1,
  xlim = c(0, lim),
  ylim = c(0, lim),
  xlab = expression(Z[j]),
  ylab = expression(tilde(Z)[j]),
  cex = 0.8,
  main = expression(paste("Knockoff pairs: (", Z[j], ", ", tilde(Z)[j],
  ↪  ")"))
)
abline(a = 0, b = 1, lty = "dashed", col = "grey50", lwd = 1.5)
segments(x0 = Th, y0 = 0 - 5, x1 = Th, y1 = Th, col = "black", lwd = 1.5)
segments(x0 = 0 - 5, y0 = Th, x1 = Th, y1 = Th, lwd = 1.5)
points(Z_j[ind_tp], Z_j[ind_tp + p], col = "red", pch = 15, cex = 0.8)
xx <- c(-Th, -Th, Th, Th, lim * 3 / 2, lim * 3 / 2)
```

```
yy <- c(lim * 3 / 2, Th, Th, 0 - Th, 0 - Th, lim * 3 / 2)
polygon(xx, yy, border = NULL, col = rgb(0.5, 0.5, 0.5, alpha = 0.1))
legend("topright",
  inset = c(-0.58, 0),
  legend = c("Null features", "Non-null features"),
  col = c("black", "red"),
  pch = c(19, 15),
  xpd = NA
)

# Testing the Code programmed from scratch with the results of the actual
# knockoff package of Candes and Barber. We needed a
# customized version in order to adapt it to the construction shown in the
↪  article.
knock <- function(X) create.fixed(Xcn, method = c("equi"), randomize = F)
stats <- function(X, X_k, y, nlambda, standardize) {
  stat.lasso_lambdasmax(
    X = Xcn, X_k = Xtil, y = y, nlambda = 2 * p, ,
    standardize = FALSE
  )
}
result <- knockoff.filter(Xcn, y, knockoffs = knock, statistic = stats,
↪  fdr = q, offset = 0)
result # vector of indices selected by knockoff procedure implemente in
↪  the package knockoff
ind_knock # vector of indices selected by the code I've programmed

# result and ind_knock are pretty much always identical vectors.
# there could still be differences since they used a computationally more
↪  efficient
# way of constructing knockoffs (using SVD), while this code is less
↪  robust
# to errors.
Wstats <- stats(Xcn, Xtil, y, nlambda = 2 * p)
W_j
Wstats
# W_j, Wstats are the vector of test statistics pretty much always
↪  identical
```

```
# for the same reasons explained above.
thresh <- knockoff.threshold(Wstats, fdr = q, offset = 0)
thresh
Th # these are the final data dependent threshold
# pretty much always identical

# the following are the essential commands to navigate and inspect
# the functions of the Knockoff package of Candes (only those that were
# interesting for this work)
# and understand its structure
# getAnywhere(stat.lasso_lambdasmax)
# getAnywhere(stat.glmnet_lambdasmax)
# getAnywhere(lasso_max_lambda)
# getAnywhere(lasso_max_lambda_glmnet)
# getAnywhere(create.fixed)
# getAnywhere(create_equicorrelated)
# get("decompose", envir = asNamespace("knockoff"))
# getAnywhere(create.fixed)
```

### B.2.4 Effect of sparsity, feature correlation and signal magnitude

The following is the code used in the section on the effect of sparsity on FDR and power to generate Figure 3.6 and Figure 3.7.

```
# Comparison of Power and FDR as functions of the sparsity level
# across six procedures: Naive, Bonferroni, Benjamini-Hochberg, Holm,
↪  Knockoff and Knockoff+
# (equicorrelated knockoffs and non-orthogonal design)
# (design matrix drawn only once, new noise at each montecarlo iteration)

rm(list = ls())
set.seed(321)
par(mfrow = c(1, 1))
library(MASS) # to load the function mvrnorm
library(glmnet) # to compute lasso penalization parameters
```

```r
# artificial generation of the design matrix
# setting: n >= 2p
n <- 200 # number of observations
p <- 100 # number of variables
tp <- seq(1, 60, by = 1) # number of true positives
A <- 3.5
rho <- 0.4
q <- 0.2 # fdr upper bound
M <- 2000 # number of Montecarlo iterations
ltp <- length(tp)
FDR_bhq <- numeric(ltp)
PW_bhq <- numeric(ltp)
FDR_knock <- numeric(ltp)
PW_knock <- numeric(ltp)
FDR_knock_plus <- numeric(ltp)
PW_knock_plus <- numeric(ltp)
FDR_naive <- numeric(ltp)
PW_naive <- numeric(ltp)
FDR_bonf <- numeric(ltp)
PW_bonf <- numeric(ltp)
FDR_holm <- numeric(ltp)
PW_holm <- numeric(ltp)
Sigma <- matrix(data = rep(rho, p^2), p, p) + diag(1 - rho, p) #
↪  covariance matrix of the data
mu <- rep(0, p) # mean vector of the data
X <- mvrnorm(n = n, mu = mu, Sigma = Sigma) # design matrix
Xc <- scale(X, center = T, scale = FALSE) # centered matrix
Xcn <- apply(X, 2, function(x) x / sqrt(sum(x^2))) # normalized design

# equicorrelated knockoffs
I <- diag(p) # identity matrix
SIG <- crossprod(Xcn) # Gram matrix
eig_min <- min(abs(eigen(SIG)$values))
s <- min(2 * eig_min, 1)
diags <- diag(rep(s, p))
invSIG <- solve(SIG) # inverse gram matrix
mat <- 2 * diags - diags %*% invSIG %*% diags + diag(1e-12, nrow = p)
```

```
C <- chol(mat) # cholesky decomposition to find C
U <- Null(Xcn)[, 1:p] # U matrix orthogonal to the span of X
Xtil <- Xcn %*% (I - invSIG %*% diags) + U %*% C # knockoff matrix
Xtilc <- scale(Xtil, center = T, scale = FALSE) # centered matrix
Xtilcn <- apply(Xtilc, 2, function(x) x / sqrt(sum(x^2)))
Xtot <- as.matrix(cbind(Xcn, Xtilcn))
nlambda <- 2 * p

for (i in 1:length(tp)) {
  ind_tp <- c(1:tp[i])
  ind_fp <- c((tp[i] + 1):p)
  FDP_iter_knock <- numeric(M)
  PW_iter_knock <- numeric(M)
  FDP_iter_bhq <- numeric(M)
  PW_iter_bhq <- numeric(M)
  FDP_iter_knock_plus <- numeric(M)
  PW_iter_knock_plus <- numeric(M)
  FDP_iter_naive <- numeric(M)
  PW_iter_naive <- numeric(M)
  FDP_iter_bonf <- numeric(M)
  PW_iter_bonf <- numeric(M)
  FDP_iter_holm <- numeric(M)
  PW_iter_holm <- numeric(M)
  betas <- matrix(c(rep(A, tp[i]), rep(0, p - tp[i])), ncol = 1)

  for (m in 1:M) {
    eps <- rnorm(n)
    y <- Xcn %*% betas + eps
    y <- y - mean(y)

    lambda_max <- max(abs(crossprod(X, y))) / n
    lambda_min <- lambda_max / 2000
    k <- (0:(nlambda - 1)) / nlambda
    lambda_val <- lambda_max * (lambda_min / lambda_max)^k

    # lasso path to compute statistics Z_j and Z_j tilde
    fit <- glmnet(Xtot, y,
```

```
  alpha = 1,
  lambda = lambda_val,
  standardize = FALSE,
  standardize.response = FALSE
)

first_nz <- function(x) match(T, abs(x) > 0)
first_nz_ind <- apply(fit$beta, 1, first_nz)
sum(is.na(first_nz_ind))

Z_j <- as.numeric(ifelse(is.na(first_nz_ind), 0,
↪  fit$lambda[first_nz_ind] * n))

# compute the statistics W_j's

W_j <- numeric(p)
ind_orig <- 1:p
W_j <- pmax(Z_j[ind_orig], Z_j[ind_orig + p]) * sign(Z_j[ind_orig] -
↪  Z_j[ind_orig + p])

# Compute the data-dependent threshold for Knockoff and Knockoff+

W <- unique(abs(W_j))
W <- W[W != 0]
FDP <- numeric(length(W))
FDP_plus <- numeric(length(W))
for (j in 1:length(W)) {
  t <- W[j]
  FDP_plus[j] <- (sum(W_j <= -t) + 1) / max(sum(W_j >= t), 1)
  FDP[j] <- sum(W_j <= -t) / max(sum(W_j >= t), 1)
}

j <- which(FDP <= q)
j_plus <- which(FDP_plus <= q)

if (length(j) == 0) {
  Th <- Inf
```

```
  ind_knock <- integer(0)
} else {
  Th <- min(W[j])
  ind_knock <- which(W_j >= Th)
}

ind_knock <- which(W_j >= Th)
TOTP_knockoff <- length(ind_knock)
TP_knockoff <- sum(ind_knock %in% ind_tp)
FP_knock <- sum(ind_knock %in% ind_fp)
FDP_iter_knock[m] <- FP_knock / max(TOTP_knockoff, 1)
PW_iter_knock[m] <- TP_knockoff / tp[i]

if (length(j_plus) == 0) {
  Th_plus <- Inf
  ind_knock_plus <- integer(0)
} else {
  Th_plus <- min(W[j_plus])
  ind_knock_plus <- which(W_j >= Th_plus)
}

ind_knock_plus <- which(W_j >= Th_plus)
TOTP_knockoff_plus <- length(ind_knock_plus)
TP_knockoff_plus <- sum(ind_knock_plus %in% ind_tp)
FP_knock_plus <- sum(ind_knock_plus %in% ind_fp)
FDP_iter_knock_plus[m] <- FP_knock_plus / max(TOTP_knockoff_plus, 1)
PW_iter_knock_plus[m] <- TP_knockoff_plus / tp[i]

# Benjamini-Hochberg
mod <- lm(y ~ Xcn - 1) # no intercept
pvalues <- coef(summary(mod))[, 4]
cutoff <- max(c(0, which(sort(pvalues) <= q * (1:p) / p)))
ind_bhq <- which(pvalues <= q * cutoff / p)
TOTP_bhq <- length(ind_bhq)
TP_bhq <- sum(ind_bhq %in% ind_tp)
FP_bhq <- sum(ind_bhq %in% ind_fp)
FDP_iter_bhq[m] <- FP_bhq / max(TOTP_bhq, 1)
```

```
  PW_iter_bhq[m] <- TP_bhq / tp[i]

  # Naive
  alpha <- q
  ind_naive <- which(pvalues <= alpha)
  totp_naive <- length(ind_naive)
  tp_naive <- sum(ind_naive %in% ind_tp)
  fp_naive <- totp_naive - tp_naive
  FDP_iter_naive[m] <- fp_naive / max(totp_naive, 1)
  PW_iter_naive[m] <- tp_naive / tp[i]

  # Bonferroni
  ind_bonf <- which(pvalues <= alpha / p)
  totp_bonf <- length(ind_bonf)
  tp_bonf <- sum(ind_bonf %in% ind_tp)
  fp_bonf <- totp_bonf - tp_bonf
  FDP_iter_bonf[m] <- fp_bonf / max(totp_bonf, 1)
  PW_iter_bonf[m] <- tp_bonf / tp[i]

  # Holm
  indices <- c(1:p)
  i0 <- min(which(sort(pvalues) > alpha / (p - indices + 1)))
  ind_holm <- which(pvalues < alpha / (p - i0 + 1))
  totp_holm <- length(ind_holm)
  tp_holm <- sum(ind_holm %in% ind_tp)
  fp_holm <- totp_holm - tp_holm
  FDP_iter_holm[m] <- fp_holm / max(totp_holm, 1)
  PW_iter_holm[m] <- tp_holm / tp[i]
}
FDR_knock[i] <- mean(FDP_iter_knock)
PW_knock[i] <- mean(PW_iter_knock)

FDR_bhq[i] <- mean(FDP_iter_bhq)
PW_bhq[i] <- mean(PW_iter_bhq)

FDR_knock_plus[i] <- mean(FDP_iter_knock_plus)
PW_knock_plus[i] <- mean(PW_iter_knock_plus)
```

```
  FDR_naive[i] <- mean(FDP_iter_naive)
  PW_naive[i] <- mean(PW_iter_naive)

  FDR_bonf[i] <- mean(FDP_iter_bonf)
  PW_bonf[i] <- mean(PW_iter_bonf)

  FDR_holm[i] <- mean(FDP_iter_holm)
  PW_holm[i] <- mean(PW_iter_holm)
}

# Comparison of Powers as functions of sparsity level
# across six multiple testing approaches (Figure 3.6)
plot(tp, PW_knock * 100,
  type = "l", col = "darkorange",
  xlab = "Sparsity Level", ylab = "Power(%)", main = "Power",
  ylim = c(0, max(PW_naive * 100)), lwd = 1.2, cex.axis = 0.8, cex.lab =
  ↪  0.8, cex.main = 0.9
)
points(tp, PW_bhq * 100, type = "l", col = "blue", lwd = 1.2)
points(tp, PW_knock_plus * 100, type = "l", col = "purple", lwd = 1.2)
points(tp, PW_naive * 100, type = "l", col = "green", lwd = 1.2)
points(tp, PW_holm * 100, type = "l", col = "red", lwd = 1.2)
lines(tp, PW_bonf * 100, lty = "dashed", col = "deepskyblue", lwd = 1.2)
legend(
  x = 44.4, y = 46,
  legend = c("Naive", "BHq", "Holm", "Bonf.", "Knockoff", "Knockoff+"),
  col = c(
    "green", "blue", "red", "deepskyblue",
    "darkorange", "purple"
  ),
  lty = c(1, 1, 1, 2, 1, 1),
  cex = 0.6,
  lwd = 1.1,
)

# Comparison of FDRs as functions of sparsity level
```

```
# across six multiple testing approaches (Figure 3.7)
plot(tp, FDR_knock * 100,
  ylim = c(0, max(FDR_naive * 100)),
  type = "l", col = "darkorange",
  xlab = "Sparsity Level", ylab = "FDR(%)", main = "False Discovery
  ↪  Rate", lwd = 1.2,
  cex.axis = 0.8, cex.lab = 0.8, cex.main = 0.9
)
abline(h = q * 100, lty = "dashed", lwd = 1.2, col = "grey50")
points(tp, FDR_bhq * 100, type = "l", col = "blue", lwd = 1.2)
points(tp, FDR_knock_plus * 100, type = "l", col = "purple", lwd = 1.2)
points(tp, FDR_naive * 100, type = "l", col = "green", lwd = 1.2)
points(tp, FDR_holm * 100, type = "l", col = "red", lwd = 1.2)
lines(tp, FDR_bonf * 100, lty = "dashed", col = "deepskyblue", lwd = 1.2)
legend("topright",
  legend = c("Naive", "BHq", "Holm", "Bonf.", "Knockoff", "Knockoff+",
  ↪  expression(alpha)),
  col = c(
    "green", "blue", "red", "deepskyblue",
    "darkorange", "purple", "grey50"
  ),
  lty = c(1, 1, 1, 2, 1, 1, 2),
  cex = 0.65,
  lwd = 1.1,
)
```

The following is the code used in the section on the effect of feature correlation on FDR and power to generate Figure 3.8 and Figure 3.9.

```
# Comparison of Power and FDR as functions of the feature correlation
# across six procedures: Naive, Bonferroni, Benjamini-Hochberg, Holm,
↪  Knockoff and Knockoff+
# (equicorrelated knockoffs and non-orthogonal design)
# design and error and response redrawn at each montecarlo iteration
# vector of betas fixed.

rm(list = ls())
```

```
set.seed(321)
par(mfrow = c(1, 1))
library(MASS) # to load the function mvrnorm
library(glmnet)
library(knockoff)

# artificial generation of the design matrix
# n >= 2p
n <- 300 # number of observations
p <- 150 # number of variables
tp <- 10 # number of true positives
A <- 3.5
q <- 0.2 # fdr upper bound
ind_tp <- c(1:tp)
ind_fp <- c((tp + 1):p)
M <- 800 # number of Montecarlo iterations
rho <- seq(0, 0.99, length.out = 30)
FDR_bhq <- numeric(length(rho))
PW_bhq <- numeric(length(rho))
FDR_knock <- numeric(length(rho))
PW_knock <- numeric(length(rho))
FDR_knock_plus <- numeric(length(rho))
PW_knock_plus <- numeric(length(rho))
FDR_naive <- numeric(length(rho))
PW_naive <- numeric(length(rho))
FDR_bonf <- numeric(length(rho))
PW_bonf <- numeric(length(rho))
FDR_holm <- numeric(length(rho))
PW_holm <- numeric(length(rho))
mu <- rep(0, p)
betas <- matrix(c(rep(A, tp), rep(0, p - tp)), ncol = 1)

for (i in 1:length(rho)) {
  FDP_iter_knock <- numeric(M)
  PW_iter_knock <- numeric(M)
  FDP_iter_bhq <- numeric(M)
  PW_iter_bhq <- numeric(M)
```

```r
FDP_iter_knock_plus <- numeric(M)
PW_iter_knock_plus <- numeric(M)
FDP_iter_naive <- numeric(M)
PW_iter_naive <- numeric(M)
FDP_iter_bonf <- numeric(M)
PW_iter_bonf <- numeric(M)
FDP_iter_holm <- numeric(M)
PW_iter_holm <- numeric(M)
Sigma <- matrix(data = rep(rho[i], p^2), p, p) + diag(1 - rho[i], p)

for (m in 1:M) {
  X <- mvrnorm(n = n, mu = mu, Sigma = Sigma)
  Xc <- scale(X, center = T, scale = FALSE) # centered matrix
  Xcn <- apply(X, 2, function(x) x / sqrt(sum(x^2)))
  y <- Xcn %*% betas + rnorm(n)
  y <- y - mean(y)

  # equicorrelated knockoffs
  I <- diag(p) # identity matrix
  SIG <- crossprod(Xcn) # Gram matrix

  eig_min <- min(eigen(SIG)$values)
  s <- min(2 * eig_min, 1)
  diags <- diag(rep(s, p))

  # Inverse gram matrix
  invSIG <- solve(SIG)
  mat <- 2 * diags - diags %*% invSIG %*% diags + diag(1e-12, nrow = p)

  # Cholesky decomposition to find C
  C <- chol(mat)
  U <- Null(Xcn)[, 1:p] # U matrix orthogonal to the span of X
  Xtil <- Xcn %*% (I - invSIG %*% diags) + U %*% C # knockoff matrix
  Xtilc <- scale(Xtil, center = T, scale = FALSE) # centered matrix
  Xtilcn <- apply(Xtilc, 2, function(x) x / sqrt(sum(x^2)))
  Xtot <- as.matrix(cbind(Xcn, Xtilcn))
```

```
nlambda <- 2 * p
lambda_max <- max(abs(crossprod(X, y))) / n
lambda_min <- lambda_max / 2000
k <- (0:(nlambda - 1)) / nlambda
lambda_val <- lambda_max * (lambda_min / lambda_max)^k

# Lasso path to compute statistics Z_j and Z_j tilde
fit <- glmnet(Xtot, y,
  alpha = 1,
  lambda = lambda_val,
  standardize = FALSE,
  standardize.response = FALSE
)
first_nz <- function(x) match(T, abs(x) > 0)
first_nz_ind <- apply(fit$beta, 1, first_nz)
sum(is.na(first_nz_ind))
Z_j <- as.numeric(ifelse(is.na(first_nz_ind), 0,
↪  fit$lambda[first_nz_ind] * n))

# compute the statistics W_j's
W_j <- numeric(p)
ind_orig <- 1:p
W_j <- pmax(Z_j[ind_orig], Z_j[ind_orig + p]) * sign(Z_j[ind_orig] -
↪  Z_j[ind_orig + p])

# Compute the data-dependent threshold for Knockoff and Knockoff+
W <- unique(abs(W_j))
W <- W[W != 0]
FDP_plus <- numeric(length(W))
FDP <- numeric(length(W))
for (j in 1:length(W)) {
  t <- W[j]
  FDP_plus[j] <- (sum(W_j <= -t) + 1) / max(sum(W_j >= t), 1)
  FDP[j] <- sum(W_j <= -t) / max(sum(W_j >= t), 1)
}

j_plus <- which(FDP_plus <= q)
```

```
j <- which(FDP <= q)

if (length(j) == 0) {
  Th <- Inf
  ind_knock <- integer(0)
} else {
  Th <- min(W[j])
  ind_knock <- which(W_j >= Th)
}

ind_knock <- which(W_j >= Th)
TOTP_knockoff <- length(ind_knock)
TP_knockoff <- sum(ind_knock %in% ind_tp)
FP_knock <- sum(ind_knock %in% ind_fp)
FDP_iter_knock[m] <- FP_knock / max(TOTP_knockoff, 1)
PW_iter_knock[m] <- TP_knockoff / tp

if (length(j_plus) == 0) {
  Th_plus <- Inf
  ind_knock_plus <- integer(0)
} else {
  Th_plus <- min(W[j_plus])
  ind_knock_plus <- which(W_j >= Th_plus)
}

ind_knock_plus <- which(W_j >= Th_plus)
TOTP_knockoff_plus <- length(ind_knock_plus)
TP_knockoff_plus <- sum(ind_knock_plus %in% ind_tp)
FP_knock_plus <- sum(ind_knock_plus %in% ind_fp)
FDP_iter_knock_plus[m] <- FP_knock_plus / max(TOTP_knockoff_plus, 1)
PW_iter_knock_plus[m] <- TP_knockoff_plus / tp

# Benjamini-Hochberg
mod <- lm(y ~ Xcn - 1) # no intercept
pvalues <- coef(summary(mod))[, 4]
cutoff <- max(c(0, which(sort(pvalues) <= q * (1:p) / p)))
ind_bhq <- which(pvalues <= q * cutoff / p)
```

```
  TOTP_bhq <- length(ind_bhq)
  TP_bhq <- sum(ind_bhq %in% ind_tp)
  FP_bhq <- sum(ind_bhq %in% ind_fp)
  FDP_iter_bhq[m] <- FP_bhq / max(TOTP_bhq, 1)
  PW_iter_bhq[m] <- TP_bhq / tp

  # Naive
  alpha <- q
  ind_naive <- which(pvalues <= alpha)
  totp_naive <- length(ind_naive)
  tp_naive <- sum(ind_naive %in% ind_tp)
  fp_naive <- totp_naive - tp_naive
  FDP_iter_naive[m] <- fp_naive / max(totp_naive, 1)
  PW_iter_naive[m] <- tp_naive / tp

  # Bonferroni
  ind_bonf <- which(pvalues <= alpha / p)
  totp_bonf <- length(ind_bonf)
  tp_bonf <- sum(ind_bonf %in% ind_tp)
  fp_bonf <- totp_bonf - tp_bonf
  FDP_iter_bonf[m] <- fp_bonf / max(totp_bonf, 1)
  PW_iter_bonf[m] <- tp_bonf / tp

  # Holm
  indices <- c(1:p)
  i0 <- min(which(sort(pvalues) > alpha / (p - indices + 1)))
  ind_holm <- which(pvalues < alpha / (p - i0 + 1))
  totp_holm <- length(ind_holm)
  tp_holm <- sum(ind_holm %in% ind_tp)
  fp_holm <- totp_holm - tp_holm
  FDP_iter_holm[m] <- fp_holm / max(totp_holm, 1)
  PW_iter_holm[m] <- tp_holm / tp
}
FDR_knock[i] <- mean(FDP_iter_knock)
PW_knock[i] <- mean(PW_iter_knock)

FDR_bhq[i] <- mean(FDP_iter_bhq)
```

```
  PW_bhq[i] <- mean(PW_iter_bhq)

  FDR_knock_plus[i] <- mean(FDP_iter_knock_plus)
  PW_knock_plus[i] <- mean(PW_iter_knock_plus)

  FDR_naive[i] <- mean(FDP_iter_naive)
  PW_naive[i] <- mean(PW_iter_naive)

  FDR_bonf[i] <- mean(FDP_iter_bonf)
  PW_bonf[i] <- mean(PW_iter_bonf)

  FDR_holm[i] <- mean(FDP_iter_holm)
  PW_holm[i] <- mean(PW_iter_holm)
}

# Comparison of Powers as functions of feature correlation
# across six multiple testing approaches (Figure 3.8)
plot(rho, PW_knock * 100,
  type = "l", col = "darkorange",
  xlab = "Feature correlation", ylab = "Power(%)", main = "Power",
  ylim = c(-2, max(PW_naive * 100)), lwd = 1.2, cex.axis = 0.8, cex.lab =
  ↪  0.8, cex.main = 0.9
)
points(rho, PW_bhq * 100, type = "l", col = "blue", lwd = 1.2)
points(rho, PW_knock_plus * 100, type = "l", col = "purple", lwd = 1.2)
points(rho, PW_naive * 100, type = "l", col = "green", lwd = 1.2)
points(rho, PW_holm * 100, type = "l", col = "red", lwd = 1.2)
lines(rho, PW_bonf * 100, lty = "dashed", col = "deepskyblue", lwd = 1.2)
legend("topright",
  legend = c("Naive", "BHq", "Holm", "Bonf.", "Knockoff", "Knockoff+"),
  col = c(
    "green", "blue", "red", "deepskyblue",
    "darkorange", "purple"
  ),
  lty = c(1, 1, 1, 2, 1, 1),
  cex = 0.57,
  lwd = 1.1,
```

```
)

# Comparison of FDRs as functions of feature correlation
# across six multiple testing approaches (Figure 3.9)
plot(rho, FDR_knock * 100,
  ylim = c(0, 100),
  type = "l", col = "darkorange",
  xlab = "Feature correlation", ylab = "FDR(%)", main = "False Discovery
  ↪  Rate", lwd = 1.2, cex.axis = 0.8, cex.lab = 0.8, cex.main = 0.9
)
abline(h = q * 100, lty = "dashed", lwd = 1.2, col = "grey50")
points(rho, FDR_bhq * 100, type = "l", col = "blue", lwd = 1.2)
points(rho, FDR_knock_plus * 100, type = "l", col = "purple", lwd = 1.2)
points(rho, FDR_naive * 100, type = "l", col = "green", lwd = 1.2)
points(rho, FDR_holm * 100, type = "l", col = "red", lwd = 1.2)
lines(rho, FDR_bonf * 100, lty = "dashed", col = "deepskyblue", lwd = 1.2)
legend(
  x = 0.75, y = 75,
  legend = c("Naive", "BHq", "Holm", "Bonf.", "Knockoff", "Knockoff+",
  ↪  expression(alpha)),
  col = c(
    "green", "blue", "red", "deepskyblue",
    "darkorange", "purple", "grey50"
  ),
  lty = c(1, 1, 1, 2, 1, 1, 2),
  cex = 0.57,
  lwd = 1.1,
)
```

The following is the code used in the section on the effect of signal magnitude on FDR and power to generate Figure 3.10 and Figure 3.11

```
# Comparison of Power and FDR as functions of the signal amplitude
# across six procedures: Naive, Bonferroni, Benjamini-Hochberg, Holm,
↪  Knockoff and Knockoff+
# (equicorrelated knockoffs and non-orthogonal design)
# Design matrix drawn only once, new noise at each montecarlo iteration
```

```r
rm(list = ls())
set.seed(321)
par(mfrow = c(1, 1))
library(MASS) # to load the function mvrnorm
library(glmnet)

# artificial generation of the design matrix

# setting: n >= 2p
n <- 100 # number of observations
p <- 50 # number of variables
rho <- 0.4 # features correlation
tp <- 8 # number of true positives
AA <- seq(from = -10, to = 10, length.out = 60) # signal amplitude vector
q <- 0.2 # FDR upper bound
ind_tp <- c(1:tp) # indices of true positives
ind_fp <- c((tp + 1):p) # indices of false positives
M <- 2000 # number of Montecarlo iterations

nlambda <- 2 * p # number of lasso penalization parameter to compute using
↪  glmnet
W_j <- numeric(p) # initializing knockoff test statistics
ind_orig <- 1:p

# initializations of vectors of Powers and FDR
FDR_bhq <- numeric(length(AA))
PW_bhq <- numeric(length(AA))
FDR_knock <- numeric(length(AA))
PW_knock <- numeric(length(AA))
FDR_knock_plus <- numeric(length(AA))
modFDR_knock_plus <- numeric(length(AA))
PW_knock_plus <- numeric(length(AA))
FDR_naive <- numeric(length(AA))
PW_naive <- numeric(length(AA))
FDR_bonf <- numeric(length(AA))
PW_bonf <- numeric(length(AA))
```

```
FDR_holm <- numeric(length(AA))
PW_holm <- numeric(length(AA))

# Knockoff construction

# design matrix X is generated by sampling each row from a N(mu,Sigma)
Sigma <- matrix(data = rep(rho, p^2), p, p) + diag(1 - rho, p) #
↪  covariance matrix
mu <- rep(0, p) # mean vector
X <- mvrnorm(n = n, mu = mu, Sigma = Sigma) # fixed design
Xc <- scale(X, center = T, scale = FALSE) # centered matrix
Xcn <- apply(X, 2, function(x) x / sqrt(sum(x^2))) # normalized design
I <- diag(p) # identity matrix
SIG <- crossprod(Xcn) # Gram matrix
# equicorrelated knockoffs construction
eig_min <- min(eigen(SIG)$values)
s <- min(2 * eig_min, 1)
diags <- diag(rep(s, p))
# inverse gram matrix
invSIG <- solve(SIG)
# Knockoff formula construction
mat <- 2 * diags - diags %*% invSIG %*% diags + diag(1e-12, nrow = p)
C <- chol(mat)
U <- Null(Xcn)[, 1:p]
Xtil <- Xcn %*% (I - invSIG %*% diags) + U %*% C # matrix of knockoffs
Xtilc <- scale(Xtil, center = T, scale = FALSE) # centered knockoffs
Xtilcn <- apply(Xtilc, 2, function(x) x / sqrt(sum(x^2))) # normalized
↪  knockoffs
Xtot <- as.matrix(cbind(Xcn, Xtilcn)) # column-wise concatanation matrix

# first for cycle used to set the true signal amplitude
for (i in 1:length(AA)) {
  A <- AA[i]
  FDP_iter_knock <- numeric(M)
  PW_iter_knock <- numeric(M)
  FDP_iter_bhq <- numeric(M)
  PW_iter_bhq <- numeric(M)
```

```r
FDP_iter_knock_plus <- numeric(M)
PW_iter_knock_plus <- numeric(M)
FDP_iter_naive <- numeric(M)
PW_iter_naive <- numeric(M)
FDP_iter_bonf <- numeric(M)
PW_iter_bonf <- numeric(M)
FDP_iter_holm <- numeric(M)
PW_iter_holm <- numeric(M)
modFDR_iter_knock_plus <- numeric(M)
betas <- matrix(data = c(rep(A, tp), rep(0, p - tp)), ncol = 1) # true
↪  regression coefficients
# sparse vector of betas all equal to A.

for (m in 1:M) {
  W_j <- numeric(p)
  eps <- rnorm(n)
  y <- Xcn %*% betas + eps # model artificially created
  y <- y - mean(y)

  lambda_max <- max(abs(crossprod(Xcn, y))) / n
  lambda_min <- lambda_max / 2000
  k <- (0:(nlambda - 1)) / nlambda
  lambda_val <- lambda_max * (lambda_min / lambda_max)^k

  fit <- glmnet(Xtot, y,
    alpha = 1,
    lambda = lambda_val,
    standardize = FALSE,
    standardize.response = FALSE
  )

  first_nz <- function(x) match(T, abs(x) > 0)
  first_nz_ind <- apply(fit$beta, 1, first_nz)

  Z_j <- as.numeric(ifelse(is.na(first_nz_ind), 0,
  ↪  fit$lambda[first_nz_ind] * n))
```

```
# compute the statistics W_j's

W_j <- pmax(Z_j[ind_orig], Z_j[ind_orig + p]) * sign(Z_j[ind_orig] -
↪  Z_j[ind_orig + p])

# Compute the data-dependent threshold (for Knockoff and Knockoff+)

W <- unique(abs(W_j))
W <- W[W != 0]
FDP <- numeric(length(W))
FDP_plus <- numeric(length(W))
for (j in 1:length(W)) {
  t <- W[j]
  FDP[j] <- sum(W_j <= -t) / max(sum(W_j >= t), 1)
  FDP_plus[j] <- (sum(W_j <= -t) + 1) / max(sum(W_j >= t), 1)
}

j <- which(FDP <= q)
j_plus <- which(FDP_plus <= q)

if (length(j) == 0) {
  Th <- Inf
  ind_knock <- integer(0)
} else {
  Th <- min(W[j])
  ind_knock <- which(W_j >= Th)
}

ind_knock <- which(W_j >= Th)
TOTP_knockoff <- length(ind_knock)
TP_knockoff <- sum(ind_knock %in% ind_tp)
FP_knock <- sum(ind_knock %in% ind_fp)
FDP_iter_knock[m] <- FP_knock / max(TOTP_knockoff, 1)
PW_iter_knock[m] <- TP_knockoff / tp

if (length(j_plus) == 0) {
  Th_plus <- Inf
```

```
    ind_knock_plus <- integer(0)
  } else {
    Th_plus <- min(W[j_plus])
    ind_knock_plus <- which(W_j >= Th_plus)
  }

  ind_knock_plus <- which(W_j >= Th_plus)
  TOTP_knockoff_plus <- length(ind_knock_plus)
  TP_knockoff_plus <- sum(ind_knock_plus %in% ind_tp)
  FP_knock_plus <- sum(ind_knock_plus %in% ind_fp)
  FDP_iter_knock_plus[m] <- FP_knock_plus / max(TOTP_knockoff_plus, 1)
  PW_iter_knock_plus[m] <- TP_knockoff_plus / tp
  modFDR_iter_knock_plus[m] <- FP_knock_plus / (TOTP_knockoff_plus + 1
  ↪  / q)

  # Benjamini-Hochberg
  mod <- lm(y ~ Xcn - 1) # no intercept
  pvalues <- coef(summary(mod))[, 4]
  cutoff <- max(c(0, which(sort(pvalues) <= q * (1:p) / p)))
  ind_bhq <- which(pvalues <= q * cutoff / p)
  TOTP_bhq <- length(ind_bhq)
  TP_bhq <- sum(ind_bhq %in% ind_tp)
  FP_bhq <- sum(ind_bhq %in% ind_fp)
  FDP_iter_bhq[m] <- FP_bhq / max(TOTP_bhq, 1)
  PW_iter_bhq[m] <- TP_bhq / tp

  # Naive
  alpha <- q
  ind_naive <- which(pvalues <= alpha)
  totp_naive <- length(ind_naive)
  tp_naive <- sum(ind_naive %in% ind_tp)
  fp_naive <- totp_naive - tp_naive
  FDP_iter_naive[m] <- fp_naive / max(totp_naive, 1)
  PW_iter_naive[m] <- tp_naive / tp

  # Bonferroni
  ind_bonf <- which(pvalues <= alpha / p)
```

```
    totp_bonf <- length(ind_bonf)
    tp_bonf <- sum(ind_bonf %in% ind_tp)
    fp_bonf <- totp_bonf - tp_bonf
    FDP_iter_bonf[m] <- fp_bonf / max(totp_bonf, 1)
    PW_iter_bonf[m] <- tp_bonf / tp

    # Holm
    indices <- c(1:p)
    i0 <- min(which(sort(pvalues) > alpha / (p - indices + 1)))
    ind_holm <- which(pvalues < alpha / (p - i0 + 1))
    totp_holm <- length(ind_holm)
    tp_holm <- sum(ind_holm %in% ind_tp)
    fp_holm <- totp_holm - tp_holm
    FDP_iter_holm[m] <- fp_holm / max(totp_holm, 1)
    PW_iter_holm[m] <- tp_holm / tp
  }

  FDR_knock[i] <- mean(FDP_iter_knock)
  PW_knock[i] <- mean(PW_iter_knock)
  FDR_bhq[i] <- mean(FDP_iter_bhq)
  PW_bhq[i] <- mean(PW_iter_bhq)
  FDR_knock_plus[i] <- mean(FDP_iter_knock_plus)
  PW_knock_plus[i] <- mean(PW_iter_knock_plus)
  modFDR_knock_plus[i] <- mean(modFDR_iter_knock_plus)
  FDR_naive[i] <- mean(FDP_iter_naive)
  PW_naive[i] <- mean(PW_iter_naive)
  FDR_bonf[i] <- mean(FDP_iter_bonf)
  PW_bonf[i] <- mean(PW_iter_bonf)
  FDR_holm[i] <- mean(FDP_iter_holm)
  PW_holm[i] <- mean(PW_iter_holm)
}

# Comparison of Powers functions (signal amplitude on the x-axis)
# across six multiple testing approaches (Figure 3.10)
plot(AA, PW_knock * 100,
  type = "l", col = "darkorange",
  xlab = "Signal Magnitude", ylab = "Power(%)", main = "Power function",
```

```
  ylim = c(0, max(PW_naive * 100)), lwd = 1.2, cex.axis = 0.8, cex.lab =
  ↪  0.8, cex.main = 0.9
)
points(AA, PW_bhq * 100, type = "l", col = "blue", lwd = 1.2)
points(AA, PW_knock_plus * 100, type = "l", col = "purple", lwd = 1.2)
points(AA, PW_naive * 100, type = "l", col = "green", lwd = 1.2)
points(AA, PW_holm * 100, type = "l", col = "red", lwd = 1.2)
lines(AA, PW_bonf * 100, lty = "dashed", col = "deepskyblue", lwd = 1.2)
legend("bottomright",
  legend = c("Naive", "BHq", "Holm", "Bonf.", "Knockoff", "Knockoff+"),
  col = c(
    "green", "blue", "red", "deepskyblue",
    "darkorange", "purple"
  ),
  lty = c(1, 1, 1, 2, 1, 1),
  cex = 0.5,
  lwd = 1.1,
)

# Comparison of FDRs as functions of the signal amplitude
# across six multiple testing approaches (Figure 3.11)
plot(AA, FDR_knock * 100,
  ylim = c(0, max(FDR_naive * 100)),
  type = "l", col = "darkorange",
  xlab = "Signal Magnitude",
  ylab = "FDR(%)", main = "FDR", lwd = 1.2, cex.axis = 0.8, cex.lab =
  ↪  0.8, cex.main = 0.9
)
abline(h = q * 100, lty = "dashed", col = "grey50", lwd = 1.2)
points(AA, FDR_bhq * 100, type = "l", col = "blue", lwd = 1.2)
points(AA, FDR_knock_plus * 100, type = "l", col = "purple", lwd = 1.2)
points(AA, FDR_naive * 100, type = "l", col = "green", lwd = 1.2)
points(AA, FDR_holm * 100, type = "l", col = "red", lwd = 1.2)
points(AA, modFDR_knock_plus * 100, type = "l", col = "darkgreen", lwd =
↪  1.2)

lines(AA, FDR_bonf * 100, lty = "dashed", col = "deepskyblue", lwd = 1.2)
```

```
legend(
  "topright",
  legend = c("Naive", "BHq", "Holm", "Bonf.", "Knockoff", "Knockoff+",
  ↪  "modif. FDR (K.+)", expression(alpha)),
  col = c(
    "green", "blue", "red", "deepskyblue",
    "darkorange", "purple", "darkgreen", "grey50"
  ),
  lty = c(1, 1, 1, 2, 1, 1, 1, 2),
  cex = 0.4,
  lwd = 1.1,
)
```

### B.2.5 HIV data application

The following is the code to recreate the entire analysis on HIV-1 Protease mutations associated with Nelfinavir resistance. This code includes both the data pre-processing phase and the assessment of competing multiple testing procedures. It is also provided the code used to generate Figure 3.12 and Figure 3.13

```
# Experiment on real data: HIV-1 resistance

# Our attenation is focused on HIV 1 resistance to Protease inhibitors, in
↪  particular
# to the antiviral NFV (Nelfinavir).
# link for the matrix predictors and responses (matrix of positions of the
↪  mutations and drug resistance measurements
# for 7 protease inhibitors) https://hivdb.stanford.edu/_wrapper/pages/pu
↪  blished_analysis/genophenoPNAS2006/DATA/PI_DATA.txt

# link for the TSM list containing relevant mutation of the HIV-1 protease
↪  regardless of the specific protein inhibitor used
# https://hivdb.stanford.edu/pages/published_analysis/genophenoPNAS2006/M
↪  UTATIONLISTS/NP_TSM/PI

rm(list = setdiff(ls(), c("PI_data", "PI_TSM")))
```

```
# position selected by treatement selected mutation (approximately the
↪  ground truth)
pos <- PI_TSM$V1
npos <- length(pos) # number of mutations selected
amm <- PI_TSM$V2

# Structure of PI_TSM: each row is a relevant position in the chain of
# HIV protease that has provably shown mutations. In the second column
# each cell is a list of mutation at a specific position in the Protease
↪  chain
# specified in the first column.

# The following is the code to create the full notations (like M46I etc
↪  ...)
# for the mutations in table PI_TSM (whose structure is not directly
# M46I, I309 etc...)

# wt is the wild type sequence (not-mutated of HIV-1) protease
# it has been taken from https://www.uniprot.org/uniprotkb/O90777/entry
wt <- unlist(strsplit("PQVTLWQRPIVTIKIGGQLKEALLDTGADDTVLEEMSLPGKWKPKMIGGI
↪  GGFIKVRQYDQVSIEICGHKAIGTVLIGPTPVNIIGRNLLTQLGCTLNF", split =
↪  ""))
names(wt) <- as.character(1:99)

amm <- strsplit(amm, " ")
amm_unlist <- unlist(amm)
namm <- length(amm_unlist)
wtmut <- list()
k <- 1
for (i in 1:npos) {
  position <- pos[i]
  amm_position <- unlist(amm[[i]])
  nmu <- length(amm_position)
  for (j in 1:nmu) {
    wtmut[[k]] <- paste(c(wt[position], position, amm_position[j]),
    ↪  collapse = "")
```

```
    k <- k + 1
  }
}
length(wtmut)
# wtmut contains the full list of mutations in TSM list for HIV-1 protease
# the list is composed by mutation with the notation such as M46I

data <- PI_data
# adjusting the column names
colnames(data) <- data[1, ] # naming the columns with the first row
data <- data[-1, ] # removing the first column of id's
n <- dim(data)[1] # the number of initial rows
# - are missing value that are structural
# the NA in the response drug resistance are important to take into
↪  account

summary(is.na(data)) # assessing the presence of NA in the drug resistance
↪  measurements
# We choose the drug Nelfinavir (among the Protease inhibitors)
# because it has the least amount of missing values.

# These are the indices of missing values in the response of drug
↪  resistance
ind_na_nfv <- which(is.na(data$NFV) == TRUE) # isolating the inidices
length(ind_na_nfv)

# The response is computed as a log-fold change, namely is the log
# base = 10 of the ratio between
# the concentration of drug to inhibit 50% of the replication of the virus
# when the HIV-1 Protease is mutated
# divided by the concentration of drug needed to reduce by 50% the
↪  replication of
# the virus when HIV 1 protease is wild type
y <- as.numeric(data$NFV[-ind_na_nfv])
# if y=1 the patient virus is resistant as wild type
# if y>1 patient virus is more resistant than wild type
# if y<1 patient virus is less resistant than wild type
```

```
# We also remove from the predictors the rows that have a missing value in
↪  the response
# of log-fold change of Nelfinavir
data <- data[-ind_na_nfv, ]
# We remove all responses (in this way dataX is the matrix of predictors)
dataX <- data[, -c(1:10)]
# structure of dataX: we have a matrix approximately 844x99
# where each columns represent a position. in this way we have for a fixed
# position/column all ammino acids mutated at that specific position
# obviously we could have several different mutations at the same
↪  position.
# We would like to have specific mutations like M46I instead of positions
↪  like P70
# as features. In dataX the value in cell (i,j) is either - if the the
↪  HIV-1
# Protease sample of the patient i doesn't show a mutation in position j
# or it is the first letter of the ammino acid that represents the
↪  mutation if the mut. is actually present.
# we would like to have only "-" or letters in dataX; We don't want any
↪  other symbol.
# therefore we replace "." with "-".
mut_list <- list()
n_X <- dim(dataX)[1]
p_X <- dim(dataX)[2]
anyNA(dataX)
for (j in 1:p_X) {
  for (i in 1:n_X) {
    if (dataX[i, j] == ".") {
      dataX[i, j] <- "-"
    }
  }
}
# In the following for cycles we create the list of all mutations in the
↪  standard notation
# that can be found in dataX. So mut_list is a list of characters such as
# M46I, I10L and so on.
k <- 1
```

```
for (j in 1:p_X) {
  for (i in 1:n_X) {
    if (dataX[i, j] != "-") {
      mut_list[k] <- paste(c(wt[j], j, dataX[i, j]), collapse = "")
      k <- k + 1
    }
  }
}
# Obviously we will have several mutations that will be exactly equal.
# therefore we take the vector of unique mutations umut.
numut <- length(unique(mut_list))
umut <- unique(mut_list)
# with the following code  we want to create a list of lists.
# more precisely we want a list for each HIV-1 sample including all
↪  mutations appearing for that
# sample. so we will have a list of n_X lists, where n_X list is the
# number of filtered patients. each list in the big list will have a
↪  different number
# of mutations since different samples have different mutations

mut_grouped_by_sample <- list() # list of list of mutations for each
↪  sample of HIV1 protease
for (i in 1:n_X) {
  mut_list_i <- list()
  k <- 1
  for (j in 1:p_X) {
    if (dataX[i, j] != "-") {
      mut_list_i[k] <- paste(c(wt[j], j, dataX[i, j]), collapse = "")
      k <- k + 1
    }
  }
  mut_grouped_by_sample[[i]] <- mut_list_i
}
str(mut_grouped_by_sample)
# now we are able to create a matrix having on the columns unique
↪  mutations in standard notations
# and on the rows the HIV-1 Protease samples. The entry in cell (i,j)
```

```
# will be either 0 or 1 depending whether sample i-th of HIV-1 Protease
↪  contains the mutation
# that labels column j-th or not.

X <- matrix(data = 0, nrow = n_X, ncol = length(umut))
colnames(X) <- umut
for (i in 1:n_X) {
  for (j in 1:length(umut)) {
    X[i, j] <- as.numeric(ifelse(umut[j] %in%
    ↪  unlist(mut_grouped_by_sample[[i]]), 1, 0))
  }
}

# We remove mutations that appears in less than 3 samples (namely we
↪  remove columns)
ind_rm <- which(apply(X, 2, sum) < 3)
length(ind_rm)
X <- X[, -ind_rm]

# we also remove duplicates in order to have a full rank matrix
X <- X[, which(!duplicated(t(X)) == T)]

dim(X)
# we standardize the response
y <- (y - mean(y)) / sd(y)
# we now apply the six methods

# Knockoff filter

library(MASS) # to load the function mvrnorm
library(glmnet)
library(knockoff)

n <- nrow(X) # number of observations
p <- ncol(X) # number of variables
q <- 0.2 # fdr upper bound
Xc <- scale(X, center = T, scale = FALSE) # centered matrix
```

```r
Xcn <- apply(Xc, 2, function(x) x / sqrt(sum(x^2)))
Xcn <- as.matrix(Xcn)
attr(Xcn, "dimnames") <- NULL

# equicorrelated knockoffs
I <- diag(p) # identity matrix
SIG <- crossprod(Xcn) # Gram matrix
anyNA(SIG)
eig_min <- min(eigen(SIG)$values)
s <- min(2 * eig_min, 1)
diags <- diag(rep(s, p))

# inverse gram matrix
invSIG <- solve(SIG)
mat <- 2 * diags - diags %*% invSIG %*% diags + diag(1e-10, nrow = p)

# cholesky decomposition to find C
C <- chol(mat)
U <- Null(Xcn)[, 1:p]
Xtil <- Xcn %*% (I - invSIG %*% diags) + U %*% C
Xtilc <- scale(Xtil, center = T, scale = FALSE) # centered matrix
Xtilcn <- apply(Xtilc, 2, function(x) x / sqrt(sum(x^2)))
Xtot <- as.matrix(cbind(Xcn, Xtilcn))

nlambda <- 2 * p
lambda_max <- max(abs(crossprod(Xcn, y))) / n
nlambda <- 2 * p
lambda_min <- lambda_max / 2000
k <- (0:(nlambda - 1)) / nlambda
lambda_val <- lambda_max * (lambda_min / lambda_max)^k

fit <- glmnet(Xtot, y,
  alpha = 1,
  lambda = lambda_val,
  standardize = FALSE,
  standardize.response = FALSE, intercept = FALSE
)
```

```
first_nz <- function(x) match(T, abs(x) > 0)
first_nz_ind <- apply(fit$beta, 1, first_nz)
Z_j <- as.numeric(ifelse(is.na(first_nz_ind), 0, fit$lambda[first_nz_ind]
↪  * n))

# compute the statistics W_j's
W_j <- numeric(p)
ind_orig <- 1:p
W_j <- pmax(Z_j[ind_orig], Z_j[ind_orig + p]) * sign(Z_j[ind_orig] -
↪  Z_j[ind_orig + p])

# Compute the data-dependent threshold
W <- unique(abs(W_j))
W <- W[W != 0]
FDP <- numeric(length(W))
FDP_plus <- numeric(length(W))
for (j in 1:length(W)) {
  t <- W[j]
  FDP[j] <- (sum(W_j <= -t)) / max(sum(W_j >= t), 1)
  FDP_plus[j] <- (sum(W_j <= -t) + 1) / max(sum(W_j >= t), 1)
}
j <- which(FDP <= q)
j_plus <- which(FDP_plus <= q)

if (length(j) == 0) {
  Th <- Inf
  ind_knock <- integer(0)
} else {
  Th <- min(W[j])
  ind_knock <- which(W_j >= Th)
}

ind_knock <- which(W_j >= Th)
TOTP_knockoff <- length(ind_knock)

# ind_knock represent the indices of the columns that have been selected.
```

```
# However mutations doesn't correspond directly to columns
# so we need to find the mutations that have been selected
# selected mutations are knock_mut.
mutations <- colnames(X)
knock_mut <- mutations[ind_knock]
# at this point we consider the set of approximately true mutations
↪  deriving from the
# TSM list. even though we have for each position the type of mutations
# and we could compare standard notation mutation with the ones selected
# we decide only to compare positions selected and not the complete
↪  mutations
# since the list of TSM is approximated and besides that it is general for
↪  PIs not
# specific for Nelfinavir.

# we therefore extract the positions of mutations selected
pos_sel_knock <- numeric(length(knock_mut))
for (i in 1:length(knock_mut)) {
  pos_sel_knock[i] <- as.numeric(gsub("\\D", "", knock_mut[i]))
}

# we take unique values of positions selected and we look for
# how many unique positions selected are also in the TSM list of positions
tp_knock_sel <- sum(unique(pos_sel_knock) %in% pos)
tot_knock_sel <- length(unique(pos_sel_knock))
tot_in_TSM <- length(pos)
fp_knock_sel <- tot_knock_sel - tp_knock_sel
# then we repeat the same thing for the other multiple testing methods
if (length(j_plus) == 0) {
  Th_plus <- Inf
  ind_knock_plus <- integer(0)
} else {
  Th_plus <- min(W[j_plus])
  ind_knock_plus <- which(W_j >= Th_plus)
}

ind_knock_plus <- which(W_j >= Th_plus)
```

```
TOTP_knockoff_plus <- length(ind_knock_plus)
knock_plus_mut <- mutations[ind_knock_plus]
pos_sel_knock_plus <- numeric(length(knock_plus_mut))
for (i in 1:length(knock_plus_mut)) {
  pos_sel_knock_plus[i] <- as.numeric(gsub("\\D", "", knock_plus_mut[i]))
}
tp_knock_plus_sel <- sum(unique(pos_sel_knock_plus) %in% pos)
tot_knock_plus_sel <- length(unique(pos_sel_knock_plus))
fp_knock_plus_sel <- tot_knock_plus_sel - tp_knock_plus_sel

# knockoff graph code (Figure 3.13)

pos_tot <- numeric()
for (j in 1:p) {
  pos_tot[j] <- as.numeric(gsub("\\D","",mutations[j]))
}
index <- which(pos_tot%in%pos)
par(mfrow=c(1,1))
par(pty="s")
plot(Z_j[ind_orig],Z_j[ind_orig+p],pch=19,
     xlab = expression(Z[j]),
     ylab = expression(tilde(Z)[j]),cex=0.5,xlim=c(0,max(Z_j[ind_orig])),
     ylim=c(0,max(Z_j[ind_orig+p])),main=expression(paste("Knockoff pairs:
     ↪  (", Z[j], ", ", tilde(Z)[j], ")")))
abline(a=0,b=1,lty="dashed",col="grey20",lwd=1)
segments(x0=Th,y0=-Th,x1=Th,y1=Th,col="grey20",lwd=1,lty="dashed")
segments(x0=-Th,y0=Th,x1=Th,y1=Th,col="grey20",lwd=1,lty="dashed")
points(Z_j[index],Z_j[index+p],pch=19,col="red",cex=0.5)

lim<-max(Z_j)
xx1 <- c(Th,Th,6,lim*3/2,lim*3/2)
yy1<-c(-Th-10,Th,6,lim*3/2,-Th-10)
xx2<-c(-Th-10,-Th-10,Th,6)
yy2<-c(lim*3/2,Th,Th,6)
xx3 <- c(-Th-10,-Th-10,Th,Th)
yy3 <- c(-Th-10,Th,Th,-Th-10)
polygon(xx1, yy1, border = NA, col = rgb(0.2, 0.6, 0.3, alpha = 0.33))
```

```
polygon(xx2, yy2, border = NA, col = rgb(0.8, 0.3, 0.2, alpha = 0.33))
polygon(xx3, yy3, border = NA, col = rgb(0.0, 0.0, 1.0, alpha = 0.33))  #
↪  muted green
legend("topright",
       inset = c(-0.5, 0),
       legend = c("Null mutations", "Non-null mutations"),
       col = c("black", "red"),
       pch = c(19, 19),
       xpd = NA,
       cex=0.6)
legend("bottomright",
       inset = c(-0.52, 0.59),
       legend = c("Selected mut.", "Non selected mut.", "Ignored mut."),
       ↪  # labels
       fill = c(rgb(0.2, 0.6, 0.3, alpha = 0.33),
                rgb(0.8, 0.3, 0.2, alpha = 0.33),
                rgb(0.0, 0.0, 1.0, alpha = 0.33)),
       xpd = NA,
       cex=0.6)

# Benjamini-Hochberg
mod <- lm(y ~ Xcn - 1) # no intercept
pvalues <- as.numeric(coef(summary(mod))[, 4])
cutoff <- max(c(0, which(sort(pvalues) <= q * (1:p) / p)))
ind_bhq <- which(pvalues <= q * cutoff / p)
TOTP_bhq <- length(ind_bhq)
bhq_mut <- mutations[ind_bhq]
bhq_mut <- mutations[ind_bhq]
pos_sel_bhq <- numeric(length(bhq_mut))
for (i in 1:length(bhq_mut)) {
  pos_sel_bhq[i] <- as.numeric(gsub("\\D", "", bhq_mut[i]))
}
tp_bhq_sel <- sum(unique(pos_sel_bhq) %in% pos)
tot_bhq_sel <- length(unique(pos_sel_bhq))
fp_bhq_sel <- tot_bhq_sel - tp_bhq_sel

# Naive
```

```
alpha <- q
ind_naive <- which(pvalues <= alpha)
TOTP_naive <- length(ind_naive)
naive_mut <- mutations[ind_naive]
naive_mut <- mutations[ind_naive]
pos_sel_naive <- numeric(length(naive_mut))
for (i in 1:length(naive_mut)) {
  pos_sel_naive[i] <- as.numeric(gsub("\\D", "", naive_mut[i]))
}
tp_naive_sel <- sum(unique(pos_sel_naive) %in% pos)
tot_naive_sel <- length(unique(pos_sel_naive))
fp_naive_sel <- tot_naive_sel - tp_naive_sel

# Bonferroni
ind_bonf <- which(pvalues <= alpha / p)
bonf_mut <- mutations[ind_bonf]
pos_sel_bonf <- numeric(length(bonf_mut))
for (i in 1:length(bonf_mut)) {
  pos_sel_bonf[i] <- as.numeric(gsub("\\D", "", bonf_mut[i]))
}
tp_bonf_sel <- sum(unique(pos_sel_bonf) %in% pos)
tot_bonf_sel <- length(unique(pos_sel_bonf))
fp_bonf_sel <- tot_bonf_sel - tp_bonf_sel

# Holm
indices <- c(1:p)
i0 <- min(which(sort(pvalues) > alpha / (p - indices + 1)))
ind_holm <- which(pvalues < alpha / (p - i0 + 1))
TOTP_holm <- length(ind_holm)
holm_mut <- mutations[ind_holm]
pos_sel_holm <- numeric(length(holm_mut))
for (i in 1:length(holm_mut)) {
  pos_sel_holm[i] <- as.numeric(gsub("\\D", "", holm_mut[i]))
}
tp_holm_sel <- sum(unique(pos_sel_holm) %in% pos)
tot_holm_sel <- length(unique(pos_sel_holm))
fp_holm_sel <- tot_holm_sel - tp_holm_sel
```

```r
# Bar plot to assess the perfomance of Multiple testing procedure
# on HIV data. The following is the code for figure (3.12)

library(tidyverse)
data_comp <- data.frame(
  Procedures = c("Naive", "Bonferroni", "Holm", "BHq", "Knockoff",
  ↪  "Knockoff+"),
  Not_in_TSM_list = c(fp_naive_sel, fp_bonf_sel, fp_holm_sel, fp_bhq_sel,
  ↪  fp_knock_sel, fp_knock_plus_sel),
  In_TSM_list = c(tp_naive_sel, tp_bonf_sel, tp_holm_sel, tp_bhq_sel,
  ↪  tp_knock_sel, tp_knock_plus_sel)
)

data_comp_long <- data_comp %>%
  gather(key = "Type", value = "Count", Not_in_TSM_list, In_TSM_list)

data_comp_long$Type <- recode(data_comp_long$Type,
  "Not_in_TSM_list" = "Not in TSM list",
  "In_TSM_list" = "In TSM list"
)

data_comp_long$Type <- factor(data_comp_long$Type, levels = c("Not in TSM
↪  list", "In TSM list"))
colors <- c("Not in TSM list" = "salmon", "In TSM list" = "blue")
ggplot(data_comp_long, aes(x = Procedures, y = Count, fill = Type)) +
  geom_bar(stat = "identity", position = "stack", alpha = 0.7, color =
  ↪  "black") +
  scale_fill_manual(values = colors, name = NULL) +
  labs(
    y = "# HIV-1 protease positions selected",
    title = "Resistance to Nelfinavir"
  ) +
  geom_hline(yintercept = length(pos), lty = "dashed", color = "grey50") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
# the grey line represents the number of positions selected in the TSM
```

```
# list. each bar represents the total amount of positions selected
# where the blue part are true positives, namely positions selected
↪  appearing in the TSM list, while
# the orange part are false positives, namely it represents for each
↪  approach the number of
# mutations selected that were not in the TSM list.
```

# Bibliography


(2006). Hiv drug resistance database: https://hivdb.stanford.edu/pages/published_analysis/genophenoPNAS2006/.

(2019). Knockoff matlab tutorial 2: https://web.stanford.edu/group/candes/knockoffs/software/knockoffs/tutorial-2-matlab.html.

(2025). Uniprot hiv-1 protease wild-type sequence: https://www.uniprot.org/uniprotkb/O90777/entry#sequences.

(2025). Wikipedia, hiv/aids: https://en.wikipedia.org/wiki/HIV/AIDS.

Barber, R. & Candès, E. (2015a). Controlling the false discovery rate via knockoffs. *The Annals of Statistics* , 2055–2085.

Barber, R. & Candès, E. (2015b). Supplement to "controlling the false discovery rate via knockoffs". *The Annals of Statistics* .

Benjamini, Y. & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: series B* , 289–300.

Billingsley, P. (1995). *Probability and Measure*. Wiley.

Breiman, L. (2001). Statistical modeling: The two cultures. *Statistical Science* , 199–231.

Casella, G. & Berger, R. (2008). *Statistical Inference*. Brooks/Cole.

Efron, B. & Hastie, T. (2021). *Computer Age Statistical Inference*. Cambridge University Press.

Gelman, A. & Loken, E. (2014). The statistical crisis in science. *American scientist* , 102(6),460–465.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics* , Vol.6, No.2,65–70.

Ioannidis, J. (2005). Why most published research findings are false. *PLoS Medicine* , 2(8),124.

Lang, S. (1987). *Linear Algebra*. Springer.

Pace, L. & Salvan, A. (2001). *Introduzione alla statistica II*. Cedam.

Rhee, S.-Y., Taylor, J., Wadhera, G., Ben-Hur, A., Brutlag, D. L. & Shafer, R. W. (2006). Genotypic predictors of human immunodeficiency virus type i drug resistance. *Proc. Natl. Acad. Sci. USA* , 17355–17360.

Solari, A. (2023). Lecture notes on multiple testing and knockoff filter. *Statistical Inference II, PhD in Economics, Statistics and Data Science* , University of Milano–Bicocca.

Çinlar, E. (2011). *Probability and Stochastics*. Springer.